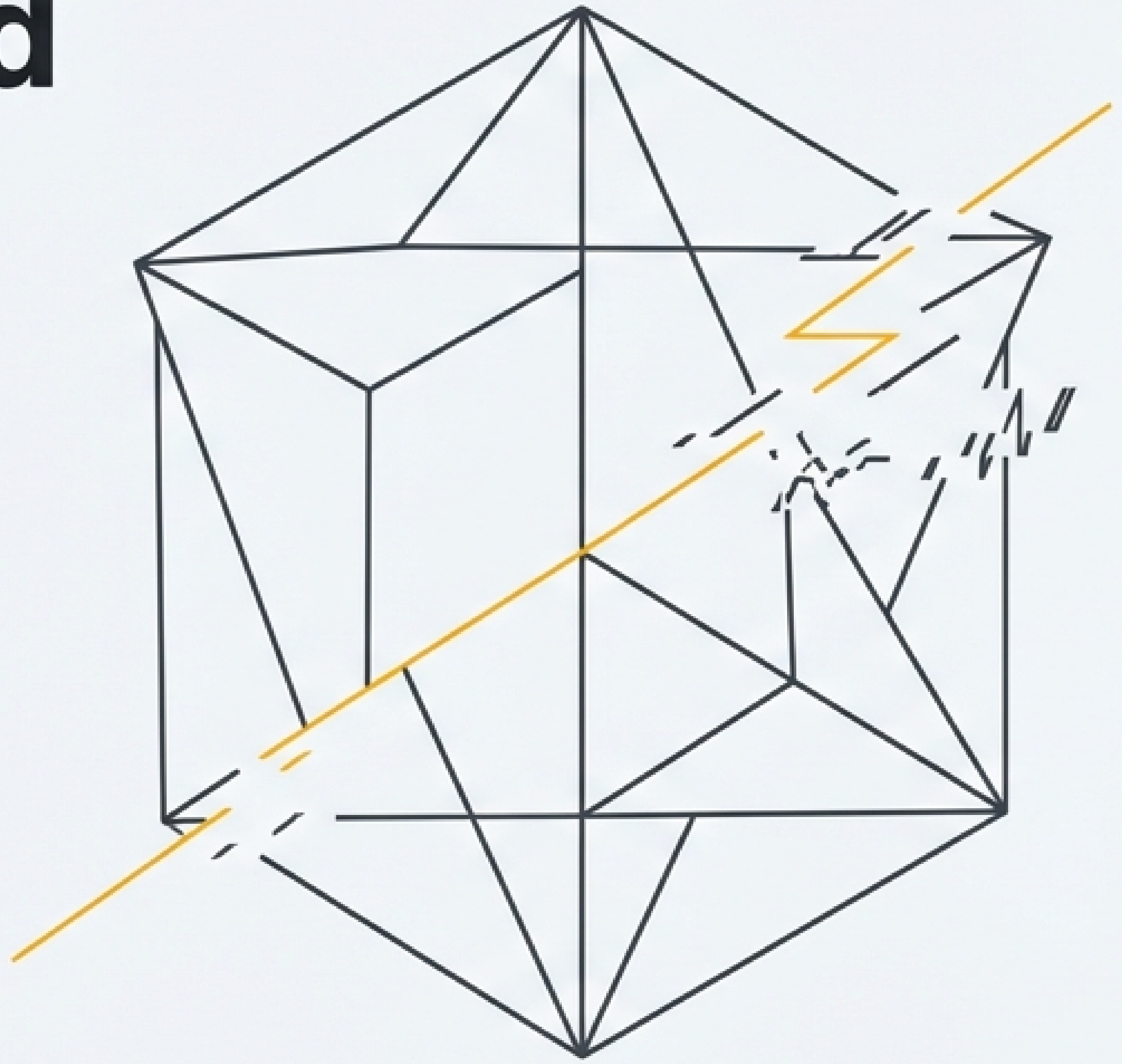
Critical Unauthenticated RCE in React Server Components

A Technical Briefing on CVE-2025-55182 & CVE-2025-66478



A **CVSS 10.0** vulnerability affecting React, Next.js, and the broader server-side React ecosystem.

