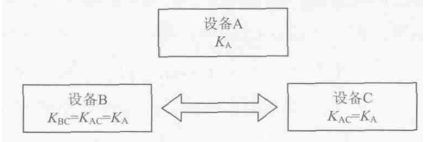
# 材料说明

现在已知的蓝牙安全漏洞包括如下几个：跳频时钟、PIN码、链路密钥欺骗、加密密钥流重复。针对这些漏洞，产生了一些针对蓝牙设备的攻击。

## 蓝牙中间人攻击原理

在两个设备之间的攻击者截获数据一方发送的数据后再转发给另一方，可在不影响双方通信的情况下获得双方通信的内容，是一种广泛应用于无线网络的攻击方式。蓝牙 4.0 版本的低功耗蓝牙技术（Bluetooth Low Energy，BLE）在设计初始时有防范中间人攻击的安全措施，但是在产品阶段考虑到产品功耗成本等因素，这方面并没有得到足够的重视，依然容易受到攻击。最常见的是用软硬件结合的蓝牙攻击设备伪造BLE 通信进行中间人攻击。中间人攻击示意图下图所示。

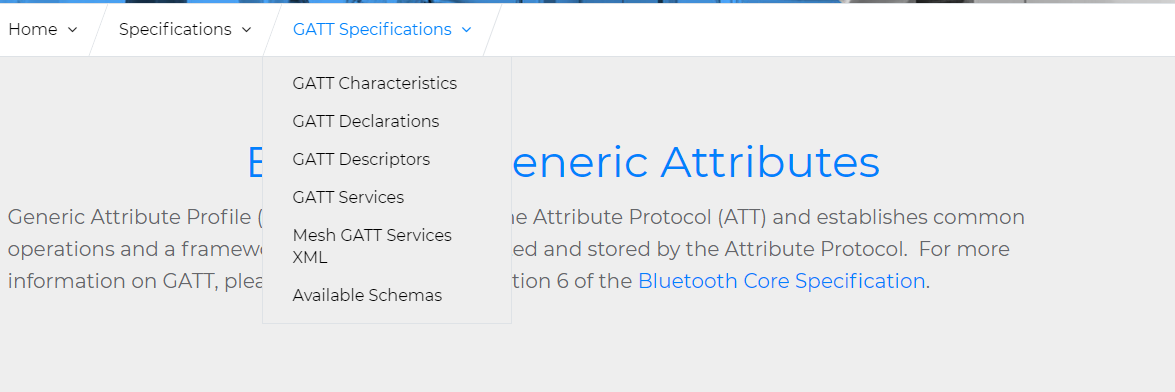


## 蓝牙数据包简析

（前面应该还有一些关于蓝牙的介绍，但是这个部分可以后面来做，也较少）

### 2.2 GATT层

关于GATT的一些服务的名称，可通过下面的网址进行查阅：

<https://www.bluetooth.com/specifications/gatt/>

蓝牙协议中包含了许多种GATT规范，每个规范适配一种用户案例，比如FindMe规范适配查找物件的场景，心率传感器规范适配心率测量场景。

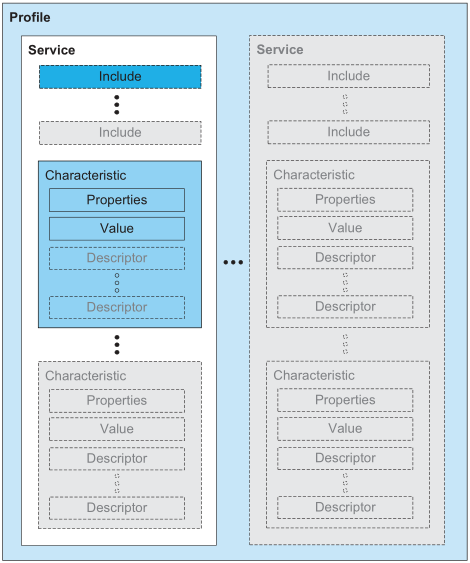
每个规范均中均有若干服务项和特征项，服务项和特征项都属于属性实体，它们携带了通信中传输的数据。

服务项分为主要服务和次要服务，主要服务可以引用（Include）另一个主要服务或次要服务，客户端设备可以通过“主要服务发现过程”获取主要服务信息。

特征项包括一个声明、配置、数据和描述符。描述符用于描述特征项的数据如何被访问和展示。

规范、服务项和特征项之间有明确的包含关系，一个GATT规范中可以包括多个服务项，一个服务项中可以包括多个特征项。

GATT的规范结构框图如下：



#### 2.2.1属性

一个属性包含句柄、UUID(类型)、值，句柄是属性在GATT表中的索引，在一个设备中每一个属性的句柄都是唯一的。UUID包含属性表中数据类型的信息，它是理解属性表中的值的每一个字节的意义的关键信息。在一个GATT表中可能有许多属性，这些属性能可能有相同的UUID。

#### 2.2.2 特性

一个特性至少包含2个属性：一个属性用于声明，一个属性用于存放特性的值。

所有通过GATT服务传输的数据必须映射成一系列的特性，可以把特性中的这些数据看成是一个个捆绑起来的数据，每个特性就是一个自我包容而独立的数据点。 例如，如果几块数据总是一起变化，那么我们可以把它们集中在一个特性里。

#### 2.2.3描述符

任何在特性中的属性不是定义为属性值就是为描述符。描述符是一个额外的属性以提供更多特性的信息，它提供一个人类可识别的特性描述的实例。

然而，有一个特别的描述符值得特别地提起：客户端特性配置描述符(Client Characteristic Configuration Descriptor，CCCD)，这个描述符是给任何支持通知或指示功能的特性额外增加的。

在CCCD中写入“1”使能通知功能，写入“2”使能指示功能，写入“0”同时禁止通知和指示功能。

#### 2.2.4服务

一个服务包含一个或多个特性，这些特性是逻辑上相关的集合体。

GATT服务一般包含几块具有相关的功能，比如特定传感器的读取和设置，人机接口的输入输出。组织具有相关的特性到服务中既实用又有效，因为它使得逻辑上和用户数据上的边界变得更加清晰，同时它也有助于不同应用程序间代码的重用。GATT基于蓝牙技术联盟(SIG)官方而设计，SIG建议根据它们的规范设计自己的profile。

