CC32xx HTTP Client Demo

Overview

Return to CC31xx & CC32xx Home Page

Return to CC31xx Sample Applications

HTTP client library can be used to communicate and exchange data with the HTTP web server.
HTTP client library support two modes

1. **Minimum mode:** HTTP client library in minimum mode supports synchronous mode, redirection handling, chunked transfer encoding, proxy and TLS. **HTTPCli_LIBTYPE_MIN** flag should be used to build the library in minimum mode.
2. **Full Mode:** HTTP Client library in full mode supports all the features of the minimal mode along with asynchronous mode and content handling and requires RTOS support. **HTTPCli_LIBTYPE_FULL** flag should be used to build the library in full mode.

This sample application demonstrates the HTTP Client library API for HTTP based application development in minimum mode. This application explain user to how to:

- Connect to an access point
- Connect to a HTTP Server with and without proxy
- Do POST, GET, PUT and DELETE
- Parse JSON data using “Jasmine JSON Parser”

To enable the secure connection (in example) user need to add the follow code snippet

```
SlDateTime_t dt;
struct HTTPCli_SecureParams sparams;

/* Set current Date to validate certificate */
dt.sl_tm_day = DATE;
dt.sl_tm_mon = MONTH;
dt.sl_tm_year = YEAR;
dt.sl_tm_hour = HOUR;
dt.sl_tm_min = MINUTE;
dt.sl_tm_sec = SECOND;
sl_DevSet(SL_DEVICE_GENERAL_CONFIGURATION,
                            SL_DEVICE_GENERAL_CONFIGURATION_DATE_TIME,
                            sizeof(SlDateTime_t), (unsigned char *)(&dt));

/* Security parameters */
sparams.method.secureMethod = SL_SO_SEC_METHOD_TLSV1_2;
sparams.mask.secureMask  = SL_SEC_MASK_TLS_RSA_WITH_AES_256_CBC_SHA;
strncpy(sparams.cafile, SL_SSL_CA_CERT, sizeof(SL_SSL_CA_CERT));
sparams.privkey[0] = 0;
sparams.cert[0] = 0;
sparams.dhkey[0] = 0;
HTTPCli_setSecureParams(&sparams);
```

And **HTTPCli_connect** function should be called with **HTTPCli_TYPE_TLS** option.

```
HTTPCli_connect(&cli, (struct sockaddr *)&addr, HTTPCli_TYPE_TLS, NULL);
```

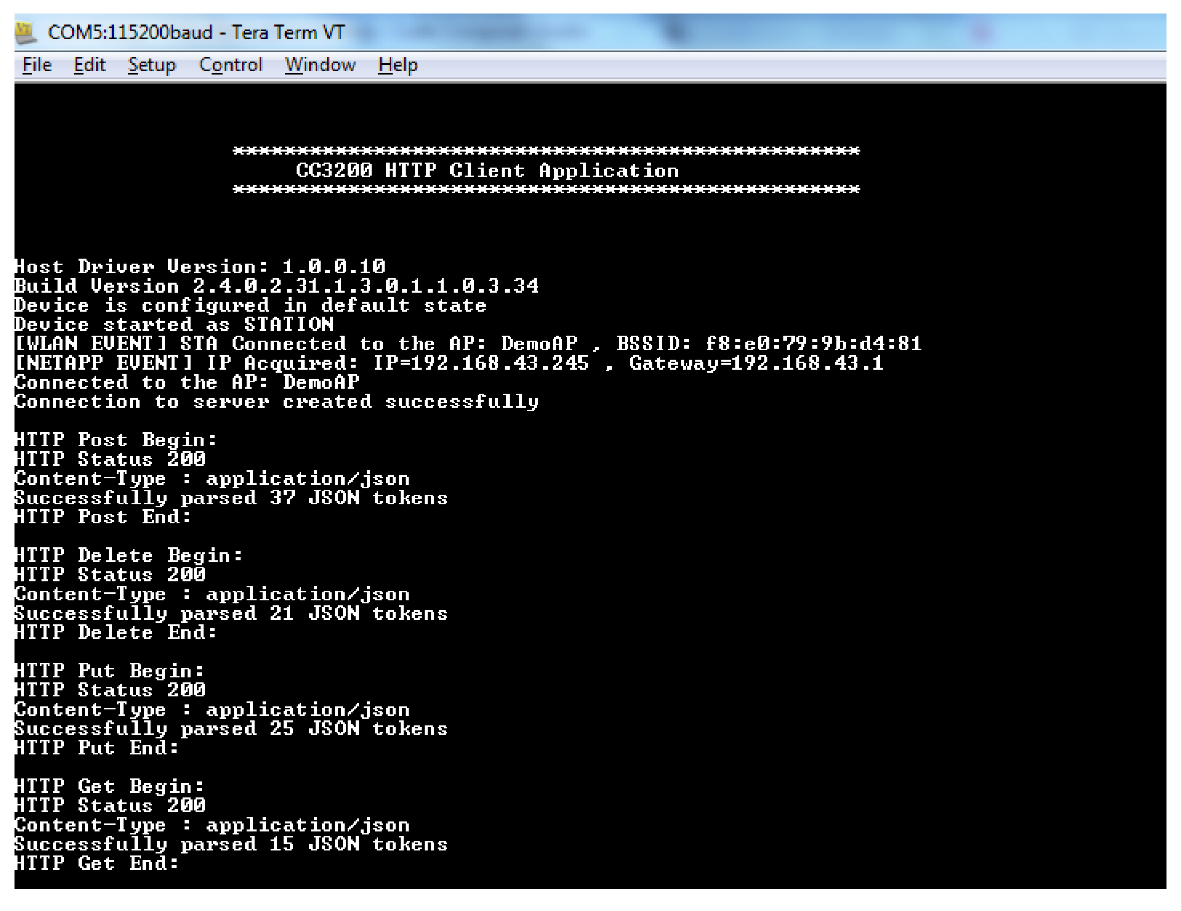
User can change the proxy by changing the MACRO 'PROXY_IP', ‘PROXY_PORT'. To enable proxy user need to define USE_PROXY macro in project properties.

