Fili Wiese

ALLES, WAS SIE FÜR DIE UMSTELLUNG AUF HTTPS WISSEN MÜSSEN (Teil 2/3)

DER AUTOR

Fili Wiese ist ein renommierter SEO-Spezialist und hat früher in leitender Funktion im Google Search Quality Team mitgearbeitet. Bei SearchBrothers.com geht er mit Erfolg gegen die Abstrafung von Websites durch Google-Penalties vor und bietet SEO-Consulting mit SEO-Audits und SEO-Workshops.

Unter dem Eindruck der Enthüllungen von Edward Snowden und angesichts der Google-Initiative, mehr Websites zur Umstellung zu bewegen, wechseln immer mehr Websites zu HTTPS. Die Umstellung auf HTTPS ist eine technische Herausforderung, die man nicht unterschätzen sollte. Unter dem Aspekt der Suchmaschinenoptimierung muss man Ressourcen bereitstellen, langfristig planen, gründlich vorbereiten und den Umzug konsequent durchziehen; Risiken sind dabei generell nicht auszuschließen.

Im ersten Teil dieser kleinen Serie in der letzten Ausgabe ging es um die Vorbereitung der Umstellung auf HTTPS. Der SEO-Experte und ehemalige Googler aus dem Search Quality Team Fili Wiese erklärt Ihnen in diesem zweiten Teil umfassend, was bei der eigentlichen Umsetzung auf HTTPS unbedingt beachtet werden sollte.

Foto: Imillian / thinkstockphotos.de

Inhalte kopieren und aktualisieren

Duplizieren Sie den gesamten Inhalt der HTTP-Version einschließlich der XML-Sitemaps und aller Dateien auf den Speicherort der HTTPS-Version. In vielen Fällen müssen Sie dazu nur den Inhalt eines Verzeichnisses auf ein anderes Verzeichnis auf demselben Server kopieren.

Danach muss die HTTPS-Version aktualisiert werden. Wenn nicht ausdrücklich auf das Gegenteil hingewiesen wird, treten die nachfolgend vorgeschlagenen Veränderungen nur auf der HTTPS-Version in Kraft, nicht aber auf der noch aktiven HTTP-Version.

Kanonische URLs

Ändern Sie alle kanonischen URLs auf der HTTPS-Version in absolute HTTPS-URLs um.

```
<link href="http://www.example.com/deep/url" rel="canonical" />
```

wird zu

```
<link href="https://www.example.com/deep/url" rel="canonical" />
```

Vermeiden Sie die Verwendung relativer URLs in den kanonischen URLs. Gibt es aktuell keine kanonischen URLs auf der Website, richten Sie zuerst welche ein, bevor Sie fortfahren. Vergessen Sie auch nicht, die kanonischen URLs auf der mobilen Version der Website auf die HTTPS-Version umzuändern.

Seiten-Nummerierung

Sind die Seiten der Website nummeriert, ändern Sie diese auf der HTTPS-Version in absolute HTTPS-URLs um.

```
<link href="http://www.example.com/deep/url?page=1" rel="prev" />
<link href="http://www.example.com/deep/url?page=3" rel="next" />
```

wird zu

```
<link href="https://www.example.com/deep/url?page=1" rel="prev" />
<link href="https://www.example.com/deep/url?page=3" rel="next" />
```

Alternierende Annotationen

Es gibt eine Reihe von alternierenden Annotationen, die auf einer Website implementiert werden können. Alle diese Annotationen müssen geändert werden.

Hreflang

Verwendet die Website Hreflang-Annotationen in den XML-Sitemaps und/oder auf der Website, müssen diese auf der HTTPS-Version in absolute HTTPS-URLs umgeändert werden.

```
<link rel="alternate" hreflang="x-default" href="http://www.example.com/" />
<link rel="alternate" hreflang="es" href="http://www.example.com/es/" />
<link rel="alternate" hreflang="fr" href="http://www.example.com/fr/" />
```

wird zu

```
<link rel="alternate" hreflang="x-default" href="https://www.example.com/" />
<link rel="alternate" hreflang="es" href="https://www.example.com/es/" />
<link rel="alternate" hreflang="fr" href="https://www.example.com/fr/" />
```

