EMERGING THREATS AND VULNERABILITIES

Understanding CVE-2025-29927: The Next.js Middleware Authorization Bypass Vulnerability

March 28, 2025

EMERGING VULNERABILITY THREAT DETECTION


Frederic Baguelin
Senior Security Researcher


Eslam Salem
Manager, Security Research


Emile-Hugo Spir
Senior Security Researcher


Matt Muir
Security Researcher

Adrian Korn
Manager, Threat Research

Understanding CVE-2025-29927: The Next.js middleware authorization bypass vulnerability

Key points and observations

- On March 21, researchers published an advisory for CVE-2025-29927, a vulnerability in Next.js middleware that allows authorization bypass through a specially crafted HTTP request that contains the internal header `x-middleware-subrequest`.
- Impacted Next.js versions include all releases earlier than 12.3.5 for 12.x, earlier than 13.5.9 for 13.x, earlier than 14.2.25 for 14.x, and earlier than 15.2.3 for 15.x.
- Exploitation allows attackers to skip critical middleware security checks, potentially exposing sensitive administrative routes and protected content.
- The vulnerability is straightforward to exploit, raising its severity and the immediacy of remediation efforts.

How to know if you're vulnerable

Your application is vulnerable under the following circumstances:

- Your deployment is self-hosted and uses the `next start` command in conjunction with the `output: standalone` configuration option (indicating that the Next.js application is using middleware).
- The application relies on middleware for critical tasks, such as authentication or security checks, and there is no validation of input later in the application.
- The version of Next.js that is running your application is earlier than 12.3.5 for 12.x, 13.5.9 for 13.x, 14.2.25 for 14.x, **or** 15.2.3 for 15.x.

You can verify your Next.js version by running:

ABOUT

You are not affected if your applications are hosted on Vercel, hosted on Netlify, or deployed as static exports (middleware not executed).

How to remediate affected applications

To remediate CVE-2025-29927, immediately:

1. **Upgrade Next.js** to versions 14.2.25, 15.2.3, or later:

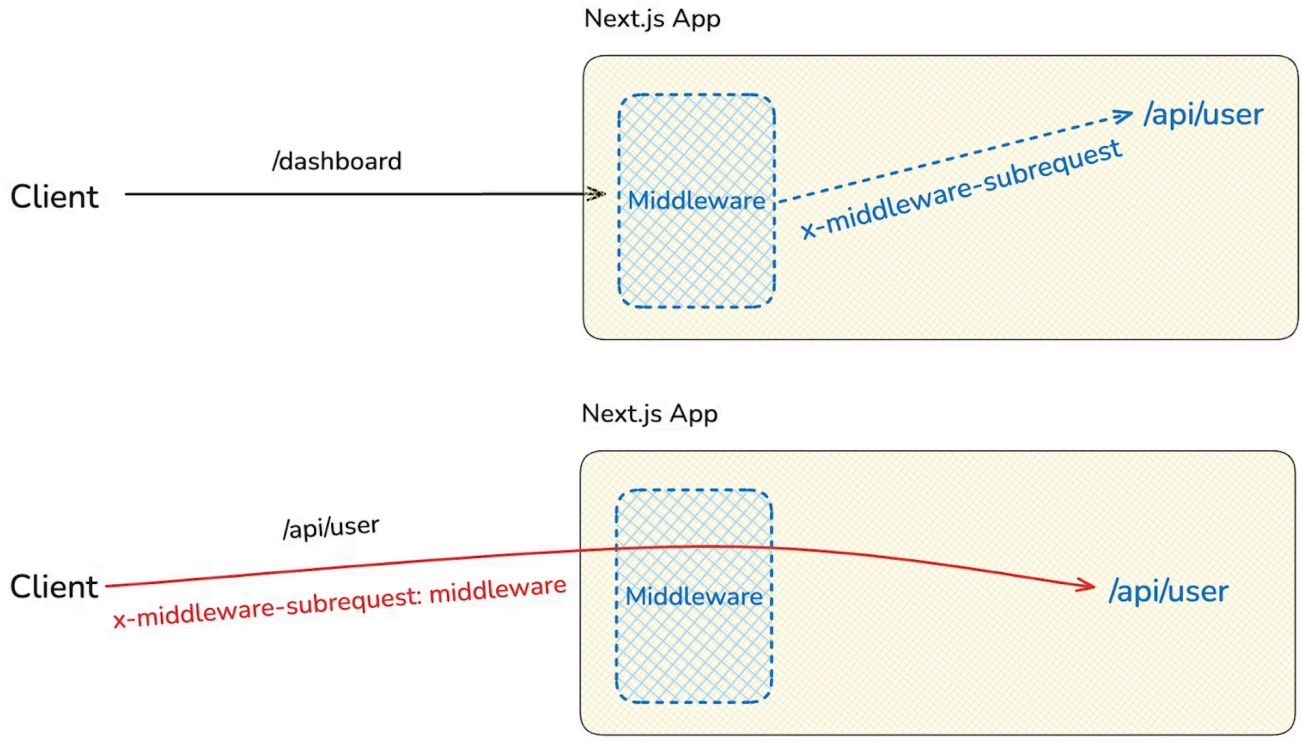
```
npm install next@latest

# or

yarn upgrade next@latest
```