#### 2.2.5 profile（数据配置文件）

一个profile文件可以包含一个或者多个服务，一个profile文件包含需要的服务的信息或者为对等设备如何交互的配置文件的选项信息。设备的GAP和GATT的角色都可能在数据的交换过程中改变，因此，这个文件应该包含广播的种类、所使用的连接间隔、所需的安全等级等信息。

需要注意的是： 一个profile中的属性表不能包含另一个属性表。

#### 2.2.6 标准的定制服务和特性

蓝牙技术联盟(SIG)已经定义一些profile、服务、特性和根据协议栈的GATT层定义的属性。但是，协议栈中只实现了一部分应用的BLE服务，那就意味着，只要协议栈支持GATT，就可能为一个应用建立一个它需要的profile和服务。

既然在一个应用中可以支持profile和服务，那么就可以在这个应用中建立一个定制的服务。

#### 2.2.7 UUID

“GATT层”中定义的所有属性都有一个UUID值，UUID是全球唯一的128位的号码，它用来识别不同的特性。

1. 蓝牙技术联盟 UUID

蓝牙核心规范制定了两种不同的UUID，一种是基本的UUID，一种是代替基本UUID的16位UUID。

所有的蓝牙技术联盟定义UUID共用了一个基本的UUID：

0x0000xxxx-0000-1000-8000-00805F9B34FB

为了进一步简化基本UUID，每一个蓝牙技术联盟定义的属性有一个唯一的16位UUID，以代替上面的基本UUID的‘x’部分。例如，心率测量特性使用0X2A37作为它的16位UUID，因此它完整的128位UUID为：

0x00002A37-0000-1000-8000-00805F9B34FB

虽然蓝牙技术联盟使用相同的基本UUID，但是16位的UUID足够唯一地识别蓝牙技术联盟所定义的各种属性。

蓝牙技术联盟所用的基本UUID不能用于任何定制的属性、服务和特性。对于定制的属性，必须使用另外完整的128位UUID。

1. 供应商特定的UUID

SoftDevice 根据蓝牙技术联盟定义UUID类似的方式定义UUID：先增加一个特定的基本UUID，再定义一个16位的UUID（类似于一个别名），再加载在基本UUID之上。这种采用为所有的定制属性定义一个共用的基本UUID的方式使得应用变为更加简单，至少在同一服务中更是如此。

使用软件nRFgo Studio非常容易产生一个新的基本UUID：

例如，在LED BUTTON示例中，采用0x0000xxxx-1212-EFDE-1523-785FEABCD123作为基本UUID。

蓝牙核心规范没有任何规则或是建议如何对加入基本UUID的16位UUID进行分配，因此你可以按照你的意图来任意分配。

例如，在LED BUTTON示例中，0x1523作为服务的UUID，0x1524作为LED特性的UUID，0x1525作为按键状态特性的UUID。

#### 2.2.8 空中操作和性质

大部分的空中操作事件都是采用句柄来进行的，因为句柄能够唯一识别各个属性。如何使用特性依据它的性质，特性的性质包括：

1）写

2）没有回应的写

3）读

4）通知

5）指示

写和没有回应的写

写和没有回应的写允许GATT客户端写入一个值到GATT服务器的一个特性中。它们之间不同的地方在于没有回应的写事件没有任何应用层上的确认或回应。

读

读性质表明一个GATT客户端可以读取在GATT服务器中特性的值。

通知和指示

通知和指示性质允许GATT服务器在其某个特性改变的时候对GATT客户端进行提醒，通知和指示之间不同之处在于指示有应用层上的确认，而通知没有。

## 针对BLE设备中间人攻击

这是一个有趣的项目，其中介绍了如何对使用蓝牙低功耗（BLE）进行数据交换的测温枪进行中间人攻击。由于测温枪使用低功耗蓝牙（Bluetooth Low Energy）（是蓝牙4.0的子集），因此从理论上讲任何BLE设备都可以控制灯泡。但是要控制灯泡，必须了解与灯泡进行通信的协议，并且本文将向您展示如何使用Bluefruit LE嗅探器和其他工具对蓝牙低功耗小工具进行中间人攻击。

我们需要如下的一些硬件：

* 蓝牙4.0 USB模块。确保模块支持低功耗蓝牙。 4.0版之前的旧版蓝牙不支持BLE！
* PC计算机。最好使用带有Linux系统的计算机，并保证计算机带有USB口，在Linux系统下能够使用一些很好用的小工具，十分便于我们调试，USB口则是其他硬件设备与计算机连接的必备接口。
* 手机（Andriod或IOS），通过手机端的软件与测温枪相连。
* 测温枪，能够测量身体的温度数据，并通过蓝牙连接到app传输温度数据给手机端。
* Bluefruit LE Sniffer。 Bluefruit LE Sniffer是Bluefruit LE friend的特殊版本，可以允许查看来自其他设备的BLE数据包。

### GATT服务初探

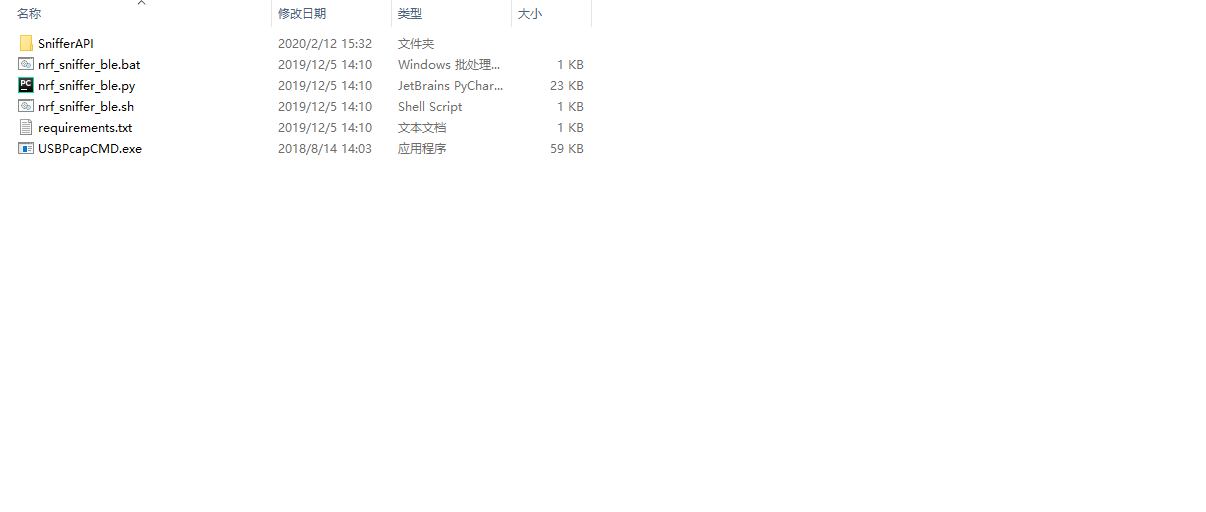
我使用了IOS上的一款名为LightBlue的蓝牙调试软件进行初步测试，这是测试图：



