PUBLIC SAMPLE / FICTIONAL CASE

Security review report

Clear priorities for leadership. Exact fixes for the engineering team.

EDITION 13 OCT 2026 / VERSION 1.0

Sample report showing what a CodeArc security review looks like. The target platform and evidence are fictional; the CVE advisories and remediation patterns are real.

Executive summary

Patch two critical framework flaws before anything else.

In this fictional case, unpatched Next.js middleware could let unauthenticated visitors reach private pages, and a vulnerable React Server Components build exposes the admin server to remote code execution. Upgrade both packages, check access close to the data in each route, and verify again.

2 critical findings	1 hardening item	3 findings total

EXAMPLE PLATFORM

Customer portal + operator admin

STATUS

Sample / fictional fixture

Inside this report

02 Scope, boundaries and risk register

03 to 05 Findings: business impact, evidence, fix and retest

06 Rollout plan and official advisories

PUBLIC SAMPLE / FICTIONAL CASE

Scope and risk register

Every review starts with a written scope: which systems are in bounds, what access was provided, and what we deliberately did not touch.

Example targets

Customer portal	`portal.demo.example` Next.js web app, route protection and browser headers
Operator admin	`admin.demo.example` React Server Components build and package inventory

What was reviewed

Package lockfiles provided by the owner, deployment and auth configuration, and live HTTP responses. Package findings come from the actual deployed build inventory rather than outside guessing.

Out of bounds

No attempts to guess passwords, destructive payloads, load tests, internal network pivoting, or customer data extraction. No live exploit was run against production.

Findings overview

ID	Finding	Priority	CVSS*	Page
CA-001	Private routes rely on vulnerable Next.js middleware	Critical	9.1	03
CA-002	Unpatched React Server Components allow server code execution	Critical	10.0	04
CA-003	Missing Content-Security-Policy header on portal	**Low**	Not scored	05

* Maintainer CVSS v3.1 base scores for the referenced CVEs. Base scores reflect theoretical severity; actual priority depends on how the app is hosted, how auth is layered, and what data the server holds.

How this differs from the free website check

Our free check of a public page inspects visible HTTPS and header settings on a single URL. Catching vulnerable server packages, auth bypasses, or business logic flaws requires an agreed review like this one.

PUBLIC SAMPLE / FICTIONAL CASE

SAMPLE FINDING CA-001

Private routes rely on vulnerable middleware

CRITICAL / PATCH IMMEDIATELY

CVE-2025-29927 / CWE-285 / Next.js Middleware

9.1 / 10 Maintainer CVSS v3.1 base score

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

What this means for the business

When route protection relies solely on affected Next.js middleware, an outside request can skip that check and reach pages meant for authenticated users. Whether private data is exposed depends on whether data queries check the session again.

Sample evidence

Fictional excerpt illustrating how we record the exact build, route, and configuration behind each finding.

```
target: https://portal.demo.example/account/invoices (fictional)
package: next@14.2.24 / self-hosted Node.js runtime
guard: src/middleware.ts -> matcher: ['/account/:path*']
handler: getInvoices() trusts middleware without re-checking session
status: affected build + single-layer guard in sample fixture
```

How to fix it

Upgrade Next.js to a patched release per the maintainer advisory. Do not use middleware as your only security gate: verify the user's session and permissions directly inside every data fetch, server action, and API route.

How to verify the fix

Confirm the deployed Next.js version in production. Request protected routes unauthenticated and across user roles (including crafted internal middleware headers) to confirm unauthorized requests fail closed.

Scope & assumptions

Sample finding based on a fictional Node.js deployment managed directly by the application team. In real assessments, applicability depends on hosting mode and whether downstream handlers enforce their own auth checks.

RETEST STATUS: PENDING FIX / SAMPLE

CVE-2025-29927 / Official advisory

CWE-285

PUBLIC SAMPLE / FICTIONAL CASE

SAMPLE FINDING CA-002

Server component build allows code execution

CRITICAL / PATCH IMMEDIATELY

CVE-2025-55182 / CWE-502 / React Server Components

10.0 / 10 Maintainer CVSS v3.1 base score

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

What this means for the business

On an affected React Server Components deployment, an unauthenticated attacker can execute arbitrary code on the application server, potentially exposing server secrets, connected databases, and internal APIs.

