Skywire® LTE Cat 1 AT&T Socket Dial Example

NimbeLink Corp

Updated: Jan 2017

Table of Content

1. Introduction

1.1. Prerequisites

This document assumes you have completed the initial setup of your modem and development kit. If you have not completed those steps, refer to the Skywire® Development Kit User Manual and complete the modem setup before proceeding. See the Development Kit User Manual for details on how to accomplish these steps.

2. Socket Dial

2.1. Overview

Socket dialing is useful for uploading or downloading information to or from a website or database via HTTP commands. Below is an example for connecting to a server, uploading data, downloading data, and then disconnecting from the server.

2.2 Setup APN

Type the following command into the terminal program:

AT%PDNSET=1,"[your apn]","IP"

where **[your apn]** is the APN for your carrier, followed by the Enter key.

The terminal should respond with:

OK

For example, if you are using an AT&T SIM and your APN is:

broadband

you would type:

AT%PDNSET=1,"broadband","IP"

followed by the Enter key.

2.2. Start Internet Service

Type the following command into the terminal program:

AT@INTERNET=1

followed by the Enter key, and the terminal should respond with:

OK

2.3. Activate Packet Switch Data Service

Type the following command into the terminal program:

AT@SOCKDIAL=1

followed by the Enter key, and the terminal should respond with:

OK

2.4. Get IP Address of Host

In the terminal program, type the following command:

AT@DNSRESVDON="example.com"

where **example.com** is the host name you'd like to connect to. After a short delay the terminal program should respond with:

@DNSRESVDON:"<ip_address>"

where <ip_address> is the host's IP address. This is the IP address we will use when making a socket connection to the host.

2.5. Create the Socket

In the terminal program, type the following command:

AT@SOCKCREAT=1

The modem should respond with:

@SOCKCREAT:1

OK

indicating that a socket with identifier 1 has been created.

2.6. Open Socket Connection to Host

Type the following command into the terminal program:

AT@SOCKCONN=1,"<ip_address>",80

where <ip_address> is the host's IP address found in Section 2.4.

Note: The TCP port (third parameter) that is typically used is port 80, which is the TCP Port for HTTP. Depending on your application, you may use a different port.

2.7. Send Data via HTTP

In order to send data to the website, you can use the HTTP POST command. The syntax of the POST command is as follows:

POST /test/demo_form.asp HTTP/1.1

followed by CR LF CR LF (CR=carriage return (ASCII 0x0D), LF=line feed (ASCII 0x0A)). **POST** is the HTTP command being issued, **/text/demo_form.asp** is the endpoint on the server, and **HTTP/1.1** is the HTTP version you will be using.

To send an HTTP POST command, type the following in the terminal:

AT@SOCKWRITE=1,<number_of_bytes>,"<hex_string>"

where <number_of_bytes> is the length of the entire POST command (including any headers and data fields) in bytes, and <hex_string> is the POST command encoded as a hexadecimal string. For example, to send the command **POST /test/demo_form.asp HTTP/1.1**, the AT command would be:

AT@SOCKWRITE=1,37,"504f5354202f746573742f64656d6f5f666f726d2e6173 37020485454502f312e310d0a0d0a"

If the socket write is successful, the modem should respond with

OK

Note: the length of the hexadecimal string will be twice as long as the <number_of_bytes> parameter, since two hexadecimal characters are used to represent a 1-byte ASCII character. Online tools such as this <u>ASCII</u> <u>to</u> <u>hex</u> <u>converter</u> can be used to convert your POST command to a hexadecimal string.

If the server sends a response, the modem will send the following unsolicited message to the terminal:

@SOCKWRITE:XX

where **XX** is the number of bytes written. This should be the same number as **<number_of_bytes>** from above. This response can be read by issuing the following command:

AT@SOCKREAD=1,1024

The modem will respond with:

AT@SOCKREAD:<number_of_bytes>,"<hex_string>"

where <number_of_bytes> is the length of the response in bytes, and <hex_string> is the response from the server encoded as a hexadecimal string.

