5 CVE-2021-22876: Automatic referer leaks credentials

Share:

TIMELINE

vsz submitted a report to curl. Feb 11th (2 ye

Summary:

When using the `--referer ';auto'` feature the current URL is copied as-is to the referrer header of the subsequent request. The recommendation [1] is to strip t (along with the URL fragment). I can imagine this may, in rare cases, result in unwanted/unexpected disclosure of credentials (e.g. them appearing in 3rd party web server logs), though the overall chances seem low (also considering that ';auto', by hunch, is likely not a widely used curl feature).

[1] https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Referer#directives

Steps To Reproduce:

Code 94 Bytes Wrap lines Copy Dow

```
1 $ curl -svLe ';auto' 'https://user:pass@curl.haxx.se#frag'  2>&1 >/dev/null | grep -i Referer:
```

Supporting Material/References:

Code 482 Bytes Wrap lines Copy Dow

```
1 $ curl -V
2 curl 7.64.1 (x86_64-apple-darwin19.0) libcurl/7.64.1 (SecureTransport) LibreSSL/2.8.3 zlib/1.2.11 nghttp2/1.39.2
3 Release-Date: 2019-03-27
4 Protocols: dict file ftp ftps gopher http https imap imaps ldap ldaps pop3 pop3s rtsp smb smbs smtp smtps telnet tftp
5 Features: AsynchDNS GSS-API HTTP2 HTTPS-proxy IPv6 Kerberos Largefile libz MultiSSL NTLM NTLM_WB SPNEGO SSL UnixSockets
```

Patch that fixes it

Code 1.84 KiB Wrap lines Copy Dow

```
1  diff --git a/lib/transfer.c b/lib/transfer.c
2  index c3b2d11a2..f63aadbaf 100644
3  --- a/lib/transfer.c
4  +++ b/lib/transfer.c
5  @@ -1567,6 +1567,9 @@ CURLcode Curl_follow(struct Curl_easy *data,
6         data->state.followlocation++; /* count location-followers */
7
8         if(data->set.http_auto_referer) {
9  +        CURLU *h;
10 +        char *url;
11 +
12           /* We are asked to automatically set the previous URL as the referer
13              when we get the next URL. We pick the ->url field, which may or may
14              not be 100% correct */
15 @@ -1576,7 +1579,40 @@ CURLcode Curl_follow(struct Curl_easy *data,
16             data->change.referer_alloc = FALSE;
17           }
18
19 -         data->change.referer = strdup(data->change.url);
20 +         /* Make a copy of the URL without the crenditals and fragment */
21 +         h = curl_url();
22 +         if(!h)
23 +           return CURLE_OUT_OF_MEMORY;
24 +
25 +         uc = curl_url_set(h, CURLUPART_URL, data->change.url, 0);
26 +         if(uc) {
27 +           curl_url_cleanup(h);
28 +           return CURLE_OUT_OF_MEMORY;
29 +         }
30 +         uc = curl_url_set(h, CURLUPART_FRAGMENT, NULL, 0);
31 +         if(uc) {
32 +           curl_url_cleanup(h);
33 +           return CURLE_OUT_OF_MEMORY;
34 +         }
35 +         uc = curl_url_set(h, CURLUPART_USER, NULL, 0);
36 +         if(uc) {
37 +           curl_url_cleanup(h);
38 +           return CURLE_OUT_OF_MEMORY;
39 +         }
40 +         uc = curl_url_set(h, CURLUPART_PASSWORD, NULL, 0);
41 +         if(uc) {
42 +           curl_url_cleanup(h);
43 +           return CURLE_OUT_OF_MEMORY;
44 +         }
45 +         uc = curl_url_get(h, CURLUPART_URL, &url, 0);
46 +         if(uc) {
```

```
50  +
51  +       curl_url_cleanup(h);
52  +
53  +       data->change.referer = url;
54          if(!data->change.referer)
55            return CURLE_OUT_OF_MEMORY;
56          data->change.referer_alloc = TRUE; /* yes, free this later */
```

I'm ready to continue this in a public PR if it seems more fitting.

