Engineer

Har hærdet applikationen med nonce-baseret CSP

Har hærdet applikationen med nonce-baseret CSP, HSTS, SameSite-cookies, least-privilege-databaseroller og fornyet kontrol af rettigheder på serveren.

2025

forsvar, der holder, når browseren er fjendtlig

Situation. Platformen rummer data om fagpersoner og organisationer, den slags, folk forventer bliver håndteret ordentligt, så en enkelt forsvarslinje ville aldrig være nok. Arbejdsantagelsen må være, at klienten er fjendtlig — at alt, hvad browseren håndhæver, kan slås fra af den, der sidder ved browseren — og sikkerheden skal holde alligevel.

Opgave. Platformen skulle hærdes i hvert eneste lag — frontend, API, database — så sikkerheden blev håndhævet af serveren, uafhængigt af hvad brugerfladen nu tilfældigvis tillod.

Handling. I frontenden sætter Next.js' middleware — `proxy.ts` — en Content-Security-Policy med en ny nonce for hver request og strict-dynamic, plus HSTS og SameSite-cookies, så browseren er stramt begrænset i, hvad den vil køre og sende. I API'et er der rate limiting, CORS, grænser for request-størrelse, inputvalidering, før noget når databasen, og logning af de sikkerhedsrelevante hændelser. I databasen logger API'et ind som en least-privilege-rolle, der kun kan EXECUTE app-funktionerne, funktionerne kører som SECURITY DEFINER, og alt er parametriseret. Og rettighederne — tier, rolle, organisation, afgrænsning til skibe — kontrolleres igen på serveren ved hver request, mens kontrollerne i frontenden kun betragtes som UX. De afgør, hvad man ser; serveren afgør, hvad man må.

Resultat. Sikkerheden afhænger ikke af, at brugerfladen opfører sig pænt. Beskyttelserne ligger i lag for at undgå afhængighed af en enkelt kontrol. Autorisationen på serveren er udformet til at fungere uafhængigt af kontrollerne på klienten, så brugerfladen ikke afgør, hvem der har adgang.

Brug erfaringen i jeres projekt

Brug »Har hærdet applikationen med nonce-baseret CSP« som udgangspunkt for en lignende udfordring i jeres organisation. Ydelserne bag arbejdet kan også bruges i jeres projekt. Den del, brugerne aldrig ser: hvor data ligger, hvilke regler der gælder, og hvordan andre systemer må tale med den.

Se vores ydelser Kontakt os →

Se bag resultatet

Læs om situationen, beslutningerne og resultatet i eksemplet. Vi kan drøfte, hvad der passer til jeres vilkår, og hvad der kræver en anden tilgang. Har hærdet applikationen med nonce-baseret CSP — forsvar, der holder, når browseren er fjendtlig

Se vores ydelser Kontakt os →

Er din applikation egentlig sikker?

Tal om jeres projekt med den ingeniør, der har ansvaret for leverancen. Vi aftaler omfang, prisgrundlag og overdragelse skriftligt, før arbejdet begynder.

Rolle i opgaven: Sikkerhedsingeniør. Det trækker på Platformarkitektur, Backend-udvikling, Frontend-udvikling, API'er & integration, Databaser og Sikkerhed. Og når I alligevel er her, er vi ret dygtige til Backend- & API-udvikling og Sikkerhed & adgangsstyring.

Kontakt os Se vores ydelser

Ét stykke af **en længere fortegnelse** — og det hele ligger her på sitet.

Hver post er skrevet på samme måde: situationen, opgaven, hvad vi gjorde, og hvad der ændrede sig.

Se hele fortegnelsen

Definér den ønskede forbedring

Beskriv arbejdsgangen, de eksisterende grænseflader og de brugere, der har brug for ændringen. Vi kan definere en testbar leverance og en overdragelse, der gør teamet i stand til at vedligeholde løsningen.

Se, hvad det koster Kontakt os →

Er din applikation egentlig sikker?

Spørg til beslutningerne bag »Har hærdet applikationen med nonce-baseret CSP«. I taler med mennesker, der kan forklare arbejdet og vurdere jeres eget projekt.

Se vores ydelser

welcome@engineer.company +45 50 10 61 84 København, Danmark

Tag denne side med dig

Visitkort

Lavet med kærlighed af D.

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

Copyright © 2025 – nu · Engineer ApS · Alle rettigheder forbeholdes.

Udskrevet fra https://engineer.company/da/portfolio/hardened-the-application-with-nonce-based-csp/