BLETOOL Manual

Version: BLETOOL _Manual_V0.5

Date: 2020-03-30

History

Version	Date	Description
0.1	2019-4-18	Initial
0.2	2019-5-06	Add connection, gatt operation commands
0.3	2020-2-06	Delete some commands and fix calibrate parameters and return values
0.4	2020-3-26	Add C/C++ API
0.5	2020-3-30	Change introductions

Table of Contents

1. Description

1.1 What's bletool

BleTool is a software develop kit for Bluetooth Low Energy (BLE) in GL-iNET's products. It provides a basic and simple method for developers to operate all the BLE functions.

Different from BlueZ which includes the full Bluetooth protocol stack in the host system, bletool is a light weight tool to operate hostless BLE modules which has fully built-in protocol stack. The module can fully operate on itself rather than depending on the host system.

To use BleTool, you need to have one of the following devices.

- GL-S1300 (Convexa-S): Smarthome gateway with beamforming Wi-Fi
- GL-S100: Smarthome gateway with 2.4G Wi-Fi
- GL-X750 (Spitz): LTE IoT gateway
- GL-B2200 (): Whole home mesh system and gateway

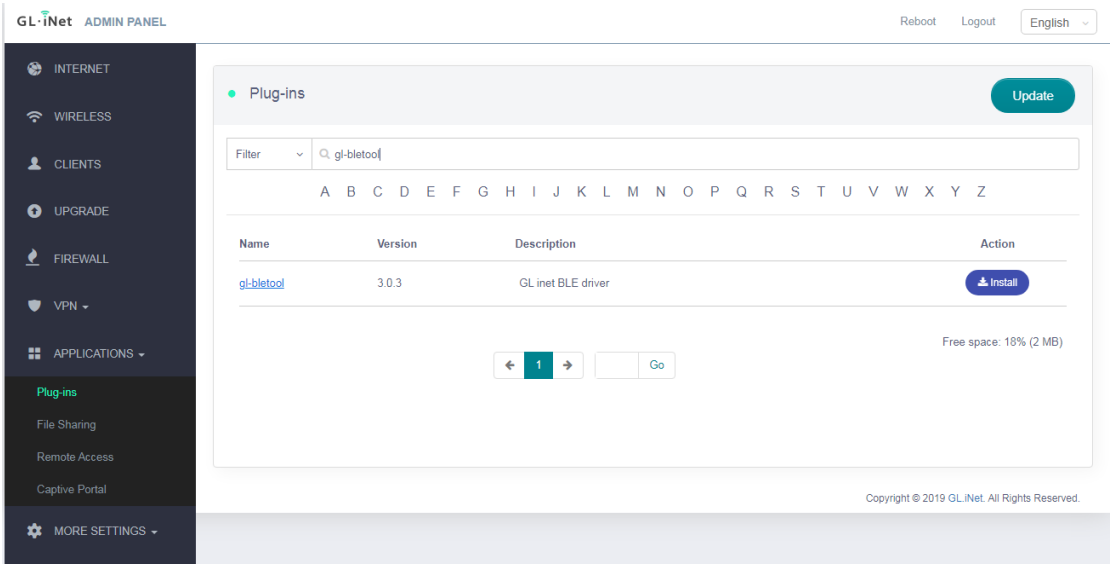
You can also use BleTool if you use Silconlabs EFR32 BLE modules which use UART/SPI to connect to your host Linux.

