Engineer

Subresource Integrity

en hash en nettleser sjekker en fil mot

Hva det er

Subresource Integrity er en mekanisme der en side på forhånd oppgir nøyaktig hvilke byte et skript eller et stilark skal inneholde, og der nettleseren kaster filen hvis det som kommer fram, er noe annet. Til daglig forkortes det til SRI. Siden skriver et kryptografisk sammendrag av filen inn i et `integrity` -attributt på `script` - eller `link` -elementet som laster den. Når svaret kommer, kjører nettleseren den samme hashfunksjonen over bytene den fikk, og sammenligner resultatet med verdien på elementet. Er de like, kjører skriptet eller virker stilarket på siden som vanlig. Er de ulike, behandler nettleseren svaret som en nettverksfeil: skriptet blir aldri utført, og stilarket blir aldri brukt.

Verdien består av navnet på en hashfunksjon, en bindestrek og sammendraget i base64. Én kommando over den ferdige filen lager den, og to kjøringer viser hvor lite som skal til for å endre 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 den første linjen med utdata verdien attributtet skal ha for det enlinjes skriptet. Ett linjeskift lagt til på slutten gir et sammendrag som ikke har noe til felles med det første, og en nettleser som holder det første, ville avvise den andre filen.

Spesifikasjonen kjenner tre hashfunksjoner, SHA-256, SHA-384 og SHA-512, alle fra SHA-2-familien, og omtaler SHA-384 som et godt utgangspunkt da teksten ble skrevet. Et attributt kan liste flere sammendrag atskilt med mellomrom. Nettleseren velger den sterkeste funksjonen blant dem og godtar filen når den stemmer med ett av sammendragene laget med den funksjonen. Slik kan en side gå over til en sterkere algoritme, eller godta to utgivelser av et bibliotek mens de byttes, uten et tidsrom der siden bare bærer feil verdier.

Attributtet finnes på `script` og `link`, og den samme verdien kan følge med i en `Link`-header i svaret som forhåndslaster en fil. Gjeldende tekst er andre utgave, et W3C-arbeidsutkast fra Web Application Security Working Group. Den første utgaven ble en W3C-anbefaling i 2016.

Bruk begrepet i en beslutning

Bruk «Hva er Subresource Integrity?» 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

En side som laster et bibliotek fra et CDN, eller fra en hvilken som helst vert eierne ikke selv drifter, gir den verten makt til å kjøre kode på siden. HTTPS beskytter filen på veien over nettet, men sier ingenting om hva verten valgte å sende i utgangspunktet. Blir lagringen hos CDN-et brutt opp, blir en konto som publiserer dit, overtatt, eller blir filen bak en kjent adresse byttet ut i det stille, kjører hver side som tar den med, den nye koden med hele sidens myndighet. Sidens egen kode har ingen måte å merke forskjellen på.

SRI flytter den avgjørelsen tilbake til siden. Sammendraget lages av eieren av nettstedet over en fil de har lest eller bygd selv, og det reiser inne i siden, fra nettstedets egen server. En vert som leverer noe annet, får filen sin avvist i stedet for kjørt. Siden mister en funksjon til noen ser på den, en synlig feil med en åpenbar årsak, og slipper å kjøre andres kode uten at noen merker det.

Sammendraget låser også en versjon. En adresse som `/lib/latest.js` kan når som helst begynne å levere en ny utgivelse under en side, og det kan ikke sammendraget følge etter. En oppgradering eieren aldri valgte, blir en avvisning på ett bestemt element, meldt gjennom en feilhendelse.

Det lønner seg å være presis om hva SRI lar være. Mekanismen sjekker en fil mot et sammendrag og vurderer aldri om filen er trygg. Et sammendrag av et bibliotek som allerede inneholder en fiendtlig linje, beskytter den fiendtlige linjen trofast. Skript skrevet rett inn i siden faller utenfor, og der er det en Content Security Policy som gjør jobben. Og mekanismen stoler på siden som bærer sammendraget, så selve siden må fortsatt komme fram uendret.

Slik ser det ut i praksis

Dette nettstedet bruker det på sitt eget stilark. Siden om hvordan vi arbeider sikkert lister Subresource Integrity på stilarket: pakken får et fingeravtrykk og en hash når nettstedet bygges, slik at nettleseren til leseren kan bekrefte at den kom fram uendret. Ingen kjørbar JavaScript publiseres, og ingen forespørsler går til tredjeparter. En leser som teller vertene i nettverkspanelet i nettleseren, finner én, nettstedets egen.

Kolofonen beskriver den samme filen sett fra bygget, som én CSS-pakke satt sammen og merket med et fingeravtrykk ved bygging og levert med en SRI-hash. Fingeravtrykket legger en hash av innholdet inn i filnavnet, så et endret stilark får en ny adresse, og en nettleser tar aldri feil av en gammel kopi i hurtigbufferen og den gjeldende. Integritetsverdien lar nettleseren sjekke bytene bak den adressen. Bygget lager begge deler fra de samme bytene i det samme trinnet, så hver utgivelse leverer stilarket og sammendraget sammen, og den vanligste feilen beskrevet nedenfor, en fil som endres etter at den ble hashet, har svært lite rom til å skje.