Once decoded as ASCII characters, the server response will look similar to the following:

HTTP/1.1 2xx OK

[text response from server]

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. If you receive a different value, please consult the HTTP/1.1 Protocol Documentation at the following URL:

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

2.8. Receive Data via HTTP

In order to receive data from a website, you must use the HTTP GET command. The syntax of the GET command is similar to the POST command, and is as follows:

GET /test/demo_form.asp HTTP/1.1

where **GET** is the HTTP command being issued, **/text/demo_form.asp** is the location of the data, and **HTTP/1.1** is the HTTP version you will be using.

To send an HTTP GET command, type the following in the terminal:

AT@SOCKWRITE=1,<number_of_bytes>,"<hex_string>"

where <number_of_bytes> is the length of the entire POST command (including any headers and data fields) in bytes, and <hex_string> is the POST command encoded as a hexadecimal string. For example, to send the command **GET /test/demo_form.asp HTTP/1.1**, the AT command would be:

AT@SOCKWRITE=1,36,"474554202f746573742f64656d6f5f666f726d2e61737 020485454502f312e310d0a0d0a"

If the socket write is successful, the modem should respond with

OK

After the server response is received, the modem will send the following unsolicited message to the terminal:

@SOCKDATAIND:1,1

which indicates that there is data available from the host server. This response can be read by issuing the following command:

> *AT@SOCKREAD=1,1024*

The modem will respond with:

> *AT@SOCKREAD:<number_of_bytes>,"<hex_string>"*

where <number_of_bytes> is the length of the response in bytes, and <hex_string> is the response from the server encoded as a hexadecimal string.

Once decoded as ASCII characters, the server response will look similar to the following:

> *HTTP/1.1 xxx OK*
>
> *[text response from server]*

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. If you receive a different value, please consult the HTTP/1.1 Protocol Documentation at the following URL:

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

2.9. Shutdown the Socket Connection

To close the socket, type the following command into the terminal program:

> *AT@SOCKCLOSE=1*

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

> *OK*

The data connection and socket connection has now been terminated.

3. Working Example 1: dweet.io

3.1. Overview

This section will provide a working example of sending and receiving data via an HTTP transfer using a Skywire NL-SW-LTE-WM14 modem using a 4G T-Mobile 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 we will be using.

3.2. Get IMEI of Modem

dweet.io requires a unique device name in order to send and receive data. For this example, we will be using the IMEI of our modem, which is unique to our modem. To display the IMEI, type the following command into the terminal program:

AT+GSN

followed by the Enter key, and the terminal will respond with something similar to:

xxxxxxxxxxxxxxx

OK

where "xxxxxxxxxxxxxxx" should be the IMEI number printed on the top of the modem label.

3.3. Enable Internet and Socket Dial Services

Send the following command to the modem to turn on the Internet service:

AT@INTERNET=1

The modem should respond with:

OK

Send the following command to turn on the socket dial service:

AT@SOCKDIAL=1

The modem should respond with:

OK

3.4. Get the IP Address for dweet.io

Using the syntax described in Section 2.4, type the following command into the terminal program:

AT@DNSRESVDON="dweet.io"

followed by the Enter key. The modem should respond with:

@DNSRESVDON:"52.7.140.191"

@DNSRESVDON:"52.71.224.50"

OK

We will use the first IP address, "52.7.140.191", to connect to dweet.io.

3.5. Create the Socket

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

AT@SOCKCREAT=1

followed by the Enter key. The modem should respond with:

@SOCKCREAT:1

OK

indicating that the socket was successfully created.

3.6. Initiate Socket Connection

Using the syntax described in Section 2.6, type the following command into the terminal program:

AT@SOCKCONN=1,"52.7.140.191",80

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

OK

In this case**, 1** is the socket we are using on the Skywire modem and **80** is the TCP port we are using (TCP port 80 is used for HTTP).