从这张图我们可以看到我们的测温枪的设备名和使用的服务名以及设备的一些信息以及使用该服务发送的一些值。

### WireShrak抓包

我们继续使用WireShark软件对测温枪的发送数据进行探究，要注意的是，在进行这些活动时，测温枪不能连接其专用的APP，否则将会无法抓包。同时，我在github上上传了名为nrf\_sniffer\_for\_bluetooth\_le\_3.0.0\_129d2b3的wireshark插件，将其中的extcap文件夹复制到wireshark对应的文件夹，即可在wireshark中使用低功耗蓝牙抓取的插件，使抓包过程更为便捷：

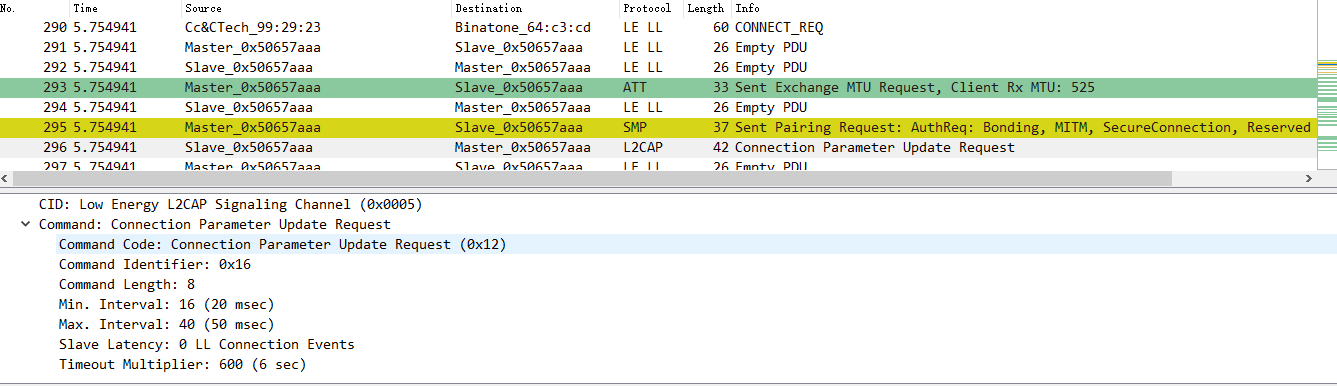


复制后的文件夹



插件示意

下面是WireShark抓包的图：



### 进行MITM攻击

#### 3.3.1接收

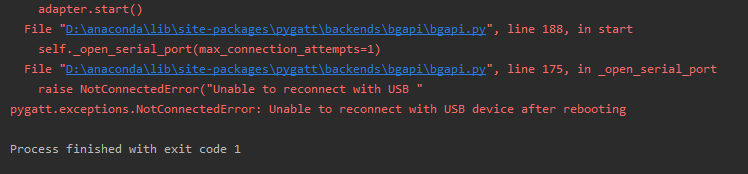
从上面的分析可知，我们这里可以实施较为初级的攻击，即首先将测温枪连接到自己的电脑，同时读取数据，得到测温枪发送的数据格式，但是这一步并不是最重要的。

因为在我们的项目中，测温枪发送温度这一最重要的步骤我们是无法探知的，这不同于其他一些蓝牙设备的交互逻辑，在我们的项目中，从设备（即测温枪）是数据的发送方，所以他发送的数据必须要连接上主设备后才开始发送，所以我们不能达到他发送温度数据的格式。但是如果是一个使用手机蓝牙控制小灯这样的应用中，就算没有连接上小灯，我们仍然可以通过手机发送命令，这就给了我们探究该命令数据格式的机会。但是所幸我们的项目发送数据必须使用规定的服务方式，所以理论上数据格式是固定的。

然后这一步也是没有成功，因为这一步需要使用蓝牙接收器来连接上测温枪接收数据并分析，但是我的电脑当蓝牙接收器连接上的时候没办法识别COM口，所以就没办法搜索设备。：



