Talos Vulnerability Report

TALOS-2022-1522

InHand Networks InRouter302 httpd port 4444 upload.cgi leftover debug code vulnerability

OCTOBER 27, 2022

CVE NUMBER

CVE-2022-29888

SUMMARY

A leftover debug code vulnerability exists in the httpd port 4444 upload.cgi functionality of InHand Networks InRouter302 V3.5.45. A specially-crafted HTTP request can lead to arbitrary file deletion. An attacker can send an HTTP request to trigger this vulnerability.

CONFIRMED VULNERABLE VERSIONS

The versions below were either tested or verified to be vulnerable by Talos or confirmed to be vulnerable by the vendor.

InHand Networks InRouter302 V3.5.45

PRODUCT URLS

InRouter302 - https://www.inhandnetworks.com/products/inrouter300.html

CVSSV3 SCORE

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

CWE

CWE-489 - Leftover Debug Code

DETAILS

The InRouter302 is an industrial LTE router. It features remote management functionalities and several security protection mechanisms, such as: VPN technologies, firewall functionalities, authorization management and several other features.

One of the ports the `httpd` protocol listens to is 4444. This port seems to have different purposes. In some APIs it offers more functionalities, in others it is used to managed the APIs in headless mode. But in some APIs, that port is used to trigger some debug code that could perform certain actions outside the scope of the API.

The `upload.cgi` API will execute mainly two functions: `upload.cgi_input` that will parse the POST request, and `upload.cgi_output` that will use the parsed input to perform the actual API request and return the output, if required. The `upload.cgi_input` function:

```
void upload.cgi_input(char* cgi_func_name,uint CONTENT_LENGTH,char *BOUNDARY)

{
  [...]

  if ((post == 0) || (BOUNDARY == (char *)0x0)) {
    syslog(4,"not POST or no boundary for multipart upload!");
  }
  else {
    boundary_len = strlen(BOUNDARY);
    if ((int)CONTENT_LENGTH < 0x400000) {
      syslog(7,"wi_upload: %s",cgi_func_name);
      if (gl_server_port != 4444) {
[1]
        webcgi_init(0,cgi_func_name);
[2]
      }
    [...]
}
```

The two main variables that are going to be parsed, and later used in the `upload.cgi_output`, are `type` and `filename`. The `upload.cgi_input` function is also responsible for creating a temporary file with the content of the provided one. The file `/tmp/<provided_filename>` is opened and filled with the provided content. The `filename` variable goes through different checks in order to prevent path traversal vulnerabilities.

Later, in `upload.cgi_output`, based on the `type` variable provided, different actions could be performed. Eventually the temporary file created will be removed. The `upload.cgi_output` function:

```
void upload.cgi_output(void)

{

  [...]
  type = (char *)webcgi_get("type");
  filename = (char *)webcgi_get("filename");
  [...]
}
```

The `httpd` webserver parses some request variables, like the ones specified in the URL. In `upload.cgi_input`, at `[1]`, it is checked if the request was for the `httpd` webserver that listens at port 4444. If this is is the case, the already-parsed variables are not removed. This can inject arbitrary values in `type` and `filename` variables, bypassing the checks performed in `upload.cgi_input`. Then the `upload.cgi_output` will use the parsed variables, without knowing that those were the ones specified in the URL and not parsed by `upload.cgi_input`. This problem can lead to a delete arbitrary file vulnerability, exploiting the cleanup procedure, due to the `upload.cgi_output` removing the “temporary” file `/tmp/<provided_filename>`. Where `<provided_filename>`, if specified in the URL, is an arbitrary value specified in the `filename` parameter.

TIMELINE

2022-06-07 - Vendor Disclosure
2022-10-25 - Vendor Patch Release
2022-10-27 - Public Release

CREDIT

Discovered by Francesco Benvenuto of Cisco Talos.

© 2022 Cisco Systems, Inc. and/or its affiliates. All rights reserved. View our Privacy Policy.