SIM7070_SIM7080_SIM7090 Series_TCPUDP(S) _Application Note

LPWA Module

SIMCom Wireless Solutions Limited
Building B, SIM Technology Building, No.633, Jinzhong Road
Changning District, Shanghai P.R. China
Tel: 86-21-31575100
support@simcom.com
www.simcom.com

Document Title:	SIM7070_SIM7080_SIM7090 Series_TCPUDP(S)_Application Note
Version:	1.02
Date:	2020.07.08
Status:	Released

GENERAL NOTES

SIMCOM OFFERS THIS INFORMATION AS A SERVICE TO ITS CUSTOMERS, TO SUPPORT APPLICATION AND ENGINEERING EFFORTS THAT USE THE PRODUCTS DESIGNED BY SIMCOM. THE INFORMATION PROVIDED IS BASED UPON REQUIREMENTS SPECIFICALLY PROVIDED TO SIMCOM BY THE CUSTOMERS. SIMCOM HAS NOT UNDERTAKEN ANY INDEPENDENT SEARCH FOR ADDITIONAL RELEVANT INFORMATION, INCLUDING ANY INFORMATION THAT MAY BE IN THE CUSTOMER'S POSSESSION. FURTHERMORE, SYSTEM VALIDATION OF THIS PRODUCT DESIGNED BY SIMCOM WITHIN A LARGER ELECTRONIC SYSTEM REMAINS THE RESPONSIBILITY OF THE CUSTOMER OR THE CUSTOMER'S SYSTEM INTEGRATOR. ALL SPECIFICATIONS SUPPLIED HEREIN ARE SUBJECT TO CHANGE.

COPYRIGHT

SIMCom Wireless Solutions Limited
Building B, SIM Technology Building, No.633 Jinzhong Road, Changning District, Shanghai P.R. China
Tel: +86 21 31575100
Email: simcom@simcom.com

For more information, please visit:
https://www.simcom.com/download/list-863-en.html

For technical support, or to report documentation errors, please visit:
https://www.simcom.com/ask/ or email to: support@simcom.com

About Document

Version History

Version	Date	Owner	What is new
V1.00	2019.10.27	Wei.zhang	First Release
V1.01	2020.02.26	Jiangting.Ding	Change AT+SHBOD
V1.02	2020.07.08	Wei.Zhang	All

Scope

This document applies to the following products

Name	Type	Size(mm)	Comments
SIM7080G	CAT-M/NB	17.6*15.7 *2.3	N/A
SIM7070G/SIM7070E	CAT-M/NB/GPRS	24*24*2.4	N/A
SIM7070G-NG	NB/GPRS	24*24*2.4	N/A
SIM7090G	CAT-M/NB	14.8*12.8*2.0	N/A

Contents

1 Introduction

1.1 Purpose of the document

Based on module AT command manual, this document will introduce TCPUDP(S) application process.

Developers could understand and develop application quickly and efficiently based on this document.

1.2 Related documents

[1] SIM7070_SIM7080_SIM7090 Series_AT Command Manual
[2] SIM7070_SIM7080_SIM7090 Series_SSL_Application Note

1.3 Conventions and abbreviations

In this document, the GSM engines are referred to as following term:

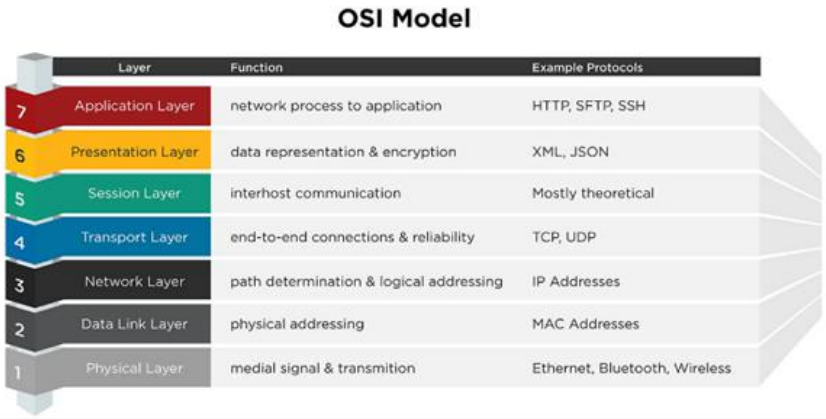
- ME (Mobile Equipment);
- MS (Mobile Station);
- TA (Terminal Adapter);
- DCE (Data Communication Equipment) or facsimile DCE (FAX modem, FAX board);

