Engineer

Hardened the application with nonce-based CSP

Hardened the application with nonce-based CSP, HSTS, SameSite cookies, least-privilege database roles and server-side entitlement re-checks.

2025

defence that holds when the browser is hostile

Situation. The platform holds professional and organizational data, the kind people expect to be handled properly, so a single line of defence was never going to be enough. The working assumption has to be that the client is hostile — that anything the browser enforces can be switched off by whoever's holding the browser — and the security has to hold up anyway.

Task. The platform had to be hardened at every layer — frontend, API, database — so that security was enforced by the server independently of whatever the interface happened to allow.

Action. On the frontend, the Next.js proxy middleware — proxy.ts — sets a Content-Security-Policy with a per-request nonce and strict-dynamic, plus HSTS and SameSite cookies, so the browser is locked down about what it'll run and send. At the API, there's rate limiting, CORS, request-size limits, input validation before anything touches the database, and logging of the security-relevant events. In the database, the API logs in as a least-privilege role that can only EXECUTE the app functions, the functions run SECURITY DEFINER, and everything is parameterised. And the entitlements — tier, role, organization, ship scoping — are re-checked on the server on every request, with the frontend gates treated as UX only. The gates decide what you see; the server decides what you can do.

Result. Security doesn't depend on the UI behaving. The protections are layered to avoid relying on a single control. Server-side authorization is designed to operate independently of client-side checks, so the interface is not the authority for access decisions.

Bring this experience to your project

Use “Hardened the application with nonce-based CSP” as a starting point for a similar challenge in your organisation. The services behind this work are available for your project. The part users never see: where the data lives, what the rules are, and how other systems are allowed to talk to it.

Explore our services Get in touch →

Look behind the result

Read the situation, decisions and result in this example. We can discuss which parts transfer to your constraints and which need another approach. Hardened the application with nonce-based CSP — defence that holds when the browser is hostile

Explore our services Get in touch →

Is your application actually secure?

Talk through your project with the engineer responsible for delivery. We agree the scope, price basis and handover in writing before work begins.

Role context: Security Engineer. It draws on Platform Architecture, Backend Engineering, Frontend Engineering, APIs & Integration, Databases and Security. And while you are here, we are rather good at Backend & API Development and Security & Access Management.

Get in touch Explore our services

One piece of **a longer record** — and it is all on this site.

Every entry is written the same way: the situation, the task, what we did and what changed.

See the whole record

Define the improvement you need

Describe the workflow, the existing interfaces and the users who need the change. We can define a testable delivery and a handover that lets your team maintain it.

See what it costs Get in touch →

Is your application actually secure?

Ask about the decisions behind “Hardened the application with nonce-based CSP”. You will speak with the people who can explain the work and assess your own project.

Explore our services

welcome@engineer.company +45 50 10 61 84 Copenhagen, Denmark

Take this page with you

Contact card

Carefully Crafted With Love by D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – present · Engineer ApS · All rights reserved.

Printed from https://engineer.company/portfolio/hardened-the-application-with-nonce-based-csp/