New issue

Jump to bottom

set_redirect & set_header are susceptible to http response splitting attack #425

Closed shouc opened this issue on Apr 10, 2020 · 4 comments

Labels bug

shouc commented on Apr 10, 2020

ref: https://owasp.org/www-community/attacks/HTTP_Response_Splitting

Analysis

```
// L2766
inline void Response::set_header(const char *key, const char *val) {
  headers.emplace(key, val);
}

inline void Response::set_header(const char *key, const std::string &val) {
  headers.emplace(key, val);
}

inline void Response::set_redirect(const char *url) {
  set_header("Location", url);
  status = 302;
}
...
// L3090
inline bool Server::write_response(Stream &strm, bool last_connection,
                                   const Request &req, Response &res) {
...
if (!detail::write_headers(bstrm, res, Headers())) { return false; }
...
// L1967
template <typename T>
inline ssize_t write_headers(Stream &strm, const T &info,
                             const Headers &headers) {
  ssize_t write_len = 0;
  for (const auto &x : info.headers) {
    ...
      // write_format never replaces/parses \r\n in x.first.c_str() or x.second.c_str()
      strm.write_format("%s: %s\r\n", x.first.c_str(), x.second.c_str());
    ...
  }
  for (const auto &x : headers) {
    ...
      // write_format never replaces/parses \r\n in x.first.c_str() or x.second.c_str()
      strm.write_format("%s: %s\r\n", x.first.c_str(), x.second.c_str());
    ...
}
```

PoC

```
#include "cpp-httplib/httplib.h"
using namespace httplib;
int main() {
    Server svr;
    svr.Get("/1", [](const Request& req, Response& res) {
        res.set_redirect("1\r\nSet-Cookie: a=1");
    });
    svr.Get("/2", [](const Request& req, Response& res) {
        res.set_header("a", "1\r\nSet-Cookie: a=1");
    });
    svr.listen("localhost", 3000);
}
```

Lastly, this library is gorgeous. Thank you!!

PixlRainbow commented on Apr 11, 2020 • edited Contributor

Fix: headers should allow only readable characters and not special codes like newlines etc.

yhirose added the bug label on Apr 11, 2020

yhirose commented on Apr 11, 2020 Owner

Great finding! I'll look into it.

yhirose commented on Apr 11, 2020 Owner

I researched some web frameworks to see how they handle this issue. So each framework takes a different way to handle it.

PHP: Treat it as a warning
https://github.com/php/php-src/blob/e1b57310b1ec15b24e5ead8b63e58f74a0dbc988/main/SAPI.c#L727-L742

Django: Raise `BadHeaderError` exception
https://github.com/django/django/blob/41a3b3d18647b258331104520e76f977406c590d/django/http/response.py#L117-L119

Rails: Just remove CRLF characters
https://github.com/rails/rails/blob/be3d9daaa39aa69603af16b602fe62ae47548469/actionpack/lib/action_controller/metal/redirecting.rb#L111

I am thinking to do this check in `set_header`, and let the function return immediately If the check detects a CR or a LF.

yhirose closed this as completed in `85327e1` on Apr 12, 2020

yhirose commented on Apr 13, 2020 • edited

Owner

@shouc, I fixed it and released v0.5.9. Could you report this issues has been resolved to the place to which you have submitted this as a vulnerability? Thanks a lot!

https://nvd.nist.gov/vuln/detail/CVE-2020-11709
https://gist.github.com/shouc/a9330df817128bc4c4132abf3de09495

yhirose added a commit that referenced this issue on May 1, 2020

`Fix #425` `1a68327`

Assignees

No one assigned

Labels

bug

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

3 participants