In application, controlling device controls the GSM engine by sending AT Command via its serial interface. The controlling device at the other end of the serial line is referred to as following term:

- TE (Terminal Equipment);
- DTE (Data Terminal Equipment) or plainly "the application" which is running on an embedded system;

2 TCP/UDP Introduction

In the TCP/IP network architecture, TCP (Transport Control Protocol) and UDP (User Data Protocol) are the two most important protocols at the transport layer, and provide upper-level users with a level of communication reliability.



3.1 Connection-oriented TCP

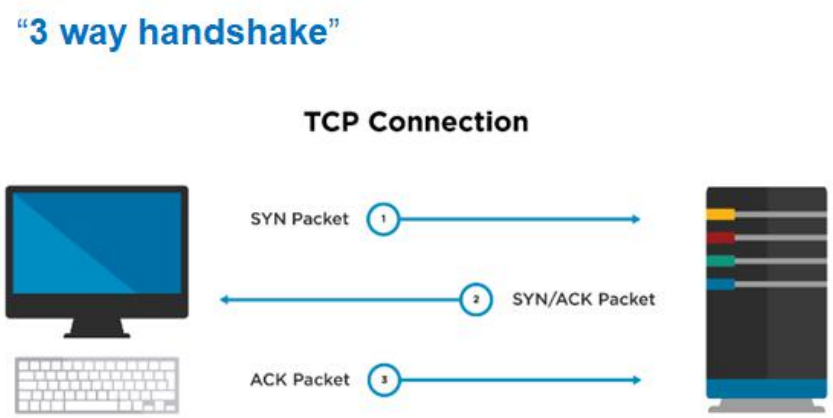
TCP (Transmission Control Protocol) is a connection-oriented protocol.
"Connection-oriented" means that a connection must be established with the other party before formal communication. It is modeled according to the telephone system. For example, if you call someone, you must wait for the line to be connected and the other party can pick up the microphone to talk to each other.

The TCP protocol is a reliable, one-to-one, connection-oriented communication protocol. TCP mainly guarantees the reliability of data transmission in the following ways:
(1) When using the TCP protocol for data transmission, it is often necessary for the client and server to establish a "channel", and this channel can only be used by the client and server, so the TCP transmission protocol can only be used for one-to-one connection.
(2) In order to ensure the accuracy of data transmission, the TCP transmission protocol divides the data packet used for transmission into several parts (the size of each part depends on the network situation at that time), and then adds a check word in their header Section. After a part of the data is received, the server will verify the integrity and accuracy of the part. After the verification, if the data is 100% complete and accurate, the server will ask the client to start Transmission of the next part of the data. If the integrity and accuracy of the data do not match the original, the server will request the client to transmit this part again.

The client and server must first establish a "channel" when using the TCP transmission protocol, and then

close the "channel" after the transmission is completed. The former can be visually referred to as a "three-way handshake", while the latter can be referred to as " Four waves."

Channel establishment-three handshake:
(1) When establishing a channel, the client first sends a SYN synchronization signal to the server.
(2) After receiving this signal, the server will send a SYN synchronization signal and an ACK confirmation signal to the client.
(3) When the server's ACK and SYN reach the client, the "channel" between the client and the server will be established.



