

Simcom Android ril User Guide V1.11

	History
2016-06-07	First release
2017-11-27	Version 1.11

1. SIMCOM Module USB Port Description

For SIM7100/SIM7200/SIM7230/SIM7250/SIM7500/SIM7600, USB VID is 0x1E0E, PID is 0x9001

For SIM5360/SIM6320/SIM5320, USB VID is 0x05C6, PID is 0x9000

7100 series module as a Slave USB device, configuration as following table

Interface number		
0	USB serial	Diagnostic Interface
1	USB serial	GPS NMEA Interface
2	USB serial	AT port Interface
3	USB serial	Modem port Interface
4	USB serial	USB Audio Interface
5	USB Net	NDIS wwan interface
6	USB adb	Android add debug port

SIM7100 could support NDIS dial up, but it is ppp dial up by default.

Based on the dial up mode, please choose the following configuration step accordingly.

2. USB Serial Driver

1> USB Serial Kernel Configuration

```
CONFIG_USB_SERIAL=y
CONFIG_USB_SERIAL_WWAN=y
CONFIG_USB_SERIAL_OPTION=y
```

If the module is SIM5360/SIM6320/SIM5320, since normally the linux kernel would support the VID/PID of these modules by default, so after doing above configuration, could jump to the fourth chapter "Ril Library Application" directly.

2> Add SIM7100 VID/PID

Modify the code in kernel source file option.c (normally the path is: drivers/usb/serial/option.c)

- If the Kernel is newer than V3.2 (include V3.2)

```
#define SIMCOM_SIM7100_VID          0x1E0E
#define SIMCOM_SIM7100_PID          0x9001

//for SIM7100 modem for NDIS
static const struct option_blacklist_info simcom_sim7100_blacklist = {
    .reserved = BIT(5),
};
```

Add below in option_ids list

```
... ..
//for SIM7100 modem for NDIS
{ USB_DEVICE(SIMCOM_SIM7100_VID, SIMCOM_SIM7100_PID),
  .driver_info = (kernel_ulong_t)& simcom_sim7100_blacklist
},
... ..
```

- If the Kernel version is lower than V3.2

```
#define SIMCOM_SIM7100_VID          0x1E0E
#define SIMCOM_SIM7100_PID          0x9001
```

Add below in option_ids list

```
{ USB_DEVICE(SIMCOM_SIM7100_VID, SIMCOM_SIM7100_PID)}, /*SIM7100 */
```

3> Reserve NDIS Port :

If use PPP dial up, no need to follow this step.

If use NDIS dial up, need to do such configuration

In option_probe of option.c, add the code below:

```
/* sim7100 */
if (serial->dev->descriptor.idVendor == SIMCOM_SIM7100_VID &&
    serial->dev->descriptor.idProduct == SIMCOM_SIM7100_PID &&
    serial->interface->cur_altsetting->desc.bInterfaceNumber == 5 )
    return -ENODEV;
```

4> Print Kernel Debug Information

If the driver is compiled successfully in Kernel, kernel would print below information after power on and connect to module,

```
usb 1-1: new high speed USB device using rt3xxx-ehci and address 2
option 1-1:1.0: GSM modem (1-port) converter detected
usb 1-1: GSM modem (1-port) converter now attached to ttyUSB0
option 1-1:1.1: GSM modem (1-port) converter detected
usb 1-1: GSM modem (1-port) converter now attached to ttyUSB1
option 1-1:1.2: GSM modem (1-port) converter detected
usb 1-1: GSM modem (1-port) converter now attached to ttyUSB2
option 1-1:1.3: GSM modem (1-port) converter detected
usb 1-1: GSM modem (1-port) converter now attached to ttyUSB3
option 1-1:1.4: GSM modem (1-port) converter detected
usb 1-1: GSM modem (1-port) converter now attached to ttyUSB4
```

dev/ttyUSB0~4 would be available by then, upper layer could communicate with module through those port(like send AT command, dial up).

