

### Introducing the Adafruit Bluefruit LE Sniffer

Created by Kevin Townsend

Capturing from \\.\pipe\wireshark_nordic_ble [Wireshark 1.12.1 (v1.12.1	.1-0-g01b65bf from master-1.12)]
<u>File Edit View Go Capture Analyze Statistics Telephony Tools</u>	s Internals <u>H</u> elp
◎ ◎ ∡ ■ ₫   🖿 🗎 🗙 🔁   역 수 수 🗛 👍 🚣	L   🗐 🖬   Q, Q, Q, 💟   🖉 🕺 🧏 🎉   🙀
Filter	Expression Clear Apply Save
No. Time Source Destination	Protocol Length Info
55593 2854.95085 slave Master	LE LL 26 Empty PDU
55594 2854.99892 Master Slave	LE LL 26 Empty PDU
55595 2855,00055 Slave Master	ATT 34 Rcvd Handle Value Notification, Handle: 0x001c
55596 2855.04974 Master Slave	LE LL 26 Empty PDU
55597 2855.05107 slave Master	LE LL 26 Empty PDU
55598 2855.09619 Master Slave	LE LL 26 Empty PDU
55599 2855.09776 Slave Master	LE LL 26 Empty PDU
55600 2855.19274 Master Slave	LE LL 26 Empty PDU
55601 2855.19395 Slave Master	LE LL 26 Empty PDU
55602 2855.29033 Master Slave	LE LL 26 Empty PDU
55603 2855.29210 slave Master	LE LL 26 Empty PDU
55604 2855.33914 Master Slave	LE LL 26 Empty PDU
55605 2855. 34070 slave Master	LE LL 26 Empty PDU
55606 2855. 38816 Master Slave	LE LL 26 Empty PDU
55607 2855.38971 slave Master	LE LL 26 Empty PDU
Frame 55595: 34 bytes on wire (272 bits), 34 byte	es captured (272 bits) on interface 0
∃ Nordic BLE sniffer meta	
Bluetooth Low Energy Link Layer	
Access Address: 0x84d81175 Data Header: 0x080a	
Data Header: 0x080a 000 = RFU: 0	
0 = More Data: False	
1 = Sequence Number: True	
0 = Next Expected Sequence Number: Fa	alse
	or a complete L2CAP message with no fragmentation (0x02)
000 = RFU: 0	or a comprete Ezew message with no tragmentation (0x02)
0 1000 = Length: 8	
Bluetooth L2CAP Protocol	
Length: 4	
CID: Attribute Protocol (0x0004)	
Bluetooth Attribute Protocol	
Opcode: Handle Value Notification (0x1b)	
Handle: 0x001c	
Value: 54	
0000 11 06 1b 01 6d 49 06 0a 01 0d 2b 2d 85 99 00	
0010 00 75 11 d8 84 0a 08 04 00 04 00 1b 1c 00 54	
0020 e9 73	.5
Bluetooth Attribute Protocol (btatt), 4 bytes Packets: 57843 · Di	Profile: Default

https://learn.adafruit.com/introducing-the-adafruit-bluefruit-le-sniffer

Last updated on 2022-12-01 02:20:21 PM EST

### Table of Contents

Introduction	5
Using with Sniffer V2 and Python3	5
<ul> <li>BLE Sniffer Hardware</li> <li>Silicon Labs VCP Driver</li> <li>Python 3</li> <li>Python Serial Support</li> </ul>	
<ul> <li>Install Wireshark</li> <li>Install BLE Sniffer Plugin</li> <li>Final Check and Test Capture</li> <li>Next Steps</li> </ul>	
Working with Wireshark	13
<ul> <li>Working with Wireshark</li> <li>Capturing Exchanges Between Two Devices</li> <li>Scan Response Packets</li> <li>Connection Request</li> <li>Write Request</li> <li>Regular Data Requests</li> <li>Notify Event Data</li> <li>Closing Wireshark and nRF-Sniffer</li> <li>Moving Forward</li> </ul>	
Using with Sniffer V2 (old)	21
<ul> <li>Nordic User Manual</li> <li>nRF Sniffer V2 Multi-Target Application</li> <li>V2 Firmware</li> </ul>	
V2 Wireshark Usage (old)	23
<ul> <li>Install Wireshark</li> <li>Install Wireshark Plugin</li> <li>Dealing With Python 2 vs 3</li> <li>Installing Dependancies</li> <li>Test Capture</li> <li>Windows Install Supplemental Information</li> </ul>	
Using with Sniffer V1 (old)	28
USB Driver Install	28
<ul><li>CP2104 Driver Requirements (Black Boards)</li><li>FTDI Driver Requirements (Blue Boards)</li></ul>	
V1 Sniffer Software	29
<ul> <li>Using the Firmware V1 Sniffer</li> <li>Nordic's nRF Sniffer Utility (Windows only)</li> <li>Python API (Cross-Platform, no Registration)</li> </ul>	
V1 Nordic nRF Sniffer • Getting the Sniffer Utility • Getting Wireshark	31

• Running the Sniffer

#### • Select the Sniffer Target

V1 OS X Support	34
V1 Python API	35
Requirements	

- Download the API
- Using the sniffer.py Wrapper
- Linux
- OS X
- Windows
- Scanning for Devices
- Locating the Log File
- Analyze Data in Wireshark

#### FAQs

### Downloads 44

- Files
- Schematic C2104 Rev

40

### Introduction

Using a special firmware image provided by Nordic Semiconductors and the open source network analysis tool Wireshark, the <u>Bluefruit LE Sniffer</u> () can be used as a low cost Bluetooth Low Energy sniffer.

NOTE: This product can only be used to sniff Bluetooth Low Energy devices. It will not work with classic Bluetooth devices or transactions.



Since nRF-Sniffer is a passive solution that is simply scanning packets over the air, there is the possibility of missing packets using this tool (or any other passive sniffing solution). In order to capture as many packets as possible, be sure to run the sniffer on a USB bus that isn't busy and avoid running it in a virtual machine since this can introduce significant latency over USB.

## Using with Sniffer V2 and Python3

Once things are all setup, usage is fairly easy. However, there are numerous separate items that need to be installed and configured. So the initial setup can be a bit cumbersome. We'll go through each step, but it can also help to have a general understanding of the overall setup.

Here's a simplified diagram of the setup:



Here's a summary of all the parts needed:

- 1. The actual BLE sniffing hardware. This guide uses the Adafruit Bluefruit LE Sniffer with V2 firmware ().
- The BLE Sniffer uses a Silicon Labs CP2104 to provide USB to serial conversion. In order for this to show up as a COM port, the Silicon Labs <u>Virtual COM Port</u> driver () is needed.
- 3. The BLE sniffing plugin uses Python ().
- 4. To talk to the virtual com port from Python, the <u>pyserial module</u> () needs to be installed.
- 5. Wireshark () is the main software front end used to facilitate BLE sniffing and decoding.
- 6. To talk to the BLE sniffer from Wireshark, the Nordic Semiconductor <u>nRF Sniffer</u> for BLE () plugin in is used.

These parts come from numerous different sources - at least 5 different vendors are shown in the diagram above. So this will be quite the journey. Here we go...

### **BLE Sniffer Hardware**

You'll need one of these:



### Bluefruit LE Sniffer - Bluetooth Low Energy (BLE 4.0) - nRF51822

Interested in learning how Bluetooth Low Energy works down to the packet level? Debugging your own BLE hardware, and trying to spot where something is going wrong? Or maybe you're...

https://www.adafruit.com/product/2269

### Silicon Labs VCP Driver

This driver allows the CP2104 chip on the Adafruit BLE Sniffer to show up as a COM port on your PC.



Once installed, a COM port should show up on your PC when the Adafruit BLE Sniffer is plugged into a USB port. It should have CP210x in the name.



This check does not require any of the other software components we install later. So if a COM port is not showing up at this point, do not proceed further until determining why.

### Python 3

If Python 3 is not already installed on your system, go to the Python main page to learn how to download and install it for your specific system:



It should now be possible to launch Python and run some simple commands:



On Windows, try using py to launch Python.

### Python Serial Support

To provide access to the COM port, install the pyserial package.



It should now be possible to launch Python and import the pyserial package:



NOTE: the import is actually **serial**, not pyserial.

### Install Wireshark

Go to the Wireshark main page to learn how to download and install Wireshark for your specific system:



Once complete, it should be possible to run Wireshark and at least get the start screen:



### Install BLE Sniffer Plugin

OK, finally, the thing we actually care about. The thing that will let us talk to the Adafruit BLE Sniffer and do some actual BLE sniffing. Let's download and install that BLE sniffing plugin!

Download Plugin from Nordic

Start by downloading the nRF Sniffer for BLE package from Nordic Semiconductor:

nRF Sniffer for Bluetooth LE

This will be a ZIP file. At the time of this guide, the version is 4.0.

Overview Downloads		
v4.x.x	Selected version 4.0.0 nrf_sniffer_for_bluetooth_le_4.0.0.zip	
	Changelog: • 4.0.0	~

Determine Wireshark Plugin Folder Location (extcap)

We need to install items from the ZIP file downloaded from Nordic into a specific Wireshark folder location. This location is different on different systems. To determine it for your system, do this:



We'll refer to this folder location as the Wireshark extcap folder.

### Install BLE Sniffer Plugin into Wireshark

To install the plugin, simply copy the files shown below from the ZIP downloaded from Nordic into the Wireshark extcap folder location determined above.

Open the ZIP file downloaded from Nordic:



We only need the contents of the extcap folder from the ZIP file.

Extract and copy all of the contents of the extcap folder to the Wireshark extcap folder location.

### Final Check and Test Capture

OK, now we can test things out with some real actual BLE sniffing! woot!

