HTTP Strict Transport Security (HSTS)

Es handelt sich hierbei um ein Verfahren zur Verbesserung der Sicherheit in HTTPS-Verbindungen.

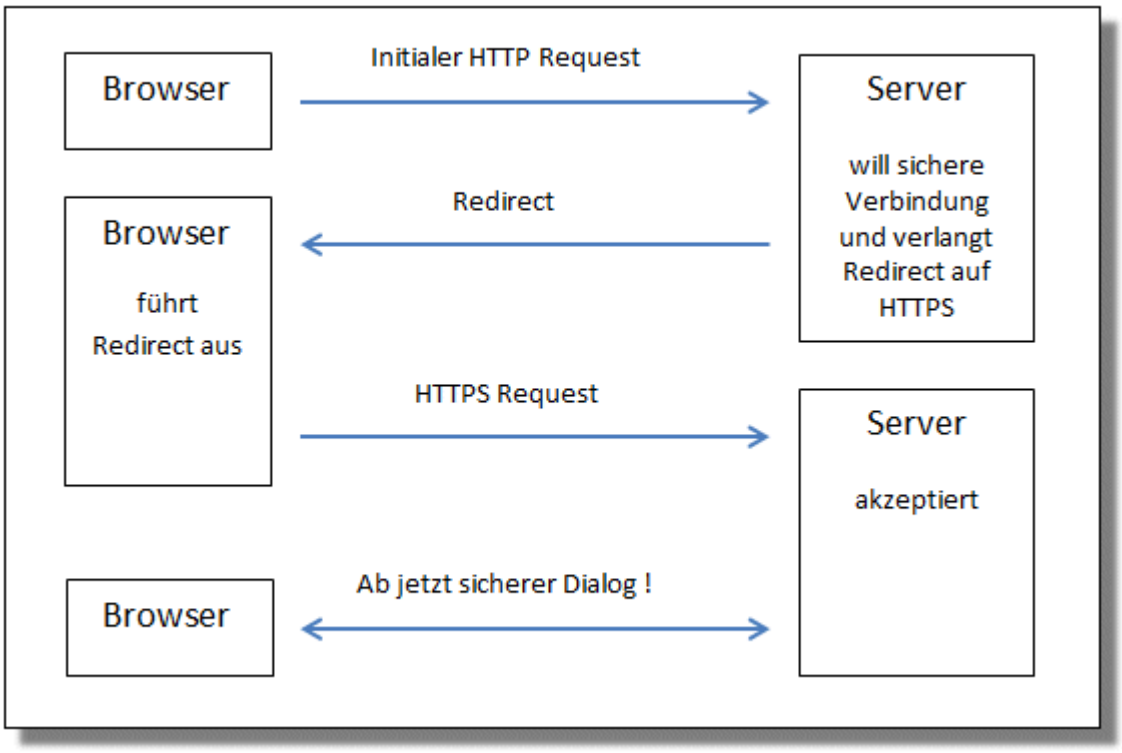
Der Web-Browser wird dabei angewiesen, die aktuelle und alle Unterseiten grundsätzlich mit HTTPS aufzurufen. Damit soll verhindert werden, dass in Zukunft ein Man-in-the-Middle (MitM) Angriff auf diese Verbindung Erfolg hat.

Der HSTS-Schutz ist nur dann erforderlich, wenn Webseiten, die eigentlich eine sichere Verbindung erfordern, beim ersten Aufruf mit einer unsicheren Verbindung gerufen werden. Das hat mit auch mit Unwissenheit oder Gleichgültigkeit auf der Nutzer-Seite zu tun.

Dieser Artikel richtet sich primär an Webserver-Administratoren. Am Ende der Seite werden aber auch für die Nutzer auf der Webclient-Seite einige Tipps für die grundsätzliche Verbesserung der Sicherheit bei Web-Verbindungen gegeben.

Typisches Szenario

1. Der Verbindungsaufbau wird über HTTP durchgeführt.
2. Der Server verlangt eine sichere Verbindung und fordert zu einem Redirect per HTTPS auf.

