Skywire® LTE Global CAT-M1 NL-SW-LTE-QBG96 Socket Dial Application Note

NimbeLink Corp

Updated: June 2018

Table of Contents

1. Introduction

1.1 Orderable Part Numbers

Orderable Device	Description	Carrier	Network Type
NL-SWDK	Skywire Development Kit	Any	Any
NL-SW-LTE-QBG96	LTE CAT M1	Any	LTE

1.2 Prerequisites

This document assumes that the initial setup of the requisite modem and development kit has been completed using the Skywire® Development Kit User Manual.

If these steps are incomplete, please refer to the link above and complete the modem setup before proceeding.

2. Default Socket Dial Procedure

2.1 Overview

Socket dials are a useful process for uploading or downloading information from a website or database using HTTP commands. This document will cover two ways to complete a socket dial, one using the default online data mode and the other using command mode. This section will cover online data mode, and Section 3 will cover command mode. Finally, three examples of a complete socket dial procedure will be covered in Section 4, Section 5 and Section 6.

2.2 Socket Configuration

First, the socket context ID must be configured. Issue this command into the terminal, followed by the enter key:

```
AT+QIACT=1
```

The terminal should respond with:

```
OK
```

2.3 Initiate Socket Dial

In the terminal program, type the following command::

```
AT+QIOPEN=1,0,"TCP","example.com",80,0,2
```

In this case, `1` is the socket context being used, `0` specifies socket service index, TCP is the HTTP protocol being used, example.com is the hostname, `80` is the TCP port being used (TCP port 80 is used for HTTP), 0 is the local port, and 2 is for transparent access mode.

The modem should respond with:

```
CONNECT
```

2.4 Send Data via HTTP

In order to send data to the web server, use the HTTP POST command. The syntax of the POST command is as follows:

```
POST /test/demo_form.asp HTTP/1.1
```

POST is the HTTP command being issued, /test/demo_form.asp is the endpoint of the server, and HTTP/1.1 is the HTTP version being used.

In the terminal program, enter the desired POST data, making sure that it is properly formatted according to the syntax listed above. Additionally, please note that the text being entered into the terminal will not be echoed, making manual data entry difficult. Accordingly, it is recommended that the command be pasted into the terminal, as opposed to typed.

Once the appropriate command has been entered, press the sequence **CTRL+M, CTRL+J, CTRL+M, CTRL+J**. This sequence will enter two new line characters, which are necessary to signal to the server that data transmission is complete. After a short delay, the terminal program should display:

```
HTTP/1.1 2xx OK

[text response from server]

OK
```

where 2xx is the successful response code of the HTTP server. 200 is the general HTTP/1.1 response of OK, meaning that the response was received properly. There are derivations of the 2xx response, however. If an unexpected value is received, please consult the HTTP/1.1 Protocol Documentation at the following URL:

http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html

2.5 Shutdown Socket Connection

In the terminal program, type the following command, followed by the enter key:

```
AT+QICLOSE=1
```

The output from the terminal should be:

```
OK
```

3. Socket Dial Using Buffer Access Mode

3.1 Overview

The following section will lay out an alternate procedure for completing a socket dial using buffer access mode.

3.2 Socket Configuration

To configure the socket, type the following command into the terminal, followed by the enter key:

```
AT+QIACT=1
```

The terminal should respond with:

```
OK
```

3.3 Initiate Socket Dial

Type the following command into the terminal to open the socket, followed by the enter key:

```
AT+QIOPEN=1,0,"TCP","example.com",80,0,0
```

In this case, `1 or 3` is the socket context being used, `0` specifies socket service index, TCP is the HTTP protocol being used, example.com is the hostname, `80` is the TCP port being used (TCP port 80 is used for HTTP), 0 is the local port, and 0 is for direct push mode.

After a short delay, the terminal program should respond with:

```
OK
```

3.4 Send Data via HTTP

To send data through the socket, type the following command, followed by the enter key:

```
AT+QISEND=0
```

The terminal should respond with:

```
>
```

Now, type or paste in the correct POST command, but do not hit the enter key. For information on the POST command or syntax, please refer to Section 2.5.