NotebookLM

Executive Summary: Immediate Action Required

The Threat:
A critical (**CVSS 10.0**) unauthenticated Remote Code Execution (RCE) vulnerability exists in the React Server Components 'Flight' protocol.

The Scope:
Affects React versions 19.x and Next.js versions 14.3.0-canary.77+, 15.x, and 16.x using App Router. Default configurations are vulnerable.

> *"This is the kind of bug that turns 'web app' into 'server shell' with one request."*

The Impact:
Allows an unauthenticated attacker to execute arbitrary code on the application server. AWS/Wiz research suggests ~39% of cloud instances could be affected.

The Cause:
Insecure deserialization of attacker-controlled payloads sent to React Server Function endpoints.

The Action:
Upgrade affected packages immediately. This is the only definitive mitigation. Temporary WAF rules can provide defence-in-depth but do not replace patching.

The Attacker's Playbook: From RCE to Cloud Compromise

An unauthenticated attacker needs only to send a single, crafted HTTP request to compromise the server. Once RCE is achieved, the following actions are predictable:

1. Environment Discovery
Enumerate runtime, file system, environment variables, and cloud metadata endpoints.

2. Secret Hunting
Steal `.env` files, deployment secrets, CI tokens, service credentials, and signing keys.

3. Persistence
Deploy webshells, modify server-side code, or implant scheduled tasks.

4. Lateral Movement
Pivot to internal services, databases, message queues, and cloud APIs.

5. Supply Chain Attack
If build or deploy credentials are stolen, pivot upstream to compromise CI/CD pipelines.

If the runtime identity is over-privileged, this becomes a cloud compromise story, not a web app story.

A Widespread Vulnerability by Default

RSC Server DOM Package Distribution

Weekly Downloads

`react-server-dom-webpack`: 672,086 (~93%)

`react-server-dom-turbopack`: 43,421 (~6%)

`react-server-dom-parcel`: 7,170 (~1%)