2. **Temporary mitigation (if upgrading is not feasible)**: Configure your web server or proxy to drop or reject requests that contain the header `x-middleware-subrequest`.
3. **Review middleware logic**: Ensure that critical security checks are reinforced beyond middleware, adding redundancy to security layers.

Background on the vulnerability and exploitation method

Middleware in Next.js applications centralizes tasks such as authentication, logging, and security enforcement across all incoming requests. CVE-2025-29927 occurs due to improper handling of the `x-middleware-subrequest` header internally by Next.js.

The header `x-middleware-subrequest` is a special header used internally by Next.js to indicate that a request is an internal subrequest initiated by the middleware, rather than a direct request from a client. This header is crucial for internal Next.js features, allowing them to function properly. It also prevents the risk of infinite recursive loops by keeping track of any called middleware. However, this functionality inadvertently makes the authentication bypass possible because the presence of this header indicates that the request should bypass certain middleware logic.



CVE-2025-29927 execution flow (click to enlarge).

Exploitation method

A threat actor sends an HTTP request to the targeted Next.js application with the header `x-middleware-subrequest` and adds the path where the middleware is located as a value within the request (for example, `pages/_middleware`, `middleware`, `src/middleware`). The naming convention and middleware path possibilities depend on which Next.js version is running.

The Next.js internal logic then reads the header and completely bypasses the middleware, knowing that this request is an internal subrequest.

```
GET /admin HTTP/1.1
Host: vulnerable-site.com
x-middleware-subrequest:middleware
```

In Next.js version 15.x, the code was changed to prevent a recursive infinite loop. If there is middleware running on a path (for example, `/api/*`), then Next.js fetches another URL that also triggers the same middleware. Next.js checks for the `x-middleware-subrequest` header and calculates how many calls are made to this middleware. If the number of calls exceeds `MAX_RECURSION_DEPTH`, set to 5 by default, it will not call the middleware anymore.

The attacker can abuse this functionality by supplying the header with the maximum number of middleware calls to hit the `MAX_RECURSION_DEPTH` condition and bypass the middleware entirely.

Example:

```
GET /admin HTTP/1.1
Host: vulnerable-site.com
x-middleware-subrequest:middleware:middleware:middleware:middleware:middleware
```

The Datadog Security Research team has created a sample vulnerable application and exploitation proof of concept (POC) for this vulnerability.

In-the-wild observed exploitation

Currently, we are seeing a relatively low number of scans for this vulnerability in the wild. Of the observed scans, we've identified a number of payloads, most of which fall into the category of normal activity.

Normal activity

Some hosting providers (such as Vercel, which initially reported the vulnerability) store data in this header, which isn't meant to bypass middleware. In the case of Vercel, this data appears to be a request or session ID (40-character hex string). Those requests use the `Vercel Edge Functions` user agent. Moreover, middleware that performs recursive calls (the main use case of the header) uses the `Next.js Middleware` user agent by default. Those payloads tend to look like a parameterized path (`app/[service]/route`), but some don't contain the "parameter" `[service]` .