3 USB NDIS NET

If use PPP dial up, could skip this step and jump to fourth chapter “Ril Library Application”

1> Kernel Configuration

Linux has integrated QMI WWAN driver in source code from Version 3.4.1. If Kernel version is newer (include) than Version 3.4.1, enable below three configuration is enough.

```
CONFIG_USB_WDM=y
CONFIG_USB_USBNET=y
CONFIG_USB_NET_QMI_WWAN=y
```

And add SIM7100 VID/PID in qmi_wwan.c

```
{ QMI_FIXED_INTF (0x1e0e, 0x9001,5)}, /* SIM7100 Modem Device */
```

If the Kernel version is lower than V 3.4.1, enable below two configuration, and use the driver we provide .

```
CONFIG_USB_WDM=y
CONFIG_USB_USBNET=y
```

We provide three files cdc-wdm.c, qmi_wwan.c and simcom_wwan.c. (packaged together in RIL Library, refer to the fourth chapter)

Please notice qmi_wwan.c and cdc-wdm.c is for SIM7100 NDIS dial up, simcom_wwan.c is for SIM7500/SIM7600 NDIS dial up.

For SIM7100 NDIS dial up, cdc-wdm.c is under the path drivers/usb/class, if there is cdc-wdm.c file already in kernel, replace it with the one we provided.

For SIM7100 NDIS dial up, place qmi_wwan.c under path drivers/net/usb, and modify Makefile

```
obj-$(CONFIG_USB_USBNET) += usbnet.o qmi_wwan.o
```

If the driver is compiled to kernel correctly, kernel would print below information after power on and connect to module.

```
qmi_wwan 1-1:1.5: cdc-wdm0: USB WDM device  
qmi_wwan 1-1:1.5: wwan0: register 'qmi_wwan' at usb-rt3xxx-1, Qualcomm Gobi  
wwan/QMI device, d6:d8:6c:10:b0:0e
```

For SIM7500 NDIS dial up, place simcom_wwan.c under path drivers/net/usb, and modify Makefile

```
obj-$(CONFIG_USB_USBNET) += usbnet.o simcom_wwan.o
```

If the driver is compiled to kernel correctly, kernel will print below message automatically after module is re-started.

```
simcom_wwan 1-1:1.5 wwan0: register 'simcom_wwan' at usb-0000:02:03.0-1,  
SIMCOM wwan/QMI device, 8a:de:f6:67:ce:1b
```

2> Use ifconfig to check NIC information, down status by default

```
wwan0      Link encap:Ethernet  HWaddr D6:D8:6C:10:B0:0E  
BROADCAST MULTICAST  MTU:1500  Metric:1  
RX packets:0 errors:0 dropped:0 overruns:0 frame:0  
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0  
collisions:0 txqueuelen:1000  
RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

4. RIL Library Application

1> Extract simcom_rilXX_XXXXXXXXX.tar.gz

The files include:

init.rc

rild

libril.so

libreference-ril.so

init.gprs-pppd (for PPP dial up)

3gdata_call.conf (for PPP dial up)

cdc-wdm.c (for SIM7100 NDIS dial up)

qmi_wwan.c (for SIM7100 NDIS dial up)

simcom_wwan.c (for SIM7500/7600 NDIS dial up)

gps.simcom.so (GPS Library)

chat (for ppp dial up)

2> Some feature supported by RIL is disabled by default. Customer need to add some attribute in android system if request those features.

The features supported currently :

GPS: rild.simcom.gps=1

enable GPS; 0 or no configuration: disable GPS

USSD: rild.simcom.ussd=1

STK: rild.simcom.stk=1

NDIS: rild.simcom.ndis=1

Only SIM7100 series support NDIS currently, SIM7500 series will support it soon.

STOPGPS: rild.simcom.stopgps=1

Close GPS when Android display screen is closed.

GPSLOG: rild.simcom.gpsloglevel=1

GPS library would disable most of the log by default. If customer meet any GPS issues and want to take GPS log for debug, could configure "GPSLOG" as 1, thus to get the complete GPS log for analysis.

NETCLOSE: rild.simcom.netclose=1

For SIM6320 module, if customer use module's internal protocol stack like TCP/HTTP/FTP, but also do external android PPP dial up. After internal protocol stack initiated successfully, would fail to do android PPP dial up. In such case, need to disable internal protocol stack.

Configure "NETCLOSE" as 1, android would disable internal protocol stack before doing PPP dial up.

CLVL: rild.simcom.clvl=* (*: 0~7)

If customer want to tune module's volume in ril, could configure the value of rild.simcom.clvl. Then Ril would configure the volume through "AT+CLVL=*" once sim card is ready.

CSDVC: rild.simcom.csdvc=*

If customer want to configure CSDVC value through ril, could use rild.simcom.csdvc. Ril would configure the value to module through "AT+CSDVC=*" after SIM card is ready.

After doing the configuration successfully, could check those property through "adb shell getprop" after power on device.

(During debug process, customer could modify the file "/system/build.prop" directly to enable/disable above features flexibly without recompiling the system)

Android 4.0/ 4.2 :

These two android versions don't have unified ril version , if need above feature for those two android versions, contact us to provide other exclusive version.

3> Init.rc file

Usually the file init.rc is in the path /devices/\$vendor_name/\$product_name/ of android source code, but some project may be different. if it is not obvious to find init.rc file, could use adb shell to check init.rc in the root directly of device, then inference which file is used for android source code.

We should modify the init.rc based on different dial up mode.

NDIS Dial up:

```
#modified by simcom
#-----
service ril-daemon /system/bin/rild -l /system/lib/libreference-ril.so
    class main
        socket rild stream 660 root radio
        socket rild-debug stream 660 radio system
#---- if need gps feature, unmark next line -----
#    socket rild-gps stream 660 radio system
    user root
    group radio cache inet misc audio sdcard_rw
```

Below figure is part of init.rc provided by simcom, transplant this part to customer's init.rc of source code.

If need GPS, enable the GPS line.

If customer use NDIS Dial up instead of PPP dial up, after finish this step, jump to step 5 (update rild and library file)

PPP Dial up :

```
#add by simcom
# change init.gprs-pppd for recovery mode
    chmod 0777 /etc/init.gprs-pppd
    chmod 0777 /etc/3gdata_call.conf
```

```

#modified by simcom
#-----
service ril-daemon /system/bin/rild -l /system/lib/libreference-ril.so
    class main
        socket rild stream 660 root radio
        socket rild-debug stream 660 radio system
#---- if need gps feature, unmark next line -----
#    socket rild-gps stream 660 radio system
    user root
    group radio cache inet misc audio sdcard_rw

service pppd_gprs /etc/init.gprs-pppd
    user root
    group radio cache inet misc
    disabled
    oneshot
#-----

```

PPP Dial Up for Android 6.0 :

```

#add by simcom
# change init.gprs-pppd for recovery mode
    chmod 0777 /etc/init.gprs-pppd
    chmod 0777 /etc/3gdata_call.conf

#modified by simcom
#-----
service ril-daemon /system/bin/rild -l /system/lib/libreference-ril.so
    class main
        socket rild stream 660 root radio system
        socket rild-debug stream 660 radio system
        socket rild-ppp stream 660 radio system
    user root
    group radio cache inet misc audio sdcard_rw

service pppd_gprs /etc/init.gprs-pppd
#class main
    user root
    group radio cache inet misc
    disabled
    oneshot

```

Above figures are from init.rc provided by simcom. Customer need to transplant those two parts into their own init.rc.

Some issues customer met before:

- a. Usually init.rc source code contains part of above configuration already, just follow above steps to modify the configuration, no need to add new one. (can't exist two service ril-daemon or two service pppd_gprs concurrently)
- b. In some customer's device, lots of path is read-only. In such case, may fail to

change the permission of `chmod 777 /etc/init.gprs-pppd`. Then need to designate the script to other path.

For instance, change the path to `/system/bin/`, then need to do the modification as below:

```
chmod 777 /system/bin/init.gprs-pppd
service pppd_gprs /system/bin/init.gprs-pppd
```

c. If change the path of `3gdata_call.conf`, then need to modify the file `init.gprs-pppd`. Since it designate the path of `3gdata_call.conf` in `init.gprs-pppd`, need to modify the path.