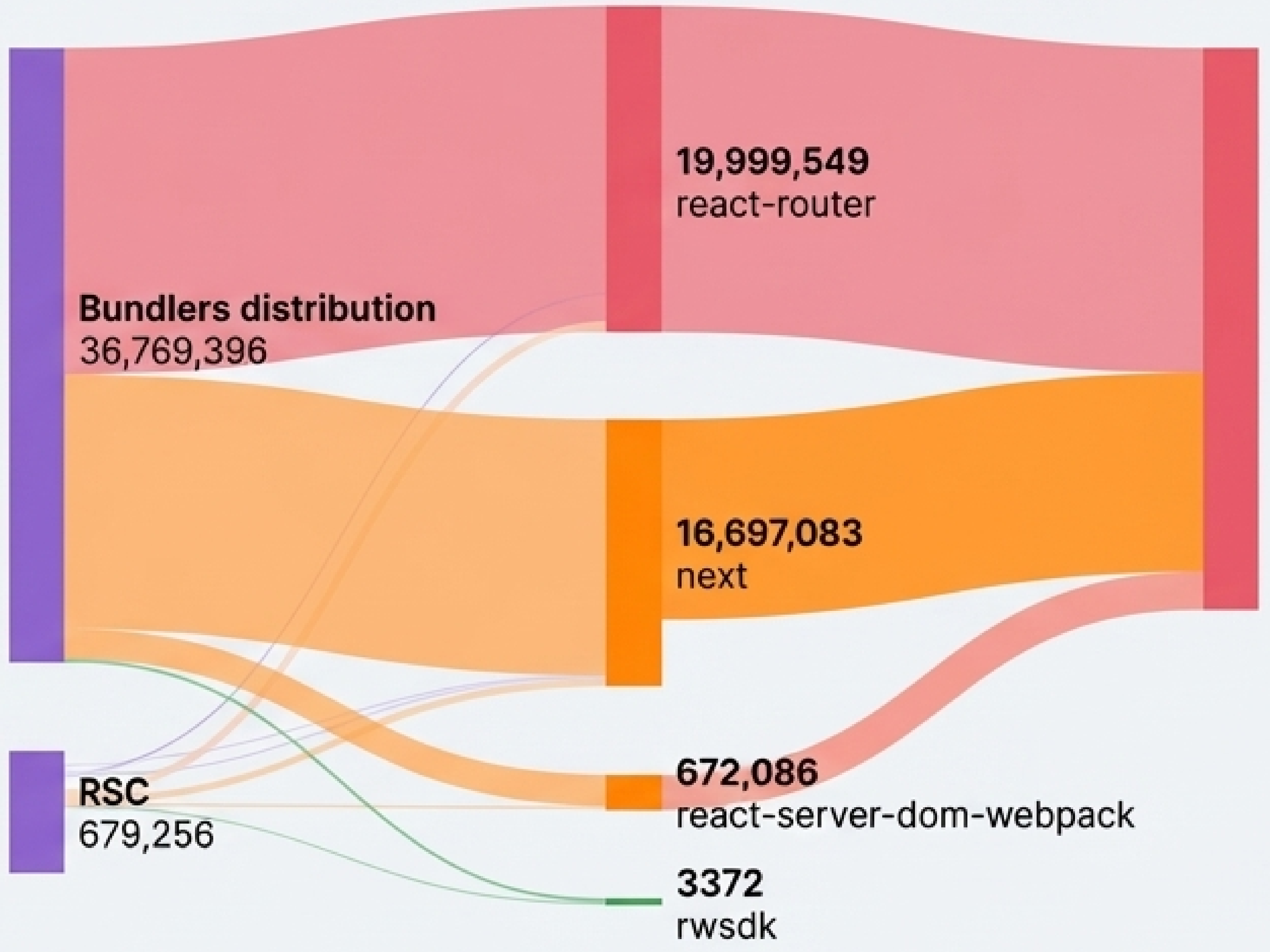
Total Weekly Downloads for Major Bundlers

Weekly Downloads

`Total:` 36,769,396

`react-router:` ~54%

`next:` ~45%



The vast majority of the React ecosystem pulls in RSC implementations transitively via frameworks like Next.js and React Router. Exposure often exists even if teams are not explicitly using Server Functions.

NotebookLM

Triage Checklist: Are You Affected?

Do you use server-side React with React Server Components (RSC)?

- If you use Next.js with the App Router, assume **YES**.
- If you use other RSC-enabled frameworks (RedwoodJS, Waku, etc.), assume **YES**.

Does your lockfile contain vulnerable package versions?

- Search `package-lock.json`, `yarn.lock`, etc., for the following:
  - Vulnerable React Packages: `react-server-dom-webpack`, `react-server-dom-parcel`, `react-server-dom-turbopack`
  - Vulnerable Versions: `19.0.0`, `19.1.0`, `19.1.1`, `19.2.0`

Are you using an affected version of Next.js?

- Check for these versions (with App Router enabled):
  - `14.3.0-canary.77` and later canary releases
  - All `15.x` versions prior to patch
  - All `16.x` versions prior to patch

The Fix: Upgrade All Affected Packages Immediately

Patching is the only definitive mitigation. Do not rely on temporary protections as a long-term solution.

For Next.js Users

Upgrade to the latest patched version in your release line.

```
# For 15.0.x
npm install next@15.0.5
# For 15.1.x
npm install next@15.1.9
# For 15.2.x
npm install next@15.2.1
# For 15.3.x
npm install next@15.3.3
# For 15.4.x
npm install next@15.4.1
# For 15.5.x
npm install next@15.5.7
# For 16.0.x
npm install next@16.0.7
```

***Warning:** If on `14.3.0-canary.77` or later, downgrade to the latest stable 14.x release: `npm install next@14`*

For General React & Other Frameworks (React Router, Waku, Vite, etc.)