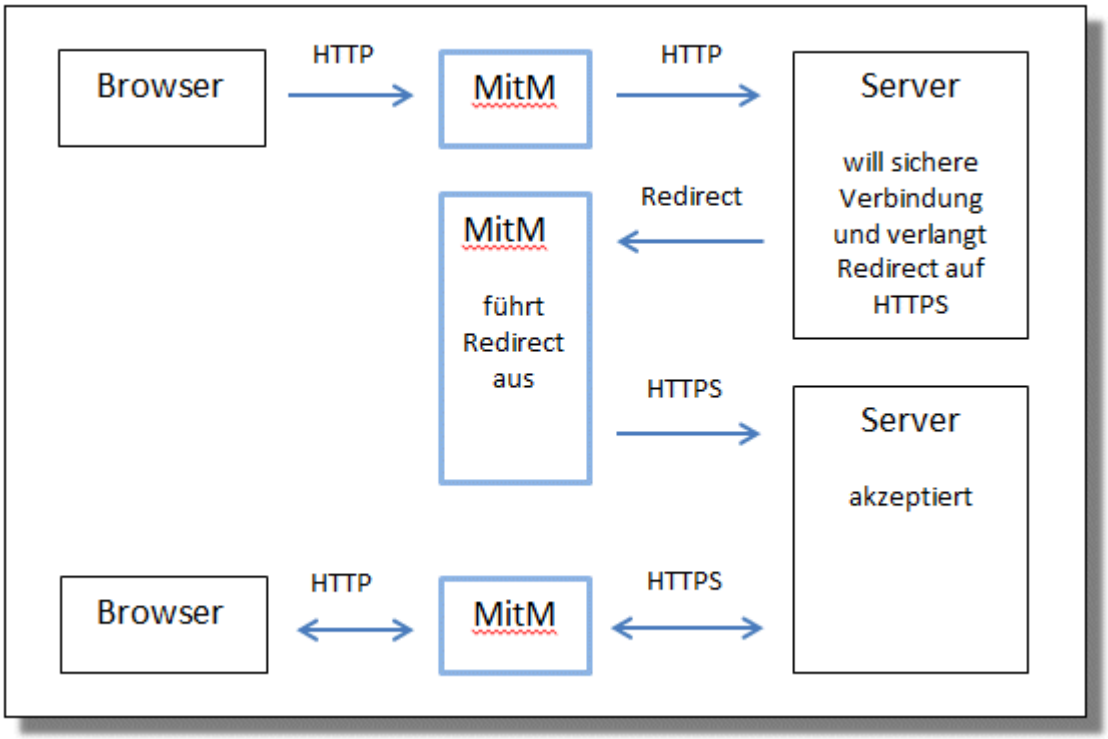


Mit SSL Stripping

Beim SSL Stripping fängt ein Man-in-the-Middle den Verbindungsaufbau ab und betreibt weiterhin zum Browser eine unsichere Verbindung, während er zum Server hin einen Browser mit sicherer Verbindung simuliert.

Wenn der Nutzer nicht erwartet, dass der Browser auf eine sichere Verbindung umschaltet, oder nichts von dieser Notwendigkeit weiß, stellt für ihn das Verbleiben in einer unsicheren Verbindung kein Problem dar - für seine Daten sehr wohl!

1. Der Verbindungsaufbau wird über HTTP durchgeführt.
2. Der Server verlangt eine sichere Verbindung und fordert zu einem Redirect per HTTPS auf.
3. MitM fängt den Redirect ab und führt ihn selbst durch.
4. MitM verkehrt mit dem Webclient über eine unsichere Verbindung und simuliert den Server während er über eine sichere Verbindung dem Server gegenüber die Rolle des Webclient einnimt.
5. MitM liest den Datenverkehr im Klartext mit.



