Engineer

HSTS

en nettlesers løfte om å bruke HTTPS neste gang

Hva det er

HSTS, kort for HTTP Strict Transport Security, er en header i HTTP-svaret som ber nettleseren om å kontakte et nettsted bare over HTTPS i et oppgitt antall sekunder, og om å avvise forbindelsen blankt når sertifikatet er feil. Headeren heter `Strict-Transport-Security`, og RFC 6797 har definert den siden november 2012. Selv krypterer den ingenting; den endrer hva en nettleser er villig til å gjøre over HTTPS.

En typisk verdi har to direktiver:

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

`max-age` er hvor lenge nettleseren husker regelen, her ett år. Med `includeSubDomains` gjelder regelen for alle navn under verten, så `api.example.com` og `old.example.com` arver det `example.com` ba om. Et tredje ord, `preload`, dukker opp i mange konfigurasjoner. Det kommer fra preload-listen som drives på hstspreload.org, og står ikke i RFC-en. Med det samtykker nettstedet i at navnet bygges inn i nettlesere.

Har en nettleser først sett headeren på et sikkert svar, gjør den tre ting for den verten til tiden er ute. Den skriver om hver `http://`-adresse til `https://` før forespørselen forlater maskinen. Den behandler enhver sertifikatfeil som fatal, uten noen knapp for å fortsette likevel. Og den starter klokken på nytt hver gang headeren kommer igjen. Et svar med `max-age=0` får den til å glemme verten.

Det vanlige sammenligningsgrunnlaget er en omdirigering fra port 80 til port 443. Den virker, men først etter at nettleseren har sendt den første forespørselen i klartekst, og hvem som helst på det samme nettverket kan

besvare den forespørselen selv og holde den besøkende på en ukryptert kopi av nettstedet. HSTS fjerner den ukrypterte forespørselen fra hvert besøk etter det første. Upgrade Insecure Requests, et direktiv i Content Security Policy, er den andre naboen: det oppgraderer forespørslene en side gjør etter sine egne skript og bilder, og overlater en besøkende som kommer fra en annens `http://` -lenke, til omdirigeringen.

Bruk begrepet i en beslutning

Bruk «Hva er HSTS?» i en konkret beslutning. Vi kan knytte begrepet til systemet dere bygger eller kjøper. Kontoer, passord og tilganger ordnet slik at folk har nøyaktig den tilgangen jobben krever, og de som slutter, mister den samme dag.

Se tjenestene våre Ta kontakt →

Undersøk antakelsene bak ordet

Bruk forklaringen og det tilknyttede arbeidet til å formulere spørsmål for prosjektet. Vi kan undersøke antakelsene sammen før et verktøy velges. Skrev 1 000 RBAC-regler for å sikre Linux-servere — 1,000 tilgangsregler skrevet for å tette et sikkerhetshull

Skrev 1 000 RBAC-regler for å sikre Linux-servere Ta kontakt →

Hvorfor det er viktig

Å slå på HSTS koster én linje konfigurasjon. Den egentlige avgjørelsen er hvor langt du tar det, fordi hvert steg beskytter flere besøkende og er vanskeligere å gå tilbake på.

Headeren alene etterlater et hull som RFC 6797 selv navngir. En nettleser som aldri har sett nettstedet, eller som har en husket policy som er utløpt, starter fortsatt med en ukryptert forespørsel. En lang `max-age` krymper den andre halvdelen av hullet, og derfor er ett år den vanlige innstillingen. Den første halvdelen lukkes bare gjennom preload-listen: vertsnavn kompilert inn i Chrome, der Firefox, Safari og Edge fører lister basert på den, slik at en nettleser rett fra installasjonen allerede vet at den skal bruke HTTPS.

Listen krever mye. Hoveddomenet trenger et gyldig sertifikat, en omdirigering fra HTTP til HTTPS på den samme verten, HTTPS på hvert underdomene og en header med `max-age` på minst 31536000 sekunder, `includeSubDomains` og `preload`. Vedlikeholderne sier rett ut at dette også gjelder interne underdomener som offentligheten aldri ser. Det som avgjør spørsmålet, er veien ut: en fjerning tar måneder å nå brukerne gjennom Chrome-oppdateringer, de andre nettleserne gir ingen garanti, og nettstedet beskriver fjerning som «slow and painful», altså tregt og smertefullt.

Valget går derfor opp en stige, ett trinn om gangen:

- **Headeren uten** `includeSubDomains` passer nesten ethvert nettsted som allerede leverer HTTPS. RFC 9325, gjeldende beste praksis for TLS, sier at webservere bør sende den.
- **Å legge til** `includeSubDomains` er trygt når noen har listet hvert navn under domenet og bekreftet at hvert av dem svarer på HTTPS, også de som en leverandør eller en annen avdeling drifter.
- **Preloading** passer et domene der eierne er sikre på at intet navn under det noen gang trenger ukryptert HTTP igjen: et innloggingsdomene, et betalingsdomene, et selskap som styrer all sin egen DNS. Et markedsføringsdomene der et kampanjeverktøy snart kan be om en CNAME, er en dårlig kandidat, og det samme er ethvert domene som deles med kontorutstyr.

Vedlikeholderne av listen angir også tempoet på klatringen: `max-age` på 300 sekunder, så en uke, så en måned, hver gang med `includeSubDomains`, og med hele levetiden til ett trinn utløpt før neste begynner. En nettleser beholder den verdien den så sist, så en feil gjort på femminuttstrinnet koster fem minutter.

