Engineer

HSTS

en webbläsares löfte att använda HTTPS nästa gång

Vad det är

HSTS, utskrivet HTTP Strict Transport Security, är ett svarshuvud som säger åt en webbläsare att under ett angivet antal sekunder bara nå en webbplats över HTTPS, och att vägra anslutningen helt när certifikatet är fel. Huvudet heter `Strict-Transport-Security`, och RFC 6797 har definierat det sedan november 2012. HSTS krypterar ingenting själv. Det ändrar vad en webbläsare är beredd att göra över HTTPS.

Ett typiskt värde innehåller två direktiv:

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

`max-age` anger hur länge webbläsaren behåller regeln, här ett år. Med `includeSubDomains` når regeln varje namn under värdnamnet, så att `api.example.com` och `old.example.com` ärver det som `example.com` bad om. Ett tredje ord, `preload`, förekommer i många konfigurationer. Det kommer från preload-listan på hstspreload.org och inte från RFC:n, och det är webbplatsens samtycke till att få sitt namn inbyggt i webbläsarna.

När en webbläsare har sett huvudet på ett säkert svar gör den tre saker för det värdnamnet tills tiden har gått ut. Den skriver om varje `http://`-adress till `https://` innan en begäran lämnar datorn. Den behandlar varje certifikatfel som ett definitivt stopp, utan någon knapp för att gå vidare ändå. Och den startar om klockan varje gång huvudet kommer på nytt. Ett svar med `max-age=0` får den att glömma värdnamnet.

Den vanliga jämförelsen är en omdirigering från port 80 till port 443. En omdirigering fungerar, men först efter att webbläsaren har skickat sin första begäran i klartext, och vem som helst i samma nätverk kan själv svara på den begäran och hålla kvar besökaren på en okrypterad kopia av webbplatsen. HSTS tar bort klartextbegäran från varje besök efter det första. Den andra grannen är Upgrade Insecure Requests, ett direktiv i Content Security Policy. Det uppgraderar de begäranden som en sida själv gör efter sina skript och bilder, och lämnar en besökare som kommer via någon annans `http://` -länk åt omdirigeringen.

Använd begreppet i ett beslut

Använd ”Vad är HSTS?” i ett konkret beslut. Vi kan knyta begreppet till systemet ni bygger eller köper. Konton, lösenord och behörigheter ordnade så att alla har exakt den åtkomst deras arbete kräver, och de som slutar förlorar den samma dag.

Se våra tjänster Hör av er →

Granska antagandena bakom ordet

Använd förklaringen och det länkade arbetet för att formulera frågor till projektet. Vi kan undersöka antagandena tillsammans innan ett verktyg väljs. Skrev 1 000 RBAC-regler som säkrar Linux-servrar — 1,000 åtkomstregler, skrivna för att täppa till en säkerhetslucka

Skrev 1 000 RBAC-regler som säkrar Linux-servrar Hör av er →

Varför det spelar roll

Att slå på HSTS kostar en rad konfiguration. Det verkliga beslutet är hur långt du ska driva det, eftersom varje steg skyddar fler besökare och är svårare att ta tillbaka.

Huvudet ensamt lämnar en lucka som RFC 6797 själv pekar ut. En webbläsare som aldrig har sett webbplatsen, eller vars sparade policy har löpt ut, börjar fortfarande med en begäran i klartext. Ett långt `max-age` krymper den andra halvan av luckan, och därför är ett år den vanliga inställningen. Den första halvan stängs bara genom preload-listan: värdnamn som kompileras in i Chrome, medan Firefox, Safari och Edge har egna listor som bygger på den, så att en webbläsare som just har installerats redan vet att den ska använda HTTPS.

Listan kräver mycket. Basdomänen behöver ett giltigt certifikat, en omdirigering från HTTP till HTTPS på samma värdnamn, HTTPS på varje underdomän och ett huvud med ett `max-age` på minst 31 536 000 sekunder, `includeSubDomains` och `preload`. Underhållarna säger rakt ut att kravet även gäller interna underdomäner som allmänheten aldrig ser. Det som avgör frågan är vägen ut: en

borttagning tar månader att nå användarna genom uppdateringar av Chrome, de andra webbläsarna lämnar inga garantier, och webbplatsen själv beskriver borttagningen som långsam och smärtsam.

Valet går alltså uppför en stege, ett steg i taget:

- **Huvudet utan** `includeSubDomains` passar nästan varje webbplats som redan levererar HTTPS. RFC 9325, den nuvarande bästa praxisen för TLS, säger att webbservrar bör skicka det.
- **Att lägga till** `includeSubDomains` är säkert först när någon har listat varje namn under domänen och bekräftat att vart och ett svarar på HTTPS, även de som en leverantör eller en annan avdelning driver.
- **Preload** passar en domän vars ägare är säkra på att inget namn under den någonsin behöver klartext-HTTP igen: en inloggningsdomän, en betaldomän, ett företag som styr hela sin egen DNS. En marknadsföringsdomän där ett kampanjverktyg snart kan be om en CNAME-post är en dålig kandidat, och detsamma gäller varje domän som delas med kontorsutrustning.

Listans underhållare anger också takten för klättringen: `max-age` på 300 sekunder, sedan en vecka, sedan en månad, varje steg med `includeSubDomains`, och ett stegs hela giltighetstid ska ha löpt ut innan nästa börjar. En webbläsare behåller det värde den såg senast, så ett misstag på femminuterssteget kostar fem minuter.