蓝牙接收器adapter

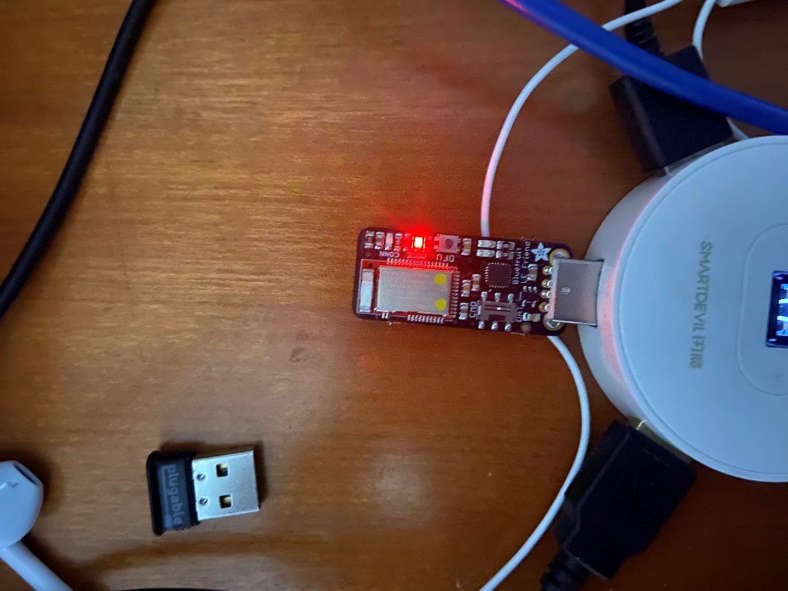


代码运行失败截图

这个部分属于硬件问题，可能出问题的地方是这个蓝牙接收器、或者是windows下这个代码的问题，因为我现在没有Linux系统的环境，所以这个问题暂时是无解的。

#### 3.3.2发送

接收到了数据，接下来我们分析一下这个数据就将它发送出去，发送的时候使用LE Friend：



LE friend

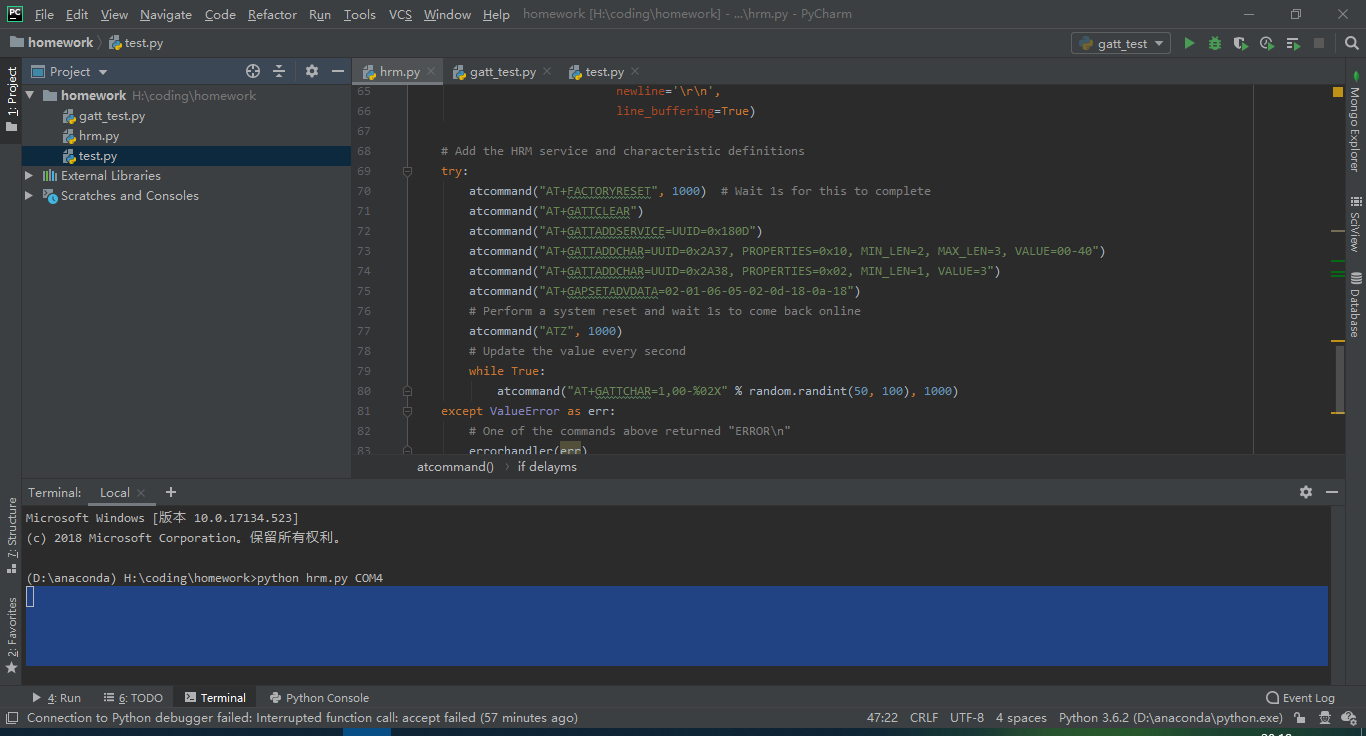
同样的，这个LE friend也是亮红灯，无法工作，但是你给的那个sniffer和这个LE friend是一模一样的东西，然后它可以使用，所以我用了你给的sniffer：



