

aio-lib / aiohttp Public

<> Code Issues 185 Pull requests 89 Discussions Actions Security and quality 33

# Commit 53b35a2



Dreamsorcerer and dhiral authored last month · ✓ 35 / 39 · Verified

Restrict reason (#12209) (#12212)

(cherry picked from commit [1851048](#))

-----

Co-authored-by: Dhiral Vyas <dhiral@users.noreply.github.com>

🔑 master (#12212) · 📦 v3.13.5 v3.13.4

1 parent [db560cf](#) commit 53b35a2 📄

6 files changed +43 -8 lines changed

↑ Top ⚙️

🔍 Filter files...

- 📁 aiohttp
  - 📄 \_http\_writer.pyx
  - 📄 http\_writer.py
  - 📄 web\_exceptions.py
  - 📄 web\_response.py
- 📁 tests
  - 📄 test\_web\_exceptions.py
  - 📄 test\_web\_response.py

6 files changed +43 -8 lines changed

🔍 Search within code ⚙️

📄 aiohttp/\_http\_writer.pyx

```

...      @@ -131,7 +131,7 @@ def _serialize_headers(str status_line, headers):
131 131      _init_writer(&writer, buf)
132 132
133 133      try:
134 -      if _write_str(&writer, status_line) < 0:
134 +      if _write_str_raise_on_nlcr(&writer, status_line) < 0:
135 135          raise
136 136          if _write_byte(&writer, b'\r') < 0:

```

```
137 137 raise
```



▼ aiohttp/http\_writer.py



```
↑... @@ -361,6 +361,7 @@ def _safe_header(string: str) -> str:
```

```
361 361
```

```
362 362
```

```
363 363 def _py_serialize_headers(status_line: str, headers: "CIMultiDict[str]") -> bytes:
```

```
364 + _safe_header(status_line)
```

```
364 365 headers_gen = (_safe_header(k) + ": " + _safe_header(v) for k, v in headers.items())
```

```
365 366 line = status_line + "\r\n" + "\r\n".join(headers_gen) + "\r\n\r\n"
```

```
366 367 return line.encode("utf-8")
```



▼ aiohttp/web\_exceptions.py



```
↑... @@ -101,6 +101,8 @@ def __init__(
```

```
101 101 "body argument is deprecated for http web exceptions",
```

```
102 102 DeprecationWarning,
```

```
103 103 )
```

```
104 + if reason is not None and ("\r" in reason or "\n" in reason):
```

```
105 + raise ValueError("Reason cannot contain \r or \n")
```

```
104 106 Response.__init__(
```

```
105 107 self,
```

```
106 108 status=self.status_code,
```



▼ aiohttp/web\_response.py



```
↑... @@ -158,8 +158,8 @@ def _set_status(self, status: int, reason: Optional[str]) -> None:
```

```
158 158 self._status = int(status)
```

```
159 159 if reason is None:
```

```
160 160 reason = REASON_PHRASES.get(self._status, "")
```

```
161 - elif "\n" in reason:
```

```
162 - raise ValueError("Reason cannot contain \n")
```

```
161 + elif "\r" in reason or "\n" in reason:
```

```
162 + raise ValueError("Reason cannot contain \r or \n")
```

```
163 163 self._reason = reason
```

```
164 164
```

```
165 165 @property
```



▼ tests/test\_web\_exceptions.py



```
↑... @@ -273,5 +273,15 @@ def test_unicode_text_body_unauthorized() -> None:
```

```
273 273
```

```
274 274
```

```

275 275     def test_multiline_reason() -> None:
276 -     with pytest.raises(ValueError, match=r"Reason cannot contain \\n"):
276 +     with pytest.raises(ValueError, match=r"Reason cannot contain"):
277 277         web.HTTPOk(reason="Bad\r\nInjected-header: foo")
278 +
279 +
280 + def test_reason_with_cr() -> None:
281 +     with pytest.raises(ValueError, match=r"Reason cannot contain"):
282 +         web.HTTPOk(reason="OK\rSet-Cookie: evil=1")
283 +
284 +
285 + def test_reason_with_lf() -> None:
286 +     with pytest.raises(ValueError, match=r"Reason cannot contain"):
287 +         web.HTTPOk(reason="OK\nSet-Cookie: evil=1")

```

tests/test\_web\_response.py

```

@@ -13,7 +13,8 @@
13 13     from multidict import CIMultiDict, CIMultiDictProxy, MultiDict
14 14     from re_assert import Matches
15 15
16 - from aiohttp import HttpVersion, HttpVersion10, HttpVersion11, hdrs
16 + from aiohttp import HttpVersion, HttpVersion10, HttpVersion11, hdrs, web
17 + from aiohttp.abc import AbstractStreamWriter
17 18     from aiohttp.helpers import ETag
18 19     from aiohttp.http_writer import StreamWriter, _serialize_headers
19 20     from aiohttp.multipart import BodyPartReader, MultipartWriter
@@ -1008,6 +1009,27 @@ def test_set_status_with_empty_reason() -> None:
1008 1009         assert resp.reason == ""
1009 1010
1010 1011
1012 + def test_set_status_reason_with_cr() -> None:
1013 +     resp = web.StreamResponse()
1014 +
1015 +     with pytest.raises(ValueError, match="Reason cannot contain"):
1016 +         resp.set_status(200, "OK\rSet-Cookie: evil=1")
1017 +
1018 +
1019 + def test_set_status_reason_with_lf() -> None:
1020 +     resp = web.StreamResponse()
1021 +
1022 +     with pytest.raises(ValueError, match="Reason cannot contain"):
1023 +         resp.set_status(200, "OK\nSet-Cookie: evil=1")
1024 +
1025 +
1026 + def test_set_status_reason_with_crlf() -> None:

```

```

1027 +     resp = web.StreamResponse()
1028 +
1029 +     with pytest.raises(ValueError, match="Reason cannot contain"):
1030 +         resp.set_status(200, "OK\r\nSet-Cookie: evil=1")
1031 +
1032 +
1011 1033     async def test_start_force_close() -> None:
1012 1034         req = make_request("GET", "/")
1013 1035         resp = StreamResponse()
@@ -1308,9 +1330,9 @@ async def test_render_with_body(buf, writer) -> None:
1308 1330     )
1309 1331
1310 1332
1311 - async def test_multiline_reason(buf, writer) -> None:
1312 -     with pytest.raises(ValueError, match=r"Reason cannot contain \\n"):
1313 -         Response(reason="Bad\r\nInjected-header: foo")
1333 + async def test_multiline_reason(buf: bytearray, writer: AbstractStreamWriter) -> None:
1334 +     with pytest.raises(ValueError, match=r"Reason cannot contain \\r or \\n"):
1335 +         web.Response(reason="Bad\r\nInjected-header: foo")
1314 1336
1315 1337
1316 1338     async def test_send_set_cookie_header(buf, writer) -> None:

```

## Comments 0



Please [sign in](#) to comment.