Engineer

HSTS

en browsers løfte om at bruge HTTPS næste gang

Hvad det er

HSTS står for HTTP Strict Transport Security og er en svarheader, hvor et website beder browseren om kun at kontakte det over HTTPS i et angivet antal sekunder og om at afbryde forbindelsen helt, hvis certifikatet ikke holder. Selve headeren hedder `Strict-Transport-Security`, og den har været defineret i RFC 6797 siden november 2012. HSTS krypterer intet selv; det ændrer, hvad browseren vil gøre over HTTPS.

En typisk værdi har to direktiver:

```
Strict-Transport-Security: max-age=31536000; includeSubDomains
```

`max-age` er, hvor længe browseren husker reglen, her et år. `includeSubDomains` lader reglen gælde for alle navne under værten, så `api.example.com` og `gammel.example.com` arver det, `example.com` bad om. Mange konfigurationer har også et tredje ord, `preload`. Det stammer ikke fra RFC'en, men fra preload-listen på hstspreload.org, og det er sitets samtykke til at få sit navn bygget ind i browsere.

Når en browser har set headeren på et sikkert svar, gør den tre ting for den vært, indtil tiden løber ud. Den skriver enhver `http://`-adresse om til `https://`, før en forespørgsel forlader maskinen. Den behandler enhver certifikatfejl som endelig, uden en knap til at fortsætte alligevel. Og den starter uret forfra, hver gang headeren kommer igen. Et svar med `max-age=0` får den til at glemme værten.

HSTS sammenlignes oftest med en omdirigering fra port 80 til port 443. Den virker også, men først efter at browseren har sendt sin første forespørgsel ukrypteret, og alle på samme netværk kan selv besvare den forespørgsel og holde den besøgende fast på en ukrypteret kopi af sitet. HSTS fjerner den ukrypterede forespørgsel fra alle besøg efter det første. Naboen på den anden side er Upgrade Insecure Requests, et direktiv i Content Security Policy. Det opgraderer de forespørgsler, en side selv sender efter scripts og billeder, men en besøgende, der kommer via et `http://` -link fra et andet site, er stadig overladt til omdirigeringen.

Brug begrebet i en beslutning

Brug »Hvad er HSTS?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Konti, adgangskoder og rettigheder sat op, så folk har præcis den adgang, jobbet kræver, og fratrådte mister den samme dag.

Se vores ydelser Kontakt os →

Undersøg antagelserne bag ordet

Brug forklaringen og det tilknyttede arbejde til at formulere spørgsmål til jeres projekt. Vi kan undersøge antagelserne sammen, før et værktøj vælges. Har skrevet 1.000 RBAC-regler til Linux-servere — 1,000 adgangsregler skrevet for at lukke et sikkerhedshul

Har skrevet 1.000 RBAC-regler til Linux-servere Kontakt os →

Hvorfor det betyder noget

At slå HSTS til koster en linje konfiguration. Den egentlige beslutning er, hvor langt man går, for hvert trin beskytter flere besøgende og er sværere at trække tilbage.

Headeren alene efterlader et hul, som RFC 6797 selv peger på. En browser, der aldrig har set sitet, eller hvis huskede politik er udløbet, begynder stadig med en ukrypteret forespørgsel. En lang `max-age` gør den sidste del af hullet mindre, og derfor er et år den almindelige indstilling. Den første del lukkes kun af preload-listen: værtsnavne, der er kompileret ind i Chrome, og som Firefox, Safari og Edge bygger deres egne lister på, så en nyinstalleret browser allerede ved, at den skal bruge HTTPS.

Listen stiller store krav. Basisdomænet skal have et gyldigt certifikat, omdirigere fra HTTP til HTTPS på samme vært, servere alle underdomæner over HTTPS og sende en header med en `max-age` på mindst `31536000` sekunder, `includeSubDomains` og `preload` . Vedligeholderne skriver ligeud, at det også gælder interne underdomæner, som offentligheden aldrig ser. Det er udgangen,

der afgør sagen: en fjernelse tager måneder om at nå brugerne via Chrome-opdateringer, for de andre browsere gives der ingen garanti, og sitet kalder processen »langsom og smertefuld«.

Beslutningen er derfor en stige, der tages ét trin ad gangen:

- **Headeren uden** `includeSubDomains` egner sig til næsten ethvert site, der allerede kører HTTPS. RFC 9325, den gældende bedste praksis for TLS, siger, at webservere bør sende den.
- `includeSubDomains` er sikkert, når nogen har listet alle navne under domænet og bekræftet, at hvert af dem svarer på HTTPS, også dem, en leverandør eller en anden afdeling driver.
- **Preload** hører hjemme på et domæne, hvor ejerne er sikre på, at intet navn under det får brug for ren HTTP igen: et login-domæne, et betalingsdomæne, en virksomhed, der selv styrer hele sin DNS. Et marketingdomæne, hvor et kampagneværktøj snart kan bede om en CNAME, er en dårlig kandidat, og det samme er ethvert domæne, der deles med kontorudstyr.

Vedligeholderne angiver også tempoet: `max-age` på 300 sekunder, derefter en uge og så en måned, hver gang med `includeSubDomains`, og den fulde levetid for ét trin skal udløbe, før det næste begynder. En browser beholder den værdi, den sidst så, så en fejl på femminuttertrinnet koster fem minutter.

