

DIE SECHS HEADER UND IHR STARTWERT

Header	Startwert	Worauf achten
Strict-Transport-Security	<code>max-age=300; includeSubDomains</code>	In Stufen steigern (unten). Ohne HTTPS ist die Seite für die Dauer von max-age unerreichbar.
Content-Security-Policy	<code>zuerst Content-Security-Policy-Report-Only: default-src 'self'</code>	Erst scharf stellen, wenn die Berichte sauber sind. Mehrere Policies gelten alle, die strengste gewinnt.
X-Content-Type-Options	<code>nosniff</code>	Eigene Skripte müssen mit JavaScript-MIME-Typ kommen, sonst blockiert.
X-Frame-Options	<code>SAMEORIGIN</code> (OWASP: DENY)	Darf <code>frame-ancestors</code> nicht widersprechen: in Chrome gewinnt die CSP.
Referrer-Policy	<code>strict-origin-when-cross-origin</code>	Nie <code>unsafe-url</code> : gibt Adressparameter an fremde Dienste weiter.
Permissions-Policy	<code>geolocation=(), camera=(), microphone=()</code>	Sperrt die Funktion auch für die eigene Seite. Nur als Header wirksam.

HSTS IN STUFEN (HSTSPRELOAD.ORG)

- `max-age=300; includeSubDomains`: alle Subdomains prüfen, auch interne
 - `max-age=604800`: eine Woche
 - `max-age=2592000`: ein Monat
 - `max-age=31536000`: ein Jahr, Minimum für Preload
 - `preload` nur, wenn jede Subdomain dauerhaft HTTPS kann
- Jede Stufe die volle max-age-Dauer laufen lassen. Zusammen gut fünf Wochen.

GEMESSEN: NGINX-FALLEN

- ! ohne `always`: auf 404 und 500 kamen 0 von 6 Headern an
- ! ein eigenes `add_header` im `location`-Block: dort 0 von 6
- ! dasselbe eine Ebene höher: `http`-Block → `server`-Block
- ✓ Abhilfe: Header wiederholen oder `add_header_inherit merge`; (ab 1.29.3)

KONFIGURATION ZUM KOPIEREN

Apache (.htaccess oder vHost, mod_headers)

```
Header always set Strict-Transport-Security "max-age=300; includeSubDomains"
Header always set X-Content-Type-Options "nosniff"
Header always set X-Frame-Options "SAMEORIGIN"
Header always set Referrer-Policy "strict-origin-when-cross-origin"
Header always set Permissions-Policy "geolocation=(), camera=(), microphone=()"
Header always set Content-Security-Policy-Report-Only "default-src 'self'"
```

nginx (server-Block mit HTTPS)

```
add_header Strict-Transport-Security "max-age=300; includeSubDomains" always;
add_header X-Content-Type-Options "nosniff" always;
add_header X-Frame-Options "SAMEORIGIN" always;
add_header Referrer-Policy "strict-origin-when-cross-origin" always;
add_header Permissions-Policy "geolocation=(), camera=(), microphone=()" always;
add_header Content-Security-Policy-Report-Only "default-src 'self'" always;
```

Netlify (Datei _headers im Veröffentlichungsordner)

```
/*
Strict-Transport-Security: max-age=300; includeSubDomains
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Referrer-Policy: strict-origin-when-cross-origin
Permissions-Policy: geolocation=(), camera=(), microphone=()
Content-Security-Policy-Report-Only: default-src 'self'
```

GEMESSEN: WAS NICHT WIRKT

- ✗ `X-Frame-Options` als `<meta>`: Seite lässt sich trotzdem einbetten
- ✗ `frame-ancestors` als `<meta>`: dito
- ✗ `Permissions-Policy` als `<meta>`: Funktion bleibt erlaubt
- ✗ `CSP-Report-Only` als `<meta>`: meldet nichts
- ✗ `X-Frame-Options: ALLOW-FROM`: jede Origin darf einbetten
- ✗ `X-XSS-Protection: 1; mode=block`: Skript läuft trotzdem
- ✓ per `<meta>` wirken nur `CSP script-src` und `<meta name="referrer">`
- ! HSTS per `<meta>` darf der Browser nicht beachten (RFC 6797)

Cloudflare

HSTS: SSL/TLS → Edge Certificates. Übrige Header: Rules → Overview → Create rule → Response Header Transform Rule → Set static.

NACHPRÜFEN

```
curl -sI https://www.example.com/
curl -sIL http://example.com/
curl -sI https://www.example.com/gibt-es-nicht
```

Die dritte Zeile zeigt fehlendes `always`. Vorher CDN-Cache leeren, im Browser „Disable cache“.