Sniffer

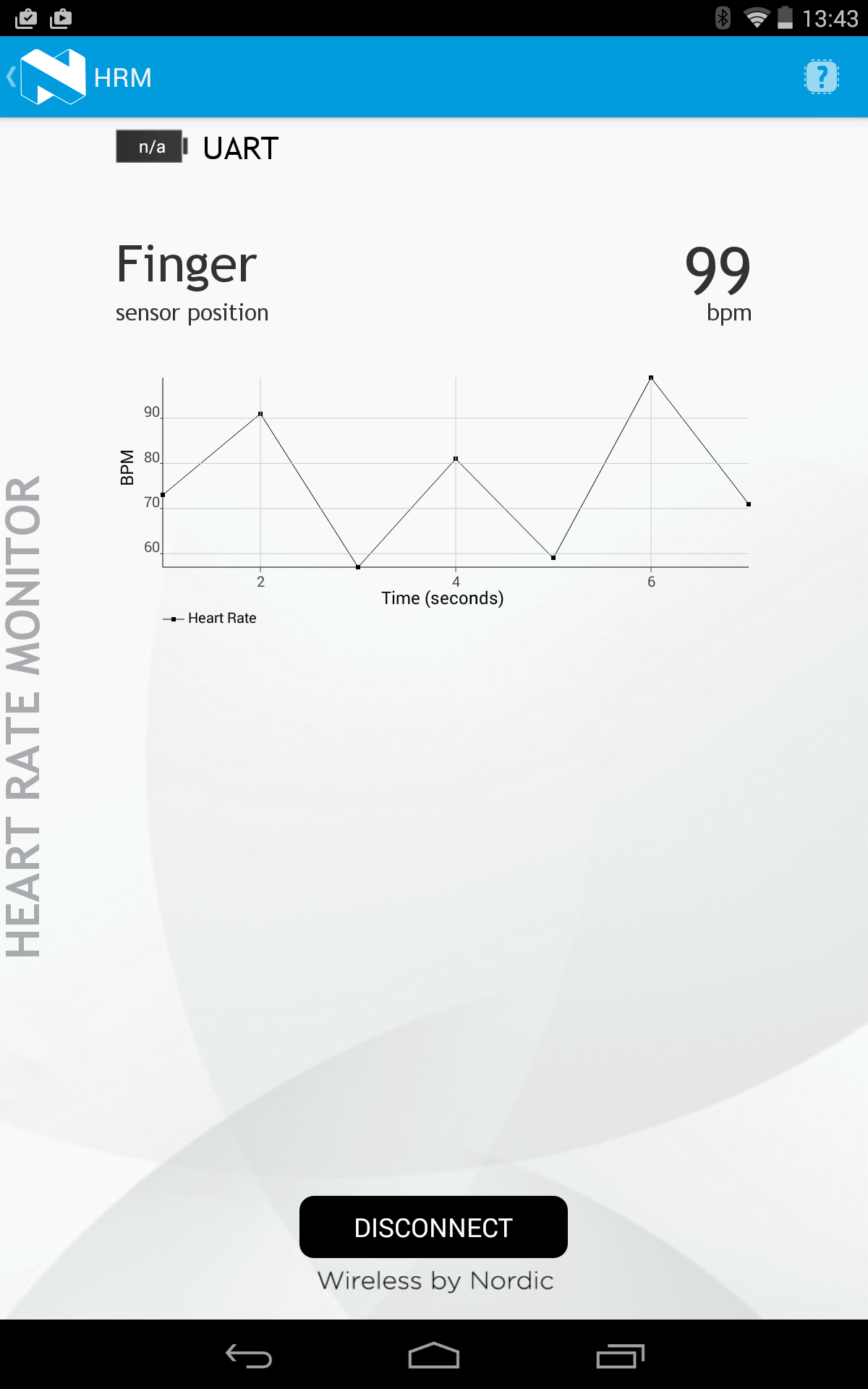
可以看到，sniffer蓝灯的部分是在进行嗅探，而黄灯的地方是在向外发送数据。

程序的运行时如下：



程序运行截图

同时，在手机端我们使用一款名叫nRF Toolbox的软件，可以看到我们自己修改的值被发送出来：



这就是在发端我们做的工作，到这里，一个完整的中间人攻击就结束了。

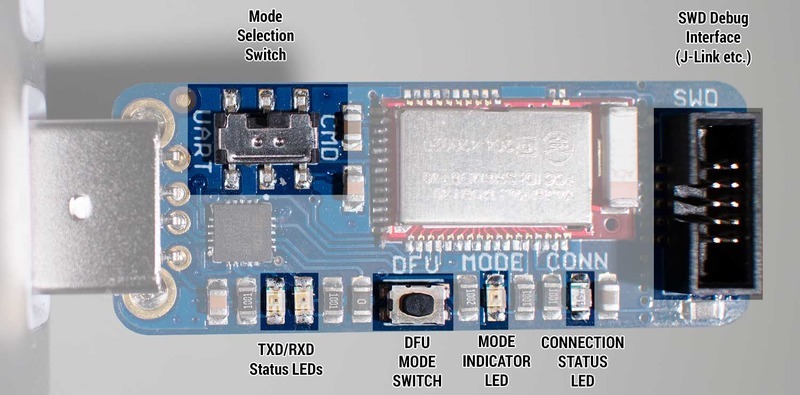
新：

接下来我们在Linux上做一下这些动作，下面的操作使用了LE friend ，因为按照官方的教程，直接使用LE friend也是可以的，所以我想试一试。

1. LE friend介绍

这是LE friend的硬件示意图，从上面我们可以清楚的看到各个指示灯的含义

后面的那个口（即SWD Debug）我们不需要用，本身硬件也没有，所以不用管，DFU是用于烧写LE friend固件的，也不用管。



Mode Selection Switch

可以在“ CMD”（命令模式）和“ UART”（数据模式）之间切换此开关，这将改变设备在终端仿真器中的行为方式。

TXD/RXD Status LEDs

提供这两个LED主要用于调试目的，以帮助您可视化USB CDC接口上的传入和传出字符。Mode Indicator LED

该指示灯用于指示设备当前正在运行的模式（数据，命令或DFU）。

Connection Status LED

当BLEFriend已成功与另一个BLE设备建立连接时，此LED将启用，并且对于调试目的很有用。

1. 操作模式

数据模式

数据模式使用BLE UART服务，并将BLEFriend变成BLE中央设备（您的手机或平板电脑）与您的PC或支持USB的设备之间的HW UART桥接器。 要使用数据模式，只需将BLEFriend模块连接到PC的USB端口，将模式选择开关设置为UART，然后使用您喜欢的终端软件以9600 bps的速度开始发送或接收数据。

如果MODE LED闪烁两次，然后延迟三秒钟，则您处于数据模式：

（视频演示了如何亮灯）

命令模式用于将配置命令发送到模块或检索有关模块本身或BLE连接另一端连接的设备的信息。 要使用命令模式，请确保将模式选择开关设置为CMD，然后使用您喜欢的终端仿真器以9600 bps的速度输入有效的Hayes AT样式命令（例如，使用``ATI''显示有关该模块的一些基本信息）。 如果MODE LED闪烁三下，然后延迟三秒钟，则您处于命令模式：