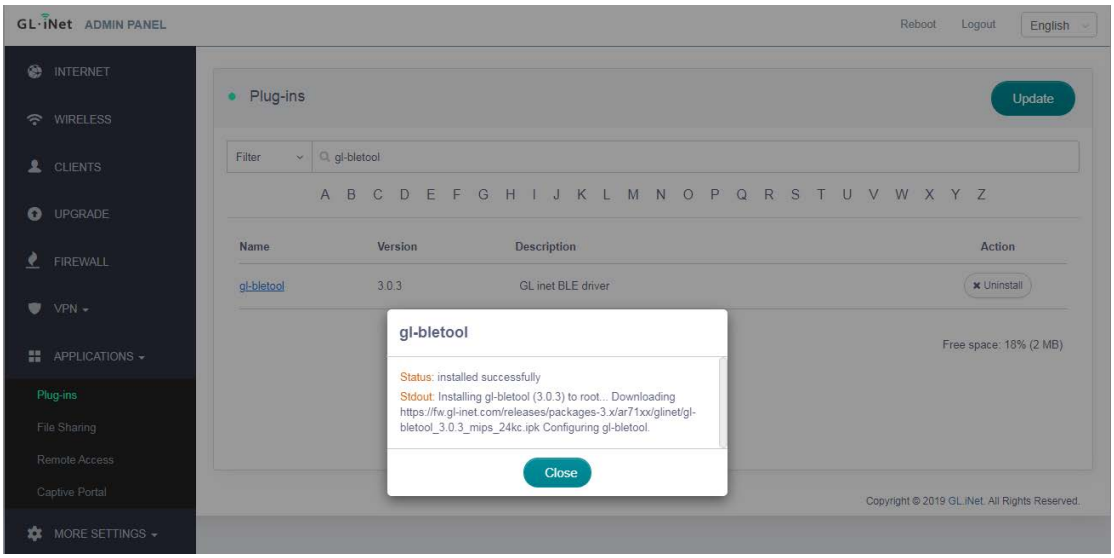
1.2 How to install

By default, BleTool is not installed on your router. You can install it using opkg if you can ssh to the router.

```
opkg update
opkg install gl-bluetool
```

Alternatively, you can install using the web UI. Login your router's web UI using your browser which is http://192.168.8.1 by default. Then go to APPLICATIONS->Plug-ins. First click "Update" to refresh your software repo then search "gl-bletool". Click "install" and wait until you got "installation successfully".





1.3 How to use

BleTool provides the following elements to handle BLE advertising, connection and GATT services.

- C/C++ APIs: This includes C functions, C header files based on which you can write your own code.
- C/C++ library: You can link this library with your own C application. You need to include the C header files in your own code to compile.
- cli (command line) tools: cli is commands that you can run in Linux terminal. You can use cli tools to test your BLE applications quickly and easily.

Here is example of how to use cli commands.

```
BusyBox v1.28.3 () built-in shell (ash)

  _______                     ________        __
 |       |.-----.-----.-----.|  |  |  |.----.|  |_
 |   -   ||  _  |  -__|     ||  |  |  ||   _||   _|
 |_______||   __|_____|__|__||________||__|  |____|
          |__| W I R E L E S S   F R E E D O M
 -----------------------------------------------------
 OpenWrt 18.06.1, r7258-5eb055306f
 -----------------------------------------------------
root@GL-X750:~# bletool enable
{ "code": 0 }
root@GL-X750:~# bletool local_address
{ "mac": "00:0b:57:f5:f2:f1", "code": 0 }
root@GL-X750:~#
```

Below is the details of the API reference as well as the cli commands.

2. API References

Note that each API function will generate a message and pass to its fixed parameter “method_handler_t cb” after been called. It’s a function pointer which in format void (*method_handler_t) (json_object* msg). This should be appointed by user to handle the message.

2.1 enable

Enable or disable the BLE hardware.

C API:

int gl_ble_enable(method_handler_t cb,int enable);

Parameters

Type	Name	Description
int	enable	0 means disable the BLE hardware; None-zero means enable the BLE hardware.

Result

Type	Name	Description
int	code	0 means success; None-zero means failed.

CLI command:

bletool enable 1

Parameters

Type	Name	Default Value*	Description
int	enable	1	0 means disable the BLE hardware; None-zero means enable the BLE hardware.

Note that must call this command or API before using any other BLE commands or functions.

**A default value means you may not set this parameter. “-” means you must set this parameter.*

2.2 local_address

Enable or disable the BLE hardware.

C API:

int gl_ble_get_mac(method_handler_t cb);

No parameter.

Result

Type	Name	Description
int	code	0 means success;

		None-zero means failed.
string	address	Local Bluetooth address like “11:22:33:44:55:66”

CLI command:

bletool local_address

2.3 set_power

Set the global power level.