Mobile Website

Gibt es eine eigene mobile Website, befinden sich wahrscheinlich alternierende mobile Annotationen auf der Website.

```
<link rel="alternate" media="only screen and (max-width: 640px)" href="http://m.example.com/page-1">
```

wird zu

```
<link rel="alternate" media="only screen and (max-width: 640px)" href="https://m.example.com/page-1">
```

Feeds

Auch alternierende Annotationen auf Atom-, RSS- oder JSON-Feeds müssen auf der Website geändert werden.

```
<link href="http://www.example.com/feed/rss/" rel="alternate" type="application/rss+xml" />
<link href="http://www.example.com/feed/atom/" rel="alternate" type="application/atom+xml" />
<link href="http://www.example.com/json.as" rel="alternate" type="application/activitystream+json" />
```

wird zu

```
<link href="https://www.example.com/feed/rss/" rel="alternate" type="application/rss+xml" />
<link href="https://www.example.com/feed/atom/" rel="alternate" type="application/atom+xml" />
<link href="https://www.example.com/json.as" rel="alternate" type="application/activitystream+json" />
```

Interne Links

Kommen auf Ihrer Website nur relative interne Links zum Einsatz, auch in Javascript und CSS-Dateien, können Sie diesen Schritt überspringen.

Interne Links sind für den Anwender und für Suchmaschinen wichtig, und die meisten Websites sind in hohem Maße

abhängig von Assets wie Javascript, CSS, Web-Fonts, Video- und Bilddateien einschließlich eines Favicons. Alle diese internen Links und Referenzen sind in der gesamten HTML-Quelle zu finden; sie können auch interne Links innerhalb der Assets beinhalten, wie beispielsweise Verweise auf Bilddateien in CSS-Dateien oder interne URLs in Javascript-Dateien.

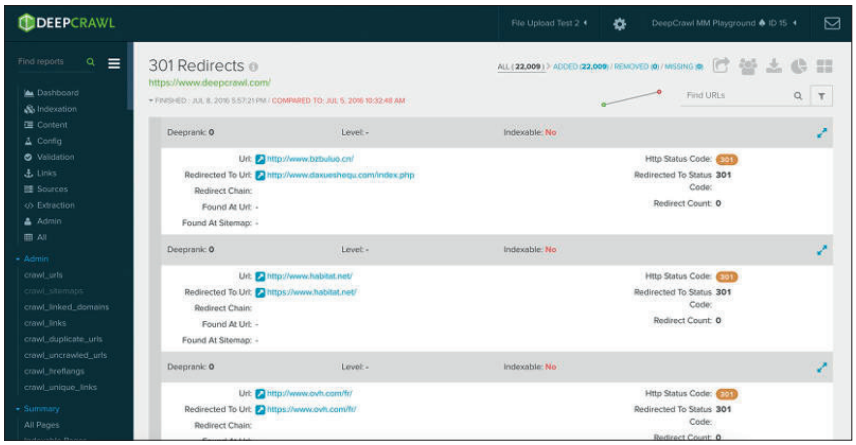

Abb. 1: Beispiel für alle von DeepCrawl gefundenen Weiterleitungen

Es müssen zum Beispiel die folgenden internen Links umgeändert werden:

» Links auf andere interne URLs im HTML-Quellcode;
» Links auf interne Bilddateien im HTML-Quellcode;
» Links auf interne Videodateien im HTML-Quellcode;
» Links auf interne Web-Fonts im HTML-Quellcode;
» Links auf interne Javascript-Dateien im HTML-Quellcode;
» Links auf andere interne URLs in den Javascript-Dateien;
» Links auf interne Bilddateien in den Javascript-Dateien;
» Links auf interne CSS-Dateien in den Javascript-Dateien;
» Links auf interne CSS-Dateien im HTML-Quellcode;
» Links auf interne CSS-Dateien im HTML-Quellcode;
» Links auf interne Bilddateien in den CSS-Dateien;
» Links auf interne Web-Fonts in den CSS-Dateien;
» sowie andere interne Links.

Um dieses Update durchzuführen, gibt es verschiedene Möglichkeiten.