Hvor det går galt

Et ledd mellom bygget og nettleseren skriver om filen. Et CDN som minifiserer skript, en proxy som setter noe inn i svarinnholdet, eller et steg i produksjonssettingen som skriver om en fil etter at sammendraget ble tatt, endrer bytene i innholdet, og nettleseren avviser resultatet selv om ingen har angrepet noe. Ny komprimering med gzip eller Brotli som bevarer innholdet, skaper ikke dette avviket: kontrollen bruker svarets dekodede innhold, ikke de komprimerte bytene som sendes over nettet. Spesifikasjonen legger ansvaret på

mellomledd som endrer innholdet: de må holde sammendraget i takt med det de leverer, eller levere originalen urørt, og opphavet bør sende `Cache-Control: no-transform` for å be dem la filen være. Den pålitelige rekkefølgen er å ta sammendraget sist, over nøyaktig de bytene i det dekodede innholdet som nettleseren vil kontrollere.

En fil fra et annet opphav kommer uten CORS. Spesifikasjonen krever at en integritetssjekk av en fil fra et annet opphav bruker CORS, og kaller et forsøk uten det en logisk feil. Elementet kan for eksempel bruke `crossorigin="anonymous"`, og verten må svare med en header som tillater deling av filen. `crossorigin="use-credentials"` bruker også CORS, med serverheadere som uttrykkelig tillater opphavet til forespørselen og legitimasjon. Uten en CORS-forespørsel og en tilsvarende serverpolicy blokkerer nettleseren filen heller enn å laste den usjekket. Regelen beskytter andre nettsteder. Fantes den ikke, kunne en angriper laste et privat svar med et gjettet sammendrag og lære noe av utfallet, f.eks. om leseren er logget inn der.

Et sammendrag som kommer over ukryptert HTTP. SRI beskytter en side mot en kompromittert vert for filer, aldri mot noen på nettverksveien. På en side levert uten HTTPS kan den som kan skrive om filen underveis, like godt skrive om eller slette sammendraget i siden, så spesifikasjonen anbefaler å levere integritetsdata bare til en sikker kontekst.

En hashfunksjon nettleseren hopper over. Spesifikasjonen kjenner bare SHA-256, SHA-384 og SHA-512 og anbefaler at nettlesere avviser kjent svake funksjoner som MD5 og SHA-1. Overraskelsen er hva som skjer med en verdi utenfor den listen: nettleseren overser den, og et attributt som bare består av oversette verdier, regnes som om det ikke bar noen data i det hele tatt, så filen lastes usjekket. Et byggverktøy som kan lage fingeravtrykk med MD5, slik Hugo kan, gjør det lett å sende dette ut ved et uhell.

En avvist fil som tar siden med seg i fallet. Når sjekken feiler, sender nettleseren en feilhendelse på elementet, og skriptet eller stilarket er rett og slett borte. For et bibliotek siden ikke klarer seg uten, foreslår spesifikasjonen å fange hendelsen og laste en reservekopi fra en kilde du stoler på, om enn tregere, f.eks. nettstedets egen server. Mangler en slik reserve, virker beskyttelsen som tenkt, og leseren får likevel en ødelagt side.

En sjekk som bare dekker elementene med et sammendrag. Et integritetsattributt på ett skript sier ingenting om det neste noen legger til uten, og en gjennomgang av skriptene som har ett, vil aldri avsløre det som mangler. Utkastet legger til en header i svaret som dekker dette fra serversiden. `Integrity-Policy` med `blocked-destinations=(script)` blokkerer hvert eksterne skript som kommer uten integritetsdata, og `Integrity-Policy-Report-Only` lister dem først, så de kan rettes før noe blir blokkert.

Les mer

- Subresource Integrity — det gjeldende utkastet, med algoritmen for verifisering, `Integrity-Policy`-headerne og merknadene om proxyer, ukryptert HTTP og lekkasjer på tvers av opphav.
- Subresource Integrity, W3C Recommendation 2016 — første utgave, den stabile anbefalingen utkastet vokste ut av.
- Fetch Standard — der integritetsdataene i en forespørsel sjekkes mot innholdet i svaret, og CORS-protokollen en sjekk på tvers av opphav er avhengig av.
- Hugo: resources.Fingerprint — hvordan en statisk nettstedsgenerator lager et fingeravtrykk av en fil ved bygging og gir tilbake den tilhørende integritetsverdien.
- FIPS 180-4, Secure Hash Standard — den føderale amerikanske standarden som definerer SHA-256, SHA-384 og SHA-512.

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 Subresource Integrity?» 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/sri/