Sample evidence

Fictional excerpt illustrating how we record the exact build, route, and configuration behind each finding.

```
target: https://admin.demo.example (fictional)
package: react-server-dom-webpack@19.1.1 (RSC enabled)
advisory: GHSA-fv66-9v8q-g76r / CVE-2025-55182
surface: public Flight endpoint reachable without sign-in
status: vulnerable version in sample lockfile (no live exploit run)
```

How to fix it

Upgrade react-server-dom-* and the hosting framework to the patched release immediately, then rebuild and redeploy every running container. WAF rules can buy time but do not replace patching the runtime.

How to verify the fix

Inspect the deployed lockfile and running container image to confirm patched package versions across all instances, then verify normal Server Actions and page transitions still work cleanly.

Scope & assumptions

Identified from the sample package inventory without executing an exploit payload. Apps that render React purely in the browser without Server Components are not affected by this CVE.

RETEST STATUS: PENDING FIX / SAMPLE

CVE-2025-55182 / Official advisory

CWE-502

PUBLIC SAMPLE / FICTIONAL CASE

SAMPLE FINDING CA-003

Portal pages ship without a content security policy

LOW / DEFENSE IN DEPTH

CA-003 / Content-Security-Policy / browser hardening

Hardening observation; no CVE or CVSS score.

What this means for the business

A missing Content-Security-Policy (CSP) is not a vulnerability on its own, but it leaves the browser without a safety net if an injection flaw or rogue external script ever slips into the portal.

Sample evidence

Fictional excerpt illustrating how we record the exact build, route, and configuration behind each finding.

```
target: GET https://portal.demo.example/ HTTP/1.1 (fictional)
status: 200 OK / Content-Type: text/html; charset=utf-8
Strict-Transport-Security: max-age=63072000; includeSubDomains
X-Content-Type-Options: nosniff
Content-Security-Policy: <missing>
```

How to fix it

Draft a CSP tailored to the scripts, styles, API origins, and frame ancestors the portal actually uses. Roll it out in Content-Security-Policy-Report-Only mode first, check for false positives, then enforce it.

How to verify the fix

Fetch representative HTML pages and redirects to confirm the enforced header is present, and run through authentication, checkout, and forms to verify no legitimate scripts or assets are blocked.

Scope & assumptions

Observed on a single sample response. A working CSP must be tuned to the application's real script and payment integrations rather than copied from a generic template.

RETEST STATUS: PENDING FIX / SAMPLE

Content-Security-Policy / Official advisory

PUBLIC SAMPLE / FICTIONAL CASE

Fix, verify, close

Assign one owner per finding and close an item only when the fix is deployed and verified in the target environment.

01 **Confirm live exposure**

Verify the deployed package versions, hosting mode, and route auth checks for CA-001 and CA-002.

02 **Patch and redeploy**

Upgrade Next.js and React Server Components packages, rebuild images, and deploy across all environments.

03 **Roll out browser hardening**

Ship the portal's Content-Security-Policy in a mode that reports violations without blocking them, verify clean checkout and authentication flows, then enforce.

04 **Retest and sign off**

Run the verification steps for each finding and record the build hash, test date, and closure status.

Because this is a fictional sample report, all three items remain open examples. No real system was tested and no financial loss is estimated.

Official advisories & standards (checked 13 Oct 2026)

Next.js security advisory / CVE-2025-29927

React security advisory / CVE-2025-55182

React blog / RSC vulnerability & upgrade guide

OWASP / Secure Headers Project

MDN / Content-Security-Policy guide

FIRST / CVSS v3.1 specification

OWASP / Web Security Testing Guide

Want a report like this for your system?

Tell us what you want tested. We agree the exact scope and price in writing before we start.

codearc.io/security info@codearc.io