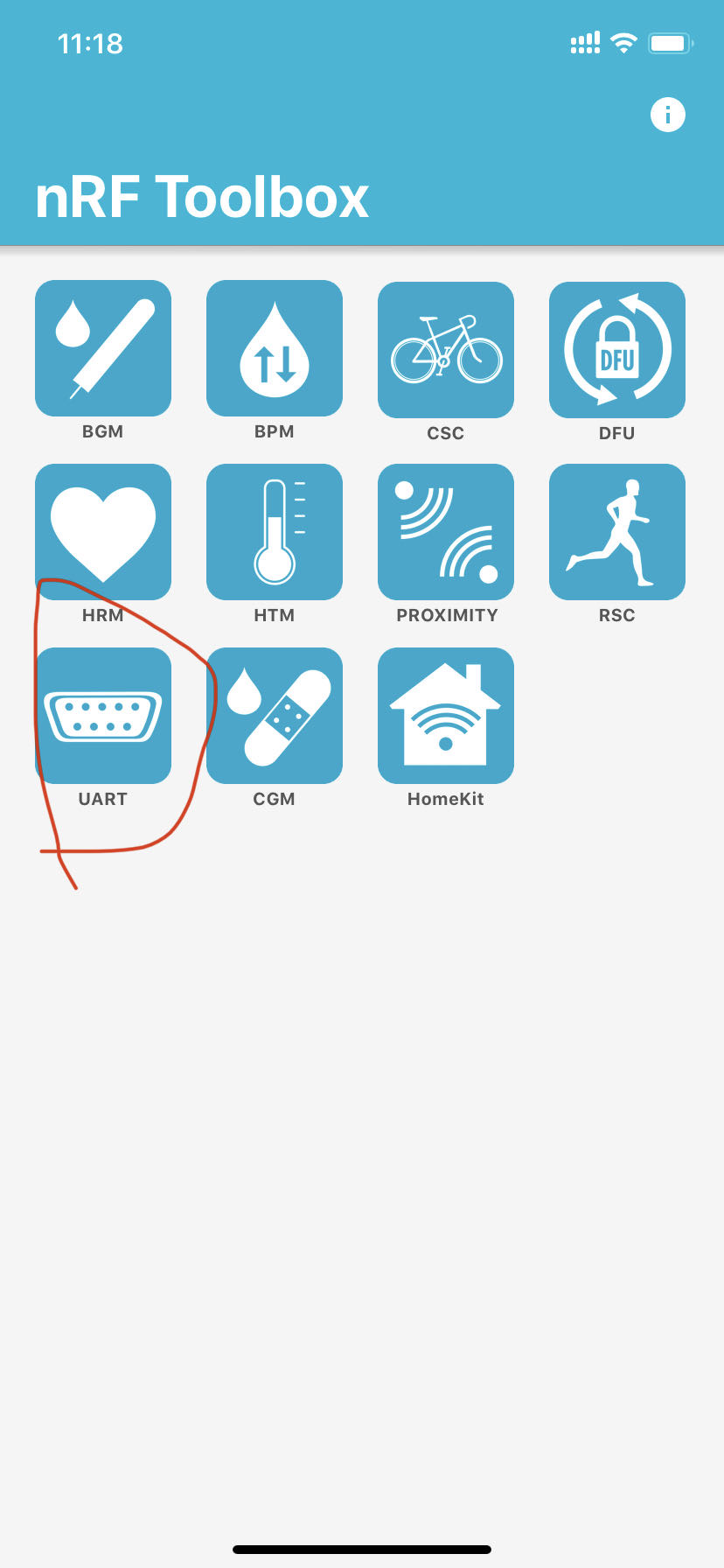
1. 测试操作（这一步只是进行测试，和我们最后做的没关系，所以这些工具可以只是作为了解或是写在论文中，不必强求能够运行）

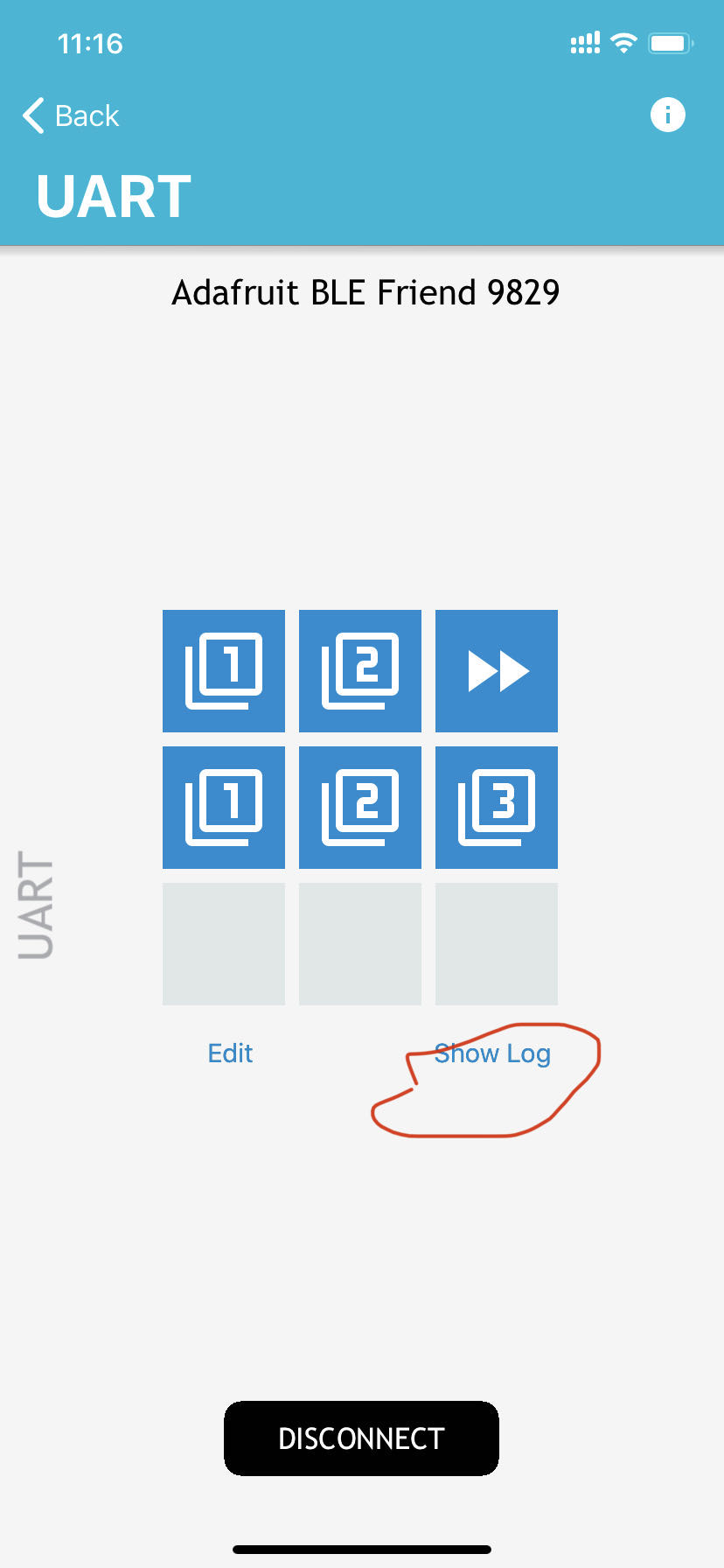
首先需要在电脑端安装CoolTerm，这是一个免费且功能齐全的终端仿真器软件包。<http://freeware.the-meiers.org/>

