Sec Bug #81122 SSRF bypass in FILTER_VALIDATE_URL

Submitted:	2021-06-10 02:37 UTC	**Modified:**	2021-07-16 22:03 UTC
From:	vi at hackberry dot xyz	**Assigned:**	cmb (profile)
Status:	Closed	**Package:**	URL related
PHP Version:	8.0.7	**OS:**	All
Private report:	No	**CVE-ID:**	2021-21705

View | Add Comment | Developer | Edit

[2021-06-10 02:37 UTC] vi at hackberry dot xyz

```
Description:
------------
In reports https://bugs.php.net/bug.php?id=77423 and https://bugs.php.net/bug.php?id=81116, it is suggested to use
FILTER_VALIDATE_URL but I have found a bypass that allows bypassing FILTER_VALIDATE_URL check.

Test script:
---------------
echo filter_var("https://example.com:\@test.com/", FILTER_VALIDATE_URL)



Expected result:
----------------
Should not validate as a valid URL given the URL.



Actual result:
--------------
Validates URL as valid. This payload in file_get_contents and parse_url would treat test.com as host.
```

Patches

Add a Patch

Pull Requests

Add a Pull Request

History

All | Comments | Changes | Git/SVN commits | Related reports

[2021-06-10 03:12 UTC] stas@php.net

```
-Status: Open
+Status: Feedback
```

[2021-06-10 03:12 UTC] stas@php.net

```
This seems to be a valid URL with username "example.com" and password "\". parse_url() parses it like that. Not sure
why is is a problem?
```

[2021-06-10 06:02 UTC] vi at hackberry dot xyz

```
-Status: Feedback
+Status: Open
```

[2021-06-10 06:02 UTC] vi at hackberry dot xyz

```
Consider the following code:

<?php
if(filter_var($_GET['url'], FILTER_VALIDATE_URL)) {
    header("location: ".$_GET['url']);
}

Now if you visit https://localhost/?url=https://example.com:\@test.com
The browser will redirect to https://example.com/@test.com. This can be used to bypass any open redirect mitigations
as well as introduce a discripency since from PHP's perspective, the host is test.com here but for the browser, host
is example.com

Now consider an SSRF protection which uses parse_url to check for test.com as host:

<?php
if(filter_var($_GET['url'], FILTER_VALIDATE_URL)) {
    if("test.com" === parse_url($_GET['url'])['host']) {
        header("location: ".$_GET['url']);
    }
}

The above will pass for the payload https://example.com:\@test.com but browser will redirect to
https://example.com/@test.com

Now another case where file_get_contents is used over username and password provided by a user:

<?php
$username = $_GET['user'];
$password = $_GET['pass'];
$loginurl = 'https://' . $username . ':' . $password . '@test.com/';

if(filter_var($loginurl, FILTER_VALIDATE_URL)) {
    echo file_get_contents($loginurl);
}

For the username as 'example.com' and password as '/', the request will be sent to
```

```
https://example.com/@test.com since ':' will indicate start of port. But port will end with '/' that indicates the
start of path.

Another case where we allow `\` in the password:

parse_url("https://user:\epass@test.com")

would return

[
     "scheme" => "https",
     "host" => "test.com",
     "user" => "user",
     "pass" => "_pass",
]

and the following:

parse_url("https://user:\\@test.com")

would return

[
    "scheme" => "https",
    "host" => "test.com",
    "user" => "user",
    "pass" => "\",
]

Another case I noticed,

parse_url("https://example.com:\/@test.com")

would return false which is completely unexpected as '\' should have escaped '/' resulting in:

[
    "scheme" => "https",
    "host" => "example.com",
    "path" => "/@test.com",
]
```

[2021-06-10 09:44 UTC] cmb@php.net

```
> This seems to be a valid URL with username "example.com" and
> password "\".

According to RFC 3986[1], it is not.

    userinfo      = *( unreserved / pct-encoded / sub-delims / ":" )
    unreserved    = ALPHA / DIGIT / "-" / "." / "_" / "~"
    sub-delims    = "!" / "$" / "&" / "'" / "(" / ")"
                  / "*" / "+" / "," / ";" / "="

The backslash would have to be percent encoded.  Since it is not,
apparently browsers interpret :\ as slash, here.

[1] <https://datatracker.ietf.org/doc/html/rfc3986>
```

