Engineer

HSTS

Zusage des Browsers, beim nächsten Mal HTTPS zu verwenden

Was es ist

HTTP Strict Transport Security, fast immer HSTS genannt, ist ein Antwortheader, der einem Browser vorgibt, eine Website für eine festgelegte Zahl von Sekunden ausschließlich über HTTPS aufzurufen und die Verbindung bei einem fehlerhaften Zertifikat vollständig abzulehnen. Der Header heißt `Strict-Transport-Security` und ist seit November 2012 in RFC 6797 definiert. Er verschlüsselt selbst nichts, sondern ändert, was ein Browser über HTTPS zu tun bereit ist.

Ein typischer Wert enthält zwei Direktiven:

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

`max-age` gibt an, wie lange der Browser die Regel behält, hier ein Jahr. Mit `includeSubDomains` gilt sie für jeden Namen unterhalb des Hosts. So übernehmen `api.example.com` und `old.example.com` die Vorgabe von `example.com`. Ein drittes Token, `preload`, erscheint in vielen Konfigurationen. Es stammt aus der von hstspreload.org betriebenen Preload-Liste, nicht aus dem RFC, und drückt die Zustimmung der Website aus, ihren Namen fest in Browser aufnehmen zu lassen.

Sobald ein Browser den Header in einer sicheren Antwort gesehen hat, tut er für diesen Host bis zum Ablauf der Frist drei Dinge. Er schreibt jede `http://`-Adresse in `https://` um, bevor eine Anfrage die Maschine verlässt. Er behandelt jeden Zertifikatsfehler als fatal, ohne Schaltfläche, mit der man trotzdem fortfahren könnte. Und er startet die Frist jedes Mal neu, wenn der Header erneut eintrifft. Eine Antwort mit `max-age=0` lässt ihn den Host vergessen.

Der übliche Vergleich ist eine Weiterleitung von Port 80 auf Port 443. Sie funktioniert, aber erst, nachdem der Browser seine erste Anfrage unverschlüsselt gesendet hat. Jeder im selben Netzwerk kann diese Anfrage selbst beantworten und den Besucher auf einer unverschlüsselten Kopie der Website halten. HSTS beseitigt diese unverschlüsselte Anfrage bei jedem Besuch nach dem ersten. Der andere benachbarte Mechanismus ist Upgrade Insecure Requests, eine Direktive der Content Security Policy: Sie stellt die Anfragen einer Seite nach ihren eigenen Skripten und Bildern auf HTTPS um. Kommt ein Besucher hingegen über einen fremden `http://` -Link, bleibt dafür die Weiterleitung zuständig.

Das Konzept für eine Entscheidung nutzen

Nutzen Sie „Was ist HSTS?“ für eine konkrete Entscheidung. Wir können helfen, das Konzept mit dem System in Verbindung zu bringen, das Sie entwickeln oder einkaufen. Konten, Passwörter und Berechtigungen werden so geordnet, dass alle genau den beruflich erforderlichen Zugriff haben und ihn am Tag ihres Ausscheidens verlieren.

Unsere Leistungen entdecken Kontakt aufnehmen →

Die Annahmen hinter dem Begriff prüfen

Nutzen Sie die Erklärung und die verlinkten Arbeiten, um Fragen für Ihr Projekt zu formulieren. Wir können die Annahmen gemeinsam mit Ihnen prüfen, bevor ein Werkzeug ausgewählt wird. 1.000 RBAC-Regeln für sichere Linux-Server — 1,000 Zugriffsregeln zum Schließen einer Sicherheitslücke

1.000 RBAC-Regeln für sichere Linux-Server Kontakt aufnehmen →

Warum es wichtig ist

HSTS einzuschalten kostet eine Konfigurationszeile. Die eigentliche Entscheidung ist, wie weit man geht, denn jede Stufe schützt mehr Besucher und lässt sich schwerer zurücknehmen.

Der Header allein lässt eine Lücke, die RFC 6797 selbst benennt. Ein Browser, der die Website noch nie gesehen hat oder dessen gespeicherte Richtlinie abgelaufen ist, beginnt weiterhin mit einer unverschlüsselten Anfrage. Ein langes `max-age` verkleinert die zweite Hälfte dieser Lücke; deshalb ist ein Jahr die übliche Einstellung. Die erste Hälfte schließt nur die Preload-Liste: in Chrome fest eingebaute Hostnamen, auf deren Grundlage auch Firefox, Safari und Edge eigene Listen führen. Ein frisch installierter Browser weiß dadurch bereits, dass er HTTPS verwenden muss.