Slik ser det ut i praksis

På en plattform med faglige og organisatoriske data arbeidet selskapet som sikkerhetsingeniør og herdet hvert lag, og HSTS var ett av dem. I proxy-mellomvaren i Next.js, `proxy.ts`, bærer svaret HSTS sammen med en Content Security Policy med en fersk nonce per forespørsel og `strict-dynamic`, og SameSite-informasjonskapsler. En proxy-mellomvare kjører foran rutene den treffer, og det holder reglene som vender mot nettleseren, samlet i én fil i stedet for spredt utover sidene.

Det samme arbeidet viser hvor HSTS slutter. Arbeidsantakelsen var at klienten er fiendtlig: alt nettleseren håndhever, kan den som sitter med nettleseren, slå av. HSTS beskytter en ærlig besøkende mot et fiendtlig nettverk. Mot en fiendtlig klient gjør det ingenting, og derfor fortsatte arbeidet under det. API-et fikk hastighetsbegrensning, CORS, grenser for forespørselsstørrelse, validering av inndata før noe nådde databasen, og logging av sikkerhetsrelevante hendelser. API-et koblet seg til databasen som en rolle som bare kunne kjøre applikasjonens funksjoner, og rettighetene ble kontrollert på nytt på serveren ved hver forespørsel, mens sperrene i frontenden ble behandlet som ren brukeropplevelse. I den stakken holder HSTS selve trafikken utenfor rekkevidde, og det er det billigste laget å legge til.

Hvor det går galt

Headeren står i konfigurasjonen, og nettlesere oppfører seg som om den manglet. Den sendes på det ukrypterte HTTP-svaret, som regel omdirigeringen, eller på et HTTPS-svar med et sertifikat nettleseren avviste. RFC 6797 ber nettlesere se bort fra den i begge tilfeller og ber servere la være å sende den på ukrypterte svar overhodet. En beslektet glipp sender `example.com` rett til `https://www.example.com` med headeren bare på `www`, slik at hoveddomenet og de andre navnene står udekket. Sett headeren på hvert HTTPS-svar, omdirigeringer medregnet, og sjekk begge dørene:

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

Den første linjen skal skrive ut headeren. Den andre hører til omdirigeringen og skal skrive ut ingenting i det hele tatt.

En gammel tjeneste under domenet slutter å laste for alle som har besøkt hovednettstedet. `includeSubDomains` på hoveddomenet dekker hvert navn under det: en statusside hos en leverandør, adminsiden til en skriver, et internt verktøy som aldri hadde sertifikat. De besøkende får en tilkoblingsfeil uten vei forbi, mens kolleger som aldri åpnet hovednettstedet, når tjenesten uten problemer, så feilmeldingene er vanskelige å gjenskape. Direktivet gikk ut før noen hadde laget listen. Lag den først, fra DNS-sonen og fra Certificate Transparency-loggene, som registrerer hvert navn et offentlig sertifikat er utstedt for. Flytt hvert navn til HTTPS eller til et annet domene, og legg så til direktivet med kort `max-age`.

En advarsel de ansatte pleide å klikke seg forbi, blir en mur. Et testmiljø med selvsignert sertifikat, en intern sertifikatutsteder som de besøkendes bærbare maskiner ikke stoler på, et sertifikat som utløp i løpet av en helg: før HSTS var hver av disse en advarselsside med en vei forbi, og etterpå tilbyr nettleseren ingen. RFC 9325 nevner selvsignerte sertifikater som et tilfelle der HSTS kan gjøre sikkerheten dårligere. Gi testmiljøet et sertifikat fra en offentlig betrodd utsteder, eller et domene policyen ikke når, og behandle fornyelse av sertifikater som noe som nå tar ned nettstedet når det svikter, uten noe valg for den besøkende.

Et domene på preload-listen trenger ukryptert HTTP tilbake. En enhet som bare snakker HTTP, en leverandør med en funksjon for eget domene som ikke leverer sertifikat, et oppkjøp som tar med egne verter inn under navnet. Oppføringen ligger allerede i nettlesere som er sendt ut, og serveren kan gjøre lite med det: en fjerning kommer først fram til brukerne etter hvert som nettleserne deres oppdateres. Legg det nye navnet under et annet domene, og bruk preload først etter en lang periode med `max-age` på ett år og `includeSubDomains` uten en eneste henvendelse til brukerstøtten.

Les mer

- RFC 6797: HTTP Strict Transport Security (HSTS) — headeren, de to direktivene og nøyaktig når en nettleser skal følge den eller se bort fra den.
- HSTS Preload List Submission — vilkårene for å komme på listen, den trinnvise utrullingen vedlikeholderne anbefaler, og hva det koster å komme seg av.
- RFC 9325, section 3.2 — gjeldende beste praksis for TLS om HSTS: klienter og servere må støtte det, og servere bør sende det.
- Upgrade Insecure Requests — W3C-direktivet som oppgraderer sidens egne forespørsler, og hvordan det deler arbeidet med HSTS.

Ordliste

Velg ut fra deres faktiske rammer

Ta med utrullingsløpet, driftsrisikoene og dem som har vakt. Vi kan avgrense en endring med kontroller, en vei tilbake og tydelig ansvar.

Se hva det koster Ta kontakt →

Gjør begrepene praktiske

Spør hvordan «Hva er HSTS?» gjelder i organisasjonen deres. Et konkret eksempel fra dagens system er nok til å starte samtalen.

Se tjenestene våre

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

Ta med deg denne siden

Visittkort

Laget med kjærlighet av D.

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

Opphavsrett © 2025 – nå · Engineer ApS · Alle rettigheter forbeholdt.

Skrevet ut fra https://engineer.company/nb/glossary/hsts/