	•						The	Wire	shar	k Net	work /	Analy	zer						
e <u>E</u> c	dit ⊻	iew g	<u>G</u> o 9	aptu	re j	Anah	yze	Stat	istics	Tele	phony	Win	eless	Tool	s <u>H</u> el	>			
	E.	0		i) (	X	6	Q	<	> :	2.10	11		Ξ			11			
Apply	y a dis	play f	ilter.	<0	trl-/>														-
	-																		
	w	elcon	ne to	Wir	esha	irk													
	Ca	ptu	re																
		ing th	is filt	er: 🗍	Ent	ter a	capt	ture f	iter.					-	All inter	faces	show	n -	
	_				-	_													
		enol										_							
		Looph	NACK	10								_							
		bluet	onth.	mani	inr.														
		nflog	oour									_	_						
	_	-									_	_							
	۰	nRF 5	iniffe	for I	aluet	ooth	LE:	(dev)	ttyUS	80-3.2	2	_							
		÷					_	-			_	_							
										re: dp	auxmo	n							
		Rande							π			_							
												_							
		SSH	ennos	e cap															
		arn	ennos	e cap															
	Le	arn							and	Answ	ers	Mail	ing Li	ists					
	Lei	arn e's G	uide	- 14	riki	Qu	esti	ions						ists					
	Lei	arn e's G	uide	- 14	riki	Qu	esti	ions			ers -			ists					
	Use You	arn are ru	uide	- Wir	fiki esha	Qu	esti	ions				3.2.3	8-1).						
Re	Lei	arn are ru	uide	- Wir	fiki esha	Qu	esti	ions				3.2.3					Profile	: Defau	dt
Re	Use You	arn are ru	uide	- Wir	fiki esha	Qu	esti	ions				3.2.3	8-1).				Profile	: Defau	<i>it</i>
Re	Use You	arn are ru	uide	- Wir	fiki esha	Qu	esti	ions				3.2.3	8-1).				Profile	: Defau	<i>it</i>
Re	Use You	arn are ru	uide	- Wir	fiki esha	Qu	esti	ions				3.2.3	8-1).				Profile	: Defau	At
Re	Les Use You ady to	arn are ru a load	or ca	- Wir	esha	Qu rk 3.	2.3 (	ions Git v	3.2.3	packa	eged as	3.2.1 No P	3-1). acket	s			Profile	: Defau	đ
Re	Use You	arn are ru a load	or ca	- Wir	esha	Qu rk 3.	2.3 (	ions Git v	3.2.3	packa		3.2.1 No P	3-1). acket	s	580-3.		Profile	: Defau	đ
•	Les Use You ady to	arn are ru a load	or ca	- Wir	esha esha	rk 3.	n na	Git v	3.2.3 iffer	packa for Bl	eged as	No P	3-1). acket	s vityu:		2	Profile	: Defau	đ
•	Lei Vou You	arn are ru b load	or ca	vir pture Capto	niki esha e	fron Ana	n na	Git v	3.2.3 iffer	packa for Bl	uetool	No P	3-1). acket	s vityu:		2	Profile	: Defau	đ
ile	Les You ady to Edit	arn are ru a load	or ca	- W g Wir pture Capto Capto	riki esha o uning ture	Providence of the second secon	n na	Git v	3.2.3 iffer	packa for Bl	uetool	No P	3-1). acket	s vityu:		2	Profile		
file	Les Vou ady to Edit	arn are ru aload View (@	or ca	wir pture capt	riki esha e uring ture <u>X</u>	Protection of the second secon	n na	Git v	iller tistics	for Bl Tele	untoor phony	No P	3-1). acket	is Nuyu Iook	s Helj	2		: Defau	
File	Les Vou You Edit	arn are ru bload	or ca go	wir pture capto capto sapto sapto sapto sapto sapto sapto sapto sapto sapto sapto sapto sapto sapto sapto sapto sapto satu sapto satu satu satu satu satu satu satu satu	riki - esha b ture X curl./	Provide the second seco	m ni lyze	Git v Git v Stal	iller istic	for Bl Tele Destin	eged as ephony ation	No P	3-1). acket	Took	s Helj	2 III ngthin	fo		-)
file      App 0,	Les Use You addy to Edit	arn er's G are ru b load	uide unnin or ca go go i fiter 719	wir pture Capt	arting ture Starting ture	Provide the second seco	m nil lyze Q	ions Git v Stal	iller tistics	for Bl Tele Destin Bread	eged as ephony ation cast	No P	3-1). acket	Tools	s Helj	2 II ngthin 63 A	fo DV NO	INCONN	-)•
file      App 0.	Les Use You ady to Edit Edit	arn er's Gu are ru b load Yiew ( () fisplay fisplay fisplay	or ca go filter	wir pture capt	arting ctri- ctri- ctri- ii:b1	• Qu rk 3. Ana 20 20 20 20 20 20 20 20 20 20 20 20 20	esti 2.3 ( lyze Q	ions Git v Sta Sta Sta Sta Sta Sta Sta Sta Sta Sta	iffer tistic	for Bl Tele Destin Broad	eged ad ephony ation cast cast	No P	3-1). acket	Tools	s Hely	2 11 63 A 63 A	fo DV_N0 DV_N0		- ) ( IN
file App o.	Les Use You ady to Edit Edit Diy a d Ti To 063 7 Do53 7	arn er's Gu are ru b load	uide unning or ca go filter 719 960 986	- Wir pture Capt	and	• Qu rk 3. Ana Ana C : : f6: : : f6: : : f6:	2.3 ( 2.3 ( lyze Q 0e:( 0e:(	ions Git v Sta Sta Sta Sta Sta Sta Sta Sta	iffer tistics	for Bi Teli Destin Broad Broad	eged as ephony ation cast cast cast	No P	3-1). acket	Proto	s Helg	2 iii iii iii iii iii iii iii i	fo 0V_N0 0V_N0 0V_N0		· ) (
file App o.	Les Use You ady to Edit Edit 042 7 063 7 064 7 064 7	arn er's Ge are ru b load <u>View</u> (@) lisplay me 6,414 6,518 6,625 6,626	uide annin or ca <u>9</u> 0 in filter 719 950 956 746	wir pture Capt	aki esha bure Ctrl-/ ource ource ource 1:b1 1:b1	• Qu rk 3. a from Ana C • • • • • • • • • • • • •	00000000000000000000000000000000000000	ions Git v Sta \$ 33:36 33:36 33:36	iffer tistic	for Bi Teli Destin Broad Broad	ation cast cast cast cast	No P	3-1). acket	Toolo Proto LE L LE L LE L LE L	s Help	2 iii iii iii iii iii iii iii i	fo 0V_N0 0V_N0 0V_N0 0V_N0	NCONN_ NCONN_ NCONN_ NCONN_	- ) ( INI INI INI
File App App 5	Les Use You ady to Edit Edit Diy a d Ti To 063 7 Do53 7	arn er's Gu are ru b load Ulew (@) lisplay me 6.625 6.626 6.625	uide annin or ca <u>9</u> 0 Filter 719 950 986 986 085	wir pture Capt	tiki esha bure Ctrl-j ource 1:b1 1:b1 1:b1 1:b1	Prov Prov	2.3 ( 2.3 ( 1/22e Q 0e:( 0e:( 0e:( 0e:( 0e:(	ions Git v Sta Sta Sta Sta Sta Sta Sta Sta	aller	for Bi Teli Destin Broad Broad	ation cast cast cast cast cast	No P	3-1). acket	Proto	s Helj	2 iii iii iii iii iii iii iii i	fo IV_N0 IV_N0 IV_N0 IV_N0 IV_N0		· ) (

Frame 1: 63 bytes on wire (504 bits), 63 bytes captured (504 bits) on interface /tmp/wires Nordic BLE Sniffer

rRF Sniffer for Blu\_apture in progress: Packets: 1068 · Displayed: 1068 (100.0%) Profile: Default

9d 06 38 01 76 05 06 0a 01 25 48 00 00 00 06 be 89 8e 42 25 36 63 0e 76 b1 61 00 01 09 20 02 18 76 24 7e 87 bc 96 5 0a 70 4b 71 2e 13 ec 85 0a 30 63 8d 2a Plug in the Adafruit BLE Sniffer. Launch Wireshark. The sniffer should show up under the available capture devices.

Double click on the sniffer capture device. This will open the device and start capturing.

If there is BLE traffic, it will be seen right away.

	<u> </u>	•		Capturi	ng fre	n ne	UF Sn	iffer f	or Bluetoo	th LE: /den	ttyUS	30-3.2		
									Telephon					
4		2 💿			3	Q	<	> 1	<b>} ⊮ -</b> 4		0 0	0 11		
I þ		a display	y filter	r ≪Ctr	6/»									
<b>o</b> .		Time		Sou	rce			D	estination		Protoco	ol Lengthin	fo	
					_									
	ŕ													
	ſ													
	ſ													
	ſ													
									progress>				hofile: De	

If there is no BLE traffic, it will look like this.

Note the device has opened properly and is sniffing, there's just nothing to be seen.

### Next Steps

Once everything is working as shown above, you are ready to move on to working with these BLE packets.

Now go here to learn how to look at BLE packets with Wireshark

## Working with Wireshark

This page will work with both V1 and V2 sniffer firmware, once you've got the software installed

## Working with Wireshark

Once Wireshark has loaded, you should see the advertising packets streaming out from the selected BLE device at a regular interval, as shown in the image below:

ile <u>E</u> dit <u>V</u> iew <u>G</u> o <u>C</u> apture <u>A</u> naly	ze Statistics Telephony Io	ls Internals <u>H</u> elp		
0 0 🛋 🗖 🧖 🗎 🕷	2 🔍 🗢 🗢 🕤 🕇		🎬 🗹 🥵 💥 🗮	
ilter: btle		Expression Clear Apply	Save	
o. Time Source	Destination	Protocol Length Info		
210 38.0421390 slave	Master	LE LL 60 ADV		
211 38.0460030 slave	Master	LE LL 60 ADV		
212 38.0478100 slave	Master	LE LL 60 ADV		
213 38.5964040 slave	Master	LE LL 60 ADV		
214 38.5991960 slave	Master	LE LL 60 ADV		
215 38.6012380 slave	Master	LE LL 60 ADV		
216 39.1499950 slave	Master	LE LL 60 ADV		
217 39.1526170 slave	Master	LE LL 60 ADV		
218 39.1543130 slave	Master	LE LL 60 ADV		
219 39.6971230 slave	Master	LE LL 60 ADV		
220 39.7004780 slave 221 39.7022140 slave	Master	LE LL 60 ADV		
	Master	LE LL 60 ADV		
222 40.2470320 slave	Master	LE LL 60 ADV		
223 40.2502480 slave 224 40.2522760 slave	Master	LE LL 60 ADV		
224 40.2522760 STave 225 40.8004240 STave	Master	LE LL 60 ADV LE LL 60 ADV		
225 40.8004240 Stave	Master Master	LE LL 60 ADV LE LL 60 ADV		
226 40.8039290 STave	Master	LE LL 60 ADV		
228 41.3522920 slave	Master	LE LL 60 ADV		
229 41.3554480 slave	Master	LE LL 60 ADV		
230 41.3572300 slave	Master	LE LL 60 ADV		
230 41.3372300 Stave	Mascer		_110	,
Frame 1: 60 bytes on wire	(480 bits), 60 bytes	antured (480 bits) on i	nterface 0	
Nordic BLE sniffer meta		apearea (400 bres) on n		
Bluetooth Low Energy Link	Layer			
00 11 06 35 01 42 70 06 0	Da 01 25 2b 00 00 26 0			
10 00 d6 be 89 8e 40 22 1				
	10 6e fb f4 f9	@n		
20 02 02 01 06 02 0a 00 1 30 e0 93 f3 a3 b5 01 00 4				
20 02 02 01 06 02 0a 00 1	0 00101415			
20 02 02 01 06 02 0a 00 1	0 00 10 14 15			

One of the key benefits of WireShark as an analysis tool is that it understands the raw packet formats and provides human-readable displays of the raw packet data.

The main way to interact with BLE data packets is to select one of the packets in the main window, and then expand the Bluetooth Low Energy Link Layer treeview item in the middle of the UI, as shown below:



Clicking on the Advertising Data entry in the treeview will highlight the relevant section of the raw payload at the bottom of the screen, but also provides human readable information about the payload that can save you a lot of time trying to debug or reverse engineer a device.

We can see, for example, that the device is advertising itself as a Bluetooth Low Energy only device ('BR/EDR Not Supported'), with a TX Power Level of OdBm, and a single service is being advertised using a 128-bit UUID (the UART service in this case).

### Capturing Exchanges Between Two Devices

If you wish to sniff data being exchanged between two BLE devices, you will need to establish a connection between the original device we selected above and a second BLE device (such as an iPhone or an Android tablet with BLE capabilities).

The nRF-Sniffer firmware is capable is listening the all of the exchanges that happen between these devices, but can not connect with a BLE peripheral or central device itself (it's a purely passive device).

### Scan Response Packets

If you open up nRF UART on an Android or iOS device, and click the Connect button, the phone or tablet will start scanning for devices in range. One of the side effects of this scanning process is that you may spot a new packet in Wireshark on an irregular basis, the 'SCAN\_REQ' and 'SCAN\_RSP' packets:

Capturing	g from \\.\p	ipe\wiresha	rk_nordic_b	ole [W	ireshark	1.12.1 (	v1.12.1-0	-g01b65	bf from r	naster-1	.12)]							- 0 <b>x</b>
Eile <u>E</u> dit	<u>V</u> iew <u>G</u> o	Capture	Analyze	Statisti	cs Tele	phony	Tools	Internal	: <u>H</u> elp									
D 0 /			X2	Q	<b>\$</b>	40 7	F 🕹			Q®		1	8	36	11			
								_										
ilter:								<ul> <li>Expr</li> </ul>	ession	Clear	Apply	Save						
	me	Source			Destina				tocol L									
	13.33929				Mast						) ADV_ 3 SCAN							
	13.340/1				Mast				LL		SCAN							
	13.3439				Mast				LL		ADV_							
3894 7	13.88351	19 slave			Mast	er		LE	LL	60	ADV_	IND						
	13.88654				Mast	er			LL		ADV_							
	13.88828	35 slave			Mast	er		LE	LL	60	) ADV_	IND						
		11																+
	3892: 38			(304	bits)	, 38	bytes o	captur	ed (30	4 bit:	s) on	inte	rface	0				
	BLE SN																	
	oth Low ss Addre			/er														
🗉 Scan	rtising Respons vertisir	e Data:	: e4:c6: 0509554			1 (e4	:c6:c7	:31:95	:11)									
⊟ Scan ⊟ Ad	Responsivertisin Device M Length Type: Device	e Data: ng Data Name: UA n: 5 Device Name:	0509554 RT Name (0)	15254		1 (e4	:c6:c7:	:31:95	:11)									
⊟ Scan ⊟ Ad	Responsivertisin Device M Length Type:	e Data: ng Data Name: UA n: 5 Device Name:	0509554 RT Name (0)	15254		1 (e4	:c6:c7:	:31:95	:11)					_				
⊟ Scan ⊟ Ad	Responsivertisin Device M Length Type: Device	e Data: ng Data Name: UA n: 5 Device Name:	0509554 RT Name (0)	15254		1 (e4	:c6:c7:	:31:95	:11)	_	_	_	_	_	_	_	_	
⊟ Scan ⊟ Ad	Responsivertisin Device M Length Type: Device	e Data: ng Data Name: UA n: 5 Device Name:	0509554 RT Name (0)	15254		1 (e4	:c6:c7	:31:95	:11)	_		_	_		_			
⊟ Scan ⊟ Ad	Responsivertisin Device M Length Type: Device	e Data: ng Data Name: UA n: 5 Device Name:	0509554 RT Name (0)	15254		1 (e4	:c6:c7	:31:95	:11)			_			_	_	_	
⊟ Scan ⊟ Ad	Responsivertisin Device M Length Type: Device	e Data: ng Data Name: UA n: 5 Device Name:	0509554 RT Name (0)	15254		1 (e4	:c6:c7:	:31:95	:11)									
⊟ Scan ⊟ Ad	Responsivertisin Device M Length Type: Device	e Data: ng Data Name: UA n: 5 Device Name:	0509554 RT Name (0)	15254		1 (e4	:c6:c7:	:31:95	:11)									
e Scan	Respons vertisin Device M Length Type: Device 0x68a41	e Data: 19 Data kame: UA 1: 5 Device Name: C	0509554 RT Name (0) JART	415254 (09)	1													
■ scan ■ Ad' ■ CRC:	Responsivertisin Device M Length Type: Device	e Data: 19 Data kame: UA 1: 5 Device Name: C 1 75 7ff 19 8e 44	0509554 RT Name (0) JART	(09)	25.0	0.00	97.00.0	00	:11)									
<ul> <li>⇒ Scan</li> <li>⇒ Ad'</li> <li>⇒</li> <li>⇒</li> <li>CRC:</li> </ul>	Responsive tisin Device N Length Type: Devicc Ox68a41	e Data: 19 Data kame: UA 1: 5 Device Name: C 1 75 7ff 19 8e 44	0509554 RT Name (0) JART	(09)	25.0	0.00	97.00.0	00	· · · · U. ·									

The Scan Response is an optional second advertising packet that some Bluetooth Low Energy periperhals use to provide additional information during the advertising phase. The normal mandatory advertising packet is limited to 31 bytes, so the Bluetooth SIG includes the possibility to request a second advertising payload via the Scan Request.

You can see both of these transactions in the image above, and the Device Name that is included in the Scan Response payload (since the 128-bit UART Service UUID takes up most of the free space in the main advertising packet).

For more information on Scan Responses and the advertising process in Bluetooth Low Energy see our Introduction to Bluetooth Low Energy Guide ().

### **Connection Request**

Once we click on the UART device in nRF UART, the two device will attempt to connect to each other by means of a Connection Request, which is initiated by the central device (the phone or tablet).

We can see this CONNECT\_REQ in the timeline in the image below:

	dic_ble [Wireshark 1.12.1 (v1.1		((),(),(),(),(),(),(),(),(),(),(),(),(),	
<u>File Edit View Go Capture Anal</u>	ze Statistics Telephony To	ols Internals Help		
0 0 🗶 🔳 🔬 📄 🗙	2 🔍 🔶 🔶 🐼 :		우, 0, 🖂   🗃 🗹 🥵 %   📜	
Filter:		<ul> <li>Expression</li> </ul>	Clear Apply Save	
lo. Time Source	Destination	Protocol		
6498 1190, 19107 Slave	Master	LE LL	60 ADV_IND	
6499 1190, 73693 slave	Master	LE LL	60 ADV_IND	
6500 1190, 73983 slave	Master	LE LL	60 ADV IND	
6501 1190,74166 slave	Master	LE LL	60 ADV_IND	
6502 1191.29056 slave	Master	LE LL	60 ADV_IND	
6503 1191, 29414 slave	Master	LE LL	60 ADV_IND	
6504 1191.29600 slave	Master	LE LL	60 ADV_IND	
6505 1191.84026 slave	Master	LE LL	60 ADV_IND	
6506 1191, 84456 slave	Master	LE LL	60 ADV_IND	
6507 1191.84624 slave	Master	LE LL	60 ADV_IND	
6508 1192,40771 Slave	Master	LE LL	60 ADV_IND	
6509 1192.41270 Slave	Master		60 CONNECT_REQ	
6510 1192.45658 Master	Slave	LE LL	26 Empty PDU	
6511 1192.45809 slave	Master	LE LL	26 Empty PDU	
6512 1192.50610 Master	slave	ATT		11.0
6513 1192. 50758 slave	Master		35 Rcvd Write Request, Handle: 0x00	JIE
6513 1192.50/58 STave	slave	LE LL	26 Empty PDU	
		LE LL	26 Empty PDU	
6515 1192.60471 slave	Master	LE LL	26 Empty PDU	
6516 1192.69980 Master	Slave	LE LL	26 Empty PDU	
6517 1192.70140 slave	Master	LE LL	26 Empty PDU	
6518 1192.74868 Master	Slave	LE LL	26 Empty PDU	
				,
E Frame 6509: 60 bytes on w	re (480 bits), 60 byt	es captured (48	30 bits) on interface 0	
Nordic BLE sniffer meta				
Bluetooth Low Energy Link	Laver			
57				
0000 11 06 35 01 ae 89 06 0	a 01 25 2c 00 00 98	00 00 E	۵/	
0010 00 d6 be 89 8e 85 22 0		35 31	%, k"1	
0020 c7 c6 e4 75 11 d8 84	f 2a c4 02 09 00 27	00 00u	*	
0030 00 bc 02 ff ff 03 e0 1	f b0 c9 cb 8f			

### Write Request

Once the connection has been established, we can see that the nRF UART application tries to write data to the BLEFriend via a Write Request to handle '0x001E' (which is the location of an entry in the attribute table since everything in BLE is made up of attributes).

Capturing from \\.\pipe\wireshark_nordic_bl	e [Wireshark 1.12.1 (v1.12.1	-0-g01b65bf from n	haster-1.12)]	- <b>-</b> X
<u>File Edit View Go Capture Analyze</u>	tatistics Telephony <u>I</u> ools	Internals Help		
0 0 <b>/ 1</b> / 1 1 1 X 2 1	् 🗢 🕈 😜 👍 🕹		Q. Q. 🖸   🗃 🕅 🥵 %   🖼	
Filter:		<ul> <li>Expression</li> </ul>	Clear Apply Save	
No. Time Source	Destination	Protocol L	ength Info	
6508 1192.40771 slave	Master	LE LL	60 ADV_IND	
6509 1192.41270 slave	Master	LE LL	60 CONNECT_REQ	
6510 1192.45658 Master	slave	LE LL	26 Empty PDU	
6511 1192.45809 Slave	Master	LE LL	26 Empty PDU	
6512 1192.50610 Master	Slave	ATT	35 Rcvd Write Request, Handle: 0x001e	
6513 1192.50758 Slave	Master	LE LL	26 Empty PDU	
6514 1192.60330 Master 6515 1192.60471 Slave	Slave Master	LE LL	26 Empty PDU	
6515 1192.604/1 STave 6516 1192.69980 Master	Slave	LE LL LE LL	26 Empty PDU	
6517 1192.70140 Slave	Master	LE LL	26 Empty PDU 26 Empty PDU	
< III III III III III III III III III I	Master	LE LL	20 Empty PD0	
Frame 6512: 35 bytes on wire	(280 bits), 35 bytes	captured (28	0 bits) on interface 0	
Nordic BLE sniffer meta				
Bluetooth Low Energy Link Lay Access Address: 0x84d81175	er			
Access Address: 0x84d811/5      Data Header: 0x0902				
CRC: 0x32acff				
Bluetooth L2CAP Protocol				
Length: 5				
CID: Attribute Protocol (0x	0004)			
Bluetooth Attribute Protocol				
Opcode: Write Request (0x12)	)			
Handle: 0x001e				
Value: 0100				
0000 11 06 1c 01 b2 89 06 0a (	03 0b 2b 02 00 37 bd	00	+7	
0010 00 75 11 d8 84 02 09 05 0		00 .u		
0020 4c 35 ff		L5.		
Bluetooth Attribute Protocol (btatt), 5 b	tes Packets: 31996 ·	Profile: Default		

What this write request is trying to do is enable the 'notify' bit on the <u>UART service's</u> <u>TX characteristic</u> () (0x001E is the handle for the CCCD or '<u>Client Characteristic</u> <u>Configuration Descriptor</u> ()'). This bit enables an 'interrupt' of sorts to tell the BLEFriend that we want to be alerted every time there is new data available on the characteristic that transmits data from the BLEFriend to the phone or tablet.

### Regular Data Requests

At this point you will start to see a lot of regular Empty PDU requests. This is part of the way that Bluetooth Low Energy works.

Similar to USB, all BLE transaction are initiated by the bus 'Main', which is the central device (the tablet or phone).

In order to receive data from the bus secondary (the peripheral device, or the BLEFriend in this particular case) the central device sends a 'ping' of sorts to the peripheral at a delay known as the 'connection interval' (not to be confused with the one-time connection highlighted earlier in this tutorial).

We can see pairs of transaction that happen at a reasonably consistent interval, but no data is exchanged since the BLEFriend (the peripheral) is saying 'sorry, I don't have any data for you':

Cile Cel	it View	Ge Canture	Analyze	Statistics Tele	heny Teel	Internals	Hele		_				
lie In	-			-	nony Tool	-							
DO	A .	🛃   🖿 🗋	XR	୍ 🔍 🗢 🔿	🤹 🐔 🛓		⊕ €		] 🛛 🖉	) 🕵 🎋	11		
ilter:						<ul> <li>Express</li> </ul>	ion C	lear App	lv Save				
0.	Time	Source		Destina	ion			igth Info	·				
		0771 Slave		Maste		LE L			V_IND				
		270 Slave		Maste		LEL			NNECT_REG	<b>_</b>			
		658 Master		Slave		LEL			pty PDU	4			
		5809 Slave		Maste					pty PDU				
		)610 Master		Slave		ATT			vd Write	Pequest	Handle	• 0x001e	
		758 Slave		Maste		LEL			pty PDU	Request	, nanore	. 00016	
		)330 Master		slave		LEL			pty PDU				
		471 Slave		Maste		LEL			pty PDU				
		980 Master		slave	-	LEL			pty PDU				
		0140 slave		Maste		LE L			DTV PDU				
		868 Master		Slave		LEL			pty PDU				
6519	1192.74	1986 Slave		Maste	r i	LEL			pty PDU				
		803 Master		slave		LE L	L		pty PDU				
6521	1192.79	943 slave		Maste	r	LE L	L		DTV PDU				
6522	2 1192.84	875 Master		slave		LE L	L	26 Em	pty PDU				
6523	3 1192.85	5002 slave		Maste	r	LE L	L		pty PDU				
6524	1192.94	425 Master		slave		LE L	L	26 Em	pty PDU				
6525	5 1192.94	553 Slave		Maste	r	LE L	L		PTY PDU				
6526	5 1193.04	087 Master		slave		LE L	L	26 Em	pty PDU				
6527	7 1193.04	208 slave		Maste	r	LE L	L	26 Em	pty PDU				
6528	8 1193.13	8933 Master		slave		LE L	L	26 Em	pty PDU				
6529	9 1193.14	1064 slave		Maste	r	LE L	L	26 Em	pty PDU				
	1193.23	8667 Master		slave		LE L	L	26 Em	DTV PDU				
				(208 bits)	26 byte	s captured	(208	bits) (	on interf	ace O			
		niffer me											
		w Energy		/er									
		ress: 0x8	d81175										
		er: 0x0009											
CR	c: 0xb5e	ef 55											
000	11 06 13	01 b9 89	06 0a	09 11 2b 07	00 99 00	00		+					
010	00 75 11	d8 84 09	00 ad	f7 aa		.u.							
N 10	\nine\wirer	hark pordic b	et clive car	ture i Packets	33336	Profile: Defau	1+						

### Notify Event Data

To see an actual data transaction, we simply need to enter some text in our terminal emulator SW which will cause the BLEFriend to send the data to nRF UART using the UART service.