Once the appropriate command has been entered, press the sequence **CTRL+M, CTRL+J, CTRL+M, CTRL+J** to enter two new line characters at the end of the POST command. Finally, press **CTRL+Z** to indicate that data transmission is complete, as per the AT commands manual.

After a short while, the terminal should respond with something similar to:

```
SEND OK
```

```
+QIURC: "recv",0,358
```

The `+QIURC` output is the socket activity notification, and signifies that there is incoming data to be read. The value of `0` corresponds to the socket being used, and the value of `358` corresponds to the number of bytes received from the server.

To read the incoming data from the server, issue this command:

```
AT+QIRD=0,1500
```

Where `0` corresponds to the socket being used, and `1500` corresponds to the number of bytes to read. Please note that the second argument can be no larger than 1500 bytes.

The terminal will now output the data received from the server. It should look something like this:

```
HTTP/1.1 2xx OK
[text response from server]
OK
```

3.5 Shutdown Socket Connection

To suspend the data connection to the socket, enter the following characters after issuing the `AT+QISEND=0` command.

```
+++
```

The terminal should respond with:

```
OK
```

To close the socket, type the following command, followed by the enter key.

```
AT+QICLOSE=0
```

The terminal will respond with:

```
OK
```

4. Working Example 1: Dweet.io

This section will provide a working example of sending and receiving data via an HTTP transfer using a Skywire NL-SW-LTE-QBG96 modem using a 4G Verizon LTE CAT M1 SIM.

dweet.io is a lightweight messaging service specifically designed for IoT (Internet of Things) devices. In addition to being lightweight, dweet.io does not require an account to get up and running. At www.dweet.io, they have an excellent “hello world” example that this example is based on.

4.1 Preliminary Setup Procedure

Before beginning the socket dial procedure, it is crucial to verify that cellular functionality is enabled. Type the following command into the terminal:

```
AT+CFUN=1
```

This command will enable cellular functionality. Next, check for cellular connectivity using the following command:

```
AT+CEREG?
```

The terminal should respond with:

```
+CEREG x,1
```

If the terminal responds with:

```
+CEREG x,2
```

wait a few seconds, and then try the `AT+CEREG?` command again. A value of `x,2` indicates that the modem is in the process of connecting to the network.

Once network connectivity has been established, proceed to the next section.

4.2 Get IMEI of Modem

dweet.io requires a unique device name in order to send and receive data. It is recommended to use a device’s IMEI as this unique indicator. To display the IMEI, type the following command into the terminal, followed by the enter key:

```
AT+CGSN
```

The terminal will respond with something similar to:

```
353238060023699

OK
```

This number should be identical to the IMEI printed on the modem label.

4.3 Socket Setup

Type the following command into the terminal, followed by the enter key:

```
AT+QIACT=1
```

The terminal should respond with:

```
OK
```

4.4 Initiate Socket Dial

Using the syntax described in Section 2.4, type the following command into the terminal program, followed by the enter key:

```
AT+QIOPEN=1,0,"TCP","dweet.io",80,0,2
```

In this case, `1` is the socket context being used, `0` specifies socket service index, TCP is the HTTP protocol being used, dweet.io is the hostname, `80` is the TCP port being used (TCP port 80 is used for HTTP), 0 is the local port, and 2 is for transparent access mode.

The expected output is:

```
CONNECT
```

4.5 Send Data via HTTP

Using the syntax described in Section 2.5, enter the following command into the terminal program:

```
POST /dweet/for/353238060023699?hello=world HTTP/1.1
```

Note: `353238060023699` is the IMEI of the modem used to create this example. This IMEI should be replaced with the IMEI of the modem being used.

Press the sequence **CTRL+M, CTRL+J, CTRL+M, CTRL+J**, and after a short delay the terminal program should respond with something similar to:

```
HTTP/1.1 200 OK

Access-Control-Allow-Origin: *

Content-Type: application/json

Content-Length: 203

Date: Tue, 15 Mar 2016 21:50:59 GMT

Connection: keep-alive


{"this":"succeeded","by":"dweeting","the":"dweet","with":{"thing":"353
238060023699","created":"2016-03-15T21:50:59.822Z","content":{"hello":
"world"},"transaction":"debcf53b-1494-4a37-852e-c4d5278f79e6"}}
```