Exploitation

We're seeing straightforward payloads based on what's known about the vulnerability. For example, these payloads include `middleware` , `src/middleware` , `pages/_middleware` , and `middleware:middleware:middleware:middleware:middleware` .

The scanning is generally small in volume, ineffective, and poorly targeted. Either attackers are no longer running scanners, or the attackers haven't yet figured out how to effectively scan for it.

We have seen scanning or exploitation activity from a number of IP addresses. We only report IP addresses we have a high confidence for, and that we have seen in at least 5 distinct environments:

```
134.122.111.207
139.162.130.199
139.162.154.240
139.162.171.103
139.162.172.244
139.162.189.169
```

```
172.104.149.38
172.104.153.103
172.104.153.129
172.104.153.227
172.104.153.232
172.104.153.235
172.104.153.246
172.104.235.170
172.104.235.194
172.104.235.232
172.104.235.237
172.104.235.59
172.104.245.212
172.105.70.14
172.105.70.162
172.105.70.18
172.105.70.216
172.105.75.116
172.105.75.218
172.105.75.235
172.105.75.95
192.46.237.147
207.180.202.75
45.79.249.191
45.79.249.36
85.90.244.219
85.90.244.8
```

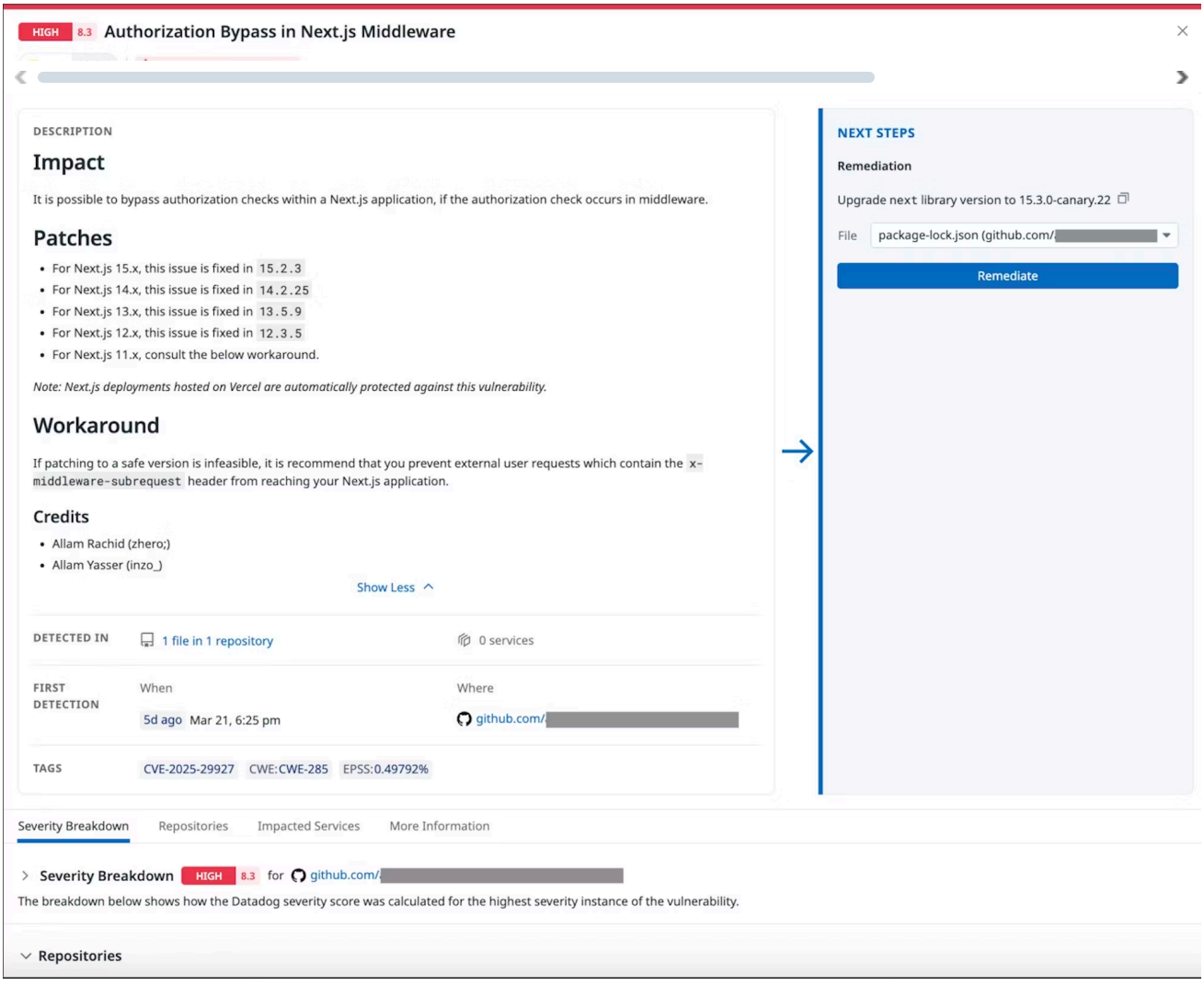
We have also seen scanning or exploitation activity from a number of User-Agents. We only report User-Agents we have a high confidence for, and that we have seen in at least 5 distinct environments:

```
Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:130.0) Gecko/20100101 Firefox/130.0
Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) C
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/112.0
Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:137.0) Gecko/20100101 Firefox/137.0
Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:93.0) Gecko/20100101 Firefox/93.0
Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/134.0.0.0
Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.
```

```
Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:86.0) Gecko/20100101 Firefox/86.0
python-requests/2.28.1
```