Entering the string 'This is a test' in the terminal emulator, we can see the first packet being sent below (only the 'T' character is transmitted because the packets are sent out faster than we enter the characters into the terminal emulator):

	lit <u>V</u> iew <u>G</u> o	<u>Capture</u> <u>Analyze</u>	Statistics Telephony Tools	Internals <u>H</u> elp		
DO	a 🔳 🙇		🔍 🗢 🔿 👍 🛂		0, 0, 🖸   🐺 🗹 🥵 %   💢	
ilter:				<ul> <li>Expression</li> </ul>	Clear Apply Save	
	Time	Source	Destination	Protocol	Length Info	
5559	3 2854.95085	slave	Master	LE LL	26 Empty PDU	
	4 2854.99892		Slave	LE LL	26 Empty PDU	
	5 2855.00055		Master	ATT	34 Rcvd Handle Value Notification, Handle: 0x001	C
	5 2855.04974		slave	LE LL	26 Empty PDU	
	7 2855.05107		Master	LE LL	26 Empty PDU	
	8 2855.09619		slave	LE LL	26 Empty PDU	
	9 2855.09776		Master	LE LL	26 Empty PDU	
	2855.19274 2855.19395		Slave Master	LE LL	26 Empty PDU	
	2855.29033		Slave	LE LL LE LL	26 Empty PDU 26 Empty PDU	
	2 2855.29033		Master	LE LL	26 Empty PDU 26 Empty PDU	
	4 2855.33914		Slave	LE LL	26 Empty PDU	
	5 2855, 34070		Master	LE LL	26 Empty PDU	
	5 2855. 38816		Slave	LE LL	26 Empty PDU	
	7 2855, 38971		Master	LELL	26 Empty PDU	
Ac	cess Address ta Header: ( 000 =	RFU: 0				
Ac Da	ccess Address ta Header: ( 000 = 0 = 0 = 10 = 000 = 0 1000 =	s: 0x84d81175 0x080a RFU: 0 More Data: F Sequence Num Next Expected LLID: Start o RFU: 0	alse ber: True d Sequence Number: Fa		L2CAP message with no fragmentation (0x02)	
	ccess Address ta Header: 0 000 = 0 = 1 = 10 = 000 = 0 1000 = c: 0x6e97ce	s: 0x84d81175 0x080a RFU: 0 More Data: F: Sequence Num Next Expecte LLID: Start ( RFU: 0 Length: 8	alse ber: True d Sequence Number: Fa		L2CAP message with no fragmentation (0x02)	
AC Da • Da	ccess Address ta Header: ( 000 = 0 = 10. = 10 = 000 = 0 1000 = c: 0x6e97ce	s: 0x84d81175 0x080a RFU: 0 More Data: F: Sequence Num Next Expecte LLID: Start ( RFU: 0 Length: 8	alse ber: True d Sequence Number: Fa		L2CAP message with no fragmentation (0x02)	
AC Da Da Da Da Da Da	ccess Address ta Header: ( 000 = 0 = 10. = 000 = 0 1000 = cc: 0x6e97ce tooth L2CAP	s: 0x84d81175 0x080a RFU: 0 More Data: F. Sequence Num Next Expecte LLID: Start RFU: 0 Length: 8 Protocol	alse ber: True d Sequence Number: Fa of an L2CAP message c		L2CAP message with no fragmentation (0x02)	
	ccess Address ta Header: ( 000 = 0 = 10. = 000 = 000 = 000 = c: 0x6e97ce tooth L2CAP ength: 4 D: Attribute	s: 0x84d81175 0x080a RFU: 0 More Data: F Sequence Num Next Expected LLID: Start 0 LEngth: 8 Protocol e Protocol (0)	alse ber: True d Sequence Number: Fa of an L2CAP message c		L2CAP message with no fragmentation (0x02)	
AC Da Da Da Da Da Da Da Da Da Da	ccess Addres: ta Header: ( 000 = 1 = 10 = 000 = 10 = 000 = cc: 0x6e7/ce ctooth L2CAP ingth: 4 D: Attribut.	<pre>s: 0x84d81175 0x080a RFU: 0 More Data: Fi Sequence Num Next Expecte. LLID: Start 0 Length: 8 Protocol e Protocol (0 bute Protocol) e Value Notif</pre>	alse ber: True d Sequence Number: Fa of an L2CAP message c		L2CAP message with no fragmentation (0x02)	
AC Da Da Da CF Blue CI Blue CI Blue CI CI CI CI CI CI CI CI CI CI CI CI CI	<pre>cess Addres: tta Header: 0 000 =  1 =  100 = 000 = 0000 = 0000 = 0000</pre>	s: 0x84d81175 0x080a RFU: 0 More Data: F. Sequence Num Next Expecte LLD: Start C RFU: 0 LLD: Start C RFU: 0 Length: 8 Protocol e Protocol (0 Dute Protocol c Sub Notific C	alse ber: True d Sequence Number: Fa of an L2CAP message o x0004)	r a complete	L2CAP message with no fragmentation (0x02) + 	
AC Da Da Da CF Blue CI Blue CI Blue CI CI CI CI CI CI CI CI CI CI CI CI CI	<pre>ccess Address ta Header: ( 000 =  1 =  1.0 = 000 =  10 = 000 =  0 1000 =  0 10000 =  0 1000 =  0 10000 =  0 10000 =  0 1000</pre>	s: 0x84d81175 0x080a RFU: 0 More Data: F. Sequence Num Next Expecte LLD: Start C RFU: 0 LLD: Start C RFU: 0 Length: 8 Protocol e Protocol (0 Dute Protocol c Sub Notific C	alse ber: True d Sequence Number: Fa of an L2CAP message of x0004) ication (0x1b) 01 0d 2b 2d 85 99 00	00mII 76 .u		

What this 4-byte 'Bluetooth Attribute Protocol' packet is actually saying is that attribute 0x001C (the location of the TX characteristic in the attribute table) has been updated, and the new value is '0x54', which corresponds to the letter 'T'.

Scrolling a bit further down we can see an example where more than one character was sent in a single transction ('te' in this case):