Die Liste stellt hohe Anforderungen. Die Basisdomain benötigt ein gültiges Zertifikat, eine auf demselben Host bleibende Weiterleitung von HTTP auf HTTPS, HTTPS für jede Subdomain und einen Header mit `max-age` von mindestens 31.536.000 Sekunden, `includeSubDomains` und `preload` . Die

Betreiber sagen ausdrücklich, dass das auch interne, öffentlich nie sichtbare Subdomains umfasst. Entscheidend für die Abwägung ist der Ausstieg: Bis eine Entfernung Nutzer über Chrome-Updates erreicht, vergehen Monate; bei den anderen Browsern gibt es keine Garantie. Die Website bezeichnet die Entfernung als „langsam und schmerzhaft“.

Die Entscheidung lässt sich daher Schritt für Schritt treffen:

- **Der Header ohne** `includeSubDomains` eignet sich für fast jede Website, die bereits HTTPS anbietet. RFC 9325, die aktuelle Empfehlung für TLS, sagt, dass Webserver ihn senden sollten.
- `includeSubDomains` **hinzuzufügen** ist sicher, sobald jemand jeden Namen unterhalb der Domain aufgelistet und bestätigt hat, dass jeder über HTTPS antwortet, einschließlich der von Anbietern oder anderen Abteilungen betriebenen Namen.
- **Preloading** eignet sich für eine Domain, deren Eigentümer sicher sind, dass kein Name darunter jemals wieder unverschlüsseltes HTTP benötigen wird: eine Login-Domain, eine Zahlungsdomain, ein Unternehmen, das sein gesamtes DNS selbst kontrolliert. Eine Marketingdomain, für die bald ein Kampagnenwerkzeug einen CNAME verlangen könnte, ist ein schlechter Kandidat, ebenso jede mit Bürogeräten geteilte Domain.

Die Betreiber der Liste geben auch das Tempo des Aufstiegs vor: `max-age` zunächst auf 300 Sekunden, dann auf eine Woche, dann auf einen Monat setzen, jeweils mit `includeSubDomains`. Vor der nächsten Stufe muss jeweils die volle Geltungsdauer der vorherigen abgewartet werden. Ein Browser behält den zuletzt gesehenen Wert; ein Fehler auf der Fünfminutenstufe kostet deshalb fünf Minuten.

Wie es in der Praxis aussieht

Auf einer Plattform mit beruflichen und organisatorischen Daten arbeitete das Unternehmen als Security Engineer und härtete jede Schicht ab; HSTS war eine davon. In der Next.js-Proxy-Middleware `proxy.ts` enthält die Antwort HSTS, zusammen mit einer Content Security Policy mit einer frischen Nonce pro Anfrage und `strict-dynamic` sowie SameSite-Cookies. Eine Proxy-Middleware läuft vor den Routen, auf die sie zutrifft. Dadurch bleiben die an den Browser gerichteten Regeln in einer Datei, statt über Seiten verteilt zu sein.

Dieselbe Arbeit zeigt, wo HSTS aufhört. Ihre Grundannahme war, dass der Client feindlich ist: Wer den Browser bedient, kann alles abschalten, was dieser durchsetzt. HSTS schützt einen ehrlichen Besucher vor einem feindlichen Netzwerk. Gegen einen feindlichen Client hilft es nicht; deshalb ging die Arbeit darunter weiter. Die API erhielt Ratenbegrenzung, CORS, Größenlimits für Anfragen, Eingabevalidierung, bevor etwas die Datenbank erreichte, und die Protokollierung sicherheitsrelevanter Ereignisse. Sie verband sich mit der Datenbank über eine Rolle, die ausschließlich die Funktionen der Anwendung ausführen durfte. Berechtigungen wurden bei jeder Anfrage erneut auf dem Server geprüft; die Zugangsschranken im Frontend galten nur als Bedienkomfort. In diesem Aufbau hält HSTS den Verkehr selbst außer Reichweite und ist die am günstigsten hinzuzufügende Schicht.

Wo es schiefgeht

Der Header steht in der Konfiguration, aber Browser verhalten sich, als fehlte er. Er wird mit der unverschlüsselten HTTP-Antwort gesendet, meist der Weiterleitung, oder mit einer HTTPS-Antwort, deren Zertifikat der Browser abgelehnt hat. RFC 6797 schreibt Browsern vor, ihn in beiden Fällen zu ignorieren, und Servern, ihn in unverschlüsselten Antworten ganz wegzulassen. Ein verwandter Fehler leitet `example.com` direkt auf `https://www.example.com` um und sendet den Header nur auf `www`. Das lässt die Basisdomain und ihre anderen Namen ungeschützt. Setzen Sie den Header in jede HTTPS-Antwort, einschließlich Weiterleitungen, und prüfen Sie beide Zugänge:

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

Die erste Zeile sollte den Header ausgeben. Die zweite gehört zur Weiterleitung und sollte gar nichts ausgeben.