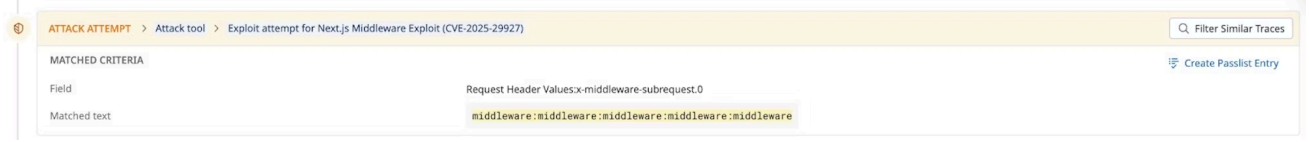
How Datadog can help

Datadog Code Security customers can use Code Security Vulnerabilities to identify any services that are vulnerable by running this query: `status:Open cve:CVE-2025-29927 is_publicly_accessible:"Accessible"`. We recommend that customers prioritize patching public-facing Next.js services.



(click to enlarge)

Datadog Application Security Management (ASM) customers can block exploitation of this vulnerability by using our WAF rule for Exploit attempt for Next.js Middleware Exploit (CVE-2025-29927).



New WAF rule (click to enlarge).

Conclusion

This Next.js middleware authorization bypass vulnerability highlights how crucial it is for developers to regularly update their dependencies and review their security architecture. Organizations should focus on patching CVE-2025-29927 for publicly exposed Next.js services to mitigate the risk.

References

- https://zhero-web-sec.github.io/research-and-things/nextjs-and-the-corrupt-middleware
- https://github.com/vercel/next.js/security/advisories/GHSA-f82v-jwr5-mffw
- https://www.nightvision.net/blog/next-js-middleware-bypass-cve-2025-29927-detection

Related Content

EMERGING THREATS AND VULNERABILITIES

Datadog threat roundup: top insights for Q4 2024

EMERGING THREATS AND VULNERABILITIES

Datadog threat roundup: Top insights for Q1 2025

EMERGING THREATS AND VULNERABILITIES

A guide to threat hunting and monitoring in Snowflake

EMERGING THREATS AND VULNERABILITIES

Attackers deploying new tactics in campaign targeting exposed Docker APIs

---