The closing of the channel-four waves:
(1) After the data transmission is completed, the client will send a FIN termination signal to the server.
(2) After receiving this signal, the server will send an ACK confirmation signal to the client.
(3) If the server does not send data to the client after that, the server will send a FIN termination signal to the client.
(4) After receiving this signal, the client will reply with a confirmation signal. After the server receives this signal, the channel between the server and the client will be closed.

The TCP protocol can provide a reliable communication connection for the application program, so that the byte stream sent by a computer can be sent to other computers on the network without errors. Data communication systems that require high reliability often use the TCP protocol to transmit data.

3.2 Connectionless UDP protocol

UDP (User Datagram Protocol) is a connectionless protocol and a simple transport layer protocol for datagrams. It provides non-connection-oriented, unreliable data streaming.

Connectionless means that it is not necessary to establish a connection with the other party before formal communication, and it is sent directly regardless of the other party's status. It is very similar to the mobile phone text message: when you send a text message, you only need to enter the other party's mobile phone

number and it is OK.

UDP transmission protocol is an unreliable, connectionless oriented communication protocol that can realize many-to-one, one-to-many and one-to-one connections. UDP does not need to establish a channel before transmitting data, nor does it need to close the channel after data transmission is completed. As long as the client sends a request to the server, the server will send all the data at once. UDP does not verify the integrity of the data when transmitting data, and does not require retransmission when data is lost or data is wrong, so it also saves a lot of time for verifying data packets, so the delay of the connection established with UDP Will have a lower latency than connections established with TCP. UDP does not control the data transmission speed according to the current network conditions, so no matter whether the network conditions are good or bad, the server will send data at a constant rate. Although this sometimes causes data loss and damage, this is very important for some real-time applications. Based on the above three points, UDP is faster in data transmission, lower in latency, and better in real-time, so it is widely used in the communication field and video websites.

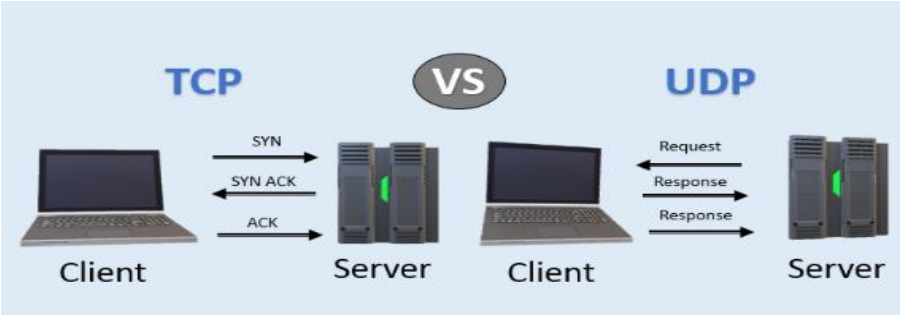
UDP is suitable for application environments that only transmit a small amount of data at a time and have low requirements on reliability. For example, we often use the "ping" command to test whether the TCP/IP communication between the two hosts is normal. In fact, the principle of the "ping" command is to send ICMP data packets to the other host, and then the other host confirms that the data packet is received. If the message of whether the packet arrives is fed back in time, then the network is connected. For example, in the default state, a "ping" operation sends 4 packets (as shown). As you can see, the number of data packets sent is 4 packets, and 4 packets are received (because the host of the other party will send back a data packet confirming receipt). This fully shows that the UDP protocol is a connection-free protocol and there is no connection establishment process. Because the UDP protocol has no connection process, its communication efficiency is high; but also because of this, its reliability is not as high as the TCP protocol. QQ uses UDP to send messages, so sometimes the message may not be received.

3.3 Differences between TCP and UDP protocols

The biggest difference between TCP/IP and UDP is that TCP/IP is connection-oriented and UDP is connectionless. The TCP and UDP protocols have their own strengths and weaknesses, and they are suitable for communication environments with different requirements. The differences between the TCP protocol and UDP protocol are shown in the table below.