HTTP Client library API details are provided in SDK under docs folder.

Usage

Prerequisite: This application requires an access-point with internet connectivity

- Connect the board to a Windows-PC and configure the terminal-program for seeing the logs - [l CC31xx_&_CC32xx_Terminal_Setting_Wiki [1]] has detailed instructions for configuring the terminal-program
- Open **sl_common.h** and and change **SSID_NAME**, **SEC_TYPE** and **PASSKEY** as per your access-point's properties. SimpleLink device will connect to this AP when the application is executed
- Open **main.c** and change **HOST_NAME** and **HOST_PORT** as per your server properties. Other HTTP request parameters may also change depending upon the server. This example is tested with **httpbin.org** host and **80** port.
- Build and launch the project, the application tries to connect to AP.
- Upon connection application tries to connect to the http host.
- Upon successful connection application will send POST, DELETE, PUT and GET request and check and parse the response.
- See the self explanatory logs on the terminal-program's console. On success, below message will be displayed on the terminal



COM5:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
                 *************************************************
                      CC3200 HTTP Client Application
                 *************************************************


Host Driver Version: 1.0.0.10
Build Version 2.4.0.2.31.1.3.0.1.1.0.3.34
Device is configured in default state
Device started as STATION
[WLAN EVENT] STA Connected to the AP: DemoAP , BSSID: f8:e0:79:9b:d4:81
[NETAPP EVENT] IP Acquired: IP=192.168.43.245 , Gateway=192.168.43.1
Connected to the AP: DemoAP
Connection to server created successfully

HTTP Post Begin:
HTTP Status 200
Content-Type : application/json
Successfully parsed 37 JSON tokens
HTTP Post End:

HTTP Delete Begin:
HTTP Status 200
Content-Type : application/json
Successfully parsed 21 JSON tokens
HTTP Delete End:

HTTP Put Begin:
HTTP Status 200
Content-Type : application/json
Successfully parsed 25 JSON tokens
HTTP Put End:

HTTP Get Begin:
HTTP Status 200
Content-Type : application/json
Successfully parsed 15 JSON tokens
HTTP Get End:
```

Limitations/Known Issues

- HTTP connection timeout is not supported.
- Only IPV4 is supported

References

[1] http://processors.wiki.ti.com/index.php/CC31xx_&_CC32xx_Terminal_Setting

Article Sources and Contributors

CC32xx HTTP Client Demo *Source*: http://processors.wiki.ti.com/index.php?oldid=194920 *Contributors*: A0132173, Jitgupta

Image Sources, Licenses and Contributors

File:Cc31xx cc32xx return home.png *Source*: http://processors.wiki.ti.com/index.php?title=File:Cc31xx_cc32xx_return_home.png *License*: unknown *Contributors*: A0221015

File:Cc32xx return sample apps.png *Source*: http://processors.wiki.ti.com/index.php?title=File:Cc32xx_return_sample_apps.png *License*: unknown *Contributors*: A0221015

Image:Http client demo.png *Source*: http://processors.wiki.ti.com/index.php?title=File:Http_client_demo.png *License*: unknown *Contributors*: Jitgupta