Option 1

Stellen Sie um auf die ausschließliche Verwendung relativer URLs; zum Beispiel:

```
<a href="http://www.example.com/">home</a>
```

wird zu

```
<a href="/">home</a>
```

Diese Option stellt möglicherweise mit internen Links auf Assets im Konflikt, vor allem, wenn diese in CSS und/oder Javascript-Dateien definiert sind. Es kann außerdem bei dieser Option sinnvoll sein, eine Base-Tag-URL oben in der KOPFZEILE des HTML-Quellcodes zu definieren.

```
<base href="https://www.example.com" />
```

Option 2

Ändern Sie das Protokoll für absolute interne URLs von HTTP auf HTTPS um, beispielsweise:

```
<a href="http://www.example.com/">home</a>
```

wird zu

```
<a href="https://www.example.com/">home</a>
```

Option 3

Löschen Sie das Protokoll für absolute interne URLs, beispielsweise:

```
<a href="http://www.example.com/">home</a>
```

wird zu

```
<a href="//www.example.com/">home</a>
```

Mit dieser Option werden die Links vom Protokoll der besuchten URL abhängig gemacht.

Für Suchmaschinen und Endanwender macht es eigentlich keinen Unterschied, welche der drei vorgenannten Optionen angewendet wird: In der Regel sind Suchmaschinen-Bots und Browser intelligent genug, die finale absolute URL zu identifizieren. Dennoch ist Option 2 die sicherste Wahl.

WordPress

Für Websites, die auf der beliebten WordPress-Plattform laufen, kann das Plug-in „Better Search Replace WordPress" oder das Script „search and replace the database" für ein schnelles Update der internen Links in der Datenbank nützlich sein. Vergessen Sie dabei aber nicht, auch die Themen-Dateien und die allgemeinen Einstellungen zu ändern. Außerdem sollten Sie sich über die offiziellen Best-Practice-Empfehlungen für die Umstellung auf HTTPS für WordPress informieren.

Interne Weiterleitungen

Wenn einer der internen Links auf eine interne Weiterleitung zu einer anderen internen URL verweist, ist es ratsam, die Weiterleitungskette so kurz wie möglich zu halten und stattdessen die interne Verlinkungsstruktur dadurch zu verbessern, dass sie direkt mit den kanonischen URLs auf der HTTPS-Empfängerseite und nicht mit der internen Weiterleitung verlinkt wird.

Zusätzlich muss die interne Verlinkungsstruktur so geändert werden, dass sie auf die korrekten URLs verweist; damit werden Weiterleitungsketten umgangen. Vermeiden Sie beispielsweise die Situation, dass eine HTTPS-URL (a) auf eine HTTP-URL (b) verlinkt, die dann auf eine andere HTTPS-URL (c) weiterleitet oder schlimmstenfalls gar auf die ursprüngliche HTTPS-URL (a).

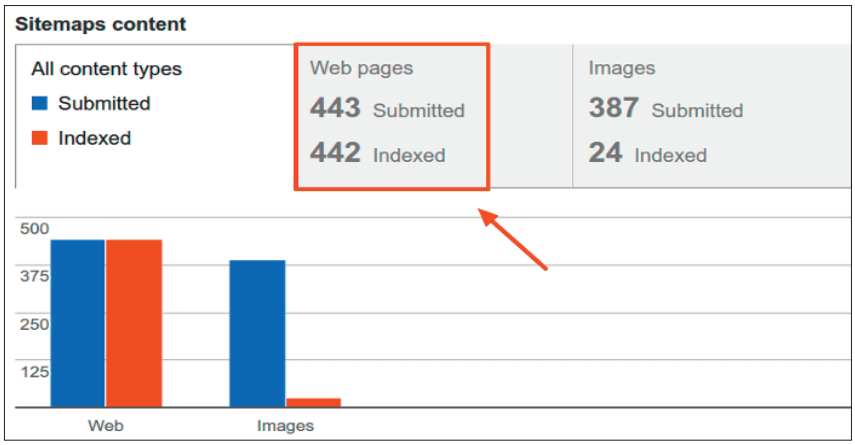

Abb. 2: Beispiel für die von Google indexierte Anzahl von URLs

Änderung von CDN-Einstellungen

Links auf Assets werden häufig zum Rendern einer URL wie beispielsweise Javascript-, Image- und CSS-Dateien verwendet, die von einem CDN (Content Distribution Network) heruntergeladen werden können, das nicht notwendigerweise unter der Kontrolle des Eigentümers der Website stehen muss. Links auf die vom CDN heruntergeladenen Assets müssen von HTTPS aus geladen werden. Auch hier kann man das Protokoll der absoluten URL löschen, siehe das Beispiel:

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js"></script>
```

wird zu

```
<script src="//ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js"></script>
```

Das heißt, dass das CDN für die Bereitstellung der Assets über HTTPS aktiviert werden muss. Wenn das CDN auf eine Subdomain der Website abgebildet wird und – was wahrscheinlich der Fall sein dürfte – unter der Kontrolle des Eigentümers der Website steht, müssen möglicherweise dieselben SSL-Zertifikate auf das CDN hochgeladen und für jede Anforderung verwendet werden (je nach Art des SSL-Zertifikats).

Ist die Original-Quelldatei für das Asset auf der HTTPS-Version verfügbar und – in den meisten Fällen – mit der CDN-Version auf HTTPS verlinkt, muss unbedingt darauf geachtet werden, die CDN-Version auf HTTPS mithilfe von HTTP-Headern zurück zur Quelldatei der Assets auf HTTPS zu kanonisieren. Im folgenden Beispiel muss das von der HTTPS-Version aus verlinkte Asset im CDN:

```
<img src="//cdn.example.com/image1.png" />
```

in der Antwort des HTTP-Headers eine Link-Referenz mit einer kanonischen URL an die Asset-Quelldatei auf der HTTPS-Version zurücksenden:

Link: `<https://www.example.com/image1.png/>; rel="canonical"`

Das zeigt den Suchmaschinen, dass die Asset-Quelldatei auf der HTTPS-Version die Originalversion ist; dies verhindert potenzielle Duplizitätsprobleme mit der CDN-Asset-Datei.

Um unerwartete potenzielle Konflikte auf der Website zu vermeiden, vergessen Sie beim Ändern der CDN-Einstellungen nicht, die Daten im Cache des CDN zu löschen.

XML-Sitemaps

Wenn noch eine XML-Sitemap von vor der Umstellung existiert (ist dies nicht der Fall, generieren/exportieren Sie eine Sitemap auf der Grundlage des ersten Crawl), muss sie von der HTTP-Version aus zugänglich sein. Bleiben die ursprünglichen XML-Sitemaps auf der HTTP-Version aktiv, haben Sie die Möglichkeit, den Indexierungsstatus auf der Google Search Console nachzuverfolgen (unter „Sitemaps"); dies ist nützlich, während die alten URLs gecrawlt und auf der HTTPS-Version neu indexiert werden.

Achten Sie darauf, dass in den XML-Sitemaps auf der HTTP-Version keine Weiterleitungen oder nicht vorhandene oder nicht indexierbare URLs aufgelistet sind, sondern nur kanonische URLs von indexierbaren Seiten. Ist dies nicht der Fall, ist die Einreichung im Vergleich zu der Anzahl der indexierten URLs nicht mehr zuverlässig.

Änderung der XML-Sitemaps

Kopieren Sie dann die XML-Sitemaps der HTTP-Version und speichern Sie sie als neue Dateien. Ändern Sie das Protokoll jeder in dem Block aufgelisteten URL in den neuen XML-Sitemaps. Beispiel:

```
<url>
    <loc>http://www.example.com/</loc>
    <xhtml:link rel="alternate" hreflang="x-default" href="http://www.example.com/" />
    <xhtml:link rel="alternate" hreflang="es" href="http://www.example.com/es/" />
    <xhtml:link rel="alternate" hreflang="fr" href="http://www.example.com/fr/" />
    <xhtml:link rel="alternate"
media="only screen and (max-width: 640px)"
href="http://m.example.com/" />
<image:image>
            <image:loc>http://www.example.com/image.jpg</image:loc>
</image:image>
<video:video>
            <video:content_loc>
http://www.example.com/video123.flv
</video:content_loc>
</video:video>
</url>
```

wird zu