Table 1 Difference between TCP and UDP

Attributes	TCP	UDP
Whether to connect	connection-oriented	no connection
Transmission reliability	Reliable	Unreliable
Application Scenarios Transfer	small amounts of data	Large amounts of data
Speed	slow	fast



In actual use, TCP is mainly used in scenarios where the accuracy of file transfer is relatively high and is not very urgent, such as e-mail and remote login. Sometimes in these application scenarios, even the loss of one or two bytes can cause irreparable errors, so these scenarios generally use the TCP transmission protocol. Because UDP can improve transmission efficiency, UDP is widely used in data transmission with large data volume and low accuracy requirements. For example, when we usually watch videos or listen to music on the website, we basically apply the UDP transmission protocol.

3 AT Commands that support TCP/UDP(S)

The module provides AT commands that can be used by device terminals as follows:

Command	Description
AT+CSSLCFG	Configure SSL parameters of a context identifier
AT+CACID	Set TCP/UDP identifier
AT+CASSLCFG	Set SSL certificate and timeout parameters
AT+CAOPEN	Open a TCP/UDP connection
AT+CASEND	Send data via an established connection
AT+CARECV	Receive data via an established connection
AT+CASEND	Send Data via an Established Connection
AT+CAACK	Query Send Data Information
AT+CASTATE	Query TCP/UDP Connection State
AT+CACLOSE	Close a TCP/UDP connection
AT+CACFG	Configure transparent transmission parameters
AT+CASWITCH	Switch to transparent transport mode

For detail information, please refer to “SIM7070_SIM7080_SIM7090 Series_AT Command Manual” .

4 Bearer Configuration

Usually module will register PS service automatically.

5.1 PDN Auto-activation

```
//Example of PDN Auto-activation.

AT+CPIN?                          //Check SIM card status
+CPIN:READY

OK
AT+CSQ                            //Check RF signal
+CSQ: 20,0

OK
AT+CGATT?                         //Check PS service. 1 indicates PS has attached.
+CGATT: 1

OK
AT+COPS?                          //Query Network information, operator and network.
+COPS: 0,0,"CHN-CT",9             //Mode 9 means NB-IOT network.

OK
AT+CGNAPN                         //Query the APN delivered by the network after the CAT-M or NB-IOT network is successfully registered.
+CGNAPN: 1,"ctnb"                 //"ctnb" is APN delivered by the CAT-M or NB-IOT network. APN is empty under the GSM network.

OK
AT+CNCFG=0,1,"ctnb"               //Before activation please use AT+CNCFG to set APN\user name\password if needed.

OK
AT+CNACT=0,1                      //Activate network, Activate 0th PDP.
OK

+APP PDP: 0,ACTIVE
```

AT+CNACT?	//Get local IP
+CNACT: 0,1,"10.94.36.44" **+CNACT: 1,0,"0.0.0.0"** **+CNACT: 2,0,"0.0.0.0"** **+CNACT: 3,0,"0.0.0.0"**	
OK	

5.2 APN Manual Configuration

If not attached automatically, could configure correct APN setting.

//Example of APN Manual configuration.	
AT+CFUN=0	//Disable RF
+CPIN: NOT READY	
OK	
AT+CGDCONT=1,"IP","ctnb"	//Set the APN manually. Some operators need to set APN first when registering the network.
OK	
AT+CFUN=1	//Enable RF
OK	
+CPIN: READY	
AT+CGATT?	//Check PS service. 1 indicates PS has attached.
+CGATT: 1	
OK	
AT+CGNAPN	//Query the APN delivered by the network after the CAT-M or NB-IOT network is successfully registered.
+CGNAPN: 1,"ctnb"	//"ctnb" is APN delivered by the CAT-M or NB-IOT network. APN is empty under the GSM network.
OK	
AT+CNCFG=0,1,"ctnb"	//Before activation please use AT+CNCFG to set APN\user name\password if needed.
OK	
AT+CNACT=0,1	//Activate network, Activate 0th PDP.
OK	

```
+APP PDP: 0,ACTIVE
AT+CNACT?                                   //Get local IP
+CNACT: 0,1,"10.94.36.44"
+CNACT: 1,0,"0.0.0.0"
+CNACT: 2,0,"0.0.0.0"
+CNACT: 3,0,"0.0.0.0"

OK
```