If the preceding text is generated, then the POST command has succeeded. To see the actual results on dweet.io, follow the link below:

https://dweet.io/get/latest/dweet/for/353238060023699

Be sure to replace the IMEI at the end of the link with the IMEI of the modem being used to follow this example. For this particular example, the following text was taken from the dweet.io webpage:

```
{this: "succeeded",by: "getting",the: "dweets",with: [{thing:
"353238060023699",created: "2016-03-15T21:51:25.173Z",content: {hello:
"world"}}]}
```

4.6 Receive Data via HTTP

Repeat Section 4.4, and enter the following text into the terminal after receiving the expected `CONNECT` output:

```
GET /get/latest/dweet/for/353238060023699 HTTP/1.1
```

Note: `353238060023699` is the IMEI of the modem used to create this example. This IMEI should be replaced with the IMEI of the modem being used.

Press the sequence **CTRL+M, CTRL+J, CTRL+M, CTRL+J**, and after a short delay the terminal program should respond with something similar to:

```
HTTP/1.1 200 OK

Access-Control-Allow-Origin: *

Content-Type: application/json

Content-Length: 152

Date: Tue, 15 Mar 2016 21:58:13 GMT

Connection: keep-alive


{"this":"succeeded","by":"getting","the":"dweet","with":[{"thing":"353
238060023699","created":"2016-03-15T21:51:25.173Z","content":{"hello":
"world"}}]}
```

5. Working Example 2: Exosite

5.1 Overview

This section will provide a working example of sending and receiving data via an HTTP transfer using a Skywire NL-SW-LTE-QBG96 modem using a 4G Verizon LTE CAT M1 SIM.

Exosite is a management service that manages sensor and device data. Exosite allows you to set up a web dashboard that you can send data to via HTTP for M2M applications.

This example assumes that you have registered your device with Exosite. If you have not, please consult the Exosite documentation located at www.exosite.com/support and docs.exosite.com.

To post to Exosite, we have created a Dashboard specifically for NimbeLink customers to walk through the complete process of sending and receiving information from Exosite. The NimbeLink Exosite Example Dashboard is located at the following URL:

https://nimbelink.exosite.com/views/2334257135/2984653781

and contains three values that we can send and read: Key1, Key2, and Key3. Key1 is of type “String”, Key2 is of type “integer”, and Key3 is of type “float” (decimal). When we send information in Section 4.4, we must send information of the correct type.

In this example, we will be changing those three values to something we specify, in order to verify that communication is working as expected.

5.2 Get CIK of Modem or Device from Exosite

Exosite requires a unique device name in order to send and receive data. This unique name—called a CIK—is created by Exosite when you register your device with them. For NimbeLink’s example page, we will be using the following CIK:

```
7cafee22ab8628b2838187a7774f5e4b3f05f877
```

Note: This CIK is unique to this example. When you register your device, you will have a different CIK.

5.3 Initiate Socket Dial

Using the syntax described in Section 2.3, type the following command into the terminal program:

```
AT+QIOPEN=1,0,"TCP","m2.exosite.com",80,0,2
```

followed by the enter key, and the terminal program should respond with

```
CONNECT
```

In this case, `1` is the socket context being used, `0` specifies socket service index, TCP is the HTTP protocol being used, dweet.io is the hostname, `80` is the TCP port being used (TCP port 80 is used for HTTP), 0 is the local port, and 2 is for transparent access mode.

5.4 Send Data via HTTP

Note: The following command must be entered correctly and in a specific order. As such, it is recommended to copy and paste the commands instead of typing them.

Note: Information entered into the terminal program at this point will not be visible on the screen.

Below is the entire command sequence we will be sending on the modem.

```
POST /onep:v1/stack/alias HTTP/1.1
Host: m2.exosite.com
X-Exosite-CIK: 7cafee22ab8628b2838187a7774f5e4b3f05f877
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Accept: application/xhtml+xml
Content-Length: 28

Key1=asdf&Key2=100&Key3=11.1
```

Below is the exact process needed to successfully send the data.

Enter the following command into the terminal program:

```
POST /onep:v1/stack/alias HTTP/1.1
```

followed by **CTRL+M and CTRL+J** on the keyboard.

