BLOG

Critical Security Vulnerability in React Server Components

December 3, 2025 by The React Team

There is an unauthenticated remote code execution vulnerability in React Server Components.

We recommend upgrading immediately.

On November 29th, Lachlan Davidson reported a security vulnerability in React that allows unauthenticated remote code execution by exploiting a flaw in how React decodes payloads sent to React Server Function endpoints.

Even if your app does not implement any React Server Function endpoints it may still be vulnerable if your app supports React Server Components.

This vulnerability was disclosed as CVE-2025-55182 and is rated CVSS 10.0.

The vulnerability is present in versions 19.0, 19.1.0, 19.1.1, and 19.2.0 of:

- react-server-dom-webpack
- react-server-dom-parcel
- react-server-dom-turbopack

Immediate Action Required

A fix was introduced in versions 19.0.1, 19.1.2, and 19.2.1. If you are using any of the above packages please upgrade to any of the fixed versions immediately.

If your app’s React code does not use a server, your app is not affected by this vulnerability. If your app does not use a framework, bundler, or bundler plugin that supports React Server Components, your app is not affected by this vulnerability.

Affected frameworks and bundlers

Some React frameworks and bundlers depended on, had peer dependencies for, or included the vulnerable React packages. The following React frameworks & bundlers are affected: next, react-router, waku, @parcel/rsc, @vitejs/plugin-rsc, and rwsdk.

See the update instructions below for how to upgrade to these patches.

Hosting Provider Mitigations

We have worked with a number of hosting providers to apply temporary mitigations.

You should not depend on these to secure your app, and still update immediately.

Vulnerability overview

React Server Functions allow a client to call a function on a server. React provides integration points and tools that frameworks and bundlers use to help React code run on both the client and the server. React translates requests on the client into HTTP requests which are forwarded to a server. On the server, React translates the HTTP request into a function call and returns the needed data to the client.

An unauthenticated attacker could craft a malicious HTTP request to any Server Function endpoint that, when deserialized by React, achieves remote code execution on the server. Further details of the vulnerability will be provided after the rollout of the fix is complete.

Update Instructions

Note

These instructions have been updated to include the new vulnerabilities:

- **Denial of Service - High Severity:** CVE-2025-55184 (CVSS 7.5)
- **Source Code Exposure - Medium Severity:** CVE-2025-55183 (CVSS 5.3)

They also include the additional case found, patched, and disclosed as CVE-2025-67779.

See the follow-up blog post for more info.

Next.js

All users should upgrade to the latest patched version in their release line:

```
npm install next@14.2.35  // for 13.3.x, 13.4.x, 13.5.x, 14.x
npm install next@15.0.7   // for 15.0.x
npm install next@15.1.11  // for 15.1.x
npm install next@15.2.8   // for 15.2.x
npm install next@15.3.8   // for 15.3.x
npm install next@15.4.10  // for 15.4.x
npm install next@15.5.9   // for 15.5.x
npm install next@16.0.10  // for 16.0.x

npm install next@15.6.0-canary.60   // for 15.x canary releases
npm install next@16.1.0-canary.19   // for 16.x canary releases
```

If you are on version `13.3` or later version of Next.js 13 (`13.3.x`, `13.4.x`, or `13.5.x`) please upgrade to version `14.2.35`.

If you are on `next@14.3.0-canary.77` or a later canary release, downgrade to the latest stable 14.x release:

```
npm install next@14
```

See the Next.js blog for the latest update instructions and the previous changelog for more info.

React Router

If you are using React Router's unstable RSC APIs, you should upgrade the following package.json dependencies if they exist:

```
npm install react@latest
npm install react-dom@latest
npm install react-server-dom-parcel@latest
npm install react-server-dom-webpack@latest
npm install @vitejs/plugin-rsc@latest
```

Expo

To learn more about mitigating, read the article on expo.dev/changelog.

Redwood SDK

Ensure you are on rwsdk>=1.0.0-alpha.0

For the latest beta version:

```
npm install rwsdk@latest
```

Upgrade to the latest `react-server-dom-webpack`:

```
npm install react@latest react-dom@latest react-server-dom-webpack@latest
```

See Redwood docs for more migration instructions.

Waku

Upgrade to the latest `react-server-dom-webpack`:

```
npm install react@latest react-dom@latest react-server-dom-webpack@latest waku@late
```

See Waku announcement for more migration instructions.

`@vitejs/plugin-rsc`

Upgrade to the latest RSC plugin:

```
npm install react@latest react-dom@latest @vitejs/plugin-rsc@latest
```

`react-server-dom-parcel`

Update to the latest version:

```
npm install react@latest react-dom@latest react-server-dom-parcel@latest
```

`react-server-dom-turbopack`

Update to the latest version:

```
npm install react@latest react-dom@latest react-server-dom-turbopack@latest
```

`react-server-dom-webpack`

Update to the latest version:

```
npm install react@latest react-dom@latest react-server-dom-webpack@latest
```

React Native

For React Native users not using a monorepo or `react-dom`, your `react` version should be pinned in your `package.json`, and there are no additional steps needed.

If you are using React Native in a monorepo, you should update *only* the impacted packages if they are installed:

- `react-server-dom-webpack`
- `react-server-dom-parcel`
- `react-server-dom-turbopack`

This is required to mitigate the security advisory, but you do not need to update `react` and `react-dom` so this will not cause the version mismatch error in React Native.

See this issue for more information.

Timeline

- **November 29th**: Lachlan Davidson reported the security vulnerability via Meta Bug Bounty.
- **November 30th**: Meta security researchers confirmed and began working with the React team on a fix.
- **December 1st**: A fix was created and the React team began working with affected hosting providers and open source projects to validate the fix, implement mitigations and roll out the fix
- **December 3rd**: The fix was published to npm and the publicly disclosed as CVE-2025-55182.

Attribution

Thank you to Lachlan Davidson for discovering, reporting, and working to help fix this vulnerability.

PREVIOUS
Denial of Service and Source Code Exposure in React Server Components

NEXT
React Conf 2025 Recap

Escape Hatches

Community

Code of Conduct

Meet the Team

Docs Contributors

Acknowledgements

More

Blog

React Native

Privacy

Terms