Ensure you upgrade the underlying React packages.

```
npm install react@latest react-dom@latest
npm install react-server-dom-webpack@latest
# Or react-server-dom-parcel, react-server-dom-turbopack
```

Patched React Versions: 19.0.1, 19.1.2, 19.2.1

Defence-in-Depth: Temporary Mitigations While You Patch

Treat this as a seatbelt, not the airbag. You still need to patch.

Edge & Network Controls

- **WAF Rules:** Deploy vendor-specific WAF rules to detect and block exploitation attempts. *Details on the following slides.*
- **Endpoint Filtering:** If possible, block or strictly filter traffic to RSC / Server Function endpoints at your load balancer or API gateway.
- **Rate Limiting:** Implement rate limiting and bot-gating on affected endpoints to slow down attacker probing.

Infrastructure Hardening

- **Reduce Blast Radius:** Enforce least-privilege IAM roles for your application runtime. Ensure it cannot modify infrastructure or access secrets it doesn't need.
- **Egress Control:** Lock down outbound network traffic from application containers to prevent easy data exfiltration.

Providers like Vercel and Firebase have deployed network-layer protections. These help reduce exposure but are **not a substitute for upgrading**.

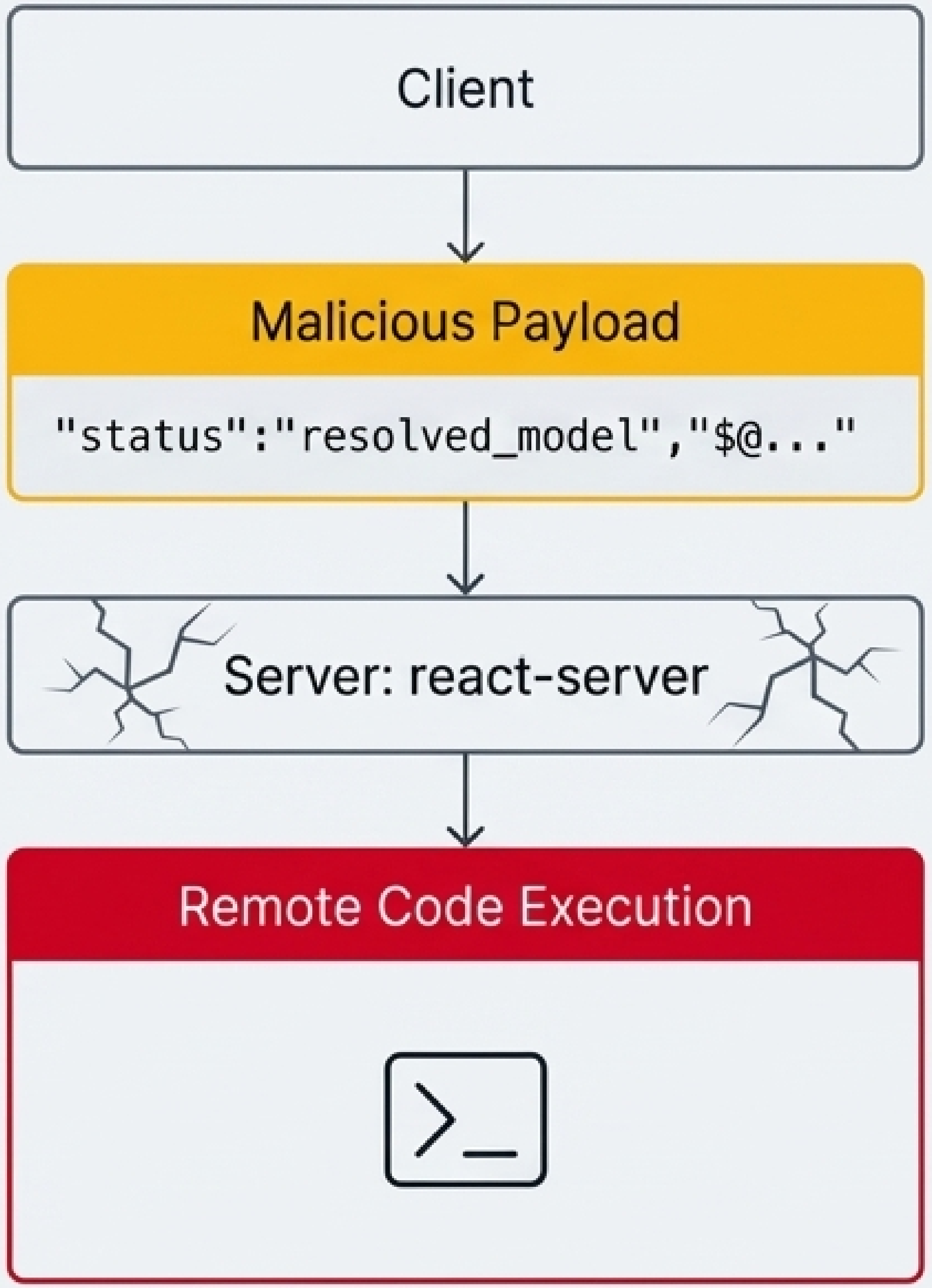
Technical Analysis: Insecure Deserialization in the 'Flight' Protocol