Ein alter Dienst unterhalb der Domain lädt für alle Besucher der Hauptwebsite nicht mehr. `includeSubDomains` auf der Basisdomain umfasst jeden Namen darunter: eine bei einem Anbieter gehostete Statusseite, die Verwaltungsseite eines Druckers, ein Intranetwerkzeug, das nie ein Zertifikat hatte. Diese Besucher erhalten einen Verbindungsfehler ohne Umgehungsmöglichkeit. Kollegen, die die Hauptwebsite nie geöffnet haben, erreichen den Dienst dagegen problemlos. Das erschwert die Reproduktion der Meldungen. Die Direktive wurde versendet, bevor jemand die Liste erstellt hatte. Erstellen Sie diese zuerst, anhand der DNS-Zone und der Certificate-

Transparency-Protokolle, die jeden Namen erfassen, für den ein öffentliches Zertifikat ausgestellt wurde. Stellen Sie jeden Namen auf HTTPS um oder verschieben Sie ihn unter eine andere Domain. Ergänzen Sie dann die Direktive mit kurzem `max-age`.

Eine Warnung, die Mitarbeitende zuvor übergehen konnten, wird zur unüberwindbaren Sperre. Eine Stagingumgebung mit selbst signiertem Zertifikat, eine interne Zertifizierungsstelle, der die Laptops der Besucher nicht vertrauen, ein übers Wochenende abgelaufenes Zertifikat: Vor HSTS war jeder dieser Fälle eine Warnseite mit einer Möglichkeit zum Fortfahren; danach bietet der Browser keine mehr. RFC 9325 nennt selbst signierte Zertifikate als Fall, in dem HSTS die Sicherheitslage verschlechtern kann. Geben Sie der Stagingumgebung ein öffentlich vertrauenswürdiges Zertifikat oder eine von der Richtlinie nicht erfasste Domain. Behandeln Sie die Zertifikatserneuerung als etwas, dessen Scheitern die Website nun ohne Wahlmöglichkeit für Besucher lahmlegt.

Eine Domain auf der Preload-Liste braucht wieder unverschlüsseltes HTTP. Ein Gerät spricht nur HTTP, ein Anbieter liefert für eigene Kundendomains kein Zertifikat, eine Übernahme bringt eigene Hosts unter den Namen. Der Eintrag steckt bereits in ausgelieferten Browsern, und der Server kann wenig dagegen tun: Ein Entfernungsantrag erreicht Nutzer erst mit ihren Browserupdates. Legen Sie den neuen Namen unter eine andere Domain und nehmen Sie Preloading erst auf, nachdem ein langer Betrieb mit einjährigem `max-age` und `includeSubDomains` ohne Supporttickets geblieben ist.

Weiterführende Lektüre

- RFC 6797: HTTP Strict Transport Security (HSTS) — Der Header, seine beiden Direktiven und die genauen Bedingungen, unter denen ein Browser ihn beachten oder ignorieren muss.
- Anmeldung zur HSTS-Preload-Liste — Die Aufnahmebedingungen, die von den Betreibern empfohlene schrittweise Einführung und die Kosten eines Ausstiegs.
- RFC 9325, Abschnitt 3.2 — Die aktuelle TLS-Empfehlung zu HSTS: Clients und Server müssen es unterstützen, und Server sollten es senden.
- Upgrade Insecure Requests — Die W3C-Direktive, die eigene Anfragen einer Seite auf HTTPS umstellt, und ihre Aufgabenteilung mit HSTS.

Glossar

Für Ihre tatsächlichen Rahmenbedingungen auswählen

Beschreiben Sie den Bereitstellungsweg, die Betriebsrisiken und die Bereitschaftsverantwortlichen. Wir können eine Änderung mit Prüfungen, einem Wiederherstellungsweg und klaren Zuständigkeiten abgrenzen.

Preise ansehen Kontakt aufnehmen →

Die Begriffe praktisch nutzbar machen

Fragen Sie, wie „Was ist HSTS?“ auf Ihre Organisation zutrifft. Ein konkretes Beispiel aus Ihrem bestehenden System reicht aus, um das Gespräch zu beginnen.

Unsere Leistungen entdecken

welcome@engineer.company +45 50 10 61 84 Kopenhagen, Dänemark

Nehmen Sie diese Seite mit

Kontaktkarte

Mit Sorgfalt und Liebe gestaltet von D.

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

Copyright © 2025 – heute · Engineer ApS · Alle Rechte vorbehalten.

Gedruckt von https://engineer.company/de/glossary/hsts/