[2021-06-13 19:18 UTC] stas@php.net

```
@cmb so should we add validation for the password part too?
```

[2021-06-13 21:51 UTC] cmb@php.net

```
-Assigned To:
+Assigned To: cmb
```

[2021-06-13 21:51 UTC] cmb@php.net

```
Yes.  I'll provide a patch on Monday.
```

[2021-06-14 11:30 UTC] cmb@php.net

```
-Assigned To: cmb
+Assigned To: stas
```

[2021-06-14 11:30 UTC] cmb@php.net

```
Formatted patch for PHP-7.3 and up:
<https://gist.github.com/cmb69/cd1a701099e0b904fd8aa4b150312bca>.

When merging that into master the SKIPIF section of the test case
should be replaced with:

    --EXTENSIONS--
    filter

> parse_url("https://example.com:\/@test.com")

This is actually

    parse_url("https://example.com:\\/@test.com")

since "\/" isn't a valid escape sequence.
```

[2021-06-16 10:41 UTC] vi at hackberry dot xyz

```
While fixing this, also take the following case in consideration:

---
parse_url("https://example.com:80\/@asdf.com");
=> [
     "scheme" => "https",
     "host" => "example.com",
     "port" => 80,
     "path" => "/@asdf.com",
   ]
---
```

```
Here \/ becomes a path separator making 80 a port number which I think is due to / getting escaped by backslash.
Tested on PHP 8.0.7
```

[2021-06-16 16:10 UTC] cmb@php.net

```
> https://example.com:80\/@asdf.com

Yes, that is an invalid URI (according to RFC 3986), and
parse_url() fails to parse it properly; I don't even consider this
as bug[1].  The fact that FILTER_VALIDATE_URL claims the URL to be
valid is a bug, but given that it is apparently interpreted as

    https://example.com:80//@asdf.com

by browsers, I don't think this is a security issue.

[1] <https://github.com/php/doc-en/commit/22fa19e2534e0749ee98b0f4dec87a3237006f8a>
```

[2021-06-21 05:03 UTC] stas@php.net

```
-CVE-ID:
+CVE-ID: 2021-21705
```

[2021-06-28 04:41 UTC] git@php.net

```
Automatic comment on behalf of cmb69 (author) and smalyshev (committer)
Revision: https://github.com/php/php-src/commit/a5538c62293fa782fcc382d0635cfc0c8b9190e3
Log: Fix #81122: SSRF bypass in FILTER_VALIDATE_URL
```

[2021-06-28 04:41 UTC] git@php.net

```
-Status: Assigned
+Status: Closed
```

[2021-07-16 01:26 UTC] kfoubert at sitecrafting dot com

```
This change might have caused an issue with validating against elastic search url and other similar URLs.

Elastic Search URL:
https://collections_name:[45 character string].us-west-2.aws.found.io:9243

Does this qualify as a valid URL?

The search page works fine on PHP 7.3.27 but no longer works with PHP 7.3.29, which Pantheon has switched to.

The FILTER_VALIDATE_URL is in the PHP Elastic Search composer package. The code below is now adding an unneeded
http://

```
/**
 * @param string $host
 *
 * @return string
 */
private function prependMissingScheme($host)
{

    if (!filter_var($host, FILTER_VALIDATE_URL)) {
        $host = 'http://' . $host;
    }