Så ser det ut i praktiken

På en plattform med yrkes- och organisationsdata arbetade företaget som säkerhetsingenjör och härdade varje lager, och HSTS var ett av dem. I middlewaren i Next.js, `proxy.ts`, bär svaret HSTS tillsammans med en Content Security Policy med en ny nonce för varje begäran och `strict-dynamic`, samt SameSite-cookies. Middlewaren körs framför de rutter den matchar, vilket samlar reglerna som riktar sig till webbläsaren i en enda fil i stället för att sprida dem över sidorna.

Samma arbete visar var HSTS tar slut. Arbetsantagandet var att klienten är fientlig: allt som webbläsaren upprätthåller kan kopplas bort av den som sitter vid webbläsaren. HSTS skyddar en ärlig besökare mot ett fientligt nätverk. Mot en fientlig klient hjälper det ingenting, och därför fortsatte arbetet i lagren under. API:et fick hastighetsbegränsning, CORS, storleksgränser för begäranden, validering av indata innan något nådde databasen och loggning av

säkerhetsrelevanta händelser. API:et anslöt till databasen som en roll som bara fick köra applikationens funktioner, och rättigheterna kontrollerades på nytt på servern vid varje begäran, medan spärrarna i frontenden enbart behandlades som användarupplevelse. I den stacken håller HSTS själva trafiken utom räckhåll, och det är det billigaste lagret att lägga till.

Var det går fel

Huvudet står i konfigurationen, och webbläsarna beter sig som om det saknades. Det skickas på det okrypterade HTTP-svaret, oftast omdirigeringen, eller på ett HTTPS-svar vars certifikat webbläsaren underkände. RFC 6797 säger åt webbläsare att ignorera det i båda fallen och åt servrar att helt hålla det borta från okrypterade svar. Ett närliggande misstag skickar `example.com` direkt till `https://www.example.com` med huvudet bara på `www`, vilket lämnar basdomänen och dess andra namn oskyddade. Sätt huvudet på varje HTTPS-svar, omdirigeringar inräknade, och kontrollera båda dörrarna:

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

Den första raden ska skriva ut huvudet. Den andra hör till omdirigeringen, och där ska utskriften vara tom.

En gammal tjänst under domänen slutar laddas för alla som har besökt huvudwebbplatsen. `includeSubDomains` på basdomänen täcker varje namn under den: en statussida som en leverantör driftar, en skrivares administrationssida, ett intranätverktyg som aldrig har haft något certifikat. De besökarna möts av ett anslutningsfel utan väg förbi, medan kollegor som aldrig har öppnat huvudwebbplatsen når tjänsten utan problem, så felrapporterna är svåra att återskapa. Direktivet gick ut innan någon hade gjort listan. Gör den först, ur DNS-zonen och ur loggarna för Certificate Transparency, som registrerar varje namn som ett publikt certifikat har utfärdats för. Flytta varje namn till HTTPS eller till en annan domän, och lägg sedan till direktivet med ett kort `max-age`.

En varning som personalen brukade klicka sig förbi blir en mur. En testmiljö med ett självsignerat certifikat, en intern certifikatutfärdare som besökarnas bärbara datorer inte litar på, ett certifikat som gick ut under en helg: före HSTS var vart och ett av dem en varningssida med en väg förbi, och efteråt erbjuder webbläsaren ingen. RFC 9325 nämner självsignerade certifikat som ett fall där

HSTS kan göra säkerheten sämre. Ge testmiljön ett publikt betrott certifikat eller en domän som policyn inte når, och behandla certifikatförnyelsen som något som numera tar ner webbplatsen när den fallerar, utan att besökaren får något val.

En domän på preload-listan behöver klartext-HTTP tillbaka. En enhet som bara talar HTTP, en leverantör vars funktion för egen domän serverar sidor utan certifikat, ett förvärv som tar med egna värdar under namnet. Posten sitter redan inbyggd i webbläsare som har levererats, och servern kan göra lite åt saken: en begäran om borttagning når användarna först när deras webbläsare uppdateras. Lägg det nya namnet under en annan domän, och skicka in domänen till preload-listan först efter en lång period med ett års `max-age` och `includeSubDomains` som har passerat utan ett enda ärende.

Läs mer

- RFC 6797: HTTP Strict Transport Security (HSTS) — huvudet, dess två direktiv och exakt när en webbläsare måste följa det eller ignorera det.
- HSTS Preload List Submission — villkoren för att komma med på listan, den stegvisa utrullning som underhållarna rekommenderar och vad det kostar att lämna den.
- RFC 9325, section 3.2 — den nuvarande bästa praxisen för TLS om HSTS: klienter och servrar måste stödja det, och servrar bör skicka det.
- Upgrade Insecure Requests — W3C-direktivet som uppgraderar en sidas egna begäranden, och hur det delar arbetet med HSTS.

Ordlista

Välj utifrån era faktiska förutsättningar

Ta med driftsättningsvägen, driftsriskerna och dem som har jour. Vi kan avgränsa en förändring med kontroller, en väg tillbaka och tydligt ansvar.

Se vad det kostar Hör av er →

Gör begreppen praktiska

Fråga hur ”Vad är HSTS?” gäller i er organisation. Ett konkret exempel från ert nuvarande system räcker för att börja samtalet.

Se våra tjänster

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

Ta med dig den här sidan

Visitkort

Omsorgsfullt skapad med kärlek av D.

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

Upphovsrätt © 2025 – nu · Engineer ApS · Alla rättigheter förbehållna.

Utskrivet från https://engineer.company/sv/glossary/hsts/