```
#/system/xbin/pppd $*
# pppd was put into /system/bin instead of /system/xbin after SDK1.6
/system/bin/pppd user $PPPD_USERNAME password $PPPD_PASSWORD connect 'chat -v -s -r "/var/log/chat.log" -
f "/etc/3gdata_call.conf"' disconnect 'chat -r "/var/log/chat.log" -t 30 -e -v "" +++ATH "NO CARRIER"'
$PPPD_DATAPORT 115200 mru 1280 mtu 1280 nodetach debug dump defaultroute usepeerdns novj novjccomp
noipdefault ipcp-accept-local ipcp-accept-remote connect-delay 500 linkname ppp0
```

4> Modify `init.rc-pppd` script, change the delay time to 500

5> Add PPP dial up script

Put `init.gprs-pppd` and `3gdata_call.conf` to the path designated in `init.rc`, and modify the permission.

```
adb push init.gprs-pppd /etc/
adb push 3gdata_call.conf /etc/
adb shell chmod 777 /etc/init.gprs-pppd
```

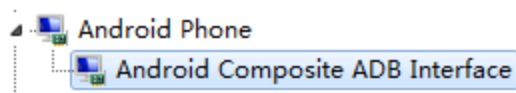
If there is no `chat` file under `/system/bin/` of android system, could push the `chat` file we provided to the path `/system/bin/`.

```
Adb push chat /system/bin/
Adb shell chmod 777 /system/bin/chat
```

Some issues customers met before:

a. `adb` is called Android Debug Bridge, mainly used to connect PC and android device. Usually we use `adb` and `logcat` together to print log, transfer file between PC and android device.

Could download `adb` package from network. Once android device is connected to PC and installed `adb` driver, could see below in device management window. Then could use `adb push` and other command.



- b. for some device, may need to call adb remount before use adb push.
- c. Some device doesn't have USB interface but use UART interface, then may could use SD card to copy the file to designated path.
- d. Some device can't operate cp command (read-only), in such case, could execute mount -o remount,rw /system under device's shell content.

6> Update Rild and library file