5 TCPUDP(S) Examples

5.1 Build a TCP/UDP connection without SSL

5.1.1 Build an ordinary TCP/UDP connection

//Example of Build an ordinary TCP/UDP connection.	
AT+CNACT?	//Check the 0th PDN/PDP local IP.
+CNACT: 0,1,"10.181.182.177"	About active PDN/PDP refer to Bearer Configuration.
+CNACT: 1,0,"0.0.0.0"	
+CNACT: 2,0,"0.0.0.0"	
+CNACT: 3,0,"0.0.0.0"	
OK	
AT+CASSLCFG=0,"SSL",0	//Set the 0th connection's SSL enable option. If TCP/UDP connection, the parameter is 0. This step can be omitted
OK	
AT+CAOPEN=0,0,"TCP","117.131.85.139",6004	//Create a TCP connection with 0th PDP on 0th connection. Return to URC the first parameter is the identifier, the second parameter is the result of the connection, and the 0 indicates success.
+CAOPEN: 0,0	
OK	
AT+CASEND=0,5	//Request to send 5 bytes of data
>	//Input data
OK	//Data sent successfully
+CASEND: 0,0,5	
CADATAIND: 0	//Data comes in on 0th connection.
AT+CARECV=0,100	//Request to get 100 byte data sent by the server.
+CARECV: 10,GET / HTTP	//Output received data
OK	
AT+CACLOSE=0	//Close the connection with an identifier of 0.

```
OK
AT+CNACT=0,0                          //Disconnect 0th data connection
OK

+APP PDP: 0,DEACTIVE
```

5.1.2 Build TCP Server

```
//Example of Build TCP Server
AT+CNACT?                             //Check the 0th PDN/PDP local IP.
+CNACT: 0,1,"10.181.182.177"          About active PDN/PDP refer to Bearer
+CNACT: 1,0,"0.0.0.0"                 Configuration.
+CNACT: 2,0,"0.0.0.0"
+CNACT: 3,0,"0.0.0.0"
OK
AT+CASERVER=0,0,"TCP",6000            //Create TCP server with 0th PDP on port 6000 of
                                      0th connection
+CASERVER: 0,0                        //Create success

OK
+CANEW: 0,1,117.131.85.139,5004       //Have a new client access on 0th connection and
                                      the client has been assigned to 1th connection.
+CADATAIND: 1                         //Date comes in on 1th connection.
AT+CARECV=1,100                       //Read 100 byte data
+CARECV: 10,GET / HTTP                //Actual output 10 byte data

OK
AT+CASEND=1,5                         //Request to send 5 bytes of data
>                                     //Input data

OK
+CASEND: 1,0,5                        //Data sent successfully
AT+CACLOSE=1                          //Close the connection with a connection identifier
                                      of 1.
OK
AT+CACLOSE=0                          //Close the connection with a connection identifier
                                      of 0.
OK
AT+CNACT=0,0                          //Disconnect 0th PDP
OK

+APP PDP: DEACTIVE
```