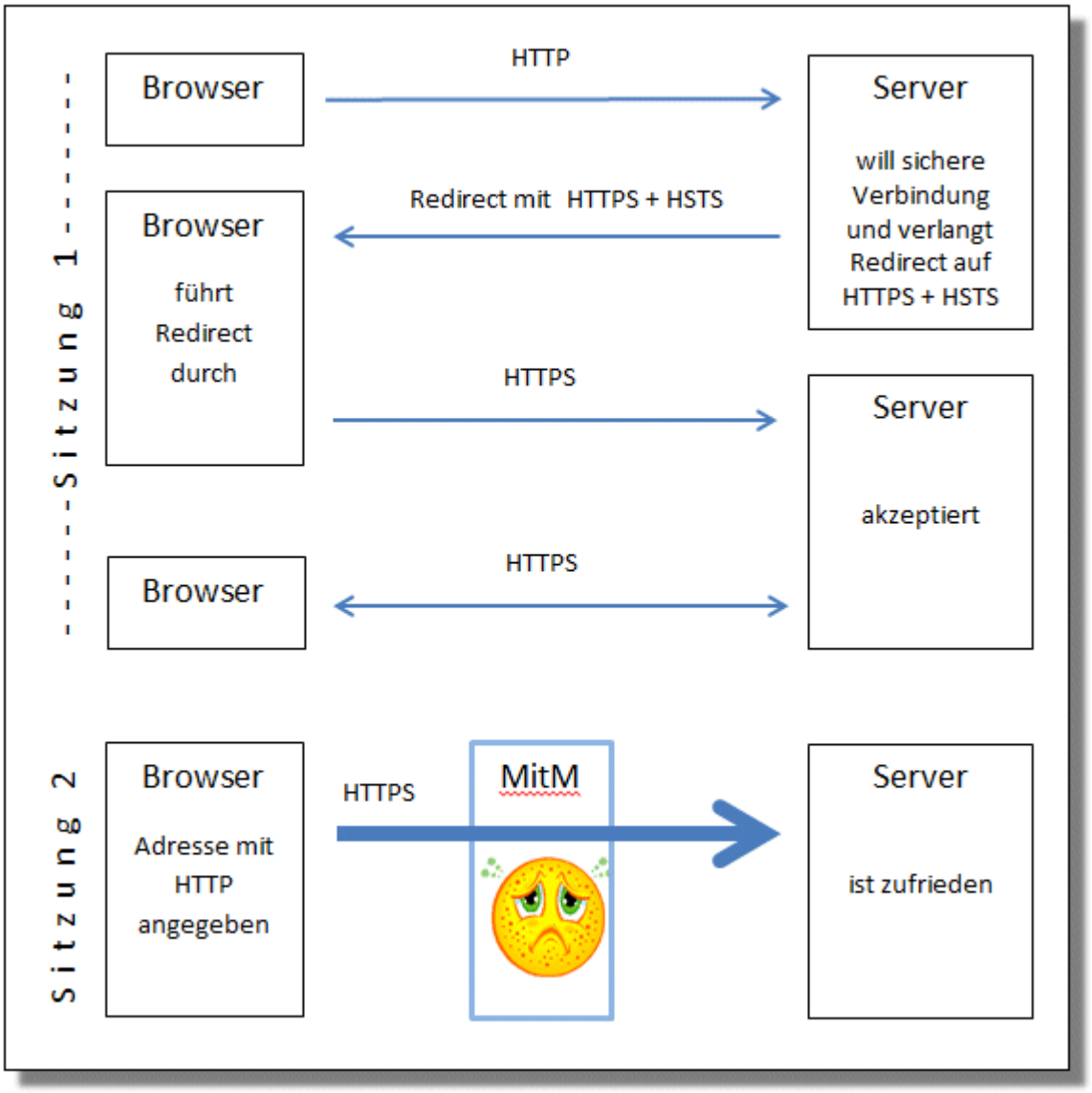
Mit HSTS

Beim HSTS-Verfahren wird mit Hilfe eines Response-Headers der Webclient dazu aufgefordert, ab sofort für eine gewisse Zeit die Verbindung zu diesem Server mit HTTPS durchführen, auch wenn der Nutzer die Adresse mit HTTP angegeben hatte.

Außerdem wird der Webclient in den Fällen, in denen er das Server-Zertifikat als „nicht

vertrauenswürdig“ einstuft, die Verbindung rigoros verhindern. Ohne HSTS würde er dem Benutzer die Entscheidung überlassen, eventuell trotzdem eine Verbindung zum Server herzustellen.

Jede HSTS-Verbindung muss von jedem Server (incl. Unterseiten) individuell abgesichert werden und kann nur greifen, wenn die initiale HTTP-Verbindung ohne MitM aufgebaut wird. Eine auf diese Weise „gehärtete“ Verbindung widersteht allerdings jedem weiteren MitM-Angriff.