Enter the following command into the terminal program:

```
Host: m2.exosite.com
```

followed by **CTRL+M and CTRL+J** on the keyboard.

Enter the following command into the terminal program:

```
X-Exosite-CIK: 7cafee22ab8628b2838187a7774f5e4b3f05f877
```

followed by **CTRL+M and CTRL+J** on the keyboard. Note that the example CIK is present in this line. This is where you would put your unique CIK in.

Enter the following command into the terminal program:

```
Content-Type: application/x-www-form-urlencoded; charset=utf-8
```

followed by **CTRL+M and CTRL+J** on the keyboard.

Enter the following command into the keyboard:

```
Accept: application/xhtml+xml
```

followed by **CTRL+M and CTRL+J** on the keyboard.

Enter the following command into the terminal program:

```
Content-Length: 28
```

followed by **CTRL+M and CTRL+J twice** on the keyboard. Typing CTRL+M and CTRL+J twice will put the necessary blank line in that the HTTP server is expecting.

Note: The value after **Content-Length:** must be equal to the number of characters that you are sending on with the following command. In this case, we have 28 characters.

Finally, enter the following command into the terminal program:

```
Key1=asdf&Key2=100&Key3=11.1
```

where `asdf` is a string of characters, `100` is an integer number, and `11.1` is a decimal number. There is no need to type anything additional to get the data sent. You may want to change these values while keeping the same type to see your specific changes on the Exosite page.

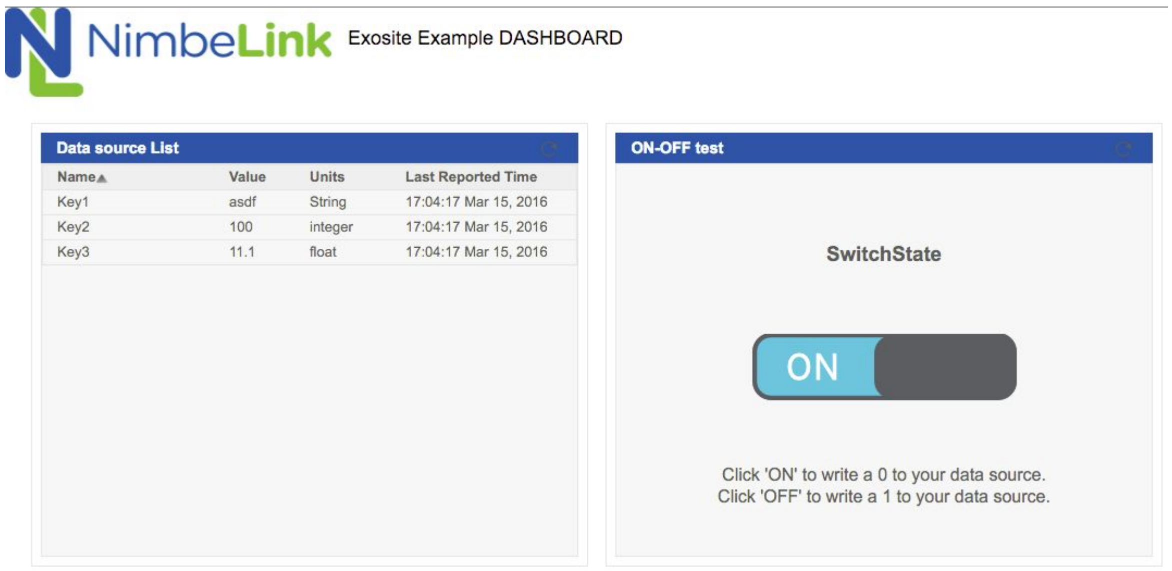
The terminal program should respond with the following:

```
HTTP/1.1 204 No Content

Date: [today's date and time]

Content-Length: 0

Server: nginx
```

If you visit the NimbeLink Exosite Example Dashboard at the following URL:

https://nimbelink.exosite.com/views/2334257135/2984653781

you will see that your values have been updated:


Exosite Example DASHBOARD

Data source List

Name	Value	Units	Last Reported Time
Key1	asdf	String	17:04:17 Mar 15, 2016
Key2	100	integer	17:04:17 Mar 15, 2016
Key3	11.1	float	17:04:17 Mar 15, 2016

ON-OFF test

SwitchState

Click 'ON' to write a 0 to your data source.
Click 'OFF' to write a 1 to your data source.