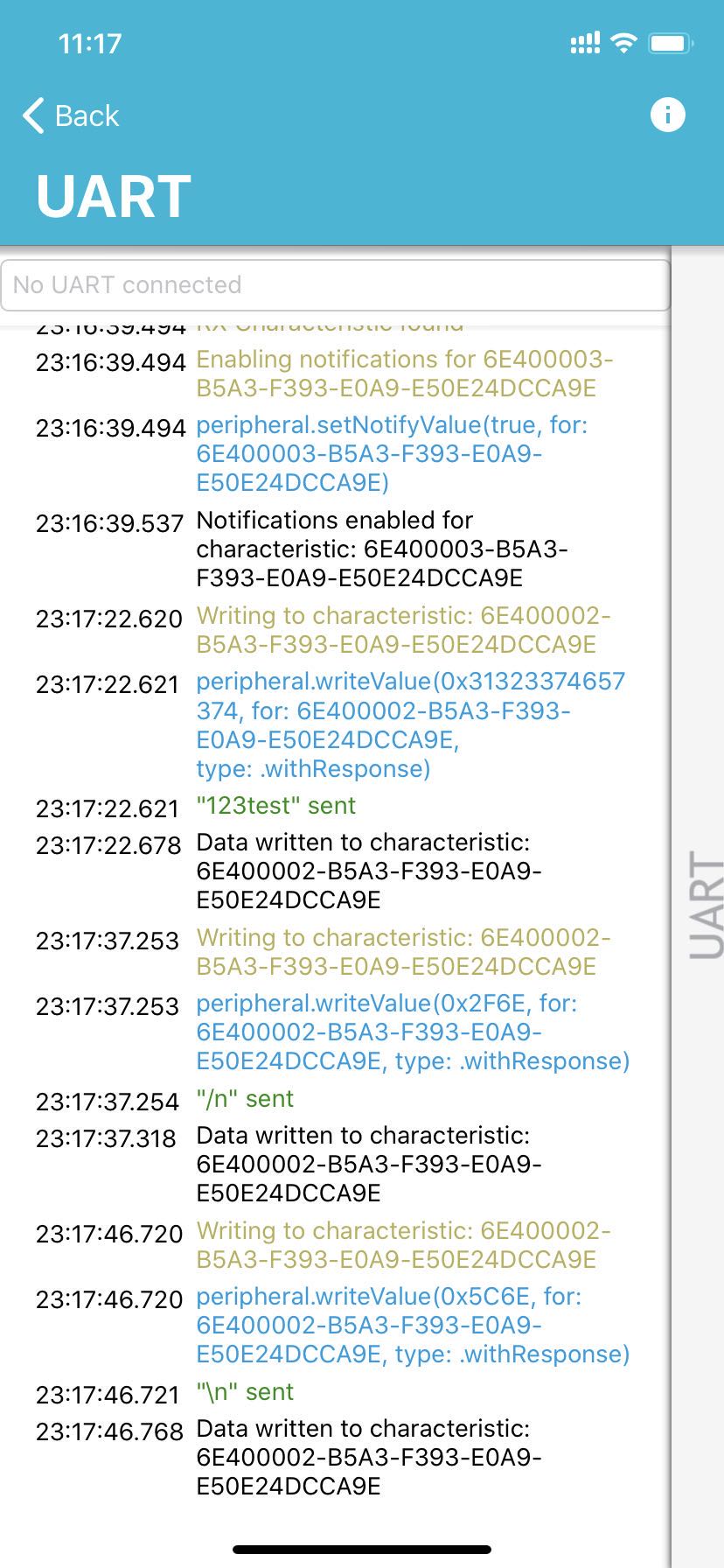
这里可以进行下载

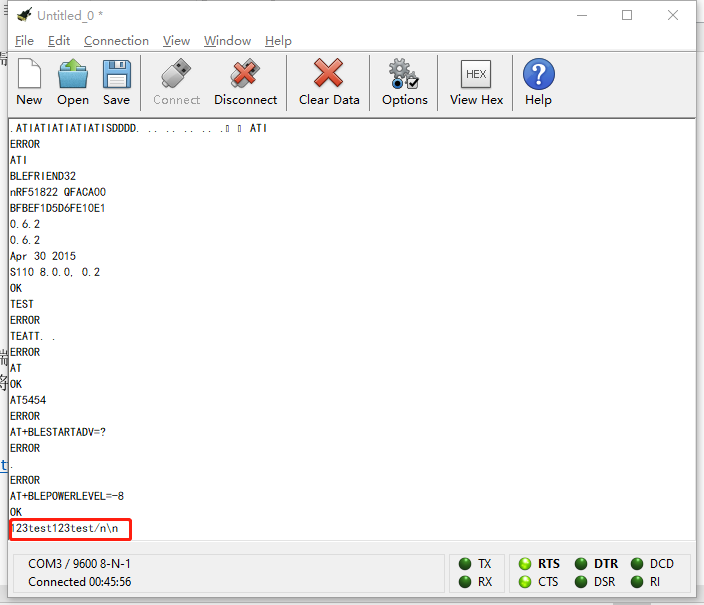
手机端需要安装

然后将LE friend设置为数据模式，手机端发送的数据就能在电脑端看到了：









将LE friend调至cmd模式，输入命令可以看到上面的一些消息打印出来

1. 模拟蓝牙

通过上面的测试我们能够知道，将LE FRIEND设置为cmd模式，我们就可以通过输入命令的方式模拟一个蓝牙设备。

使用linux系统

我们在这里想要使用bluez平台，接下来我们将纤细介绍这个工具，它包含了许多实用的蓝牙工具并且便于使用。不幸的是，没有跨Windows，Mac，Linux等平台的跨平台蓝牙协议栈或API。因此，如果想使用其他平台，则需要查看该平台的蓝牙低能耗协议栈和API。

Bluez

BlueZ是Linux官方蓝牙协议栈。它是一个基于GNU General Public License (GPL)发布的开源项目，从Linux2.4.6开始便成为Linux 内核的一部分。其基础代码均是由就职于Qualcomm(高通)的Maxim Krasnyansky完成的。包括：HCI，L2CAP，RFCOMM和基本socket的实现。Marcel Holtmann开发层的协议和应用，包括：BNEP, CMTP等。当然，这些中也有Maxim Krasnyansky的参预。有部分代码由Nokia提供的。

