Engineer

Subresource Integrity

et hash en browser tjekker en fil mod

Hvad det er

Subresource Integrity lader en side sige på forhånd, præcis hvilke bytes et script eller et stilark skal indeholde, og får browseren til at kassere filen, hvis det, der kommer frem, er anderledes. Det forkortes som regel til SRI. Siden skriver en kryptografisk hashværdi af filen i en `integrity`-attribut på det `script`- eller `link`-element, der henter den. Når svaret kommer, kører browseren den samme hashfunktion over de bytes, den har modtaget, og sammenligner resultatet med værdien på elementet. Er de ens, kører filen eller styler siden som normalt. Er de forskellige, behandler browseren svaret som en netværksfejl: scriptet bliver aldrig kørt, og stilarket bliver aldrig anvendt.

Værdien består af navnet på en hashfunktion, en bindestreg og selve hashværdien i base64. Én kommando over den færdige fil laver den, og kører man den to gange, ser man, hvor lidt der skal til for at ændre svaret:

```
$ printf "alert('Hello, world.');" > hello.js
$ openssl dgst -sha384 -binary hello.js | openssl base64 -A
H8BRh8j48O9oYatfu5AZzq6A9RINhZO5H16dQZngK7T62em8MUt1FLm52t+eX6xO
$ echo >> hello.js
$ openssl dgst -sha384 -binary hello.js | openssl base64 -A
qu8P7hCLFcBpas8BOnGmaANCkfXJIpFttDEaxOj0Pq9H6Q8NYjsMb+PVqwuJygVe
```

Med `sha384-` foran er det første output attributtens værdi for det lille script på én linje. Et enkelt linjeskift tilføjet til sidst giver en hashværdi, der intet har til fælles med den første, og en browser med den første værdi ville afvise den anden fil.

Specifikationen kender tre hashfunktioner, SHA-256, SHA-384 og SHA-512, alle fra SHA-2-familien, og kalder SHA-384 et godt udgangspunkt i skrivende stund. En attribut kan rumme flere hashværdier adskilt af mellemrum. Browseren vælger den stærkeste funktion blandt dem og godtager filen, når den stemmer med en hvilken som helst værdi lavet med den funktion. Det lader en side skifte til en stærkere algoritme eller godtage to udgaver af et bibliotek under et skifte, uden et tidsrum hvor siden kun bærer forkerte værdier.

Attributten findes på `script` og `link`, og samme værdi kan følge med i en `Link`-header i svaret, der forudindlæser en fil. Specifikationens gældende tekst er anden udgave, et W3C Working Draft fra arbejdsgruppen Web Application Security. Første udgave blev en W3C-anbefaling i 2016.

Brug begrebet i en beslutning

Brug »Hvad er Subresource Integrity?« 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

En side, der henter et bibliotek fra et content delivery network eller fra en hvilken som helst vært, som ejerne ikke selv driver, giver den vært magt til at køre kode på siden. HTTPS beskytter filen på vej over nettet, men siger intet om, hvad værten valgte at sende i første omgang. Bliver der brudt ind i CDN'ets lager, bliver en konto, der udgiver til det, overtaget, eller bliver filen bag en velkendt adresse skiftet ud i stilhed, så kører hver side, der henter den, den nye kode med sidens fulde myndighed. Sidens egen kode har ingen mulighed for at se forskel.

SRI flytter den beslutning tilbage til siden. Hashværdien beregnes af sitets ejer over en fil, de har læst eller bygget, og den rejser inde i siden fra sitets egen server. En vært, der sender noget andet, får sin fil afvist i stedet for kørt. Siden mister en funktion, indtil nogen kigger efter, og det er en synlig fejl med en åbenlys årsag i stedet for en andens kode, der kører i stilhed.

Hashværdien låser også en version fast. En adresse som `/lib/latest.js` kan begynde at levere en ny udgave under en side når som helst, men hashværdien følger ikke med. En opgradering, ejeren aldrig valgte, bliver til en afvisning på ét bestemt element, meldt via en error-hændelse.

Det er værd at være præcis om, hvad SRI ikke gør. Den tjekker en fil mod en hashværdi, men den vurderer ikke, om filen er sikker. En hashværdi af et bibliotek, der allerede indeholder en fjendtlig linje, beskytter trofast den fjendtlige linje. Den lader inline-script være, for det er en opgave for en Content Security Policy. Og den stoler på den side, der bærer værdien, så selve siden skal stadig nå frem uændret.

Sådan ser det ud i praksis

Dette site bruger det på sit eget stilark. Sikkerhedssiden nævner Subresource Integrity på stilarket: bundtet får et fingeraftryk og en hash, når sitet bygges, så læserens browser kan verificere, at det kom frem uændret. Der offentliggøres ingen JavaScript, som kan køres, og der er ingen tredjepartsforespørgsler. En læser, der tæller værterne i browserens netværksfane, finder én, sitets egen.