Image: Source       Destination         Source       Destination         Source       Destination         Source       Destination         Source       Source         Display /255.//961 bilave       Master         Source       Slave         Socrest       Slave         Socrest       Socrest	Expression C     Protocol Ler     LE LL     ATT     LE LL     LE LL	20 Empty PUU 26 Empty POU 35 Revd Handle Value Notification, Handle: 0x001c 26 Empty POU 26 Empty POU 26 Empty POU 35 Revd Handle Value Notification, Handle: 0x001c 26 Empty POU 26 Empty Andle Value Notification, Handle: 0x001c 27 Empty POU 26 Empty POU
Bitter         Source         Destination           3>b1_y 4253.//y81 JaVE         MaStEr         Slave           3>b21_y 4253.//y81 JaVE         MaStEr         Slave           55620         2855.82724 Master         Slave           55620         2855.82724 Master         Slave           55621         2855.87881 Master         Slave           55622         2855.87481 Master         Slave           55622         2855.97485 Master         Slave           55622         2855.97485 Master         Slave           55622         2856.07432 Slave         Master           55622         2856.07432 Slave         Master           55622         2856.07432 Slave         Master           55622         2856.07432 Slave         Master           55623         2856.12020 Master         Slave           55631         2856.12020 Master         Slave           55632         2856.12180 Master         Slave           55632         2856.21800 Master         Slave           55632         2856.21800 Master         Slave           55632         2856.21800 Master         Slave           55632         2856.21800 Master         Slave           55632	Expression C     Protocol Ler     LE LL     ATT     LE LL     LE LL	Clear Apply Save might Info 20 Empty PUU 26 Empty PDU 35 Rcvd Handle value Notification, Handle: 0x001c 26 Empty PDU 26 Empty PDU 26 Empty PDU 35 Rcvd Handle value Notification, Handle: 0x001c 26 Empty PDU 26 Empty PDU 27 Empty PDU 28 Empty PDU 29 Empty PDU 20 Empty P
Time         Source         Destination           >>>b1         2535.7/96151aVE         Master           >>>b502         2655.822451ave         Master           S5621         2655.822451ave         Master           S5621         2655.822451ave         Master           S5622         2655.822451ave         Master           S5622         2655.87881         Master           S5622         2655.97489         S1ave           S5622         2655.97489         S1ave           S5622         2856.024325         S1ave           S5622         2856.024325         S1ave           S5622         2856.024325         S1ave           S5622         2856.024325         S1ave           S5622         2856.07305         Master           S5622         2856.07325         S1ave           S5622         2856.024325         S1ave           S5623         2856.12020         Master           S5632         2856.12020         Master           S5632         2856.12804         Master           S5632         2856.12804         Master           S5632         2856.12804         Master           S5632         S162.1	Protocol Ler LE LL ATT LE LL LE LL	ngth Info 26 Empty PDU 26 Empty PDU 35 Revd Handle value Notification, Handle: 0x001c 26 Empty PDU 26 Empty PDU 27 Empty PDU 28 Empty PDU 29 Empty PDU 20 Empty P
>>bit         >>bit         >>bit         >>bit           >>>bit         >bit         >bit         >bit           >>bit         >bit         >bit         >bit           >>>bit         >bit         >bit         >bit           >>>>>bit         >bit         >bit         >bit           >>>>>>>>>>>>>>>>>>>>>>>>>>>>	LE LL LE LL ATT LE LL LE LL LE LL LE LL LE LL LE LL LE LL LE LL LE LL LE LL	20 Empty PUU 26 Empty POU 35 Revd Handle Value Notification, Handle: 0x001c 26 Empty POU 26 Empty POU 26 Empty POU 35 Revd Handle Value Notification, Handle: 0x001c 26 Empty POU 26 Empty Andle Value Notification, Handle: 0x001c 27 Empty POU 26 Empty POU
55621 2855. 82924 Slave         Master           55622 2855. 87881 Master         Slave           55622 2855. 87881 Master         Slave           55624 2855. 97459 Slave         Master           55624 2855. 97459 Slave         Master           55625 2855. 02432 Slave         Master           55626 2856. 02432 Slave         Master           55626 2856. 02432 Slave         Master           55626 2856. 02432 Slave         Master           55627 2856. 02432 Slave         Master           55628 2856. 07432 Slave         Master           55639 2856. 07432 Slave         Master           55631 2856. 12050 Master         Slave           55631 2856. 12051 Master         Slave           55632 2856. 12060 Master         Slave           55632 2856. 12081 Slave         Master           Frame 55627: 35 bytes on wire (280 bits), 35 bytes           Nordic BLE sniffer meta           Bluetoth Low Energy Link Layer           Access Address: 0x4641175	LE LL ATT LE LL LE LL LE LL LE LL LE LL LE LL LE LL LE LL LE LL ATT 	26 Empty PDU 35 Revd Handle value Notification, Handle: 0x001c 26 Empty PDU 26 Empty PDU 26 Empty PDU 35 Revd Handle value Notification, Handle: 0x001c 26 Empty PDU 26 Empty PDU 26 Empty PDU 26 Empty PDU 26 Empty PDU 26 Empty PDU 35 Revd Handle value Notification, Handle: 0x001c
55622         22855.87881         Master           55623         2855.88012         Slave           55624         2855.97345         Master           55627         2856.02266         Master           55627         2856.02325         Slave           55627         2856.07305         Master           55627         2856.07305         Master           55627         2856.07305         Master           55637         2856.12020         Master           55632         2856.21800         Master           55632         2856.21800         Master           55632         2856.21805         Slave           55632         2856.2180         Slave           5632         2856.2180         Slave	LE LL LE LL LE LL LE LL LE LL LE LL LE LL LE LL LE LL ATT	26 Empty PDU 26 Empty PDU 26 Empty PDU 26 Empty PDU 35 Rxvd Handle Value Notification, Handle: 0x001c 26 Empty PDU 26 Empty PDU 26 Empty PDU 26 Empty PDU 26 Empty PDU 26 Empty PDU 27 Empty PDU 28 Empty PDU 29 Fampty PDU 29 Fampty PDU 20 Empty PDU
Si622 2855.88012 Elave         Master           Si624 2855.9745 Master         Slave           Si625 2855.9745 9 slave         Master           Si626 2856.02266 Master         Slave           Si627 2856.02432 Elave         Master           Si628 2856.07305 Master         Slave           Si629 2856.0732 Elave         Master           Si629 2856.0732 Elave         Master           Si631 2856.12020 Master         Slave           Si631 2856.12155 Slave         Master           Si632 2856.12000 Master         Slave           Si631 2856.12020 Master         Slave           Si632 2856.21801 Master         Slave           Si632 2856.21801 Master         Slave           Si632 2856.21801 Master         Slave           Frame Si627: 35 bytes on wire (280 bits), 35 bytes           Nordic BLE sniffer meta           Bluetoth Low Energy Link Layer           Access Address: Ox46412175	LE LL LE LL LE LL ATT LE LL LE LL LE LL LE LL LE LL ATT	26 Empty PDU 26 Empty PDU 26 Empty PDU 35 Rcvd Handle Value Notification, Handle: 0x001c 26 Empty PDU 26 Empty PDU 27 Empty PDU 28 Empty PDU 29 Empty PDU 20 Empt
55624         2855, 07345         Share         Slave           55625         2856, 07489         Slave         Master           55626         2856, 02266         Master         Slave           55627         2855, 07435         Slave         Master           55627         2856, 02432         Slave         Master           55627         2856, 07435         Slave         Master           55627         2856, 07435         Slave         Master           55637         2856, 12020         Master         Slave           55637         2856, 12020         Master         Slave           55637         2856, 12020         Master         Master           55637         2856, 12050         Slave         Master           55637         35         Slave         Master           5637         55627:         35         Slave           56471         55	LE LL LE LL ATT LE LL LE LL LE LL LE LL LE LL ATT	26 Empty PDU 26 Empty PDU 26 Empty PDU 37 Revd Handle Value Notification, Handle: 0x001c 26 Empty PDU 26 Empty PDU 26 Empty PDU 26 Empty PDU 25 Revd Handle Value Notification, Handle: 0x001c
SS622 2855, 07/489 Slave         Master           SS622 2856, 02/266 Master         Slave           SS627 2856, 02/265 Slave         Master           SS628 2856, 07/325 Slave         Master           SS630 2856, 12/2020 Master         Slave           SS631 2856, 12/2020 Master         Slave           SS631 2856, 12/2020 Master         Slave           SS631 2856, 12/2020 Master         Slave           SS632 2856, 12/2020 Master         Slave           SS632 2856, 12/2020 Master         Slave           SS632 2856, 12/2021 Master         Master           Frame SS627: 35 bytes on wire (280 bits), 35 bytes         Nordic BLes niffer meta           Bluetoth Low Energy Link Layer         Access Address: 0x46/12/5	LE LL ATT LE LL LE LL LE LL LE LL LE LL ATT	26 Empty PDU 26 Empty PDU 27 Empty PDU 28 Empty PDU 29 Empty PDU 20
55626         2836.02266 Master         Slave           55627         2856.02325 Slave         Master           55628         2856.07305 Master         Slave           55629         2856.07305 Master         Slave           55629         2856.07305 Master         Slave           55629         2856.07325 Slave         Master           55631         2856.12020 Master         Slave           55632         2856.12020 Master         Slave           55632         2856.12000 Master         Slave           76740         Premes         Slave           76740         2460         Master           76740         Premes         Slave           810etcoth         Low Energy Lifk Layer           Access Address:         Not4612175	LE LL ATT LE LL LE LL LE LL LE LL LE LL ATT	26 Empty PDU 35 Rcvd Handle Value Notification, Handle: 0x001c 26 Empty PDU 26 Empty PDU 26 Empty PDU 26 Empty PDU 25 Empty PDU 35 Rcvd Handle Value Notification, Handle: 0x001c
S5627         Z856,02432         Slave         Master           S5628         Z856,07305         Master         Slave           S5628         Z856,07305         Master         Slave           S5630         Z856,12020         Master         Slave           S5631         Z856,12155         Slave         Master           S5632         Z856,12020         Master         Slave           S5631         Z856,12020         Master         Slave           S5632         Z856,13080         Master         Slave           Frame         S10000         Master         Slave           Frame         S5627         35         Bytes on wire (280 bits), 35         bytes           Frame         S160000         Slave         Master         Slave           Access         Address:         Ox40481175         Slave         Slave	ATT LE LL LE LL LE LL LE LL LE LL ATT	35 Rcvd Handle Value Notification, Handle: 0x001c 26 Empty PDU 26 Empty PDU 26 Empty PDU 26 Empty PDU 26 Empty PDU 35 Rcvd Handle Value Notification, Handle: 0x001c
55628 2856.07305 Master         Slave           55629 2856.07325 Slave         Master           55630 2856.12020 Master         Slave           55631 2856.12155 Slave         Master           55632 2856.12000 Master         Slave           6100 Master         Slave           6100 Master         Slave           8100 Master         Slave           8100 Master         Slave           8100 Master         Slave           Access Address:         Not4d8l175	LE LL LE LL LE LL LE LL ATT	26 Empty POU 26 Empty POU 26 Empty POU 26 Empty POU 26 Empty POU 35 Revd Handle Value Notification, Handle: 0x001c
55629         2856.07432         Patter           55630         2856.12020         Master         Slave           55631         2856.12020         Master         Slave           55631         2856.12020         Master         Slave           55632         2856.21800         Master         Slave           55631         2856.2180         Master         Master           Frame         55637         355         Master           Frame         55627         35         bytes on wire (280         bits), 35         bytes           Frame         55627         135         bytes on wire (280         bits), 35         bytes           Bluetooth         Low Energy Link         Layer         Access Address: 0x8461175	LE LL LE LL LE LL ATT	26 Empty PDU 26 Empty PDU 26 Empty PDU 26 Empty PDU 35 Revd Handle Value Notification, Handle: 0x001c
55630         2856.12020 Master         Slave           55631         2856.12155 Slave         Master           55632         2856.21800 Master         Slave           Frame         55627: 35 bytes on wire (280 bits), 35 bytes           Nordic BLE sniffer meta         Bluetoth Low Energy Link Layer           Access Address:         0x84d81175	LE LL LE LL LE LL ATT	26 Empty PDU 26 Empty PDU 26 Empty PDU 35 Rcvd Handle Value Notification, Handle: 0x001c
5561 2856.12155 5lave     Master       5563 2856.21800 Master     Slave       5563 2856.21801 slave     Master       771 077 2170 100 ministration     1       Frame 55627: 35 bytes on wire (280 bits), 35 bytes       Nordic BLE sniffer meta       Bluetooth Low Energy Link Layer       Access Address: 0x84681175	LE LL ATT	26 Empty PDU 26 Empty PDU 35 Rcvd Handle value Notification, Handle: 0x001c
55632 2856.21800 Master Slave 55633 2856.21981 Slave Master Frame 55627: 35 bytes on wire (280 bits), 35 bytes Nordic BLE sniffer meta Bluetooth Low Energy Link Layer Access Address: 0x84d81175	LE LL ATT	26 Empty PDU 35 Rcvd Handle Value Notification, Handle: 0x001c
55633 2856, 21981 Slave Master Frame 55627: 35 bytes on wire (280 bits), 35 byte: Nordic BLE sniffer meta Bluetooth Low Energy Link Layer Access Address: 0X846B175	ATT	35 Rcvd Handle Value Notification, Handle: 0x001c
Frame 55627: 35 bytes on wire (280 bits), 35 bytes Nordic BLE sniffer meta Bluetooth Low Energy Link Layer Access Address: 0x84d81175		37 Participanti
Frame 55627: 35 bytes on wire (280 bits), 35 byte: Nordic BLE sniffer meta Bluetooth Low Energy Link Layer Access Address: 0x84d81175	s captured (280	D bits) on interface O
0 = More Data: False 0 = Sequence Number: False 1 = Next Expected Sequence Number: Tr 10 = LLID: Start of an L2CAP message or 000 = RFU: 0 0101 = Length: 9		2CAP message with no fragmentation (0x02)
Bluetooth L2CAP Protocol		
Length: 5		
CID: Attribute Protocol (0x0004)		
Bluetooth Attribute Protocol Opcode: Handle Value Notification (0x1b) Handle: 0x001c		
Value: 7465		
000 11 00 1- 01 0J 40 00 0- 01 10 2L 42 05 07 00		
000 11 06 1c 01 8d 49 06 0a 01 10 2b 42 85 97 00 010 00 75 11 d8 84 06 09 05 00 04 00 1b 1c 00 74 020 18 fc 6a		
Value (btatt.value), 2 bytes Packets: 64085 - Di	Profile: Default	

The results of this transaction in the nRF UART application can be seen below:



### Closing Wireshark and nRF-Sniffer

When you're done debugging, you can save the session to a file for later analysis, or just close Wireshark right away and then close the nRF-Sniffer console window to end the debug session.

## Moving Forward

A sniffer is an incredibly powerful and valuable tool debugging your own hardware, reverse engineering existing BLE peripherals, or just to learn the ins and outs of how Bluetooth Low Energy actually works on the a packet by packet level.

You won't learn everything there is to know about BLE in a day, but a good book on BLE, a copy of the Bluetooth 4.1 Core Specification and a sniffer will go a long way to teaching you most of the important things there is to know about BLE in the real world.

## Using with Sniffer V2 (old)

This page is deprecated. It is being left here for reference and for anyone requiring a Python 2 setup.

In mid 2018, Nordic release new Bluetooth LE sniffer firmware - this firmware works way better with Wireshark.

As of August 2018 we are only selling Sniffers pre-prorgrammed with Firmware version 2

If you have a firmware V1 (packaging doesn't say firmware V2, or you bought before August 2018) see the previous sections!

## Nordic User Manual

You can grab the 'official' user manual from Nordic at <u>https://www.nordicsemi.com/</u> <u>eng/Products/Bluetooth-low-energy/nRF-Sniffer</u> () we include a mirror of the v2.1 instructions below

nRF\_Sniffer\_User\_Guide\_v2.1.pdf

## nRF Sniffer V2 Multi-Target Application

When downloading the desktop/Wireshark Sniffer tool, make sure to download BETA 1, which matches the firmware image below. You can select this version via the drop down selector shown below:

Changelog:	
• 2.0.0-beta-1 Multi-Platform	$\checkmark$
2.0.0-beta-2 Multi-Platform	~
2.0.0-beta-3 Multi-Platform	~

For convenience sake, the extcap folder contents for BETA 1 are also available for download here using the button below. See the user guide on how to install this in the correct location for Wireshark.

extcap.zip (BETA 1)

## V2 Firmware

You need a J-Link or other SWD programming jig in order to install/change the sniffer firmware!

If by chance you have an nRF51822 board you want to load the firmware on, here's a hex that does not require the 32khz crystal (but does require the 16 mhz crystal)

This firmware is from the nrf\_sniffer\_2.0.0-beta-1\_51296aa package

sniffer\_pca10028\_51296aa.hex

Again, we don't have a guide or tutorial on loading the firmware onto an nRF51. We don't have the original source code, that hex is from Nordic and they only release hex files

## V2 Wireshark Usage (old)

This page is deprecated. It is being left here for reference and for anyone requiring a Python 2 setup.

For the V2 firmware, its recommended that you use Wireshark - the V1 had various methods such as a Python API but really they were all mediocre compared to the abilities of Wireshark



## Install Wireshark

Start by installing Wireshark, a great cross-platform monitoring tool

Visit <a href="https://www.wireshark.org/">https://www.wireshark.org/</a> () and download the latest version of Wireshark for your operating system

When installing on windows, check the box to also install WinPcap

## Install Wireshark Plugin

Next up, you'll need the Nordic plugin software. We need to work with a specific release - 2.0.0 Beta 1. For convenience sake, the extcap folder contents for BETA 1 are available for download using the button below.

extcap.zip (BETA 1)

Download that zip file to your computer and see below for how to install the files into the necessary Wireshark folder.

For complete reference, the Nordic main page for the plugin software is here:

## nRF Sniffer Software (reference only)

If you decide to go there, be sure to select the correct version for download.

Changelog:	
2.0.0-beta-1 Multi-Platform	~
2.0.0-beta-2 Multi-Platform	$\sim$
2.0.0-beta-3 Multi-Platform	~

WARNING this file is huge - over 200MB! We suggest just using the extcap.zip file linked above.

Now to find the Wireshark folder location to unzip these files into.