Header-Syntax

Die Header-Anweisung, die der Server an den Webclient sendet, hat folgendes Format:

```
strict-transport-security max-age=xxx;includeSubDomains
```

Die Gültigkeitsdauer „max-age“ sollte sehr groß gewählt werden, möglichst ein Jahr (31536000) oder länger.

Mit dem zusätzlichen Parameter „includeSubDomains“ schließt man auch alle Subdomains mit ein.

Ein Original-Header sieht dann z.B. so aus:

```
HTTP/1.1 200 OK
Date                        Fri, 28 Jun 2013 06:10:04 GMT
Server                      Apache/2.2.22 (Win32) mod_ssl/2.2.22
OpenSSL/0.9.8t mod_jk/1.2.23
Strict-Transport-Security  max-age=31622400;includeSubDomains
Last-Modified               Thu, 01 Jan 1970 00:00:00 GMT
Keep-Alive                  timeout=5, max=100
Connection                  Keep-Alive
Transfer-Encoding           chunked
Content-Type                text/html
```

Einstellung für Apache 2

1. Das Header-Modul muss in httpd.conf geladen werden:

```
LoadModule headers_module modules/mod_headers.so
```

2. Der Host muss in extra/httpd-ssl.conf die Header-Anweisung erhalten:

```
<VirtualHost _default_:443>

#   SSL Engine Switch:
#   Enable/Disable SSL for this virtual host.
SSLEngine on

# HSTS Headerfür 366 Tage incl. Subdomains
Header always set Strict-Transport-Security "max-
age=31622400;includeSubDomains"

...

</VirtualHost>
```

Geht's noch sicherer?

Antwort: Auf jeden Fall!

Das vorliegende Verfahren HSTS (darauf wurde oben schon hingewiesen) ist ja darauf angelegt, beim Aufruf einer Webseite dem Webclient mitzuteilen, dass diese Verbindung in Zukunft immer mit HTTPS herzustellen ist, auch wenn der Benutzer nur die unsichere Variante HTTP wählt.

Damit ist sofort klar, dass der erste (unsichere!) Verbindungsaufbau von einem Man-in-the-Middle (MitM) abgefangen werden kann. In diesem Fall ist HSTS machtlos, denn der MitM wird die HSTS-Anweisung **garantiert nicht** an den Webclient weiterreichen.