Sådan ser det ud i praksis

På en platform med professionelle og organisatoriske data arbejdede virksomheden som sikkerhedsingeniør og hærdede hvert lag, og HSTS var et af dem. I Next.js' middleware, `proxy.ts`, får svaret HSTS sammen med en Content Security Policy med en ny nonce for hver forespørgsel og `strict-dynamic` samt SameSite-cookies. En middleware kører foran de ruter, den dækker, så reglerne for browseren ligger i én fil i stedet for at være spredt ud over siderne.

Samme stykke arbejde viser, hvor HSTS holder op. Arbejdsantagelsen var, at klienten er fjendtlig: alt, hvad browseren håndhæver, kan slås fra af den, der sidder med browseren. HSTS beskytter en ærlig besøgende mod et fjendtligt netværk. Mod en fjendtlig klient gør det intet, og derfor fortsatte arbejdet længere nede. API'et fik rate limiting, CORS, grænser for forespørgslernes størrelse, inputvalidering, før noget nåede databasen, og logning af sikkerhedsrelevante hændelser. API'et loggede ind i databasen som en rolle, der kun kunne udføre applikationens funktioner, og rettighederne blev kontrolleret

igen på serveren ved hver forespørgsel, mens frontendens egne spærringer kun tjente brugeroplevelsen. I den stak holder HSTS selve trafikken uden for rækkevidde, og det er det billigste lag at tilføje.

Hvor det går galt

Headeren står i konfigurationen, og browserne opfører sig, som om den var væk. Den bliver sendt på det ukrypterede HTTP-svar, typisk omdirigeringen, eller på et HTTPS-svar, hvis certifikat browseren afviste. RFC 6797 beder browsere ignorere den i begge tilfælde og beder servere om helt at lade være med at sende den over ren HTTP. En beslægtet fejl sender `example.com` direkte videre til `https://www.example.com` med headeren kun på `www`, så basisdomænet og dets øvrige navne står udækket. Sæt headeren på alle HTTPS-svar, også omdirigeringerne, og tjek begge døre:

```
curl -sI https://example.com/ | grep -i strict-transport-security
curl -sI http://example.com/ | grep -i strict-transport-security
```

Første linje skal udskrive headeren. Den anden hører til omdirigeringen og skal slet intet udskrive.

En gammel tjeneste under domænet holder op med at åbne for alle, der har besøgt hovedsitet. `includeSubDomains` på basisdomænet dækker hvert navn under det: en statusside hos en leverandør, en printers administrationsside, et intranetværktøj, der aldrig har haft et certifikat. De besøgende får en forbindelsesfejl uden vej forbi, mens kolleger, der aldrig har åbnet hovedsitet, når tjenesten uden problemer, så fejlen er svær at genskabe. Direktivet blev sendt ud, før nogen lavede listen. Lav den først, ud fra DNS-zonen og ud fra Certificate Transparency-logs, der registrerer hvert navn, et offentligt certifikat er udstedt til. Flyt hvert navn til HTTPS eller til et andet domæne, og tilføj så direktivet med en kort `max-age`.

En advarsel, medarbejderne plejede at klikke sig forbi, bliver til en mur. Staging med et selvsigneret certifikat, en intern CA, som de besøgendes bærbare ikke stoler på, et certifikat, der udløb en weekend: før HSTS gav hver af dem en advarselsside med en vej forbi, og bagefter tilbyder browseren ingen. RFC 9325 nævner selvsignerede certifikater som et tilfælde, hvor HSTS kan gøre sikkerheden dårligere. Giv staging et offentligt betroet certifikat eller et

domæne, politikken ikke rækker til, og behandl fornyelse af certifikater som noget, der nu tager sitet ned, når det fejler, uden at den besøgende har noget valg.

Et domæne på preload-listen får brug for ren HTTP igen. En enhed, der kun taler HTTP, en leverandør, hvis funktion til eget domæne serverer uden certifikat, et opkøb, der bringer sine egne værter ind under navnet. Posten ligger allerede i browsere, der er sendt ud, og serveren kan gøre meget lidt ved det: en anmodning om fjernelse når først brugerne, når deres browsere opdateres. Læg det nye navn under et andet domæne, og søg først om preload efter en lang periode med en `max-age` på et år og `includeSubDomains`, der har kørt uden en eneste fejlmelding.

Læs mere

- RFC 6797: HTTP Strict Transport Security (HSTS) — headeren, dens to direktiver og præcis hvornår en browser skal følge eller ignorere den.
- HSTS Preload List Submission — kravene for at komme på listen, den trinvise udrulning, vedligeholderne anbefaler, og hvad det koster at komme af den igen.
- RFC 9325, afsnit 3.2 — den gældende bedste praksis for TLS om HSTS: klienter og servere skal understøtte det, og servere bør sende det.
- Upgrade Insecure Requests — W3C-direktivet, der opgraderer en sides egne forespørgsler, og hvordan det deler arbejdet med HSTS.

Ordliste

Vælg ud fra jeres faktiske vilkår

Medbring udrulningsforløbet, driftsrisiciene og dem, der har vagten. Vi kan afgrænse en ændring med kontroller, en vej tilbage og tydeligt ansvar.

Se, hvad det koster Kontakt os →

Gør begreberne praktiske

Spørg, hvordan »Hvad er HSTS?« gælder i jeres organisation. Et konkret eksempel fra jeres nuværende system er nok til at begynde samtalen.

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/glossary/hsts/