Open that directory up, then copy over the files within the extcap.zip extcap folder into the Wireshark extcap folder

In the end, your Wireshark/extcap directory should contain nrf\_sniffer.bat, nrf\_sniffer.p y and SnifferAPI folder.

File Home Share View		~ 0		
🕂 🚽 👻 🛧 📙 « Local Disk (C:) > Progra	m Files > Wireshark > extcap v Ö Search extcap	۹		
extcap ^ Name	Date modified Type	Size		
PCBs	This folder is empty.			
T Drophov (adafruit)	📙   🕑 📃 =   extcap		- 0	
1 · · · · · · · · · · · · · · · · · · ·	File Home Share View			$\sim$
🐉 Dropbox (Personal	└ ← · ↑   > nrf_sniffer_2.0.0-beta-1_51296aa > extcap	ڻ - v	Search extcap	\$
Computer	Move to extcap	Date modified	Туре	Siz
3D Objects	A Quick access			312
Desktop	Desktop # SnifferAPI	8/11/2018 2:07 PM	File folder	
Documents	🕒 Libert	8/11/2018 2:07 PM	Windows Batch File	
Downloads	Development #	8/11/2018 2:07 PM	PY File	
Music	S Fritzing-Libra #			
Pictures				
Videos	a_ Common ≉ Dropbox (ada: *			

Now quit Wireshark so we can get things tested!

## Dealing With Python 2 vs 3

Nordic's sniffer code is Python 2 only, so if you have Python 3 your default (which, by now, you probably do) you'll need to install Python 2.

The best way to test is to go to the extcap directory in your terminal software and try running nrf\_sniffer.bat (Windows) or python nrf\_sniffer.py (Mac/Linux)

If you get the error on the print "LOGGING FAILED" line



or the "No module named 'Logger'" error



Then you'll need to trick the sniffer software into using Python 2

For Windows, at least, I installed Python 2.7 into C:\Python27 (the default) and then edited the nrf\_sniffer.bat file to say:

```
@echo off
C:\Python27\python "%~dp0nrf_sniffer.py" %*
```

note the explicit path!

### Installing Dependancies

Once you get past that part, you can try rerunning the bat/py script and you may get other missing module errors like No module named serial



You'll need to install these with pip

Warning! Because you have to use Python2 here, make sure you're using pip2 or on windows, use the full path C:\python27\Scripts\pip2.exe

e.g. C:\python27\Scripts\pip2.exe install pyserial

C:\Program Files\Wireshark\extcap>C:\python27\Scripts\pip2.exe install pyserial
Collecting pyserial
Cache entry deserialization failed, entry ignored
Cache entry deserialization failed, entry ignored
Cache entry deserialization failed, entry ignored
Downloading https://files.pythonhosted.org/packages/0d/e4/2a744dd9e3be04a0c0907414e2a01a7c88bb3
915cbe3c8cc06e209f59c30/pyserial-3.4-py2.py3-none-any.whl (193kB)
100% ###################################
Installing collected packages: pyserial
Successfully installed pyserial-3.4
Cache entry deserialization failed, entry ignored
You are using pip version 9.0.1, however version 18.0 is available.
You should consider upgrading via the 'python -m pip installupgrade pip' command.
C:\Program Files\Wireshark\extcap>

Eventually you'll get No arguments given! which means the script is, at least, fully running



## Test Capture

OK finally once that works, start Wireshark again.

This time you'll see the nRF Sniffer capture device!

Welcome to Wireshark	
Open	
C:\Users\ladyada\Dropbox\RF\zigbeesniffer\outfile2.txt (not found)	
C:\Users\ladyada\Dropbox\RF\zigbeesniffer\outfile.txt (664 Bytes)	
Capture	
using this filter: 📙 Enter a capture filter	✓ All interfaces shown ▼
Local Area Connection 4 J	
In the connection 4 Julian      In the connection of the conneconnection of the connection of the	
S NRF SNITTER COIVIOS	

Double Click on that line to start the Capture!

Now go here to learn how to look at BLE packets with Wireshark

### Windows Install Supplemental Information

Adafruit forums user @TomHildebrand put together a nice write up on their experience installing and setting up everything on Windows 10. It's generally the same

info as above, but may have some more explicit info that is useful. Checkout it out here:

Adafruit BLE Sniffer installation on Windows Step-by-Step

## Using with Sniffer V1 (old)

The original Bluetooth LE sniffer firmware from Nordic had some restrictions such as only being usable by Wireshark 1.

As of August 2018 we are only selling Sniffers pre-prorgrammed with Firmware version 2

However, we'll keep this documentation up in case its useful for people with old boards

You need a J-Link or other SWD programming jig in order to install/change the sniffer firmware!

If by chance you have an nRF51822 board you want to load the firmware on, here's a hex that does not require the 32khz crystal (but does require the 16 mhz crystal)



Again, we don't have a guide or tutorial on loading the firmware onto an nRF51. We don't have the original source code, that hex is from Nordic and they only release hex files

## **USB** Driver Install

# CP2104 Driver Requirements (Black Boards)

The latest version of the sniffer uses the CP2104 USB to Serial bridge and drops the SWD connector, allowing us to sell the boards at a significant discount compared to

version 1.0. To use these boards, though, you will need to install the  $\frac{CP2104 \ VCP}{VCP}$  driver from Silicon Labs ():



Legacy OS software and driver package download links and support information >

## FTDI Driver Requirements (Blue Boards)

Before you can start talking to the sniffer, you'll need to install a standard FTDI driver for the FT231x located on the device.

Find the appropriate FTDI VCP installer on the FTDI Driver Download Page (), install it on you system, and then insert the sniffer in any USB port on your system.

			Processor Archite	ecture					
Operating System	Release Date	x86 (32-bit)	x64 (64-bit)	PPC	ARM	MIPSII	MIPSIV	SH4	Comments
Windows*	2014-09-29	Available as se Contact support1@ftdichip.com driv	if looking to create cusomised		-		•	-	2.12.00 WHQL Certified Available as <u>setup executable</u> Release Notes
Linux	2009-05-14	1.5.0	1.5.0	•		-	•	-	All FTDI devices now supported in Ubuntu 11.10, kernel 3.0. 19 Refer to <u>TN-101</u> if you need a custom VCP VID/PID in Linux
Mac OS X	2012-08-10	2.2.18	2.2.18	2.2.18		1.1		1.	Refer to TN-105 if you need a custom VCP VID/PID in MAC OS

## V1 Sniffer Software

This page is for the V1 Sniffer firmware only! If you have V2, check the other page - the process has changed between versions.

## Using the Firmware V1 Sniffer

There are currently two ways to use the sniffer:

### Nordic's nRF Sniffer Utility (Windows only)

If you are on Windows, the best user experience will be had by using the official Nordic nRFSniffer application, available as a download from Nordic Semiconductors after creating a 'My Pages' account, and registering your device using the product ID located on the Bluefruit LE Sniffer packaging.

🔇 BLE Sniffer 1.0.1				- 0 ×
Commands: 1 arrow keys L#1 or ENTER e w x/q c v b b a p o h s u CTRL-R Available devi	Select a device Like ENTER, but Start Wireshark, Exit Display filter: Display filter: Remove display filter: Remove display filter: Passkey entry 00B key entry Define new adv h Get support Launch User Guid Re-program firmw	<pre>ice list. Use Ef to sniff from 1; sniffer will on the primary vie Nearest devices Nearest devices ilter. op sequence. e (pdf)</pre>	TER to select.	
# public	c name	RSSI	device address	
[ ] 0 "" [ ] 1 "" [ ] 2 "" Scanning for dev	vices.	-90 dBm -65 dBm -46 dBm	14:99:e2:05:29:cf 68:48:98:b8:e5:2b e4:c6:c7:31:95:11	
Sent Key value (	to sniffer			~

More information on using Nordic's nRF Sniffer application ().

### Python API (Cross-Platform, no Registration)

If you are not using Windows, or don't wish to create a MyPages account, the alternative is to use a Python interface to communicate with the nRFSniffer firmware, which will log any traffic to a libpcap file that can be opened directly in Wireshark. This has been tested on OS X 10.10, Ubuntu 14.04 and Windows 7, but it currently doesn't support streaming data directly into Wireshark via named pipes (though this is possible with some platform-specific effort).

More information on using the Python API ().



## V1 Nordic nRF Sniffer

This page is for the V1 Sniffer firmware only! If you have V2, check the other page - the process has changed between versions.

The following guide will walk you through downloading, installing and using the official nRF Sniffer application for Nordic Semiconductors.

## Getting the Sniffer Utility

The Bluefruit LE Sniffer comes pre-flashed with the special sniffer firmware image, but you'll need to go to Nordic's website and download the nRF-Sniffer package to capture the data on Windows and push it out into Wireshark for packet by packet analysis.

Go to the <u>nRF Sniffer product page downloads tab</u> (), then download the latest V1 version of the utility, and unzip it.

Inside this downloaded file you'll find the sniffer executable, which will open up the command-line tool when you click on it.

## **Getting Wireshark**

In order to use the sniffer utility you'll also need to <u>download Wireshark</u> (), preferably verison 1.12.1 (the same one used in this tutorial).

You may need to explore the download mirrors, such as  $\frac{https://1.na.dl.wireshark.org/}{1}$  () to find the download link since they dont have a direct v1 link

Simply select the 32-bit or 64-bit Windows Installer and install it on your machine using the default settings:



Make sure that you install the libpcap library when installing Wireshark. Any log files captured by the python library are in libpcap format, and will require this library to work.

## Running the Sniffer

Now that everything is installed, you can get started using the Bluefruit LE Sniffer and the sniffer bridge SW that pushes any sniffed data out into Wireshark ...

## Select the Sniffer Target

The nRF-Sniffer can only sniff one device at a time, so the first step is getting the sniffer running and then selecting the device that you want to debug.

Start nRF-Sniffer by running the ble-sniffer\_win executable (for example: ble-sniffer\_win\_1.0.1\_1111\_Sniffer.exe).

This will try to detect the device running the nRF-Sniffer firmware over a UART COM port.

If the board isn't detected right away type 'f' to erase any previous com port settings, or try removing and then re-inserting the sniffer while the console application is running.

Once the sniffer is found, you should see a list of all BLE devices that were detected in listening range:

If you see a warning in the application about your firmware being out of date and requesting to update it, IGNORE THE WARNING. The Adafruit boards run a slightly modified version of the sniffer firmware, which causes the tool to think it is out of date.

BLE Sniffer 1.0.1				_ <b>D</b> X
Commands: 1 arrow keys [#] or ENTER e w x/q c v b a p o h s u CTRL-R Available devi	Select a device Like ENTER, but Start Wireshark, Exit Display filter: Display filter: Display filter: Remove display f Passkey entry OOB key entry OOB key entry Define new adv h Get support Launch User Guid Re-program firmw	ice list. Use EN to sniff from li sniffer will onl the primary vie Nearest devices Nearest devices ilter. op sequence. e (pdf)	TER to select.	nts.
# public	c name	RSSI	device address	
[ ] 0 "" [ ] 1 "" [ ] 2 "" Scanning for de	vices.	-65 dBm	68:48:98:b8:e5:2b	public public random
Sent Key value	to sniffer			Ŧ

In this particular case, we'll select device number 2, which is a BLEFriend running the standard UART firmware.

Type the device number you want to sniffer (in this case '2'), and you should see the device highlighted in the list, similar to the image below:

💽 BLE Sniffer 1.0.1				- • ×
arrow keys	List the dev Navigate the Select a dev Like ENTER, Start Wiresh Exit Display filt Display filt Display filt Desplay filt OB key entr OB key entr Define new a Get support Launch User	ices available ff device list. Use ice to sniff fro but sniffer will ark, the primary er: Nearest devic er: Nearest devic ay filter. y dv hop sequence.	≥ ENTER to select. n list. only follow advertisem viewer for the sniffer es (RSSI > -50 dBm). es (RSSI > -70 dBm). es (RSSI > -90 dBm).	
Available devid # publid		RSSI	device address	
[ ] 0 "" [ ] 1 "" -> [X] 2 "" Sniffing device			14:99:e2:05:29:cf 68:48:98:b8:e5:2b	public

At this point you can type 'w', which will try to open wireshark and start pushing data out via a dedicate pipe created by the nRF-Sniffer utility.