3.7. Send Data via HTTP

To send data, we will be issuing an HTTP POST command with the following syntax:

POST /dweet/for/xxxxxxxxxxxxxxx?hello=world HTTP/1.1

In this example, we will be using RapidTables ASCII to hex converter to convert our POST command to a hexadecimal string. As a hexadecimal string, this command can be represented as:

**504f5354202f64776565742f666f722f78787878787878787878787878787
83f68656c6c6f3d776f726c6420485454502f312e310d0a0d0a**

Note: when using the RapidTables ASCII to hex converter, newlines are represented as a single LF character (0x0A), but the HTTP 1.1 specification specifies using both a CR and LF character (0x0D and 0x0A) for each line ending. You will have to manually add in the CR characters.

Using the syntax described in Section 2.7, enter the following command into the terminal program:

AT@SOCKWRITE=1,56,"504f5354202f64776565742f666f722f78787878
78787878787878787878783f68656c6c6f3d776f726c6420485454502f31
2e310d0a0d0a"

The modem will respond with:

@SOCKWRITE:56

OK

<u>Note</u>: in the command above, an IMEI of "xxxxxxxxxxxxxxx" was used when generating the hexadecimal string. You should use your device's IMEI when sending data to dweet.io in order to avoid device name conflicts.

To get the response from dweet.io, send the following command:

AT@SOCKREAD=1,1024

The modem should respond with something similar to the following:

@SOCKREAD:368,"485454502F312E3120323030204F4B0D0A4163636
573732D436F6E74726F6C2D416C6C6F772D4F726967696E3A202A0D
0A436F6E74656E742D547970653A206170706C69636174696F6E2F6A
736F6E0D0A436F6E74656E742D4C656E6774683A203230330D0A4461
74653A205765642C203037205365702032303136203230A35313A3430
20474D540D0A436F6E6E656374696F6E3A206B6565702D616C697665
0D0A0D0A7B2274686973223A2273756363656564656422C226279223
A226477656574696E67222C22746865223A2264776565742C227769
7468223A7B227468696E67223A22787878787878787878787878787878
222C2263726561746564223A22323031362D30392D30375432303A353
13A34302E3232355A222C22636F6E74656E74223A7B2268656C6C6F2
23A22776F726C64227D2C227472616E73616374696F6E223A22663664
66643262372D313136382D346634322D616637352D3065323737396464
39313139227D7D"

The decoded hexadecimal string should be similar to the following:

HTTP/1.1 200 OK

*Access-Control-Allow-Origin: **

Content-Type: application/json

Content-Length: 150

Date: [today's date]

Connection: keep-alive

{"this":"succeeded","by":"dweeting","the":"dweet","with":{"thing":"
xxxxxxxxxxxxxxx","created":"[today's
date]","content":{"hello":"world"}}}

where **xxxxxxxxxxxxxx** is your IMEI. By going to the website https://dweet.io/get/latest/dweet/for/xxxxxxxxxxxxxx, replacing the *x*'s at the end with your IMEI, you will see the following text:

```
{"this":"succeeded","by":"getting","the":"dweets","with":[{"thing":"xxxxxxxxxx
xxxx","created":"[today's date]","content":{"hello":"world"}}]}
```

This text is the same as the data section of the response received by te modem, indicating a successful HTTP transaction.

3.8. Receive Data via HTTP

To receive data for our device, we will be issuing an HTTP GET command with the following syntax:

GET /get/latest/dweet/for/xxxxxxxxxxxxxx HTTP/1.1

As a hexadecimal string, this command can be represented as:

474554202f6765742f6c61746573742f64776565742f666f722f787878787878787878787878787878787820485454502f312e310d0a0d0a

Using the syntax described in Section 2.8, enter the following command into the terminal program:

AT@SOCKWRITE=1,54,"474554202f6765742f6c61746573742f647765655742f666f722f7878787878787878787878787878787820485454502f312e310d0a0d0a"

The modem will respond with:

@SOCKWRITE:56

OK

To get the response from dweet.io, send the following command:

AT@SOCKREAD=1,1024

The modem should respond with something similar to the following:

@SOCKREAD:317,"485454502F312E3120323030204F4B0D0A4163636573732D436F6E74726F6C2D416C6C6F772D4F726967696E3A202A0D0A436F6E74656E742D547970653A206170706C69636174696F6E2F6A736F6E0D0A436F6E74656E742D4C656E6774683A203135320D0A4461746553A205765642C203037205365702032303136203231313A31333A3437204742D540D0A436F6E6E656374696F6E3A206B6565702D616C6976650D0A0D0A7B2274686973223A22737563636565646564222C226279223A2267657474696E67222C22746865223A2264776565747322C22776974682A5B7B227468696E67223A2278787878787878787878787878787822C22637265617465642223A22323031362D30392D30375432303A

35313A34302E3232355A222C22636F6E74656E74223A7B2268656C6C6F223A22776F726C64227D7D5D7D"

The decoded hexadecimal string should be similar to the following:

HTTP/1.1 200 OK

*Access-Control-Allow-Origin: ***

Content-Type: application/json

Content-Length: 152

Date: **[today's date]**

Connection: keep-alive

{"this":"succeeded","by":"getting","the":"dweets","with":[{"thing":" **xxxxxxxxxxxxxxx** *","created":"[today's date]","content":{"hello":"world"}}]}*

where **xxxxxxxxxxxxxxx** is your IMEI.

This indicates that we successfully received the HTTP information.

4. Working Example 2: Exosite

4.1. Overview

This section will provide a working example of sending and receiving data via an HTTP transfer using a Skywire NL-SW-LTE-WM14 modem using a 4G T-Mobile LTE 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 4.7, 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.

4.2. 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.

4.3. Enable Internet and Socket Dial Services

Send the following command to the modem to turn on the Internet service:

AT@INTERNET=1

The modem should respond with:

OK

Send the following command to turn on the socket dial service:

AT@SOCKDIAL=1

The modem should respond with:

OK

4.4. Get the IP Address for Exosite

Using the syntax described in Section 2.4, type the following command into the terminal program:

AT@DNSRESVDON="m2.exosite.com"

followed by the Enter key. The modem should respond with:

@DNSRESVDON:"52.8.0.240"

OK

We will use the IP address "52.8.0.240" to connect to Exosite.

4.5. Create the Socket

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

AT@SOCKCREAT=1

followed by the Enter key. The modem should respond with:

@SOCKCREAT:1

OK

indicating that the socket was successfully created.

4.6. Initiate Socket Connection

Using the syntax described in Section 2.6, type the following command into the terminal program:

AT@SOCKCONN=1,"52.8.0.240",80

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

OK

In this case**, 1** is the socket we are using on the Skywire modem and **80** is the TCP port we are using (TCP port 80 is used for HTTP).

4.7. Send Data via HTTP

Below is the entire POST command we will be sending using the Skywire™ 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
```

Note that the example CIK is present in this command. This is where you would put in your unique CIK.

As a hexadecimal string, this is:

504f5354202f6f6e65703a76312f737461636b2f616c69617320485454502f312e3 10d0a486f73743a206d322e65786f736974652e636f6d0d0a582d45786f736974

652d43494b3a20376361666665533232616238363238623238333831383761373737
3734663565346233663035663837370d0a436f6e74656e742d547970653a206120
70706c69636174696f6e2f782d7777772d666f726d2d75726c656e636f64656443
b20636861727365743d7574662d380d0a4163636570743a206170706c69636361617
4696f6e2f7868746d6c2b786d6c0d0a436f6e74656e742d4c656e6774683a2032
380d0a0d0a4b6579313d61736466264b6579323d313030264b6579333d31312
e31

Note: when using the RapidTables ASCII to hex converter, newlines are represented as a single LF character (0x0A), but the HTTP 1.1 specification specifies using both a CR and LF character (0x0D and 0x0A) for each line ending. You will have to manually add in the CR characters.

To send this POST command and data to Exosite, send the following command to the modem:

*AT@SOCKWRITE=1,260,"504f5354202f6f6e65703a76312f737461636b
2f616c69617320485454502f312e310d0a486f73743a206d322e65786f73
6974652e636f6d0d0a582d45786f736974652d43494b3a2037636166656
5323261623836323862323838333831383736137373734663565346233663630
35663837370d0a436f6e74656e742d547970653a206170706c696361746
96f6e2f782d7777772d666f726d2d75726c656e636f64656443b206368661
27365743d7574662d380d0a4163636570743a206170706c6963617469696f
6e2f7868746d6c2b786d6c0d0a436f6e74656e742d4c656e6774683a203
2380d0a0d0a4b6579313d61736466264b6579323d313030264b6579333
d31312e31"*

The modem should respond with:

@SOCKWRITE:230

OK

To read the response from Exosite, send the following command:

AT@SOCKREAD=1,1024

The modem should respond with something similar to the following:

*@SOCKREAD:98,"485454502F312E3120323034204E6F20436F6E7465
6E740D0A446174653A205468752C2030382053657702032230313620313
43A32323A343620474D540D0A436F6E74656E742D4C656E6774683A2
0300D0A5365727665723A206E67696E780D0A0D0A"*

OK

Decoding this hexadecimal string, we get the following response from Exosite:

```
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:

4.8. Receive Data via HTTP

Below is the entire GET command we will be sending using the Skywire™ modem (including the 3 new lines at the end):

```
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
```

Note that the example CIK is present in this command. This is where you would put in your unique CIK.

As a hexadecimal string, this command is:

474554202f6f6e65703a76312f737461636b2f616c6961733f4b657931264b6579 32264b65793320485454502f312e310d0a4163636570743a206170706c696361 74696f6e2f782d7777772d666f726d2d75726c656e636f6465643b20636861727 365743d7574662d380d0a4163636570742d456e636f64696e673a20677a69702 c206465666c6174650d0a486f73743a206d322e65786f736974652e636f6d0d0a 436f6e6e656374696f6e3a20436c6f73650d0a582d45786f736974652d43494b3 a20376361666565323261623836323862323833383138376137373734663565 34623366303566383737370d0a0d0a0d0a

Note: when using the RapidTables [ASCII](#) [to](#) [hex](#) [converter](#), newlines are represented as a single LF character (0x0A), but the HTTP 1.1 specification specifies using both a CR and LF character (0x0D and 0x0A) for each line ending. You will have to manually add in the CR characters.

To send this POST command and data to Exosite, send the following command to the modem:

> *AT@SOCKWRITE=1,242,"474554202f6f6e65703a76312f737461636b2f 616c6961733f4b657931264b657932264b65793320485454502f312e310 d0a4163636570743a206170706c69636174696f6e2f782d7777772d666f7 26d2d75726c656e636f6465643b20636861727365743d7574662d380d0a 4163636570742d456e636f64696e673a20677a69702c206465666c61746 50d0a486f73743a206d322e65786f736974652e636f6d0d0a436f6e6e656 374696f6e3a20436c6f73650d0a582d45786f736974652d43494b3a20376 3616665653232616238363238623238333831383761373737346635653 4623366303566383737370d0a0d0a0d0a"*

The modem should respond with:

> *@SOCKWRITE:242*

OK

To read the response from Exosite, send the following command:

AT@SOCKREAD=1,1024

The modem should respond with something similar to the following:

@SOCKREAD:138,"485454502F312E3120323030204F4B0D0A4461746 53A205468752C203038205365702032323031362031343A33373A3139204 74D540D0A436F6E74656E742D4C656E6774683A2032380D0A436F6E 6E656374696F6E3A20636C6F73650D0A5365727665723A206E67696E 780D0A0D0A4B6579333D31312E31264B6579323D313030264B657931 3D61736466"

OK

Decoding this hexadecimal string, we get the following response from Exosite:

```
HTTP/1.1 200 OK
Date: [today's date and time]
Content-Length: 28
Connection: close
Server: nginx

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

The values for each of the keys are the same values we sent to Exosite in Section 4.7.