```
<url>
    <loc>https://www.example.com/</loc>
    <xhtml:link rel="alternate" hreflang="x-default" href="https://www.example.com/" />
    <xhtml:link rel="alternate" hreflang="es" href="https://www.example.com/es/" />
    <xhtml:link rel="alternate" hreflang="fr" href="https://www.example.com/fr/" />
    <xhtml:link rel="alternate"
    media="only screen and (max-width: 640px)"
    href="https://m.example.com/" />
    <image:image>
            <image:loc>https://www.example.com/image.jpg</image:loc>
    </image:image>
    <video:video>
            <video:content_loc>
    https://www.example.com/video123.flv
    </video:content_loc>
    </video:video>
    </url>
```

Dieses Beispiel gilt unter der Annahme, dass der gesamte Inhalt auf die HTTPS-Version umgestellt wurde. Ist das nicht der Fall, ändern Sie die entsprechenden URLs erst jetzt.

Erstellen Sie, soweit möglich, XML-Sitemaps für jede Subdomain oder jeden Unterabschnitt der Website auf der HTTP-Version wie auch auf der HTTPS-Version; ändern Sie dann die HTTPS-Version entsprechend. Bei großen Websites ist es überlegenswert, die verschiedenen XML-Sitemaps für jede Subdomain oder jeden Unterabschnitt mithilfe von XML-Sitemap-Indexdateien in Gruppen zusammenzufassen. Das wird Ihnen später dabei helfen, die Indexierungsnummern der HTTP- und der HTTPS-Versionen nachzuverfolgen.

Resource Hints

Arbeitet die Website mit Resource Hints wie beispielsweise dns-prefetch, preload, preconnect, prerender, prefetch etc., müssen auch diese aktualisiert werden. Beispiel:

```
<link rel="preconnect" href="http://cdn.example.com" pr="0.42">
```

wird zu

```
<link rel="preconnect" href="https://cdn.example.com" pr="0.42">
```

oder

```
<link rel="preconnect" href="//cdn.example.com" pr="0.42">
```

Prüfen Sie sorgfältig, ob Resource Hints in den HTTP-Headern, im HEAD von HTML oder im Javascript-Code enthalten sind.

CSS und Javascript

Assets sind für die meisten Websites von elementarer Bedeutung, beispielsweise CSS für Stil und Javascript für Interaktion. Viele SEOs denken beim Migrieren von Inhalten nicht daran, dass derartige Assets oft andere Assets importieren oder laden können, beispielsweise andere CSS- oder Javascript-Dateien auf dem gleichen oder anderen Servern.

```
@import ‚http://fonts.googleapis.com/css?family=Open+Sans‘;
```

wird zu

```
@import ‚https://fonts.googleapis.com/css?family=Open+Sans‘;
```

Durchsuchen Sie alle CSS- und Javascript-Dateien nach dem Pattern „http://“ und probieren Sie aus, ob Sie dieses durch „https://“ ersetzen können. Steht bei einem geladenen Asset „//“ am Anfang des URL-Patterns, ist dieses Asset sowohl auf HTTP wie auch auf HTTPS verfügbar; der Browser wird automatisch die Version anfordern, der derjenigen der geladenen Website entspricht (nach Abschluss der Umstellung auf die HTTPS-Version).

Nicht alle Assets sind standardmäßig auf HTTPS verfügbar; das gilt vor allem für die von externen Quellen geladenen oder importierten Assets. Prüfen Sie in einem solchen Fall, ob über die externe Quelle eine andere URL auf HTTPS verfügbar ist. Eine andere Möglichkeit ist, das Asset auf denselben Server zu kopieren wie die Website und es von dort in die Codebasis zu importieren/zu laden und/oder eine andere externe Quelle oder ein anderes Asset zu finden, die bzw. das Sie importieren/laden können.

HTTP-Header

HTTP-Header sind ein extrem mächtiges Tool für die Weitergabe von SEO-Signalen an Suchmaschinen und bedeuten dabei minimalen Mehraufwand in der Codebasis. In vielen Fällen werden die Link-Annotationen in PHP-Code oder in den .htaccess-Dateien von Apache-Servern etc. gespeichert. Für .htaccess kann man beispielsweise wie folgt vorgehen:

```
<Files testPDF.pdf >
```

Header mit zugefügtem Link `,http://www.example.com/>; rel="canonical"‘`