    return $host;
}
```
```

[2021-07-16 03:40 UTC] vi at hackberry dot xyz

```
According to RFC 2396 (https://www.ietf.org/rfc/rfc2396.txt), https://collections_name:[45 character string].us-west-
2.aws.found.io:9243 is not a valid URL. Because the : character in the host separates the port from host in hostport.
(See the section 3.2.2)


3.2.2. Server-based Naming Authority

   URL schemes that involve the direct use of an IP-based protocol to a
   specified server on the Internet use a common syntax for the server
   component of the URI's scheme-specific data:

      <userinfo>@<host>:<port>

   where <userinfo> may consist of a user name and, optionally, scheme-
   specific information about how to gain authorization to access the
   server.  The parts "<userinfo>@" and ":<port>" may be omitted.

      server        = [ [ userinfo "@" ] hostport ]

   The user information, if present, is followed by a commercial at-sign
   "@".

      userinfo      = *( unreserved | escaped |
                         ";" | ":" | "&" | "=" | "+" | "$" | "," )

   Some URL schemes use the format "user:password" in the userinfo
   field. This practice is NOT RECOMMENDED, because the passing of
   authentication information in clear text (such as URI) has proven to
   be a security risk in almost every case where it has been used.

   The host is a domain name of a network host, or its IPv4 address as a
   set of four decimal digit groups separated by ".".  Literal IPv6
   addresses are not supported.

      hostport      = host [ ":" port ]
      host          = hostname | IPv4address
      hostname      = *( domainlabel "." ) toplabel [ "." ]
      domainlabel   = alphanum | alphanum *( alphanum | "-" ) alphanum
      toplabel      = alpha | alpha *( alphanum | "-" ) alphanum
```

[2021-07-16 19:27 UTC] kfoubert at sitecrafting dot com

To clarify the actual URL being validated is basic authentication. Is that a valid URL?

https://[username]:[password]@cfd83a4879dc445d84dd990baf771358.us-west-2.aws.found.io:9243

[2021-07-16 19:31 UTC] kfoubert at sitecrafting dot com

Here's a better explanation, the actual URL being validated is basic authentication. Is that a valid URL?

Pattern (which I think matches the previous description hackberry provided)
https://[username]:[password]@cfd83a4879dc445d84dd990baf771358.us-west-2.aws.found.io:9243

here's what the elastic search url looks like, with no password.
https://wshs_collections:[password]@cfd83a4879dc445d84dd990baf771358.us-west-2.aws.found.io:9243

For PHP 7.3.29, FILTER_VALIDATE_URL is returning false. I'm guessing it should return true.

[2021-07-16 19:34 UTC] stas@php.net

```
-Assigned To: stas
+Assigned To: cmb
```

[2021-07-16 19:34 UTC] stas@php.net

@cmb could you check?

[2021-07-16 19:35 UTC] stas@php.net

I suspect a particular password may contain characters that are not allowed in the URL. That would make it an invalid URL.

[2021-07-16 20:58 UTC] cmb@php.net

> Elastic Search URL:
> https://collections_name:[45 character string].us-west-2.aws.found.io:9243
>
> Does this qualify as a valid URL?

No, because verbatim spaces are not allowed in an URL. Unless you
provide a verbatim URL, *nobody* can tell.

[2021-07-16 21:18 UTC] stas@php.net

Just for the sake of clarity, please do not post any of the actual passwords here, please generate a fake one that demonstrates the problem instead.

[2021-07-16 21:51 UTC] kfoubert at sitecrafting dot com

To: stas and cmb

here's what the elastic search url looks like, with no password (i made sure previous posts had no password). It's basic authentication.

https://[username]:[password]@cfd83a4879dc445d84dd990baf771358.us-west-2.aws.found.io:9243

The password contains a caret character and an exclamation mark, everything else is alphanumberic.

I also placed an issue for Elasticsearch-PHP for older versions, since their most recent releases don't use FILTER_VALIDATE_URL.

Thank you!

[2021-07-16 22:02 UTC] stas@php.net

Exclamation mark is fine, but caret is not:

```
 Other characters are excluded because gateways and other transport
   agents are known to sometimes modify such characters, or they are
   used as delimiters.

   unwise      = "{" | "}" | "|" | "\" | "^" | "[" | "]" | "`"
```

[2021-07-16 22:03 UTC] cmb@php.net

> The password contains a caret character and an exclamation mark,
> everything else is alphanumberic.

According to RFC 3986[1], a caret is not valid in the password
which is a part of the userinfo:

```
    userinfo      = *( unreserved / pct-encoded / sub-delims / ":" )
    unreserved    = ALPHA / DIGIT / "-" / "." / "_" / "~"
    sub-delims    = "!" / "$" / "&" / "'" / "(" / ")"
                  / "*" / "+" / "," / ";" / "="
```

The caret would need to be percent-encoded as %5E.

```
[1] <https://datatracker.ietf.org/doc/html/rfc3986>
```

[2021-11-09 16:07 UTC] cmb@php.net

```
Related To: Bug #81604
```