C API:

int gl_ble_set_power(method_handler_t cb,int power);

Parameters

Type	Name	Description
int	power	TX power in 0.1dBm steps, for example the value of 10 is 1dBm and 55 is 5.5dBm

Result

Type	Name	Description
int	code	0 means success; None-zero means failed.
int	power	Actual adopted power level.

CLI command:

bletool set_power 80

Parameters

Type	Name	Default Value	Description
int	power	-	Power level

2.4 listen

Listen to events generated from the BLE module.

C API:

int gl_ble_subscribe(method_handler_t cb);

This function will subscribe events generate from BLE module. Note that it must be followed by uloop_run(), it will continuously pass events to function cb.

int gl_ble_unsubscribe(void);

This function will unsubscribe the BLE events.

CLI command:

bletool listen

This command will not return. It will continuously print events generated from BLE module.

2.5 adv_data

Act as BLE slave, set customized advertising data

C API:

int gl_ble_adv_data(method_handler_t cb, int flag, char* data);

Parameters

Type	Name	Description
int	flag	Adv data flag. This value selects if the data is intended for advertising packets, scan response packets or advertising packet in OTA. • **0:** Advertising packets • **1:** Scan response packets • **2:** OTA advertising packets • **4:** OTA scan response packets
string	data	Customized advertising data. Must be hexadecimal ASCII. Like "020106"

Result

Type	Name	Description
int	code	0 means success; None-zero means failed.

CLI command:

bletool adv_data –f 0 –v 020106

Parameters

Type	Name	Default Value	Description
int	flag -f	-	Adv data flag.
string	data -v	-	Customized advertising data.

2.6 adv

Set the advertising parameters and start advertising act as BLE slave.

C API:

int gl_ble_adv(method_handler_t cb, int phys, int interval_min,int interval_max,int discover,int connect);

Parameters

Type	Name	Description
int	phys	The PHY on which the advertising packets are transmitted on. • **1:** LE 1M PHY • **4:** LE Coded PHY
int	interval_min	Minimum advertising interval. Value in units of 0.625 ms • Range: 0x20 to 0xFFFF • Time range: 20 ms to 40.96 s
int	interval_max	Maximum advertising interval. Value in units of 0.625 ms • Range: 0x20 to 0xFFFF • Time range: 20 ms to 40.96 s • Note: interval_max should be bigger than interval_min
int	discover	Define the discoverable mode. • **0:** Not discoverable • **1:** Discoverable using both limited and general discovery procedures • **2:** Discoverable using general discovery procedure • **3:** Device is not discoverable in either limited or generic discovery procedure, but may be discovered by using the Observation procedure • **4:** Send advertising and/or scan response data defined by the user. The limited/general discoverable flags are defined by the user.
int	connect	Connectable mode. • **0:** Non-connectable non-scannable • **1:** Directed connectable (RESERVED, DO NOT USE) • **2:** Undirected connectable scannable (This mode can only be used in legacy advertising PDUs) • **3:** Undirected scannable (Non-connectable but responds to scan requests) • **4:** Undirected connectable non-scannable. This mode can only be used in extended advertising PDUs

Result

Type	Name	Description
int	code	0 means success; None-zero means failed.

CLI command:

bletool adv

Parameters

Type	Name	Default Value	Description
int	phys -p	1	The PHY on which the advertising packets are transmitted on.

int	interval_min -n	160 (100ms)	Minimum advertising interval.
int	interval_max -x	160 (100ms)	Maximum advertising interval.
int	discover -d	2	Discoverable mode.
int	connect -c	2	Connectable mode.

2.7 adv_stop

Set the advertising parameters and start advertising act as BLE slave.

C API:

int gl_ble_stop_adv(method_handler_t cb);

No parameter.

Result

Type	Name	Description
int	code	0 means success; None-zero means failed.

CLI command:

bletool adv_stop

2.8 send_notify

Act as GATT server, send Notification to remote device.

C API:

int gl_ble_send_notify(method_handler_t cb,int connection,int char_handle, char* value);

Parameters

Type	Name	Description
int	connection	Connection handle
int	char_handle	GATT characteristic handle
string	value	Data value to be sent.