5.5 Receive Data via HTTP

Note: The following command must be entered correctly and in a specific order. As such, it is recommended to copy and paste the commands instead of typing them.

Note: Information entered into the terminal program at this point will not be visible on the screen.

Below is the entire command sequence we will be sending on the modem.

```
GET /onep:v1/stack/alias?Key1&Key2&Key3 HTTP/1.1

Accept: application/x-www-form-urlencoded; charset=utf-8

Accept-Encoding: gzip, deflate
```

```
        Host: m2.exosite.com
        Connection: Close
        X-Exosite-CIK: 7cafee22ab8628b2838187a7774f5e4b3f05f877
```

Below is the exact process needed to successfully receive data from the Exosite Dashboard.

Enter the following command into the terminal program:

```
GET /onep:v1/stack/alias?Key1&Key2&Key3 HTTP/1.1
```

followed by **CTRL+M and CTRL+J** on the keyboard.

Enter the following command into the terminal program:

```
Accept: application/x-www-form-urlencoded; charset=utf-8
```

followed by **CTRL+M and CTRL+J** on the keyboard.

Enter the following command into the terminal program:

```
Accept-Encoding: gzip, deflate
```

followed by **CTRL+M and CTRL+J** on the keyboard.

Enter the following command into the terminal program:

```
Host: m2.exosite.com
```

followed by **CTRL+M and CTRL+J** on the keyboard.

Enter the following command into the keyboard:

```
Connection: Close
```

followed by **CTRL+M and CTRL+J** on the keyboard.

Finally, enter the following command into the terminal program:

```
X-Exosite-CIK: 7cafee22ab8628b2838187a7774f5e4b3f05f877
```

followed by **CTRL+M and CTRL+J three times** on the keyboard. Typing CTRL+M and CTRL+J three times will put the necessary blank lines in that the HTTP server is expecting. Note that the example CIK is present in this line. This is where you would put your unique CIK in.

The terminal program should respond with the following:

```
HTTP/1.1 200 OK
Date: Tue, 15 Mar 2016 22:08:05 GMT
Content-Length: 28
Connection: close
Server: nginx

Key3=11.1&Key2=100&Key1=asdf
```

6. Working Example 3: Dweet.io Using Buffer Access Mode

This section will provide a working example of sending and receiving data via an HTTP transfer using a Skywire NL-SW-LTE-QBG96 modem and a 4G Verizon LTE CAT M1 SIM.

This example is nearly identical to Section 4, except that it uses buffer access mode instead of the transparent access mode.

6.1 Preliminary Setup Procedure

Before beginning the socket dial procedure, it is crucial to verify that cellular functionality is enabled. Type the following command into the terminal:

```
AT+CFUN=1
```

This command will enable cellular functionality. Next, check for cellular connectivity using the following command:

```
AT+CEREG?
```

The terminal should respond with:

```
+CEREG x,1
```

If the terminal responds with:

```
+CEREG x,2
```

wait a few seconds, and then try the `AT+CEREG?` command again. A value of `x,2` indicates that the modem is in the process of connecting to the network.

Once network connectivity has been established, proceed to the next section.

6.2 Get IMEI of Modem

dweet.io requires a unique device name in order to send and receive data. It is recommended to use a device's IMEI as this unique indicator. To display the IMEI, type the following command into the terminal, followed by the enter key:

```
AT+CGSN
```

The terminal will respond with something similar to:

```
357353080053541

OK
```

This number should be identical to the IMEI printed on the top of the modem label.

6.3 Socket Setup

Next, configure the socket connection by issuing the following command, followed by the enter key:

```
AT+QIACT=1
```

Where `1` is the socket context ID being used. Please refer to Section 3.3 for further information.

The terminal should respond with:

```
OK
```

6.4 Initiate Socket Dial

Using the syntax described in Section 3.4, type the following command into the terminal program, followed by the enter key:

```
AT+QIOPEN=1,0,"TCP","dweet.io",80,0,0
```

Where `1` is the socket context being used, `0` specifies socket service index, TCP is the HTTP protocol being used, dweet.io is the hostname, `80` is the TCP port being used (TCP port 80 is used for HTTP), 0 is the local port, and 0 is for buffer access mode.