Impact

The best I can think of is if an attacker gets hold of web server logs that includer referrer info with credentials leaked into them. It's a privacy/sensitive info-leak vulnerability at best. Can't readily think of a way to actively exploit this.

vsz changed the scope from **None** to **https://github.com/curl/curl**. Feb 11th (2 ye

bagder curl staff posted a comment. Feb 12th (2 ye

Thanks for your report. I think I agree that this needs to be treated as a security issue.

bagder curl staff posted a comment. Feb 12th (2 ye

To reduce the number of cleanup + returns in that code, maybe it could be written like this:

Code 408 Bytes Wrap lines Copy Dow

```
1   uc = curl_url_set(h, CURLUPART_URL, data->change.url, 0);
2   if(!uc)
3     uc = curl_url_set(h, CURLUPART_FRAGMENT, NULL, 0);
4   if(!uc)
5     uc = curl_url_set(h, CURLUPART_USER, NULL, 0);
6   if(!uc)
7     uc = curl_url_set(h, CURLUPART_PASSWORD, NULL, 0);
8   if(!uc)
9     uc = curl_url_get(h, CURLUPART_URL, &url, 0);
10   if(uc) {
11     curl_url_cleanup(h);
12     return CURLE_OUT_OF_MEMORY;
13   }
14
15   curl_url_cleanup(h);
16
```

vsz posted a comment. Feb 12th (2 ye

Thank you for looking into this. This is indeed a much cleaner way to code it. Also made a successful test with it.

bagder curl staff posted a comment. Feb 12th (2 ye

This flaw was introduced here: https://github.com/curl/curl/commit/f30ffef477, landed in curl 7.1.1. Released on August 21 2000...

bagder curl staff posted a comment. Feb 12th (2 ye

Since this issue has been around for two decades and is in a fairly unused feature, I don't think we need to rush out a fix for this. I propose we publish the fix and adv coordinated with the scheduled 7.76.0 release on March 31, 2021.

vsz posted a comment. Feb 12th (2 ye

Agreed, this sounds reasonable.

Slighly (un)related (and not a security issue), it may also make sense to skip setting the initial referrer value if it is empty, e.g. with the input value ';auto'. Not sure if counts as breaking change, but assigning an empty value to the referrer header doesn't seem to give much benefit over just omitting it. Patch:

Code 429 Bytes Wrap lines Copy Dow

```
1  diff --git a/src/tool_getparam.c b/src/tool_getparam.c
2  index d187643a7..0e82b3c8b 100644
3  --- a/src/tool_getparam.c
4  +++ b/src/tool_getparam.c
5  @@ -1545,7 +1545,8 @@ ParameterError getparameter(const char *flag, /* f or -long-flag */
6          }
7          else
8            config->autoreferer = FALSE;
9  -       GetStr(&config->referer, nextarg);
10 +       if(*nextarg)
11 +         GetStr(&config->referer, nextarg);
12       }
13       break;
14       case 'E':
```

bagder curl staff posted a comment. Feb 12th (2 ye

Yeah, I think that's sensible.

bagder curl staff posted a comment. Feb 14th (2 ye

according to our process)

I'll get a first draft of a security advisory done soonish and post here.

bagder curl staff posted a comment. Feb 16th (2 ye

Here's my first take at an advisory for this issue.

Automatic referer leaks credentials

Project curl Security Advisory, March 31st 2021 -
Permalink

VULNERABILITY

libcurl does not strip off user credentials from the URL when automatically populating the `Referer:` HTTP request header field in outgoing HTTP requests, and therefore risks leaking sensitive data to the server that is the target of the second HTTP request.

libcurl automatically sets the `Referer:` HTTP request header field in outgoing HTTP requests if the `CURLOPT_AUTOREFERER` option is set. With the curl tool, it is enabled with `--referer ";auto"`.

We are not aware of any exploit of this flaw.

INFO

This flaw has existed in libcurl since commit
f30ffef477 in libcurl 7.1.1,
released on August 21, 2000.

The Common Vulnerabilities and Exposures (CVE) project has assigned the name CVE-2021-XXXX to this issue.

CWE-359: Exposure of Private Personal Information to an Unauthorized Actor

Severity: Low

AFFECTED VERSIONS

- Affected versions: libcurl 7.1.1 to and including 7.75.0
- Not affected versions: libcurl < 7.1.1 and libcurl >= 7.76.0

Also note that libcurl is used by many applications, and not always advertised as such.

THE SOLUTION

If a provided URL contains credentials, they will be blanked out before the URL is used to populate the header field.