1. BlueZ支持蓝牙核心层和协议，它灵活、高效，以模块化方式实现，具有以下特点：

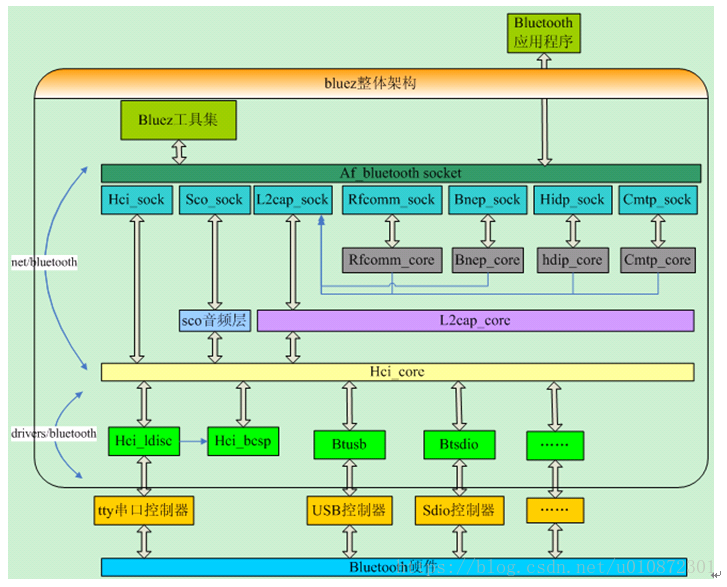
* 完整的模块化实现
* 均衡的多处理安全
* 支持多线程数据处理
* 支持多个蓝牙设备
* 硬件抽象
* 提供所有层的标准socket接口
* 支持设备级和服务级安全保证

1. BlueＺ包含多个相互独立的模块：

* Linux内核蓝牙子系统核心
* L2CAP 和 SCO 音频内核层
* RFCOMM, BNEP, CMTP 和 HIDP内核实现
* HCI UART, USB, PCMCIA 和虚拟设备驱动
* 通用的蓝牙和SDP库及守护进程
* 配置和测试小工具
* 协议解析和分析工具

我们主要使用了上面标红的这些模块

下图是bluez的一个架构示意图：



Bluez的安装

在ubuntu14.04 下编译安装Bluez-5.52

1、安装相关依赖库：

sudo apt-get install libusb-dev libdbus-1-dev libglib2.0-dev automake libudev-dev libical-dev libreadline-dev

2、下载Bluez-5.28压缩包并解压:

wget http://www.kernel.org/pub/linux/bluetooth/bluez-5.52.tar.xz

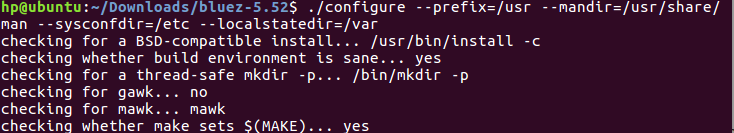
tar xvf bluez-5.52.tar.xz

进入解压后的目录：

cd bluez-5.52

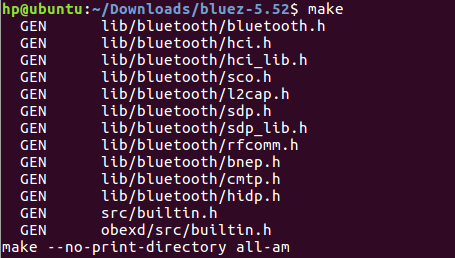
3、执行配置文件

./configure --prefix=/usr --mandir=/usr/share/man --sysconfdir=/etc --localstatedir=/var

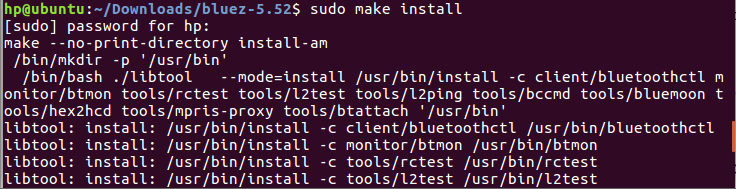


4、执行make和install

make



sudo make install



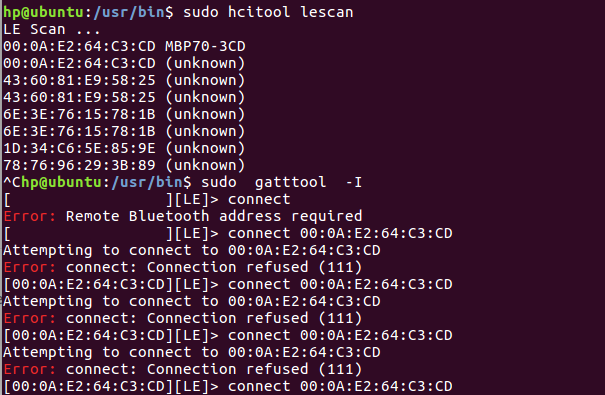
没有提示任何错误的话，安装完毕。

Bluez的使用

1. lescan扫描低功耗蓝牙设备

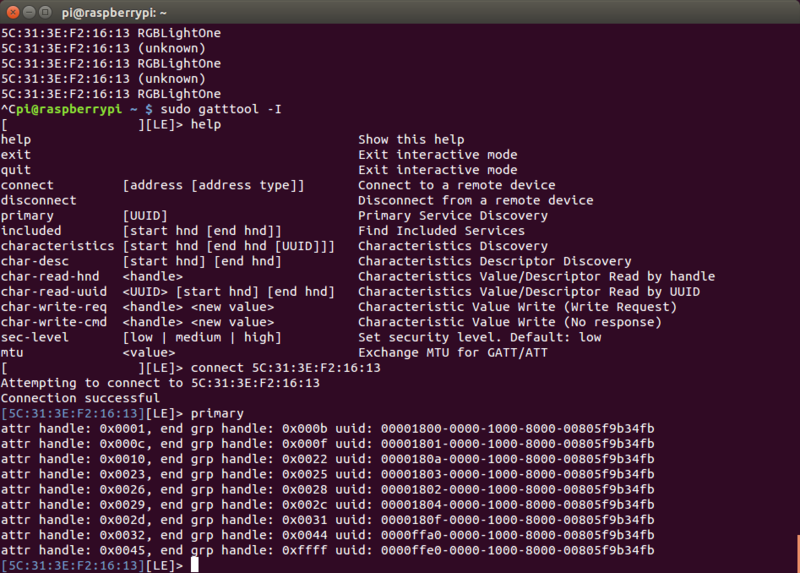


1. connect   //连接低功耗设备



1. primary

在连接成功后输入这个命令，可以看到该低功耗蓝牙设备能够支持的服务，和uuid



1. char-desc 0x0028 0x0028

可以运行的另一个命令是char-desc命令，以获取有关特定特征的详细信息。我运行以下命令来查询0x0028特性：