Result

Type	Name	Description
int	code	0 means success; None-zero means failed.

2.9 discovery

Act as master, set and start the BLE discovery.

C API:

int gl_ble_discovery(method_handler_t cb,int phys,int interval,int window,int type,int mode);

Note that after call this function, BLE packets will be continuously pass to callback function registered by gl_ble_subscribe();

Parameters

Type	Name	Description
int	phys	The scanning PHY. • 1: LE 1M PHY • 4: LE Coded PHY
int	interval	Scan interval. • Time = Value x 0.625 ms • Range: 0x0004 to 0xFFFF • Time Range: 2.5 ms to 40.96 s
int	window	Scan window. • Time = Value x 0.625 ms • Range: 0x0004 to 0xFFFF • Time Range: 2.5 ms to 40.96 s
int	type	Scan type. Values: • **0:** Passive scanning • **1:** Active scanning • In passive scanning mode, the device only listens to advertising packets and does not transmit packets. • In active scanning mode, the device sends out a scan request packet upon receiving an advertising packet from a remote device. Then, it listens to the scan response packet from the remote device
int	mode	Bluetooth discovery Mode. • **0:** Discover only limited discoverable devices • **1:** Discover limited and generic discoverable devices • **2:** Discover all devices

Result

Type	Name	Description
int	code	0 means success; None-zero means failed.

CLI command:

bletool discovery

Note that you have to using command "*bletool listen"* to receive BLE advertising packets after this command.

Parameters

Type	Name	Default Value	Description
int	phys -p	1	The scanning PHY.
int	interval -i	16 (10ms)	Scan interval.
int	window -w	16 (10ms)	Scan window.
int	type -t	0	Scan type.
int	mode -m	1	Bluetooth discovery Mode.

2.10 stop

Act as master, stop discovery procedure.

C API:

int gl_ble_stop(method_handler_t cb);

No parameter.

Result

Type	Name	Description
int	code	0 means success; None-zero means failed.

CLI command:

bletool stop

2.11 connect

Act as master, start connect to a remote BLE device.

C API:

int gl_ble_connect(method_handler_t cb,char* address,int address_type,int phy);

Parameters

Type	Name	Description
string	address	Remote BLE device address. Like “11:22:33:44:55:66”
int	address_type	Advertiser address type. Values: • **0:** Public address • **1:** Random address • **2:** Public identity address resolved by stack • **3:** Random identity address resolved by stack
int	phy	The initiating PHY.

		• **1:** LE 1M PHY • **4:** LE Coded PHY

Result

Type	Name	Description
int	code	0 means success; None-zero means failed.
int	connection	Handle of new connection
int	address	Remote device address
int	address_type	Remote device address type
int	master	Device role in connection. Values: • **0:** Slave • **1:** Master
int	bonding	Bonding handle if the remote advertising device has previously bonded with the local device. Values: • **0xff:** No bonding • **Other:** Bonding handle
int	interval	Connection interval
int	latency	Slave latency
int	timeout	Connection timeout
int	security_mode	Connection security mode. Values: • **0:** No security • **1:** Unauthenticated pairing with encryption • **2:** Authenticated pairing with encryption • **3:** Authenticated Secure Connections pairing with encryption using a 128-bit strength encryption key
int	txsize	Maximum Data Channel PDU Payload size the controller can send in an air packet

CLI command:

bletool connect –a 11:22:33:44:55:66 –t 0

Parameters

Type	Name	Default Value	Description
string	address -a	-	Remote BLE device address.
int	address_type -t	-	Advertiser address type.
int	phy -p	1	The initiating PHY.

2.12 disconnect

Act as master, disconnect with remote device.

C API:

int gl_ble_disconnect(method_handler_t cb,int connection);

Parameters

Type	Name	Description
int	connection	Connection handle

Result

Type	Name	Description
int	code	0 means success; None-zero means failed.
int	connection	Connection handle
int	reason	Connection disconnect reason

CLI command:

bletool disconnect 1

Parameters

Type	Name	Default Value	Description
int	connection	-	Connection handle

2.13 get_rssi

Act as master, get rssi of connection with remote device.