The terminal should respond with:

```
OK
```

6.5 Send Data via HTTP

To send data through the socket, type the following command, followed by the enter key:

```
AT+QISEND=0
```

The terminal should respond with:

```
>
```

Next, type or paste the following text into the terminal prompt, but do not press enter:

```
POST /dweet/for/357353080053541?hello=world HTTP/1.1
```

Note: `357353080053541` is the IMEI of the modem used to create this example. This IMEI should be replaced with the IMEI of the modem being used.

Press the sequence **CTRL+M, CTRL+J, CTRL+M, CTRL+J** to add two new line characters. Finally, press **CTRL+Z** to signal the end of data transmission. Please refer to Section 3.5 for information regarding POST commands and related escape sequences.

The terminal should respond with something similar to this:

```
SEND OK
+QIURC:"recv",0,358
```

Where `+QIURC:` is the socket activity notification, `0` is the socket being used, and `358` is the number of bytes received in the HTTP response. If desired, read the information sent by the server through the socket by issuing the following command:

```
AT+QIRD=0,1500
```

This command tells the modem to read the first 1500 bytes of incoming data on socket number 0. The output to the terminal will be something similar to this:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
Content-Length: 203
Date: Mon, 07 May 2018 21:44:47 GMT
Connection: keep-alive


{"this":"succeeded","by":"dweeting","the":"dweet","with":{"thing":"357
353080053541","created":"2018-05-07T21:44:47.635Z","content":{"hello":
"world"},"transaction":"edf62c02-181d-4724-aebf-d2bd30b3b9a5"}}
```

If the preceding text is generated, then the POST command has succeeded. To see the actual results on dweet.io, follow the link below:

https://dweet.io/get/latest/dweet/for/357353080053541

Be sure to replace the IMEI at the end of the link with the IMEI of the modem being used to follow this example. For this particular example, the following text was taken from the dweet.io webpage:

```
{"this":"succeeded","by":"getting","the":"dweets","with":[{"thing":"35735308005
3541","created":"2018-05-07T21:44:47.635Z","content":{"hello":"world"}}]}
```

6.6 Receive Data via HTTP

To send data through the socket, type the following command, followed by the enter key:

```
AT+QISEND=0
```

The terminal should respond with:

```
>
```

Next, type or paste the following text into the terminal prompt, but do not press enter:

```
GET /get/latest/dweet/for/357353080053541 HTTP/1.1
```

Note: `357353080053541` is the IMEI of the modem used to create this example. This IMEI should be replaced with the IMEI of the modem being used.

Press the sequence **CTRL+M, CTRL+J, CTRL+M, CTRL+J, CTRL+Z**. The terminal should respond with something similar to this:

```
OK

+QIURC: "recv",0,358
```

To read the data sent by the server through the socket, issue the following command:

```
AT+QIRD=0,317
```

Make sure to specify enough bytes to fully read the received text. In this case, the second argument to the following command must be greater than or equal to 317 in order to completely read all of the received data. In this example, the terminal responded with:

```
+QIRD: 0,317

HTTP/1.1 200 OK

Access-Control-Allow-Origin: *

Content-Type: application/json

Content-Length: 152

Date: Mon, 07 May 2018 22:43:48 GMT

Connection: keep-alive


{"this":"succeeded","by":"getting","the":"dweets","with":[{"thing":"35
7353080053541","created":"2018-05-07T22:32:38.306Z","content":{"hello"
:"world"}}]}
```

7. Troubleshooting

7.1 Serial Clients

If the POST sequences are being inputted manually into a serial program such as PuTTY or TeraTerm, some serial clients may interpret the "Carriage Return" (CTRL+M) and "Line Feed" (CTRL+J) characters differently. For instance, if issues are encountered when sending information using an HTTP POST, try sending four "Line Feed" characters instead of the sequence of "Carriage Return" and "Line Feed".

For instance, instead of sending:

CTRL+M, CTRL+J, CTRL+M, CTRL+J

Send:

CTRL+J, CTRL+J, CTRL+J, CTRL+J