The vulnerability is a classic insecure deserialization flaw in the server-side logic that processes React Server Components payloads.

1. **Client-Server Interaction:** React Server Functions allow a client to call a function on the server. This communication is serialised using the 'Flight' protocol.
2. **Crafted Payload**: An unauthenticated attacker sends a malicious HTTP request containing a specially crafted Flight payload to any any Server Function endpoint.
3. **Deserialization Flaw**: The `react-server` package on the server receives the payload. Its decoding logic fails to properly validate the structure of the attacker's malformed data.
4. **Code Execution**: This validation failure allows attacker-controlled data to influence server-side execution logic, resulting in **Remote Code Execution** in the server's context.

Key Characteristics

Attack Vector: Remote, Network	Complexity: Low
Authentication: Not required	Affected Configuration: Default



Mitigation Recipe: Custom Rule for AWS WAF

Managed Rule Update:

⚠️ The default version (1.24) of AWSManagedRulesKnownBadInputsRuleSet now includes an updated rule for this issue. Ensure it is enabled.

Custom Rule for Defence-in-Depth:

As an interim protection, deploy the following custom rule. **We recommend testing in COUNT mode first before setting the action to BLOCK.**

```
{
  "Name": "ReactJSRCE_CUSTOM",
  "Priority": 99,
  "Statement": {
    "AndStatement": [
          { "RegexMatchStatement": { "RegexString": "POST", "FieldToMatch": { "Method": {} }, "TextTransformations": [{ "Priority": 0,
"Type": "NONE" }] } },
          { "RegexMatchStatement": { "RegexString": "(?i)(?:next-action|rsc-action-id)", "FieldToMatch": { "Headers": { "MatchPattern": {
"All": {} }, "MatchScope": "KEY", "OversizeHandling": "CONTINUE" } }, "TextTransformations": [{ "Priority": 0, "Type": "NONE" }] } },
          { "RegexMatchStatement": { "RegexString": "(?i)\"status\"\s*:\s*\"resolved_model\", "FieldToMatch": { "Body": {
 "OversizeHandling": "CONTINUE" } }, "TextTransformations": [{ "Priority": 0, "Type": "URL_DECODE_UNI" }, { "Priority": 1, "Type":
"JS_DECODE" }, { "Priority": 2, "Type": "UTF8_TO_UNICODE" }] } },
          { "RegexMatchStatement": { "RegexString": "\\$\@", "FieldToMatch": { "Body": { "OversizeHandling": "CONTINUE" }
  }, "TextTransformations": [{ "Priority": 0, "Type": "URL_DECODE_UNI" }, { "Priority": 1, "Type": "JS_DECODE" }, { "Priority": 2,
"Type": "UTF8_TO_UNICODE" }] } }
      ]
    }
  },
  "Action": { "Block": {} },
  "VisibilityConfig": { "SampledRequestsEnabled": true, "CloudWatchMetricsEnabled": true, "MetricName": "ReactJS_Custom" }
}
```

Mitigation Recipe: Preconfigured Rule for Google Cloud Armor

A new preconfigured WAF rule, `cve-canary`, is available to detect and block exploitation attempts. Deploy it as a temporary mitigation while patching.