5.1.3 Build UDP Server

//Example of Build UDP Server.	
AT+CNACT?	//Check the 0th PDN/PDP local IP.
+CNACT: 0,1,"10.181.182.177"	About active PDN/PDP refer to Bearer Configuration.
+CNACT: 1,0,"0.0.0.0"	
+CNACT: 2,0,"0.0.0.0"	
+CNACT: 3,0,"0.0.0.0"	
OK	
AT+CASERVER=0,0,"UDP",6000	//Create UDP server with 0th PDP on port 6000 of 0th connection
+CASERVER: 0,0	//Create success
OK	
+CADATAIND: 0	//Date comes in on 0th connection.
+CANEW: 0,0,117.131.85.139,5001	//Have a new client access on 0th connection and the client has been assigned to 0th connection.
AT+CARECV=0,100	//Read 100 byte data on 0th connection
+CARECV: 10,GET / HTTP	//Actual output 10 bytes data
OK	
AT+CACFG="REMOTEADDR",1,117.131.85.139,6014	//Set remote address information for send to.
OK	
AT+CASEND=0,5	//Request to send 5 bytes of data
>	//Input data
OK	//Data sent successfully
+CASEND: 0,0,5	
AT+CACLOSE=0	//Close the connection with a connection identifier of 0.
OK	
AT+CNACT=0,0	//Disconnect 0th PDP
OK	
+APP PDP: DEACTIVE	

5.2 Build a TCP/UDP connection with SSL

The following example is to visit Baidu web as an example.

5.2.1 Configure SSL parameters and Certificate

About SSL parameters configure and certificate convert please refer to “SIM7070_SIM7080_SIM7090 Series_SSL_Application Note”.

5.2.2 Build a one-way authentication SSL(TLS) connection

Because of modules can only serve as clients. When you need to establish a one-way authentication connection, you need to import the root certificate of the server. If no certificate is imported, the module will default that all the servers can be trusted.

//Example of Build a one-way authentication SSL(TLS) connection .	
AT+CNACT?	//Check the 0th PDN/PDP local IP.
+CNACT: 0,1,"10.181.182.177" **+CNACT: 1,0,"0.0.0.0"** **+CNACT: 2,0,"0.0.0.0"** **+CNACT: 3,0,"0.0.0.0"**	//About active PDN/PDP refer to Bearer Configuration.
OK	
AT+CSSLCFG="SSLVERSION",0,3	//Set the protocol type of SSL with an identifier of 0. In this example,3 indicate TLS1.2
OK	
AT+CASSLCFG=0,"SSL",1	//Whether to use SSL, 1 means to turn on the SSL function.
OK	
AT+CASSLCFG=0,"CRINDEX",0	//Select SSL configure file Configure SSL. Identifier for AT+CSSLCFG corresponding SSL configuration.
OK	
AT+CASSLCFG=0,"CACERT","root.pem"	//Set root certificate. The root certificate must be a certificate that has been converted through AT+CSSLCFG. This item can be omitted. If omitted, all server certificates are trusted by default.
OK	
AT+CAOPEN=0,0,"TCP","117.131.85.139",6005	//Create a SSL connection with 0th PDP on 0th

	connection identifier
+CAOPEN: 0,0	//Connection success
OK	//Data come in on 0th connection. When a connection is successfully established or data is successfully sent, the module actively reads the data once, and if the server data is received, the URC is reported. If no data is received, the URC will not be reported.
+CADATAIND: 0	//After open a connection successfully, if module receives data, it will report "+CADATAIND: <cid>" to remind user to read data
AT+CARECV=0,100	//Read 100 byte data
+CARECV: 10,GET / HTTP	//Output data
OK	
AT+CASEND=0,5	//Request to send 5 bytes of data
>	Input data
OK	//Data sent successfully
+CASEND: 0,0,5	
AT+CACLOSE=0	//Close the connection with an identifier of 0.
OK	
AT+CNACT=0,0	//Close the connection with an identifier of 0.
OK	
+APP PDP: DEACTIVE	

5.2.3 Build a two-way authentication SSL(TLS) connection

To establish a two-way authentication SSL connection, you need to set up a client certificate. The client certificate needs to be transformed through "AT+CSSLCFG" first.

The certificate format that the module can support is .PEM, .DER and .P7B.

