December 5, 2025

React2Shell, a CVSS 10.0 RCE Vulnerability in React Server Components (CVE-2025-55182)

Key Insights:

- CVE-2025-55182 (React2shell) is a maximum-severity (CVSS 10.0) remote code execution flaw affecting applications built with modern frameworks like React and Next.js.
- The exploit is triggered by an insecure deserialization flaw within React Server Components. It can be executed by an unauthenticated attacker sending a single malicious HTTP request, requiring no prior access or special permissions.
- A vast number of applications are potentially vulnerable due to the widespread global adoption of the React and Next.js ecosystems, creating an enormous attack surface for adversaries.
- The recommended first step for immediate, temporary protection is to deploy a web application firewall (WAF). Radware published signatures to block incoming exploit attempts through our Security Update Services (SUS).
- A widespread circulation of invalid proofs of concept (PoCs) creates a significant risk, as developers and security teams might incorrectly conclude their applications are safe based on unsuccessful PoC runs.

A critical remote code execution (RCE) vulnerability affecting the widely used web development frameworks React and Next.js was disclosed on December 3. The vulnerability, codenamed React2shell by its discoverer, Carapace Security Innovation Lead Lachlan Davidson, has been assigned CVE-2025-55182. The related Next.js vulnerability (CVE-2025-66478) was rejected as a duplicate and is now tracked under upstream CVE-2025-55182.

The vulnerability carries the maximum possible severity rating of CVSS 10.0, reflecting its critical nature and ease of exploitation. An attacker requires no authentication or special access. It only needs network access to a vulnerable application to send a crafted HTTP request and achieve remote code execution. The security community has responded rapidly, with Meta’s React team releasing patches that must be applied without delay. As an interim defense, major cloud and security providers have deployed protective web application firewall (WAF) rules to block exploit attempts while organizations work to deploy the permanent fix.

Background: Why This Vulnerability Has a Massive Attack Surface

React and Next.js are popular open-source frameworks used for building user interfaces and web applications. Their efficiency, flexibility and robust ecosystems have made them a default choice for developers worldwide, powering everything from mobile applications and simple websites to complex, enterprise-grade platforms. This widespread reliance means a single critical flaw can have cascading consequences for a significant portion of modern web infrastructure.

Data from across the security industry underscores the scale of the potential impact. Research from Palo Alto Networks Unit 42 has identified over 968,000 servers running these modern frameworks, highlighting a lucrative target for attackers. Furthermore, cloud security firm Wiz found that 39% of cloud environments contain instances vulnerable to these CVEs. This data paints a clear picture: a substantial number of applications across public and private clouds are immediately exploitable, necessitating urgent and widespread action.

Deconstructing the "React2shell" Exploit

The React2shell exploit is not a complex, multi-stage attack; it is a direct and powerful primitive that abuses the core functionality of React Server Components (RSC). The root cause of the vulnerability is a critical flaw in how the React Server Flight protocol handles serialized payloads sent to React Server Function endpoints. The exploit works when an unauthenticated attacker sends a specially crafted HTTP request to a vulnerable server. While processing this request, the server deserializes the malicious payload insecurely. This process is susceptible to prototype pollution, in which an attacker can tamper with JavaScript object prototypes, altering the application's behavior at runtime. When combined with specific execution paths in the React Server Components, this primitive can be escalated to achieve remote code execution on the server. The attack requires no login, special configuration, or prior access, making it trivial for an adversary to execute.

Invalid Proofs of Concept

While proof-of-concept (PoC) exploits have been published since the disclosure, Davidson, the discoverer, indicated that many of the circulating PoCs are invalid because they rely on the developer explicitly exposing dangerous functionality, a precondition that rarely exists in production. Common examples of invalid PoCs include vm#runInThisContext, child_process#exec, and fs#writeFile. These would only be exploitable if a developer had deliberately allowed clients to invoke them, which would be dangerous no matter what. The disclosed vulnerability does not have this constraint. Furthermore, in Next.js, the list of server functions is managed by the framework and does not include the aforementioned.

The widespread circulation of "fake" PoCs creates significant risks in itself, as developers might run a PoC that attempts to call, for example, child_process#exec, see it fail because Next.js doesn't bundle it by default, and incorrectly conclude that their application is safe. Furthermore, security teams may focus on blocking specific calls, such as vm or exec, rather than patching the underlying deserialization flaw, leaving them exposed to React2Shell.

Mitigation: A Multi-Layered Defense Strategy

Effective mitigation for CVE-2025-55182 requires a two-pronged approach: immediate, temporary containment to block active threats, followed by permanent remediation to eliminate the underlying vulnerability. Both steps are non-negotiable for achieving comprehensive protection.