C API:

int gl_ble_get_rssi(method_handler_t cb,int connection);

Parameters

Type	Name	Description
int	connection	Connection handle

Result

Type	Name	Description
int	code	0 means success; None-zero means failed.
int	connection	Connection handle
int	rssi	Rssi of the specified connection (dBm)

CLI command:

bletool get_rssi 1

Parameters

Type	Name	Default Value	Description
int	connection	-	Connection handle

2.14 get_service

Act as master, get service list of a remote GATT server.

C API:

int gl_ble_get_service(method_handler_t cb, int connection);

Parameters

Type	Name	Description
int	connection	Connection handle

Result

Type	Name	Description
int	code	0 means success; None-zero means failed.
int	connection	Connection handle
jsonArray	service_list	Array of service list

service_list

Type	Name	Description
int	service_handle	service hanle
int	UUID	UUID of service

CLI command:

bletool get_service 1

Parameters

Type	Name	Default Value	Description
int	connection	-	Connection handle

2.15 get_char

Act as master, Get characteristic list of a remote GATT server.

C API:

int gl_ble_get_char(method_handler_t cb, int connection, int service_handle);

Parameters

Type	Name	Description

int	connection	Connection handle
int	service_handle	service handle

Result

Type	Name	Description
int	code	0 means success; None-zero means failed.
int	connection	Connection handle
jsonArray	characteristic_list	Array of characteristics

characteristic_list

Type	Name	Description
int	characteristic_handle	characteristic handle
int	UUID	UUID of characteristic
int	properties	Characteristic properties

CLI command:

bletool get_char –c 1 –h 10789

Parameters

Type	Name	Default Value	Description
int	connection -c	-	Connection handle
int	service_handle -h	-	Service handle

2.16 set_notify

Act as master, Enable or disable the notification or indication of a remote gatt server.

C API:

int gl_ble_set_notify(method_handler_t cb, int connection, int char_handle,int flag);

Parameters

Type	Name	Description
int	connection	Connection handle
int	char_handle	Characteristic handle
int	flag	Notification flag. • **0:** disable • **1:** notification • **2:** indication

Result

Type	Name	Description
int	code	0 means success; None-zero means failed.

CLI command:

bletool set_notify –c 1 –h 10789 –f 1

Parameters

Type	Name	Default Value	Description
int	connection -c	-	Connection handle
int	char_handle -h	-	Characteristic handle
int	flag -f	-	Notification flag.

2.17 read_value

Act as master, Read value of specified characteristic in a remote gatt server.

C API:

int gl_ble_read_char(method_handler_t cb, int connection, int char_handle);

Parameters

Type	Name	Description
int	connection	Connection handle
int	char_handle	Characteristic handle

Result

Type	Name	Description
int	code	0 means success; None-zero means failed.
int	connection	Connection handle
int	char_handle	Characteristic handle
int	att_opcode	Attribute opcode which informs the GATT transaction used.
int	offset	Value offset
string	value	Characteristic value. In hexadecimal ASCII. Like “00560aff”

CLI command:

bletool read_value –c 1 –h 10789

Parameters

Type	Name	Default Value	Description
int	connection -c	-	Connection handle
int	char_handle	-	Characteristic handle

	-h		

2.18 write_value

Act as master, Write value to specified characteristic in a remote gatt server.

C API:

int gl_ble_write_char(method_handler_t cb, int connection, int char_handle,char* value,int res);

Parameters

Type	Name	Description
int	connection	Connection handle
int	char_handle	Characteristic handle
string	value	Value to be written. Must be hexadecimal ASCII. Like “00010203”
int	res	Response flag. • **0:** Write with no response • **1:** Write with response

Result

Type	Name	Description
int	code	0 means success; None-zero means failed.
int	sent_len	Bytes be written successfully

CLI command:

bletool write_value –c 1 –h 10789 –v 00000000 –r 0

Parameters

Type	Name	Default Value	Description
int	connection -c	-	Connection handle
int	char_handle -h	-	Characteristic handle
string	value -v		Value to be written
int	res -r	0	Response flag