//Example of Build a two-way authentication SSL(TLS) connection.	
AT+CNACT?	//Check the 0th PDN/PDP local IP.
+CNACT: 0,1,"10.181.182.177"	//About active PDN/PDP refer to Bearer Configuration.
+CNACT: 1,0,"0.0.0.0"	
+CNACT: 2,0,"0.0.0.0"	
+CNACT: 3,0,"0.0.0.0"	
OK	

AT+CSSLCFG="SSLVERSION",0,3	//Set the protocol type of SSL with an identifier of 0. 3 indicate TLS1.2
OK	
AT+CASSLCFG=0,"SSL",1	//Set the 0th connection ' s SSL enable option. //Whether to use SSL, 1 means to turn on the SSL function.
OK	
AT+CASSLCFG=0,"CRINDEX",0	//Identifier for AT+CSSLCFG corresponding SSL configuration.
OK	
AT+CASSLCFG=0,"CACERT","root.pem"	//Set root certificate. The root certificate must be a certificate that has been converted through AT+CSSLCFG. This item can be omitted. If omitted, all server certificates are trusted by default.
OK	
AT+CASSLCFG=0,"CERT","client.pem"	//Set up client certificates. //The root certificate must be converted to a certificate that can be directly used by AT+CSSLCFG.
OK	
AT+CAOPEN=0,0,"TCP","117.131.85.139",6005	//Create a SSL connection with 0th PDP on 0th //connection identifier. //Connection success
+CAOPEN: 0,0	
OK	
AT+CASEND=0,5	//Request to send 5 bytes of data
>	//Input data
OK	//Data sent successfully
+CASEND: 0,0,5	
AT+CACLOSE=0	//Close the connection with a connection identifier of 0.
OK	
AT+CNACT=0,0	//Disconnect 0th PDP
OK	
+APP PDP: 0,DEACTIVE	

5.2.4 Build a PSK authentication SSL (DTLS) connection

To establish PSK DTLS connection, you need to set up a psktable. The psktable needs to be transformed through "AT+CSSLCFG" first.

//Example of Build a PSK authentication SSL (DTLS) connection.	
AT+CNACT?	//Check the 0th PDN/PDP local IP.
+CNACT: 0,1,"10.181.182.177" **+CNACT: 1,0,"0.0.0.0"** **+CNACT: 2,0,"0.0.0.0"** **+CNACT: 3,0,"0.0.0.0"**	//About active PDN/PDP refer to Bearer Configuration.
OK	
AT+CSSLCFG="SSLVERSION",0,5	//Set the protocol type of SSL with an identifier of 0. 5 indicate DTLS1.2
OK	
AT+CASSLCFG=0,"SSL",1	//Whether to use SSL, 1 means to turn on the SSL function.
OK	
AT+CASSLCFG=0,"CRINDEX",0	//Select SSL configure file Configure SSL. //Identifier for AT+CSSLCFG corresponding SSL configuration
OK	
AT+CASSLCFG=0,"PSKTABLE","psktable.secrets"	//Select psktable configure file. The psktable must be a file that has been converted through AT+CSSLCFG. This item does not be omitted, If the server uses PSK Cipher Suites.
OK	
AT+CAOPEN=0,0,"UDP","117.131.85.139",6013	//Create a SSL connection with 0th PDP on 0th connection identifier.
+CAOPEN: 0,0	//Connection success
OK	
+CADATAIND: 0	//Data comes in on 0th connection. When a connection is successfully established or data is successfully sent, the module actively reads the data once, and if the server data is received, the URC is reported. //If no data is received, the URC will not be reported.
AT+CARECV=0,100	//Read 100 byte data
+CARECV: 10,GET / HTTP	//Output data
OK	
AT+CACLOSE=0	//Close the connection with an identifier of 0.
OK	
AT+CNACT=0,0	//Disconnect data connection
OK	

```
+APP PDP: DEACTIVE
```