WAFs as Immediate and Interim Defense

While patches are being tested and deployed, a WAF serves as a critical interim defense against real-time exploit attempts. A WAF can inspect incoming HTTP traffic and identify the malicious patterns associated with the React2shell exploit, preventing them from reaching the vulnerable application. Radware's Security Update Service provides customers with signatures to protect against CVE-2025-55182 and CVE-2025-66478 using our products and services.

Prioritize Patching as the Permanent Solution

The most effective and essential defense is to apply the security patches provided by the React team. This is the only way to permanently resolve the vulnerability at its source.

Affected React packages react-server-dom-parcel, react-server-dom-turbopack and react-server-dom-webpack in React versions 19.0.0, 19.1.0, 19.1.1 and 19.2.0 were fixed in React versions 19.0.1, 19.1.2, 19.2.1.

Affected Next.js versions include 15.x and 16.x using the App Router, and the issue was fixed in Next.js versions 15.0.5, 15.1.9, 15.2.6, 15.3.6, 15.4.8, 15.5.7 and 16.0.7.

The vulnerability also affects experimental canary releases starting with 14.3.0-canary.77. Users on any of the 14.3 canary builds should either downgrade to a 14.x stable release or 14.3.0-canary.76.

Are My Radware Products Affected?

Radware is evaluating the impact of this vulnerability on its own products while, at the same time, providing protection through our cyber defense products and services. This allows us to block malicious actors from exploiting the vulnerability.

Radware's security researchers are continuing to investigate the vulnerability and its impact and will update the guidance provided to customers as new information becomes available. Please make sure to check this advisory on the customer portal for ongoing updates on products and mitigation.

Summary and Conclusion: The Imperative for Immediate Action

CVE-2025-55182 poses a clear and present danger to a wide range of web applications. It is a critical, easily exploitable vulnerability with a massive footprint across the internet, driven by the popularity of the React and Next.js frameworks. The nature of this vulnerability, abusing core framework functionality in a widespread ecosystem, is indicative of an emerging class of threats that security teams must prepare to address in modern, component-based web architectures.

The path to remediation is equally clear and requires a two-step response: first, deploy updated WAF signatures for immediate, temporary protection against active exploits; second, test and apply the necessary vendor patches to fully and permanently eliminate the threat.

While the proliferation of "fake" PoCs, some of which were generated using the latest LLMs, may initially distract defenders or lead to false negatives in their exposure assessments, the availability of genuine exploit scripts is only a matter of when, not if. Once this happens, attackers can leverage AI to automate remote code execution, making unpatched systems immediate targets.

Given the maximum severity rating and the significant number of servers exposed, the window for action is expected to remain narrow. Security teams are strongly advised to act without delay to apply mitigations to protect their applications and infrastructure from compromise.

EFFECTIVE DDOS PROTECTION ESSENTIALS

Hybrid DDoS Protection – Use on-premises and cloud DDoS protection for real-time DDoS attack prevention that also addresses high-volume attacks and protects from pipe saturation

Behavioral-Based Detection – Quickly and accurately identify and block anomalies while allowing legitimate traffic through

Real-Time Signature Creation – Promptly protect against unknown threats and zero-day attacks

Web DDOS Tsunami Protection – Automated immediate detection and mitigation of Web DDOS encrypted high RPS and morphing attacks

A Cybersecurity Emergency Response Plan – Turn to a dedicated emergency team of experts who have experience with Internet of Things security and handling IoT outbreaks

Intelligence on Active Threat Actors – High fidelity, correlated and analyzed data for preemptive protection against currently active known attackers

For further network and application protection measures, Radware urges companies to inspect and patch their network to defend against risks and threats.

EFFECTIVE WEB APPLICATION SECURITY ESSENTIALS

Full OWASP Top-10 coverage against defacements, injections, etc.

Low false positive rate using negative and positive security models for maximum accuracy

Auto-policy generation capabilities for the widest coverage with the lowest operational effort

Bot protection and device fingerprinting capabilities to overcome dynamic IP attacks and achieve improved bot detection and blocking

Securing APIs by filtering paths, understanding XML and JSON schemas for enforcement, and using activity tracking mechanisms to trace bots and guard internal resources

Flexible deployment options including on-premises, out-of-path, virtual or cloud-based

LEARN MORE AT RADWARE'S SECURITY RESEARCH CENTER

To know more about today's attack vector landscape, understand the business impact of cyberattacks, or learn more about emerging attack types and tools, visit Radware's Security Research Center. Additionally, visit Radware's Quarterly DDoS & Application Threat Analysis Center for quarter-over-quarter analysis of DDoS and application attack activity based on data from Radware's cloud security services and threat intelligence.