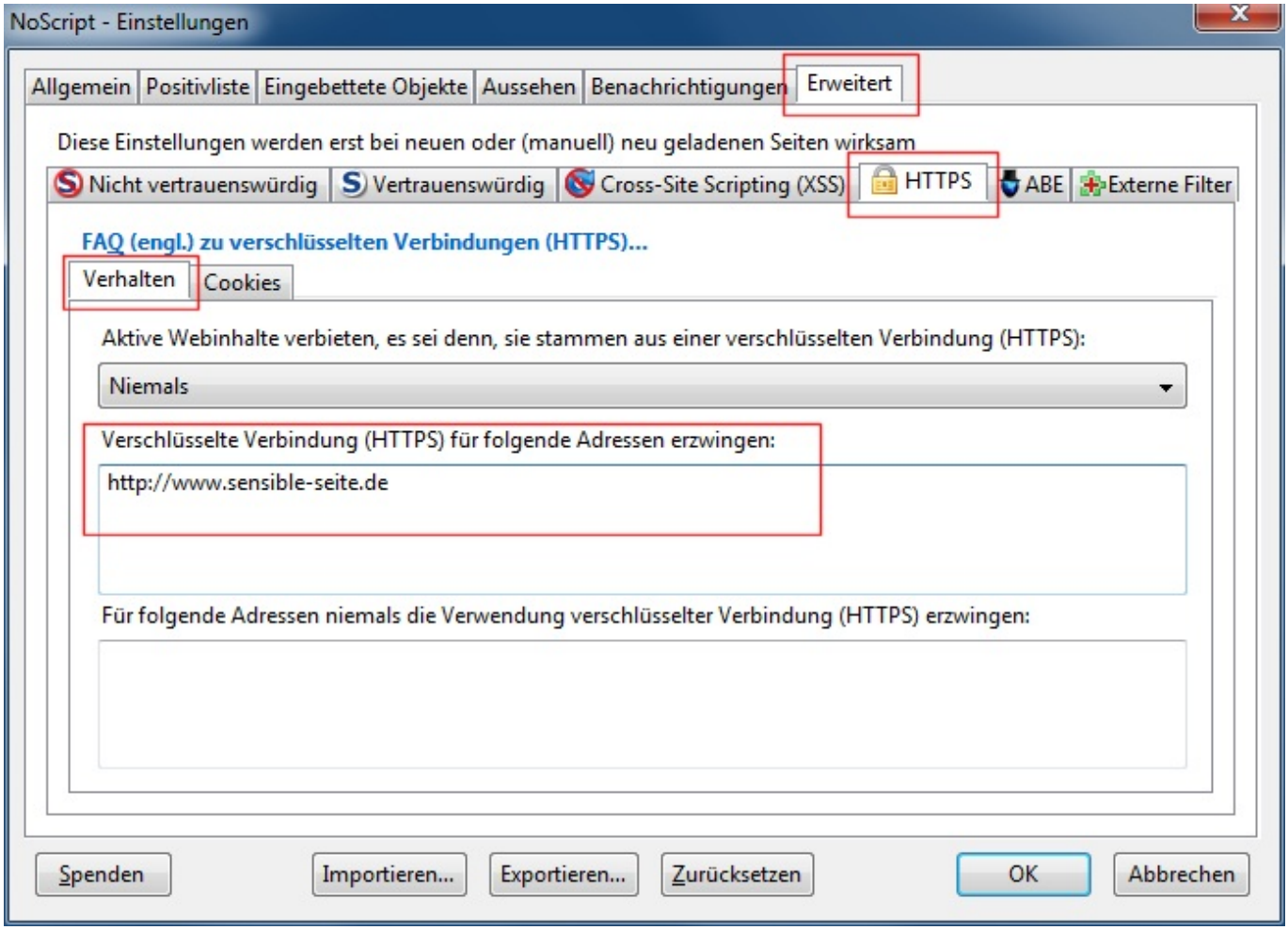
Deshalb hier einige Tipps für die Nutzer:

- Achten Sie darauf, dass Sie sensible Verbindungen - dazu gehören grundsätzlich solche, bei denen ein Login erforderlich ist - von vornherein mit HTTPS aufrufen.
- Achten Sie auf das Vorhängeschloss im Adressfeld. Es signalisiert eine verschlüsselte Verbindung.
- Verwenden Sie im sensiblen Umfeld Browser, die HSTS unterstützen. Das Whitepaper in den Quellenangaben [1] am Ende der Seite enthält eine Liste der HSTS-fähigen Browser.
- Nutzen Sie - falls möglich - Browser-AddOns, die sichere Verbindungen erzwingen können. Damit entfällt die problematische Phase des ersten unsicheren Aufrufs, bevor HSTS greift (s. nächstes Kapitel!).

Das Firefox-AddOn "NoScript"

Das Firefox-AddOn „NoScript“ ist primär dazu gedacht, den Nutzer vor unerwünschtem JavaScript zu schützen. Es ist aber auch noch auf einigen weiteren Gebieten der Web-Sicherheit aktiv, z.B. gegen Cross-Site Scripting (CSS) und kann eben auch HSTS gewissermaßen überflüssig machen, indem man an einer bestimmten Stelle Webadressen einträgt, zu denen eine HTTPS-Verbindung erzwungen werden muss.

Die zum vorliegenden Thema passende Konfigurationsseite sieht so aus:



Die Herstellerseite: **http://noscript.net/**

Diese Software ist sogar eine Spende wert!

Quellen

Zu empfehlen:
[1] http://www.securenet.de/fileadmin/papers/HTTP_Strict_Transport_Security_HSTS_Whitepaper.pdf

[2] http://blog.securenet.de/2012/11/02/ssl-stripping-die-ignorierte-gefahr/
[3] http://www.securenet.de/fileadmin/papers/HTTP_Strict_Transport_Security_HSTS_Verbreitung.pdf
[4] https://www.owasp.org/index.php/HTTP_Strict_Transport_Security
[5] https://www.zendas.de/themen/server/hsts_header.html

Sicherheit - Artikelübersicht, Netz - Artikelübersicht, Server - Artikelübersicht, Artikel zum tag: protokolle