A [fix for CVE-2021-XXXX](insert link)

RECOMMENDATIONS

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

A - Upgrade libcurl to version 7.76.0

B - Apply the patch to your local version

C - Avoid `CURLOPT_AUTOREFERER` and `--referer ";auto"`,

TIMELINE

This issue was reported to the curl project on February 12, 2021.

This advisory was posted on March 31st 2021.

CREDITS

This issue was reported and patched by Viktor Szakats.

Thanks a lot!

vsz posted a comment. Feb 16th (2 ye

Thank you @bagder!

I've committed the empty-referrer patch:
https://github.com/curl/curl/commit/cdb630655db39ff1a319ace871e75389072deeb9

Regarding this one, here an updated (slightly further simplified) patch

Code 1.49 KiB Wrap lines Copy Dow

```
1  diff --git a/lib/transfer.c b/lib/transfer.c
2  index ae414cfc5..2ad79a57c 100644
3  --- a/lib/transfer.c
4  +++ b/lib/transfer.c
5  @@ -1581,6 +1581,9 @@ CURLcode Curl_follow(struct Curl_easy *data,
```

```
9  +        CURLU *u;
10 +         char *referer;
11 +
12           /* We are asked to automatically set the previous URL as the referer
13              when we get the next URL. We pick the ->url field, which may or may
14              not be 100% correct */
15 @@ -1590,9 +1593,27 @@ CURLcode Curl_follow(struct Curl_easy *data,
16             data->change.referer_alloc = FALSE;
17           }
18
19 -         data->change.referer = strdup(data->change.url);
20 -         if(!data->change.referer)
21 +         /* Make a copy of the URL without crenditals and fragment */
22 +         u = curl_url();
23 +         if(!u)
24 +           return CURLE_OUT_OF_MEMORY;
25 +
26 +         uc = curl_url_set(u, CURLUPART_URL, data->change.url, 0);
27 +         if(!uc)
28 +           uc = curl_url_set(u, CURLUPART_FRAGMENT, NULL, 0);
29 +         if(!uc)
30 +           uc = curl_url_set(u, CURLUPART_USER, NULL, 0);
31 +         if(!uc)
32 +           uc = curl_url_set(u, CURLUPART_PASSWORD, NULL, 0);
33 +         if(!uc)
34 +           uc = curl_url_get(u, CURLUPART_URL, &referer, 0);
35 +
36 +         curl_url_cleanup(u);
37 +
38 +         if(uc || referer == NULL)
39             return CURLE_OUT_OF_MEMORY;
40 +
41 +         data->change.referer = referer;
42           data->change.referer_alloc = TRUE; /* yes, free this later */
43         }
44       }
```

It checks out okay in a limited local test, but I'd appreciate taking a critical eye on it.

vsz posted a comment. Feb 16th (2 ye

Your write-up looks spot on to me. 👍

bagder curl staff posted a comment. Feb 23rd (2 ye

@vsz do you have a test case that verifies the strip-credentials-from-referer functionality after this fix or do you want me to make one?

vsz posted a comment. Feb 24th (2 ye

@bagder: Hi Daniel, please find my attempt attached. It does the job of testing this, but it's quite possibly not the most minimal or ideal/perfect iteration. Tried to change '1080' (the test I used as a base for this) to other numbers, but got failures, so I'm still missing some details here). I hope it helps anyway!

1 attachment:
F1208224: test9999

bagder curl staff posted a comment. Feb 25th (2 ye

Looks excellent to me. Thanks!

bagder curl staff posted a comment. Feb 26th (2 ye

I made it into test2081 in my local branch and updated it slightly to use that number internally as well. Works with the patch, fails without it.

1 attachment:
F1209820: test2081

bagder curl staff updated CVE reference to CVE-2021-22876. Mar 8th (2 ye

Mar 8th (2 years ago)
bagder curl staff
changed the report title from **curl -L --referer ';auto' https://user:pass@example.com includes credentials in referrer when following redirects** to **CVE-2021-22876: Automatic referer leaks credentials**.

url rewarded vsz with a $800 bounty. Mar 8th (2 ye

The curl security team has decided to reward hacker @vsz with the amount of 800 USD for finding and reporting this issue. Many thanks for your great work!

bagder curl staff changed the status to **o Triaged**. Mar 10th (2 ye

This is now just waiting for publication.

bagder curl staff closed the report and changed the status to **o Resolved**. Mar 31st (2 ye

This report has been disclosed.

Apr 30th (2 ye