Now go here to learn how to look at BLE packets with Wireshark

## V1 OS X Support

This page is for the V1 Sniffer firmware only! If you have V2, check the other page - the process has changed between versions.

If you are running OS X 10.9 or higher, you can also use the sniffer on OS X using the nrf-ble-sniffer-osx () package from Roland King. (Make sure you have the latest version, as of 20 June 2015, which is now compatible with the FTDI chip used on the Adafruit board.)

Setup instructions are available on the wiki page () for the project.

Be sure to	download	Wireshark	version	2.0.x NOT	the nev	v 2.2.7	that w	as
released J	une 2017							

Shiffer Information  Status: Shiffing <unknown> Packet Count: 15 358 USB Device: FTDI - FT231X USB UART [s/n: DN009WNO]  Currently Shiffing  Device Name : <unknown> Address: dd:2b:48:7f:b5:8c (random) RSSI : -54 Event Count: 0  Advertisers  Mame: <unknown> Address: 46:8e:86:db:97:c1 (random) Name: <unknown> Address: dd:2b:48:7f:b5:8c (random) List Devices Shiff Device Capture to Wireshark Capture To PCAP Enter Passkey</unknown></unknown></unknown></unknown>		usbserial-DN009WNO
Packet Count : 15 358 USB Device : FTDI - FT231X USB UART [s/n: DN009WNO] Currently Sniffing Device Name : <unknown> Address : dd:2b:48:7f:55:8c (random) RSSI : -54 Event Count : 0 Advertisers Name : <unknown> Address : dd:2b:46:7f:55:8c (random) Name : <unknown> Address : dd:2b:46:7f:55:8c (random) Adverss : dd:2b:46:7f:55:8c (random)</unknown></unknown></unknown>	Sni	ffer Information
USB Device : FTDI - FT231X USB UART [s/n: DN009WN0 ]  Currently Sniffing Device Name : <unknown> Address : dd:2b:48:7f:b5:8c (random) RSSI : -54 Event Count : 0  Advertisers Name : <unknown> Address : 6d:2b:48:7f:b5:8c (random) Name : <unknown> Address : dd:2b:48:7f:b5:8c (random) Adverss : dd:2b:48:7f:b5:8c (random)</unknown></unknown></unknown>	►	Status : Sniffing <unknown></unknown>
Currently Sniffing Device Name : <unknown> Address : dd:2b:48:7f:b5:8c (random) RSSI : -54 Event Count : 0 Advertisers Name : <unknown> Address : d6:8e8.db:97:c1 (random) Name : <unknown> Address : dd:2b:48:7f:b5:8c (random)</unknown></unknown></unknown>		Packet Count: 15 358
Device Name : <unknown> Address : dd:2b:48:7f:b5:8c (random) RSSI : -54 Event Count : 0 Advertisers Name : <unknown> Address : 46:8e:86:db:97:c1 (random) Name : <unknown> Address : dd:2b:46:7f:b5:8c (random)</unknown></unknown></unknown>		USB Device : FTDI - FT231X USB UART [ s/n: DN009WNO ]
Address : dd:2b:48:7f:b5:8c (random) RSSI : -54 Event Count : 0 Advertisers Name : - uuknown> Address : 46:8e:86:db:97:c1 (random) Name : - uuknown> Address : dd:2b:48:7f:b5:8c (random)	Cu	rrently Sniffing
RSSI: -54 Event Count : 0 Advertisers Name : <unknown> Address : 46:8e:86:do:97:c1 (random) Name : <unknown> Address : dd:2b:48:7f:b5:8c (random)</unknown></unknown>		Device Name : <unknown></unknown>
Event Count : 0 Advertisers Name : <ur> <li>vunknown&gt;</li> <li>Address : 48.8e.86.db.97:c1 (random)</li> <li>Name : <ur> <ur> <li>Name : <ur></ur></li></ur></ur></li></ur>		Address : dd:2b:48:7f:b5:8c (random)
Advertisers Name : <unknown> Address : 46:8e:86:db:97:c1 (random) Name : <unknown> Address : dd:2b:48:7f:b5:8c (random)</unknown></unknown>		RSSI: -54
Name : <unknown> Address : 48:86:db:97:c1 (random) Name : <unknown> Address : dd:2b:48:7f:b5:8c (random)</unknown></unknown>		Event Count: 0
Name : <unknown> Address : 48:86:db:97:c1 (random) Name : <unknown> Address : dd:2b:48:7f:b5:8c (random)</unknown></unknown>		
Address : 46:8e:86:db:97:c1 (random) Name : cunknown> Address : dd:2b:46:7f:b5:8c (random)	Adv	ertisers
Address : dd:20:48:7f:b5:8c (random)		
List Device Shiff Device Capture to Wireshark Capture To PCAP Enter Passkey		
List Device Sniff Device Capture to Wireshark Capture To PCAP Enter Passkey		
List Device Sniff Device Capture to Wireshark Capture To PCAP Enter Passkey		
List Devices Sniff Device Capture to Wireshark Capture To PCAP Enter Passkey		
List Devices Sniff Device Capture to Wireshark Capture To PCAP Enter Passkey		
List Devices Sniff Device Capture to Wireshark Capture To PCAP Enter Passkey		
List Devices Shiff Device Capture to Wireshark Capture To PCAP Enter Passkey		
	Lis	t Devices Sniff Device Capture to Wireshark Capture To PCAP Enter Passkey

Please note that there can be a long delay (30-60 seconds) before Wireshark shows up using the tool, due to the X11 startup time, etc.

	X	Capturing from /var/folders/86/	b2vp14n5_5_yvdz_z8w9x_c000	0gn/T/BTLEyJUGqJ	88 [Wireshark 1.12.1 (v1.1	2.1-0-g01b65bf from master-1.12)]	
<u>Eile</u>	dit <u>V</u> iew <u>G</u> o <u>C</u>	apture <u>A</u> nalyze <u>S</u> tatist	ics Telephony <u>T</u> ools <u>I</u> r	nternals <u>H</u> elp			
0 6	) 📶 🔳 🧕	🖻 🗎 🗙 😂 🔍	🗢 🌳 🔏 🛃 🎚	<b>. . . .</b>	R. C. 🖭 📓 🕅	🎦 💥   🗮	
Filter:			- Expression	Clear Apply Sa	ave		
No.	Time	Source	Destination	Protocol L	ength Info		
	78 13.758362000	dd:2b:48:7f:b5:8c	 dcast>	LE LL	56 ADV_IND		
	79 14.302589000	dd:2b:48:7f:b5:8c	 broadcast>	LE LL	56 ADV_IND		
	80 14.303255000 81 14.303919000	dd: 2b: 48: 7f: b5: 8c dd: 2b: 48: 7f: b5: 8c	 dcast> dcast>	LE LL LE LL	56 ADV_IND		
	81 14.303919000	dd: 2b: 48: 7f: b5: 8c dd: 2b: 48: 7f: b5: 8c	<pre><broadcast> <broadcast></broadcast></broadcast></pre>	LE LL	56 ADV_IND 56 ADV_IND		
	83 14.850082000	dd: 2b: 48: 7f: b5: 8c	<pre><broadcast></broadcast></pre>	LE LL	56 ADV_IND		[
	84 14.850746000	dd: 2b: 48: 7f: b5:8c	<pre> droadcast&gt;</pre>	LE LL	56 ADV_IND		
	85 15.401586000	dd: 2b: 48: 7f: b5: 8c	<pre>shroadcast&gt;</pre>	LE LL	56 ADV_IND		
	86 15.402252000	dd:2b:48:7f:b5:8c	 dcast>	LE LL	56 ADV IND		
	87 15.402916000	dd:2b:48:7f:b5:8c	 dcast>	LE LL	56 ADV IND		
	88 15.948595000	dd:2b:48:7f:b5:8c	 dcast>	LE LL	56 ADV_IND		
	89 15.949262000	dd:2b:48:7f:b5:8c	 dcast>	LE LL	56 ADV_IND		
	90 15.949926000	dd:2b:48:7f:b5:8c	 dcast>	LE LL	56 ADV_IND		
-	91 16.499165000	dd:2b:48:7f:b5:8c	 dcast>	LE LL	56 ADV IND		
Nordi ⇒ Bluet Acc ▷ Pac Adb ⇒ Adb ▷ ↓	ic BLE Sniffer Me cooth Low Energy ress Address: OxE ket Header: OxIe vertising Address vertising Data Flags TX Power Level 128-bit Service ( Length: 17 Type: 128-bit S	ta Link Layer e89bed6 40 (PDU Type: ADV_IND, T 40 (PDU Type: ADV_IND, T 1: dd:2b:48:7f:b5:8c (dd:	2b:48:7f:b5:8c)	interface 0			
P CR	: 0x47e7e5	ecauc240ee5a5e05515a5050,	004008				
0010 0 0020 0	2         06         31         01         67         41           0         d6         be         89         8e         40           2         0a         00         11         07         9e           5         01         00         40         6e         e2	1e 8c b5 7f 48 2b dd 02 ca dc 24 0e e5 a9 e0 93	01 06@H+				
0 10	ustom UUID (btc	ommon.eir Packets: 1	89 · Displayed: 189 (100.0	)%)		Profile: Defa	ult

If Wireshark doesn't show up and X11 has been installed correctly, try forcing X11 closed and trying a second time. The startup process can sometimes stall.

## V1 Python API

This page is for the V1 Sniffer firmware only! If you have V2, check the other page - the process has changed between versions.

The Python interface requires a custom Wireshark library for Linux. We're currently working on adding support for this. Please use the Windows or OS X utility until the update is available.

Nordic provides a Python API for their sniffer firmware that makes it possible for us to use the sniffer on any platform, and we've put together a basic wrapper for this API to help you get started.

We've tested this wrapper with Python 2.7 on the following platforms:

- OS X 10.10
- Windows 7 x64
- Ubuntu 14.04

To stream live data into Wireshark the way the <u>official Windows app</u> () from Nordic does you will need to compile a Wireshark utility that creates a name pipe that data gets pushed through.

To keep things simple, though, you can also just log sniffed traffic directly to a libpcap file, which can be opened directly in Wireshark when you are done, which is the easiest solution and what we'll be demonstrating here:

C:\Windows\system32\cmd.exe
Logging data to C:\Users\ktown\AppData\Roaming\Nordic Semiconductor\Sniffer\logs \capture.pcap Connecting to sniffer on COM8 Scanning for BLE devices (5s) Found 2 BLE devices:
[1] "" (E7:0C:E1:BE:87:66, RSSI = -51) [2] "" (14:99:E2:05:29:CF, RSSI = -91)
Select a device to sniff, or '0' to scan again
Attempting to follow device E7:0C:E1:BE:87:66
· · · · · · · · · · · · · · · · · · ·

## Requirements

To use the example we provide for the Python API, you will require the following utilities:

- Python 2.7.x () (we tested with 2.7.6)
- pySerial ()

If you're new to Python and pySerial, have a look at our <u>Instaling Python and PySerial</u> ( ) guide by Simon Monk.

## Download the API

Once you have Python and pySerial installed on your system, you will need to download a copy of the Python API.

The latest version of the API is always <u>available on Github</u> (), but you can also download a .zip file of the latest code directly using the button below:

### Download the Python API from Github

Unzipping the file should give you a file structure resembing the image below:

퉬 SnifferAPI	28/11/2014 15:34	Dossier de fichiers		
wireshark_dissector_source	27/11/2014 09:29	Dossier de fichiers		
DS_Store	26/11/2014 22:05	Fichier DS_STORE	7 Ko	
API Manifest.txt	27/11/2014 09:29	Document texte	1 Ko	
💿 documentation.html	27/11/2014 09:29	Chrome HTML Do	12 Ko	
尾 example.py	26/11/2014 22:20	JetBrains PyCharm	2 Ko	
LICENSE.txt	27/11/2014 09:29	Document texte	2 Ko	
🔁 Nordic Semiconductor Sniffer API Guide	27/11/2014 09:29	Adobe Acrobat D	468 Ko	
readme.md	27/11/2014 09:29	Fichier MD	2 Ko	
尾 sniffer.py	28/11/2014 15:48	JetBrains PyCharm	7 Ko	
sniffer_uart_protocol.xlsx	27/11/2014 09:29	Feuille de calcul	21 Ko	

### Using the sniffer.py Wrapper

To help you get started, we've made an easy to use wrapper called sniffer.py:

It takes a single argument, the COM port location, which will be something like 'COM15' on Windows, '/dev/ttyACM\*' on Linux, or '/dev/tty.usbserial\*' on OS X.

