Reportbasierte CSP-Erzeugung

Abschlussvortrag Projektgruppe

Malte Klaassen

2016-10-26

Inhaltsverzeichnis

Content Security Policy

- HTTP Header `Content-Security-Policy` und `Content-Security-Policy-Report-Only`
- Whitelist an Ressourcen, die ein Browser laden bzw. ausführen darf
- Verhinderung von XSS
- Browser generiert Reports für Verstösse

Ziel

- Tool zur Generierung von Content Security Policies aus Reports
- Reports können bspw. während Testläufen erstellt werden
- Unnabhängig vom genutzten Webserver etc.

Implementierung - Module

4 Module:

- Reverse Proxy zur Generierung/Sammlung von Reports
- Reverse Proxy zum Test einer Policy
- FastCGI Report Collector
- Generator

Implementierung - Reverse Proxies

Generierung

- Basiert auf Docker-Container `FROM nginx`
- Virtual Server mit
  - `fastcgi_pass 127.0.0.1:9000;` für `/csprg_collector.php`
  - `proxy_pass $SERVER;` für `/`
  - `add_header "Content-Security-Policy-Report-Only" "default-src 'none'; report-uri /csprg_collector.php;"`
- Port Forwarding vom Host auf Port 8080

Implementierung - Reverse Proxies

Testing/Production

- Basiert auf Docker-Container `FROM nginx`
- Virtual Server mit
  - `fastcgi_pass 127.0.0.1:9000;` für `/csprg_collector2.php`
  - `proxy_pass $SERVER;` für `/`
  - `add_header "Content-Security-Policy" $CSP`
- Port Forwarding vom Host auf Port 80

Implementierung - Collection

- Basiert auf Docker-Container `FROM php:fpm`
- Erhält Reports von Proxies und gibt diese an Generator weiter
- Nutzt Datei in Shared Memory für IPC

Reportstruktur - Chrome

```
{"csp-report":
  { "document-uri":"http://localhost:8080/wiki/
     Main_Page"
  , "referrer":""
  , "violated-directive":"default-src 'none'"
  , "effective-directive":"script-src"
  , "original-policy":"default-src 'none';
     report-uri /csprg_collector.php"
  , "blocked-uri":"inline"
  , "status-code":200
  }
}
```

Implementierung - Generierung

```
Input: Reports, Blacklist, Whitelist
=====================================
Reduce reports to
        ( report[effective-directive]
        , report[blocked-uri])
Check for and insert keywords
Check for blacklist matches
Starting with the whitelist fold reports into a
   Map (effective-directive -> List blocked-uri)
Flatten the Map into a policy
Return policy
```

Demo!

Inline Skripte

- `script-src 'unsafe-inline'` sollte unter allen Umständen vermieden werden
- Alternativen sind Noncen oder Hashes
- Noncen erforden Eingriff in übertragenes HTML
- Hashes sind nicht in Reports enthalten

$\Longrightarrow$ können keine Policies für Seiten mit Inline-Skripten erstellen

Reportquellen

- Bei Generierung aus Reports gesendet von normalen Seitenbesuchern: Angreifer kann beliebige Reports generieren
- Reports müssten also verifiziert werden
  $\Longrightarrow$ braucht exakte Kenntnisse der Seite
  $\Longrightarrow$ könnten Policy direkt erstellen
- Können also nur Reports aus vertrauenswürdigen Quellen verwenden

$\rightarrow$ Lösung hier: Dedizierter Server für Generierung

Persistent XSS

- Wird Generierung auf von Persistent XSS bereits betroffenen Seiten durchgeführt so wird dies durch CSP nicht verhindert
- Generierung muss auf sauberen Daten erfolgen
  - Testserver/-datensätze $\Longrightarrow$ Aufwand/nicht immer verfügbar
  - Generierung auf Seitensubset $\Longrightarrow$ erfasst möglicherweise nicht alle genutzten Ressourcen

Weitere Probleme

- Änderungen in 3^{rd} Partey Ressourcen
- Reportinkompatiblitäten
- Tools ohne CSP-Support

Zusammenfassung

- Haben Werkzeug zur Reportbasierten Erzeugung von Content Security Policies
- Für Grossteil von Websiten grundlegend reportbasierte Generierung jedoch nicht möglich
- Komplexere (reportbasierte) Generierungsmethoden häufig anfällig

$\Longrightarrow$ Keine wirkliche Alternative zur Generierung von Hand/durch direkte Analyse der Seite


GitHub https://github.com/malteklaassen/pg_ss2016


Content Security Policy Level 2; W3C Candidate Recommendation https://www.w3.org/TR/CSP2/ (Abgerufen 2016-06-15)