```
adb push rild /system/bin/
adb shell chmod 777 /system/bin/rild
adb push libreference-ril.so /system/lib/
adb push libril.so /system/lib/
```

7> The problem that exclamation mark shows on signal display icon in android 5.0 and above version

Android system would detect network through captive_portal_detection, it would send a HTTP request to server (the server by default is <http://connectivitycheck.android.com/>), since have no access to this server in China, thus would exist exclamation problem.

Another caused problem: if there is no data interactivity with network, android system would regard that current network is dysfunctional, thus redo the PPP dial up, or reinitialize the ril and wireless module.

Solution 1:

In the android source code

frameworks/base/packages/SettingsProvider/res/values/defaults.xml, add one attribute:

<bool name="def_captive_portal_detection_enabled">false</bool>. It would disable network detection feature.

Solution 2:

If customer want to keep the network detection feature, add the attribute captive_portal_server in defaults.xml. The value of captive_portal_server should be an accessible server. Or could modify the server directly in NetworkMonitor.java.

8> GPS

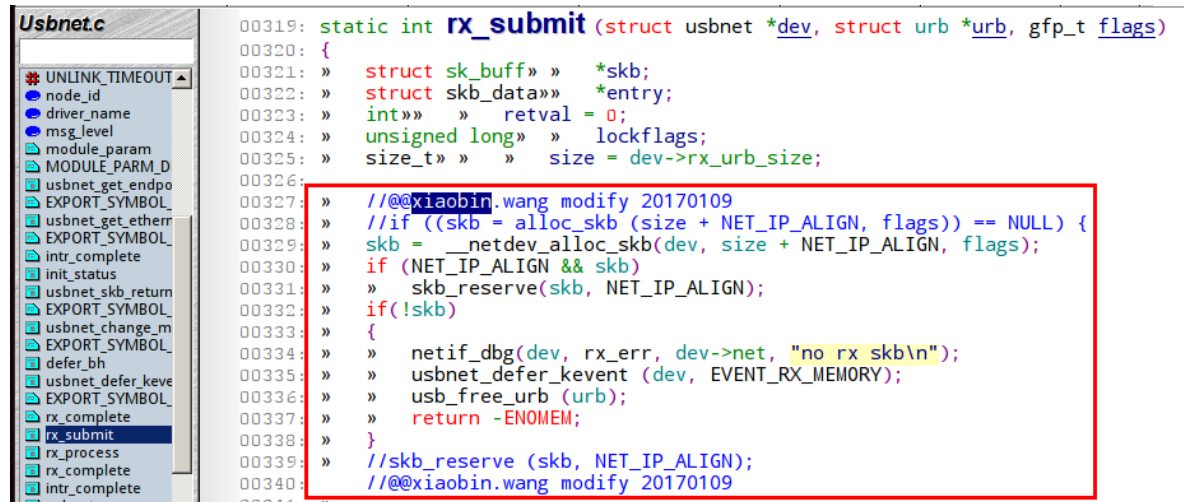
The library file for GPS we provided is gps.simcom.so, customer should modify the name based on their own system. Usually could enter /system/lib/hw to check previous library file name, then replace it

accordingly.

Adb push gps.simcom.so /system/lib/hw/gps.xxxx.so

9> For the problem to acquire IP in Kernel 3.0 and previous version

Compare customer's usbnet.c with the usbnet.c SIMCom provided, normally



```
00319: static int rx_submit (struct usbnet *dev, struct urb *urb, gfp_t flags)
00320: {
00321:     struct sk_buff *skb;
00322:     struct skb_data *entry;
00323:     int retval = 0;
00324:     unsigned long lockflags;
00325:     size_t size = dev->rx_urb_size;
00326:
00327:     //@@xiaobin.wang modify 20170109
00328:     //if ((skb = alloc_skb (size + NET_IP_ALIGN, flags)) == NULL) {
00329:     skb = __netdev_alloc_skb(dev, size + NET_IP_ALIGN, flags);
00330:     if (NET_IP_ALIGN && skb)
00331:         skb_reserve(skb, NET_IP_ALIGN);
00332:     if(!skb)
00333:     {
00334:         netif_dbg(dev, rx_err, dev->net, "no rx skb\n");
00335:         usbnet_defer_kevent (dev, EVENT_RX_MEMORY);
00336:         usb_free_urb (urb);
00337:         return -ENOMEM;
00338:     }
00339:     //skb_reserve (skb, NET_IP_ALIGN);
00340:     //@@xiaobin.wang modify 20170109
```

customer could check \kernel\drivers\net\usb\usbnet.c, then modify as below marked in red frame.

10> Catch ril log

If any issues after transplant the RIL, provide us the ril log:

adb logcat -b radio -v time >radio.txt

adb logcat -v time >main.txt

if device is connected with PC through UART interface, after enter into shell, execute below

Logcat -b radio -v time

Logcat -v time

Save the log printed as txt file.

5. FAQ

1. Wrong APN configuration :

If followed all the above to configure step by step, network is available but still can't dial up successfully, it is possible that forget to configure apn or apn configuration is wrong.

- Firstly check the network mode that whether it is CDMA/EVDO mode. If not using SIM7100CE/6320 and CDMA/EVDO sim card, then no need to consider CDMA/EVDO mode.

- b. Non-CDMA/EVDO mode: enter into android configuration interface, check if there is apn, whether it is activated. If not, then add the apn and activate it.
- c. CDMA/EVDO mode: in such case, the raw android configuration usually doesn't show apn configuration menu. Could judge whether apn is configured from radio log(also applicable to non-CDMA/EVDO network mode). Use UltraEdit to open radio log, search apn and get all the lines include apn. If still display null, then apn is not configured successfully.
- d. If confirmed the problem is apn configuration failed:
Then need to modify for add some configuration in /etc/apns-conf.xml of android system, then delete database file:/data/data/com.android.providers.telephony/databases/telephony.db, reset android, android would re-configure the apn menu normally.
For Chinese market, there are two types of telecom(CDMA) sim cards(46003 46011). 46011 is 4G card usually, need to add the configuration accordingly based on own card. Additionally, have to add username and password.
If it is private network sim card, need to fill in user name and password provided by operator. If it is public network sim card, just configure it as card card.
- e. If radio log still shows null after d step, then check whether configure the apn successfully. To manage and use apn, android doesn't read or operate apns-conf.xml, but to read database (telephony.db). so we could check in database whether the configuration we did exist or not.
Extract telephony.db to PC, then use SQLite Expert to check. If not extract the file, could use sqlite3 database command to check.

2. Port Property issue

In some customer's android system, ttyUSB port can't be used normally due to very limited access of ttyUSB* in their system.

Usually we could modify the ttyUSB access in rc file which should be under the subdirectory of device in android system. If execute "grep -rn ttyUSB" directly under device directory, would display a line which indicate the rc file, as well as access value. Normally the value should be 666, otherwise, find the rc file and modify the value as 666.