### Linux

To run the sniffer wrapper on Linux, enter the following command (changing the serial port as necessary):

\$ sudo python sniffer.py /dev/ttyACM0

### OS X

To run the sniffer wrapper on OS X, enter the following command (changing the serial port as necessary):

```
$ python sniffer.py /dev/tty.usbserial-DN009MP6
```

### Windows

To run the sniffer wrapper on Windows, enter the following command (changing the serial port as necessary):



python sniffer.py COM30

### Scanning for Devices

If the wrapper was able to connect to the Bluefruit LE Sniffer, it will perform a 5 second scan for Bluetooth Low Energy devices in range, and ask you which device you want to listen to:

```
$ sudo python sniffer.py /dev/ttyACM0
[sudo] password for ktown:
Logging data to logs/capture.pcap
Connecting to sniffer on /dev/ttyACM0
Scanning for BLE devices (5s) ...
Found 2 BLE devices:
   [1] "" (E7:0C:E1:BE:87:66, RSSI = -52)
[2] "" (14:99:E2:05:29:CF, RSSI = -94)
Select a device to sniff, or '0' to scan again
>
```

Once you select a device, it will start scanning that specific device, and you will see an update every second of the number of packets 'sniffed' from the device (where each '.' represents a packet):

```
Select a device to sniff, or '0' to scan again
> 1
Attempting to follow device E7:0C:E1:BE:87:66
```

### Locating the Log File

Once you've sniffed enough data, simply type CTRL+C to stop, and locate the libpcap log file at the path mentionned by the tool. This will normally be:

- Windows: 'C:\Users\ktown\AppData\Roaming\Nordic Semiconductor\Sniffer\logs \capture.pcap' (this will of course change based on your username)
- OS X/Linux: 'logs/capture.pcap' (relative to the location of the Python API)

### Analyze Data in Wireshark

At this point, you simply need to open the capture.pcap file in Wireshark, and you can analyze the sniffed data!

The image below shows an advertising packet from a factory default Bluefruit LE Friend () board:

	corespects [min	eshark 1.12.1 (v1.12.1	-0-g01b65bf from master-1.12	)]				
Eile	<u>E</u> dit <u>V</u> iew <u>G</u>	o <u>C</u> apture <u>A</u> nalyze	<u>Statistics</u> Telephony <u>T</u> oo	ls Internals <u>H</u> elp				
0 0	) 🧉 🔳 🖉	(  🖻 🗎 🗶 🗧	9   9, 4 4 9 🖗 🕯		ର୍ ତ୍ 🖭   👹 🖾	🍢 💥 🛛 😹		
Filter:				<ul> <li>Expression</li> </ul>	Clear Apply Save			
0.	Time	Source	Destination	Protocol L	ength Info			
	28 13.34712		Master	LE LL	60 ADV_IND			
	29 13.34844		Master	LE LL	60 ADV_IND			
	30 13.89326		Master	LE LL	60 ADV_IND			
	31 13.89627 32 13.89870		Master	LE LL	60 ADV_IND			
	32 13.898/0		Master	LE LL LE LL	60 ADV_IND 60 ADV_IND			
	34 14.44967		Master	LE LL	60 ADV_IND			
	35 14.45134		Master	LE LL	60 ADV_IND			
	36 14.99882		Master	LE LL	60 ADV_IND			
								•
En	amo 2221 60	but on wine	(480 bits), 60 bytes	contured (480	hitc)			
	ndic BLE sr		(400 bits), oo bytes	captured (480	0103)			
		Energy Link L	aver					
		ess: 0x8e89bed						
			U Type: ADV_IND, TXAd	ld=false, RxAdd	=false)			
			c:e1:be:87:66 (e7:0c:					
	Advertising	Data						
Ę	000 0 0. 1 1 	<pre> = Simultane  = BR/EDR NO 1. = LE Genera .0 = LE Limite Level</pre>	ous LE and BR/EDR to ous LE and BR/EDR to t Supported: true (Ox l Discoverable Mode: d Discoverable Mode: (0xOa)	Same Device Ca 01) true (0x01)			)	
	Length: Type: 1 Custom	17 28-bit Service UUID: 9ecadc24	Class UUIDs (0x07) 0ee5a9e093f3a3b501004	06e				
-	Length: Type: 1 Custom Custom RC: 0xf922 Expert 1	17 28-bit Service UUID: 9ecadc24 9f nfo (Chat/Prot	class UUIDs (0x07)	06e				
-	Length: Type: 1 Custom RC: 0xf922 [Expert 1 [correc	17 28-bit Service UUID: 9ecadc24 9 <b>f</b> Info (Chat/Prot t]	Class UUIDS (0x07) 0ee5a9e093f3a3b501004 ocol): correct]	06e				
-	Length: Type: 1 Custom RC: 0xf922 [Expert 1 [correc [Severi	17 28-bit Service UUID: 9ecadc24 9f (nfo (Chat/Prot t] ty level: Chat	Class UUIDS (0x07) 0ee5a9e093f3a3b501004 ocol): correct]	06e				
-	Length: Type: 1 Custom RC: 0xf922 [Expert 1 [correc [Severi	17 28-bit Service UUID: 9ecadc24 9 <b>f</b> Info (Chat/Prot t]	Class UUIDS (0x07) 0ee5a9e093f3a3b501004 ocol): correct]	06e				
-	Length: Type: 1 Custom CCSC 0xf922 [Expert 1 [correc [Severi [Group: 08 06 35 00 d6 b0 02 02 01	17 28-bit Service UUID: 9ecadc24 9f info (Chat/Prot t] ty level: Chat Protocol] 01 dd 0b 06 0a	Class UUIDS (0x07) 0ee5a9e09373a3b501004 ocol): correct] ] 01 27 34 00 00 98 0 87 be el 0c e7 03 1 07 96 ca do 24 0e c	2 00	<sup>1</sup> 4 			
0000 010 020 030	Length: Type: 1 Custom CC: 0xf922 [Expert 1 [correc [severi [Group: 08 06 35 00 d6 be 02 02 01 e0 93 f3	17 28-bit Service UJD: 9ecadc24 19f info (chat/Prot t] 1ty level: chat Protocol] 01 dd 0b 06 0a 89 8e 40 22 66 06 02 0a 00 11 a3 b5 01 00 40	Class UUIDS (0x07) 0ee5a9e093f3a3b501004 ocol): correct] ] 01 27 34 00 00 98 0 57 be el 0c e? 01 1 07 90 ca de 24 0e e 6e 9f 44 f9	2 00	· · · · · · · · · · · · · · · · · · ·			

Note that the utility will start sniffing data as soon as you connect to the Bluefruit LE Sniffer, so early packets in the log file might contain advertising packets from other devices in range. It will only start filtering packets once you select a specific device via the selection dialogue.

For information on how to use Wireshark, have a look at the <u>notes on the official nRF</u> <u>Sniffer utility</u> (), which describes some of the packet types you might encounter working with Bluetooth Low Energy.

## FAQs

# I'm using the V2 (BETA 1) firmware, but can't seem to connect in Wireshark?

There are a number of possible issues here, mostly revolving around the fact that the system depends on a Python script piping data into Wireshark. You may find the following post from a user help to try to debug this, the cause being a potential conflict with multiple instances of Python on your system: https://forums.adafruit.com/viewtopic.php?f=53&t=146215#p726333 ()

# When I connect to a Central device, I don't see any connection data, but when I disconnect I see the advertising packets again. How do I capture data with a connected peripheral?

This is a limitation of the sniffer firmware from Nordic. Advertising in Bluetooth Low Energy happens on three dedicated channels, each running at it's own frequency. For the sniffer to 'follow' the connection it needs to be looking at the right channel when the connection happens, and there is a 2/3 chance that it is looking at another channel at any given moment.

To capture the connection and see data exchanges post connection, you may need to connect several times until the channels are aligned between the sniffer and the BLE peripheral+central devices.

# How do I convert between Sniffer and Bluefruit LE firmware using SWD?

Reflashing Bluefruit LE modules over SWD (ex. switching to the sniffer firmware and back) is at your own risk and can lead to a bricked device, and we can't offer any support for this operation! You're on your own here, and there are unfortunately 1,000,000 things that can go wrong, which is why we offer two separate Bluefruit LE Friend boards -- the sniffer and the normal Bluefruit LE Friend board with the non-sniffer firmware, which provides a bootloader with fail safe features that prevents you from ever bricking boards via OTA updates.

AdaLink (SWD/JTAG Debugger Wrapper)

Transitioning between the two board types (sniffer and Bluefruit LE module) is unfortunately not a risk-free operation, and requires external hardware, software and know-how to get right, which is why it isn't covered by our support team.

That said ... if you're determined to go down that lonely road, and you have a Segger J-Link () (which is what we use internally for production and development), or have already erased your Bluefruit LE device, you should have a look at AdaLink (), which is the tool we use internally to flash the four files required to restore a Bluefruit LE module. (Note: recent version of AdaLink also support the cheaper STLink/V2 (http://adafru.it/2548), though the J-Link is generally more robust if you are going to purchase a debugger for long term use.)

To go from the sniffer to Bluefruit LE firmware the mandatory Intel Hex files are available in the Bluefruit LE Firmware repo (). You will need to flash:

• An appropriate bootloader image

- An appropriate SoftDevice image
- The Bluefruit LE firmware image
- The matching signature file containing a CRC check so that the bootloader accepts the firmware image above (located in the same folder as the firmware image)

The appropriate files are generally listed in the version control .xml file () in the firmware repository.

If you are trying to flash the sniffer firmware (at your own risk!), you only need to flash a single .hex file, which you can find here (). The sniffer doesn't require a SoftDevice image, and doesn't use the fail-safe bootloader -- which is why changing is a one way and risky operation if you don't have a supported SWD debugger.

### Adafruit\_nF51822\_Flasher

We also have an internal python tool available that sits one level higher than AdaLink (referenced above), and makes it easier to flash specific versions of the official firmware to a Bluefruit LE module. For details, see the Adafruit\_nRF51822\_Flasher () repo.

### Why isn't the Firmware V1 plugin working in Wireshark?

The Sniffer Firmware V1 plugin was written for Wireshark 1.12.x and won't work with older versions of the tool or the new 2.x family. Be sure to download an appropriate version (for example 1.12.1, which is the version used in this guide).

# Why am I being warned my Sniffer V1 firmware is out of data but updates fail?

The Adafruit board has a small difference compared to the original Nordic HW that Nordic wrote their sniffer firmware for. To keep the cost as low as possible, we don't populate the optional 32.768KHz RTC crystal on our boards, whereas it is present on the more expensive Nordic development kit.

Because the startup code in the sniffer firmware from Nordic uses this crystal, we had to request a custom version from Nordic that uses the internal 16MHz RC oscillator instead. When providing us the custom firmware, they changed the version number slightly, which is the reason for the warning message.

You can safely ignore the firmware update warning and use the device as normal, and in fact updating to a firmware from Nordic won't work unless you also solder the optional 32.768KHz crystal on the bottom of your PCB as well.

### How can I check that the sniffer is outputting data?

If you think there is a problem with your sniffer, you should look at the LED closest to the black SWD connector box at the end of the board. It should flash every time Bluetooth Low Energy activity is detected when the serial port is open.

You can also open a Terminal Emulator (Putty, RealTerm, etc.) with the following settings, and you should see data coming out almost as soon as you plug the sniffer in:

- Baud Rate: 460800
- HW Flow Control: RTS + CTS Enabled

# What is the difference between blue boards and black board)?

The Black boards (hardware v3) uses the much cheaper CP2104 USB to Serial bridge, and drops the SWD connector which had to be manually placed during the manufacturing process. This allows us to offer the sniffer board at a significant discount compared to the original, without sacrificing functionality that 99% of customers required. (The SWD pins are still available as pads on the bottom of the PCB if you need them!).

If you have a Blue board - you definitely have hardware version 1 and Firmware version 1

If you have a Black board - you definitely have hardware version 3. You may have firmware version 1 or version 2 depending on when you purchased it. Check your order receipt to know!



## Downloads

## Files

- EagleCAD PCB files on GitHub ()
- Bluetooth LE module Datasheet ()

## Schematic C2104 Rev

Click to embiggen