Kolofonen beskriver den samme fil fra byggets side, som ét CSS-bundt, samlet og fingeraftrykt ved bygning og leveret med en SRI-hash. Fingeraftrykket lægger en hash af filens indhold ind i dens navn, så et ændret stilark får en ny adresse, og en browser aldrig forveksler en gammel kopi i cachen med den gældende. Integritetsværdien lader samtidig browseren tjekke de bytes, der ligger bag adressen. Bygget laver begge dele ud fra de samme bytes i samme trin, så hver udgivelse sender stilarket og dets hashværdi af sted sammen, og den hyppigste fejl nedenfor, en fil der ændres efter hashingen, har meget lidt plads til at opstå.

Hvor det går galt

Noget mellem buildet og browseren omskriver filen. Et CDN, der minificerer scripts, en proxy, der indsætter noget i svarindholdet, eller et udrulningstrin, der omskriver en fil efter hashberegningen, ændrer indholdets bytes, og browseren afviser resultatet, selv om ingen har angrebet noget. Genkomprimering med gzip eller Brotli, som bevarer indholdet, skaber ikke den uoverensstemmelse: kontrollen bruger det afkodede svarindhold, ikke de komprimerede bytes, der sendes over netværket. Specifikationen lægger ansvaret hos mellemled, der ændrer indholdet: de skal holde hashværdien i trit med det, de leverer, eller levere originalen urørt, og den server, der leverer filen, bør sende `Cache-Control:`

`no-transform` for at bede dem lade den være. Den sikre rækkefølge er at tage hashværdien til sidst, over præcis de bytes i det afkodede indhold, som browseren kontrollerer.

En fil fra en anden oprindelse kommer uden CORS. Specifikationen kræver, at et integritetstjek af en fil fra en anden oprindelse bruger CORS, og kalder et forsøg uden det en logisk fejl. Elementet kan for eksempel bruge `crossorigin="anonymous"`, og værten skal svare med en header, der tillader deling af filen. `crossorigin="use-credentials"` bruger også CORS, med serverheadere, der udtrykkeligt tillader den forespørgende oprindelse og legitimationsoplysninger. Uden en CORS-forespørgsel og en tilsvarende politik på serveren blokerer browseren filen i stedet for at indlæse den uden kontrol. Reglen beskytter andre sites: uden den kunne en angriber indlæse et privat svar med en gættet hashværdi og ud fra udfaldet lære for eksempel, om læseren er logget ind dér.

Hashværdien kommer over almindelig HTTP. SRI beskytter en side mod en kompromitteret filvært, aldrig mod nogen på netværksvejen. På en side, der leveres uden HTTPS, kan den, der kan omskrive filen undervejs, lige så godt omskrive eller slette hashværdien i siden, så specifikationen anbefaler kun at levere integritetsdata til en sikker kontekst.

Hashfunktionen er en, browseren springer over. Specifikationen kender kun SHA-256, SHA-384 og SHA-512 og anbefaler, at browsere afviser kendt svage funktioner som MD5 og SHA-1. Overraskelsen er, hvad der sker med en værdi uden for den liste: browseren ignorerer den, og en attribut, der kun består af ignorerede værdier, regnes for slet ikke at have nogen metadata, så filen indlæses ukontrolleret. Et byggeværktøj, der kan lave fingeraftryk med MD5, som Hugos kan, gør det let at sende sådan en værdi ud ved et uheld.

En afvist fil trækker siden med sig. Når tjekket fejler, sender browseren en error-hændelse på elementet, og scriptet eller stilarket mangler simpelthen. For et bibliotek, siden ikke kan køre uden, foreslår specifikationen at fange hændelsen og hente en reservekopi fra en betroet, om end langsommere, kilde, fx sitets egen server. Uden en sådan reserve virker beskyttelsen som tiltænkt, og læseren får alligevel en ødelagt side.

Kun de elementer, der bærer en hashværdi, bliver tjekket. En `integrity`-attribut på ét script siger intet om det næste, som nogen tilføjer uden, og en gennemgang af de scripts, der har en, afslører aldrig det, der mangler den. Udkastet tilføjer en svarheader, der dækker det fra serverens side. `Integrity-`

`Policy` med `blocked-destinations=(script)` blokerer hvert eksternt script, der kommer uden integritetsdata, og `Integrity-Policy-Report-Only` lister dem først, så de kan rettes, før noget bliver blokeret.

Læs mere

- Subresource Integrity — det gældende udkast med verifikationsalgoritmen, `Integrity-Policy`-headerne og noterne om proxyer, almindelig HTTP og lækager på tværs af oprindelser.
- Subresource Integrity, W3C-anbefaling 2016 — første udgave, den stabile anbefaling, som udkastet voksede ud af.
- Fetch Standard — hvor en forespørgsels integritetsdata tjekkes mod svarets indhold, og CORS-protokollen, som et tjek på tværs af oprindelser afhænger af.
- Hugo: resources.Fingerprint — hvordan en statisk sitegenerator laver fingeraftryk af en fil ved bygning og giver den tilhørende integritetsværdi tilbage.
- FIPS 180-4, Secure Hash Standard — den amerikanske føderale standard, der definerer SHA-256, SHA-384 og SHA-512.

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 Subresource Integrity?« 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/sri/