```
/dev/spi0ev2.0 0000 system system
/dev/ttyUSB* 0666 radio radio
/dev/ttySAC0 0666 system system
```

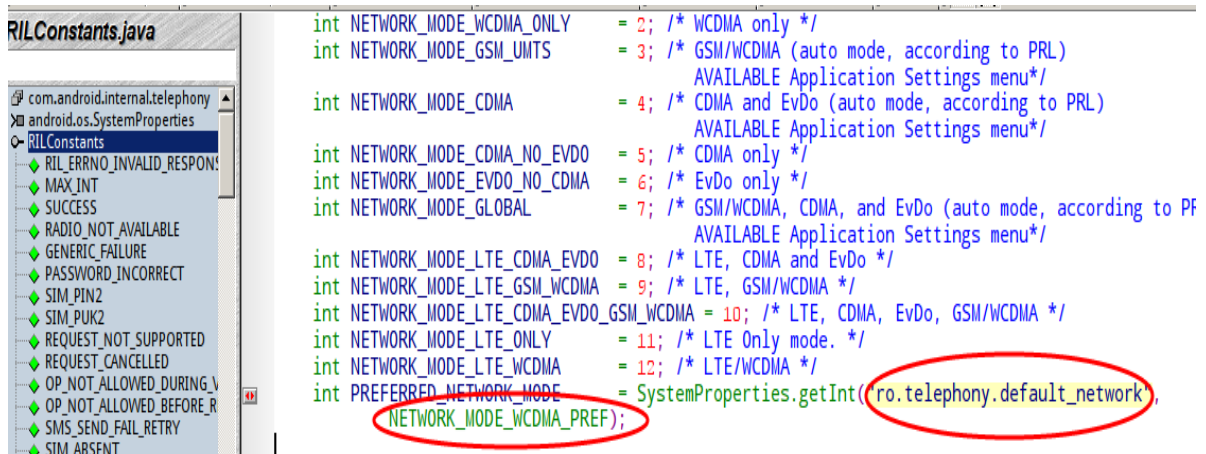
3. Add network mode in configuration menu

Some customer's android configuration menu of network mode may have only two options : 2G mode and 3G mode, doesn't have 4G mode.

a. Open android source code packages/services/Telephony/res/values/config.xml, modify the value of config_enabled_lte as true, then 4G mode would be available in configuration menu.

b. **Modify the default network mode as 4G mode:**

Open android source code



frameworks/base/telephony/java/com/android/internal/telephony/RILConstants.java
modify the value of PREFERRED_NET_MODE as NETWORK_MODE_LTE_GSM_WCDMA.

c. **Java source code for network mode configuration:**

packages/services/Telephony/src/com/android/phone/MobileNetworkSettings.java
if problem still exist after following step a and step b, could try to analyze the source code for further analysis.

4. Relative issues about APN configuration

Some android device may initiate MMS connection automatically, not initiated by subscriber proactively, it may be relative with android system itself.

Once RIL receive the MMS connection, would disconnect current normal data connection, and redo the dial up with MMS's APN to connect to network. After finishing MMS operation, ril may wait for android's new request for data network connection again. So the network would disconnect for a while since it would take some time for dial up to connect to network.

Solution: if customer doesn't need the feature like MMS, email, we recommend to remove the APN configuration for MMS, email.

Lots of Android devices reserve the APN for MMS, email. We could see a couple of APN options in APN configuration interface, recommend to reserve only one APN.

For the three operators in China, could reserve the APN as below:

China Mobile: cmnet

China Unicom: 3gnet

China Telecom: ctnet

Below modification method for reference:

Before modifying APN:

```
<apn carrier="China Mobile" mcc="460" mnc="00" apn="cmnet" type="default,supl" />
<apn carrier="China Mobile" mcc="460" mnc="02" apn="cmnet" type="default,supl" />
<apn carrier="中国移动 (China Mobile) GPRS" mcc="460" mnc="07" user="cmnet" password="cmnet" />
<apn carrier="China Mobile MMS" mcc="460" mnc="00" apn="cmwap" proxy="10.0.0.172" port="80" mmsc="http://mmsc.monternet.com" />
<apn carrier="China Mobile MMS" mcc="460" mnc="02" apn="cmwap" proxy="10.0.0.172" port="80" mmsc="http://mmsc.monternet.com" />
<apn carrier="中国移动彩信 (China Mobile)" mcc="460" mnc="07" apn="cmwap" proxy="10.0.0.172" port="80" mmsc="http://mmsc.monternet.com" />
<apn carrier="China Mobile CMWAP" apn="CMWAP" mcc="460" mnc="00" user="wap" password="wap" />
<apn carrier="China Mobile CMWAP" apn="CMWAP" mcc="460" mnc="02" user="wap" password="wap" />
<apn carrier="China Mobile CMWAP" apn="CMWAP" mcc="460" mnc="07" user="wap" password="wap" />
<apn carrier="China Unicom 3G" mcc="460" mnc="01" apn="3gnet" port="80" type="default,supl" />
<apn carrier="中国联通 3g 彩信 (China Unicom)" mcc="460" mnc="01" apn="3gwap" mmsc="http://mmsc.myuni.com.cn" />
<apn carrier="China Unicom MMS" mcc="460" mnc="01" apn="uniwap" mmsc="http://mmsc.myuni.com.cn" />
<apn carrier="China Telecom" apn="CTNET" mcc="460" mnc="03" user="ctnet@mycdma.cn" password="vnetcm" />
<apn carrier="China Telecom wap" apn="CTWAP" mcc="460" mnc="03" user="ctwap@mycdma.cn" password="vnetcm" />
<apn carrier="China Telecom Mms" apn="CTWAP" mcc="460" mnc="03" user="ctwap@mycdma.cn" password="vnetcm" />
```

Then remove the parts marked with cross and underline in red.

After modification:

```
<apn carrier="China Mobile_00" mcc="460" mnc="00" apn="cmnet" type="default" />
<apn carrier="China Mobile_02" mcc="460" mnc="02" apn="cmnet" type="default" />
<apn carrier="China Unicom_01" mcc="460" mnc="01" apn="3gnet" type="default" />
<apn carrier="China Unicom_06" mcc="460" mnc="06" apn="3gnet" type="default" />
<apn carrier="China Telecom03" apn="ctnet" mcc="460" mnc="03" user="ctnet@mycdma.cn" password="vnetcm" />
<apn carrier="China Telecom11" apn="ctnet" mcc="460" mnc="11" user="ctnet@mycdma.cn" password="vnetcm" />
```