First, deploy the rule in **preview mode** to monitor its impact on legitimate traffic without blocking. Once verified, disable preview mode to enforce the **block**.

Match Condition Expression

```
(has(request.headers['next-action']) ||
has(request.headers['rsc-action-id'])
|| request.headers['content-type'] ==
'multipart/form-data'
|| request.headers['content-type'] ==
'application/x-www-form-urlencoded'
) &&
evaluatePreconfiguredWaf('cve-
canary',{'sensitivity': 0,
'opt_in_rule_ids': ['google-mrs-
v202512-id000001-rce']})
```

gcloud CLI Command

```
gcloud compute security-policies
rules create PRIORITY_NUMBER \
  --security-policy
  SECURITY_POLICY_NAME \
  --expression "(...expression from
above...)" \
  --action=deny-403
```

Terraform Syntax

```
rule {
  action = "deny(403)"
  priority = "PRIORITY_NUMBER"
  match {
    expr {
      expression = "(...expression
from above...)"
    }
  }
  description = "Applies protection
for CVE-2025-55182 (React/Next.JS)"
}
```

A Coordinated Disclosure Timeline

November 29, 2025
Vulnerability responsibly disclosed by researcher Lachlan Davidson via Meta's Bug Bounty program.

November 30, 2025
Meta security researchers confirm the issue and begin working with the React team on a fix.

December 1, 2025
A fix is developed. Coordination begins with affected hosting providers and major open-source projects to validate the fix and prepare for rollout.

December 3, 2025
Patched versions are published to npm, and the vulnerability is publicly disclosed.

Reference: Comprehensive List of Affected Packages

React Packages (CVE-2025-55182)

- react-server-dom-webpack: 19.0.0, 19.1.0, 19.1.1, 19.2.0
- react-server-dom-parcel: 19.1.0, 19.1.1, 19.2.0
- react-server-dom-turbopack: 19.0.0, 19.1.0, 19.1.1, 19.2.0

**Includes all associated canary releases.*

Frameworks & Bundlers (Potentially Affected)

- Next.js (CVE-2025-66478)
  - 14.3.0-canary.77 and later canaries (with App Router)
  - All 15.x versions
  - All 16.x versions
- Others:
  - React Router (in RSC preview mode)
  - Waku
  - Parcel RSC plugin (@parcel/rsc)
  - Vite RSC plugin (@vitejs/plugin-rsc)
  - RedwoodJS (rwsdk)

Guidance for Security, DevOps, and Leadership Teams

Treat this as a security incident, not a routine upgrade.

- **Verify Reality:** “Merged the PR” is not remediation. Confirm that patched versions are running in production builds and deployed containers.
- **Prioritise Ruthlessly:** Focus on public-facing endpoints first, then internal applications reachable from untrusted networks. Map your exposure.
- **Hunt for Malice:** Review edge and application logs for suspicious spikes, malformed payloads, or anomalous outbound traffic from affected services, especially around the disclosure date (Dec 3).
- **Rotate Credentials:** If you suspect any service was exposed prior to patching, rotate all secrets, keys, and credentials accessible to that service’s runtime environment.
- **Improve Posture:** Use this event to validate your Application Security Posture Management (ASPM). Can you quickly answer “Where is this vulnerable component running, and who owns it?”

Key Identifiers & Official Resources

Vulnerability Information

- Primary CVE: CVE-2025-55182
- Next.js CVE: CVE-2025-66478 (Rejected as duplicate)
- CVSS Score: 10.0 (Critical)

Official Advisories

- React Team Security Advisory
- Next.js Security Notice
- AWS Security Bulletin: AWS-2025-030
- Google Cloud Security Advisory

Acknowledgements

This vulnerability was discovered and responsibly disclosed by Lachlan Davidson.

The rapid response was a coordinated effort between the researcher, Meta, the React team, and numerous hosting providers and open-source projects.