```
</Files>
```

Denken Sie immer daran, die HTTP-Header der Website nach Links zu durchsuchen; alle gefundenen Links müssen dann entsprechend geändert werden, beispielsweise:

Link: `<http://www.example.com/es/>; rel="alternate"; hreflang="es"`

Link: `<http://www.example.com/>; rel="canonical"`

Link: `<http://example.com>; rel=dns-prefetch`

wird zu

Link: `<https://www.example.com/es/>; rel="alter-`

```
nate“; hreflang=“es“
```

Link: `<https://www.example.com/>; rel=“canonical“`

Link: `<https://example.com>; rel=dns-prefetch`

oder

Link: `<//www.example.com/es/>; rel=“alternate“; hreflang=“es“`

Link: `<//www.example.com/>; rel=“canonical“`

Link: `<//example.com>; rel=dns-prefetch`

Strukturierte Daten

Suchmaschinen bevorzugen strukturierte Daten, und in vielen Fällen kommen die SEOs diesem Wunsch gern nach – in der Hoffnung, dass die Suchmaschinen die Inhalte dann besser verstehen und die Sichtbarkeit der Website bei der organischen Suche erhöhen. Zum jetzigen Zeitpunkt ist Schema.org die Standardoption und der wichtigste Speicher für strukturierte Daten. Glücklicherweise wird der Inhalt von Schema.org sowohl auf HTTP als auch auf HTTPS unterstützt und kann somit in der Codebasis verwendet werden.

Ändern Sie alle Verweise auf absolute URLs in den auf Ihrer Website eingesetzten strukturierten Daten sowie alle Schema.org-Verweise auf HTTPS. Beispielsweise:

```
{
    „@context“: „http://schema.org“,
    „@type“: „WebSite“,
    „name“: „Your WebSite Name“,
    „alternateName“: „An alternative name for your WebSite“,
    „url“: „http://www.your-site.com“
}
```

wird zu

```
{
    „@context“: „https://schema.org“,
    „@type“: „WebSite“,
    „name“: „Your WebSite Name“,
    „alternateName“: „An alternative name for your WebSite“,
    „url“: „https://www.example.com“
}
```

Man kann „//“ für die URL in diesem Beispiel verwenden, nicht aber für den Kontext-Verweis. Diese Variante hier funktioniert z. B. auch:

```
{
    „@context“: „https://schema.org“,
    „@type“: „WebSite“,
    „name“: „Your WebSite Name“,
    „alternateName“: „An alternative name for your WebSite“,
    „url“: „//www.example.com“
}
```

Das Tool Google Structured Data Testing lässt jedoch dies nicht zu:

```
{
    „@context“: „//schema.org“,
    „@type“: „WebSite“,
    „name“: „Your WebSite Name“,
    „alternateName“: „An alternative name for your WebSite“,
    „url“: „//www.example.com“
}
```

Suchen Sie nach JSON-LD, Microdata, RDFa oder anderen möglichen Verweisen auf strukturierte Daten in der Codebasis; sind welche vorhanden, ändern Sie das Protokoll jeder URL, auf die in den strukturierten Daten verwiesen wird, auf HTTPS.

RSS-/Atom-Feeds

Die RSS- und/oder Atom-Feeds einer Website werden ebenfalls häufig übersehen. Wenngleich die Nutzung von RSS-/Atom-Programmen seit der Einstellung des Google Reader zurückgegangen ist, kommen auch heute noch eine Reihe von alternativen Feed-Reader-Programmen zum Einsatz oder andere Programme, die mit Syndikation arbeiten.

Suchen Sie nach Feeds auf der Website. Wenn welche vorhanden sind, prüfen Sie auf der HTTPS-Version, ob die HREF-Annotationen (der Link auf den Artikel) zum Inhalt und die Verweise auf In-Content-Links (ein Link in einem Artikel) auf HTTPS geändert wurden. Ist dies nicht der Fall, müssen Sie möglicherweise – je nachdem, auf welcher Plattform die Feeds generiert werden – mit den Web-Developern oder dem IT-Team reden und alle Links auf absolute HTTPS-URLs umstellen (hier reicht es nicht, einfach „//“ zu verwenden, weil man nicht weiß, wohin der Inhalt syndiziert wurde und ob diese Website unter HTTP oder HTTPS läuft).

Accelerated Mobile Pages

Ist die Website AMP-fähig, müssen die Link-Verweise auf AMP-URLs im Quellcode auf die absolute HTTPS-Version umgestellt werden. Beispiel:

```
<link rel=“amphtml“ href=“http://www.example.com/amp/“>
```

wird zu

```
<link rel=“amphtml“ href=“https://www.example.com/amp/“>
```

Auch etwaige interne Links, Link-Verweise, kanonische URLs, Asset-Links etc. im Quellcode der AMP-Seiten müssen auf die entsprechende HTTPS-Version umgestellt werden.

Nähere Informationen über AMP finden Sie unter *„AMP Project“*.

Cookies

Es ist auch wichtig, dass keine ungesicherten Cookies versendet werden. Wird dieses Gebot missachtet, kann es passieren, dass die Daten in einem Cookie, z. B. die Authentifizierungsdaten, für jedermann im Klartext sichtbar sind. Prüfen Sie die Servereinstellung sorgfältig, damit Ihre Cookies sicher sind. Im folgenden Beispiel wird die Datei php.ini mit PHP geprüft:

```
session.cookie_secure = True
```

Nehmen Sie in der Datei web.config die folgende Einstellung mit ASP.NET vor:

```
<httpCookies requireSSL="true" />
```

Zur Prüfung der Einstellung gehen Sie auf eine Seite, die ein neues Cookie setzt, und suchen in den HTTP-Headern den folgenden Code:

```
Set-Cookie: mycookie=abc; path=/secure/; Expires=12/12/2018; secure; httpOnly;
```

Umstellung auf HTTPS

Sie haben den Inhalt vorbereitet und für die HTTPS-Version geändert; nun kann die Website umgestellt werden.

HTTPS-Version crawlen

Bevor Sie die Umstellung von der HTTP- auf die HTTPS-Version abschließen und die HTTPS-Version Ihrer Website für die ganze Welt einschließlich Googlebot live gehen lassen, müssen Sie eine Sicherheitsüberprüfung durchführen.

Crawlen Sie die gesamte HTTPS-Version und halten Sie dabei Ausschau nach:

- CSS-, Bild-, Javascript-, Flash-, Video-, Audio-, Iframe-Dateien oder Fonts, die ungesichert über HTTP und nicht über HTTPS geladen werden;
- Weiterleitungen auf die HTTP-Version;
- Interne Links, kanonische URLs, hreflang und/oder strukturierte Daten etc., die auf die HTTP-Version verweisen;
- 40x- oder 50x-Fehler in den Server-Log-Dateien für die HTTPS-Version.

Stellen Sie keine Fehler fest, machen Sie mit dem nächsten Schritt weiter.

Wichtig: Crawlen Sie ausschließlich die HTTPS-Version, nicht die HTTP-Version.

XML-Sitemap aktualisieren (schon wieder)

Extrahieren Sie alle beim Crawlen der HTTPS-Version gefundenen URLs und vergleichen Sie diese Liste mit den in den XML-Sitemaps aufgeführten URLs. Stellen Sie fest, welche URLs auf der HTTPS-Version aktiv und indexierbar sind, aber nicht in den XML-Sitemaps vorkommen. Ergänzen Sie die XML-Sitemaps mit den fehlenden indexierbaren URLs, die Sie beim Crawlen der HTTPS-Version gefunden haben.

Verweise

Nachdem der gesamte Inhalt geändert und auf den neuen Standort verschoben wurde, müssen neue Weiterleitungsregeln implementiert werden, damit der gesamte HTTP-Traffic zu den entsprechenden HTTPS-Versionen geleitet wird. Als einfache pauschale Lösung für Apache bietet sich im .htaccess der HTTP-Version an:

```
RewriteEngine On
RewriteRule ^(.*)$ https://www.example.com/$1 [R=301,L]
```

Damit werden zugrunde liegende Patterns auf der HTTP-Version auf die HTTPS-Version umgeleitet. Das aber führt dazu, dass Sie nicht mehr auf robots.txt und die XML-Sitemaps auf der HTTP-Version zugreifen können, weil diese pauschale Weiterleitungsregel jede Anforderung dieser Dateien zur HTTPS-Version umleitet. Um das zu vermeiden, muss zusätzlich eine Ausnahmeregel programmiert werden. In .htaccess für Apache auf der HTTP-Version kann das beispielsweise so aussehen:

```
RewriteEngine On
RewriteRule (robots.txt|sitemap.xml)$ - [L]
RewriteRule ^(.*)$ https://www.example.com/$1 [R=301,L]
```

Dies leitet jede Anfrage an die HTTP-Version zu der HTTPS-Version weiter, mit Ausnahme von Anfragen nach robots.txt und Sitemap.xml-Dateien.

Umstellung mithilfe kanonischer URLs

Warten Sie bei der Umstellung von Inhalten mithilfe kanonischer URLs mit der Implementierung der Weiterleitungsregeln, bis ein ausreichender Anteil des kritischen Inhalts indexiert ist und von der HTTPS-Version bereitgestellt wird. Sobald Googlebot festgestellt hat, dass der Inhalt komplett oder größtenteils auf HTTPS ist, kann die Weiterleitungsregel live geschaltet werden, damit Googlebot und der Besucher automatisch auf die HTTPS-Version gelangen.

Weiterleitungsketten möglichst klein halten

Prüfen Sie bei der Implementierung der neuen Weiterleitungsregeln die alten Regeln gründlich und aktualisieren Sie sie so, dass sie direkt auf das endgültige Ziel unter HTTPS verweisen. Vermeiden Sie Weiterleitungsketten wie: HTTP A verweist auf HTTP B, die wiederum zu HTTPS C weiterleitet.

Denken Sie auch daran, dass manche Systeme nachgestellte Schrägstriche hinzufügen oder löschen, indem sie sie zu der anderen Variation auf demselben Protokoll weiterleiten, was zusätzliche Weiterleitungen zur Folge hat. Beispiel:

http://www.examples.com/dir leitet weiter auf *http://www.examples.com/dir/* wird weitergeleitet auf *https://www.examples.com/dir*, leitet wiederum weiter zum endgültigen Ziel *https://*

www.examples.com/dir/.

Die Weiterleitung auf eine der folgenden URLs ist effizienter:

http://www.examples.com/dir

http://www.examples.com/dir/

https://www.examples.com/dir

Direkt auf:

https://www.examples.com/dir/

Wenn Sie die neuen Weiterleitungsregeln definieren, prüfen Sie, ob es sinnvoll ist, den nachgestellten Schrägstrich im Standard-Ausdruck für die Weiterleitungsregel optional zu machen. Beispiel:

```
RewriteEngine On
RewriteRule ^(dir[\/]?)$ https://www.example.com/dir/ [R=301,L]
```

„Nackte" Domain oder WWW

Wählen Sie beim Schreiben der neuen Weiterleitungsregeln einen primären Host-Namen und legen Regeln für die Weiterleitung der nicht primären auf die primäre Version auf HTTPS fest. Wenn die primäre HTTPS-Version beispielsweise der WWW Host-Name ist, sollten Sie auch alle „nackten" Domain-URLs auf der HTTPS-Version auf den primären WWW Host-Namen auf der HTTPS-Version weiterleiten:

```
RewriteEngine On
RewriteCond %{HTTP_HOST} ^example.com [NC]
RewriteRule ^(.*)$ https://www.example.com/$1 [R=301,L]
```

Sind alle neuen Weiterleitungsregeln auf der HTTPS-Version live, machen Sie mit dem nächsten Schritt weiter.

HTTP-Version aktualisieren (schon wieder)

Jetzt müssen Sie die zu einem früheren Zeitpunkt extrahierten URLs der Server-Log-Dateien, XML-Sitemaps und des Crawlens der HTTP-Version finden. Suchen Sie nach Dateinamen wie in den Beispielen unten:

» logs_extracted_urls.csv
» sitemap_extracted_urls.csv
» crawl_extracted_urls.csv

Crawlen Sie mithilfe eines Crawlers wie Screaming Frog SEO Spider jede URL und vergewissern Sie sich, dass alle Weiterleitungen wunschgemäß funktionieren und jede URL auf der HTTP-Version korrekt auf die entsprechende HTTPS-Version weiterleitet.

Funktioniert alles richtig, machen Sie mit dem nächsten Schritt weiter.

robots.txt ersetzen

Jetzt müssen Sie die Datei robots.txt auf der HTTPS-Version aktualisieren.

Kopieren Sie die Datei robots.txt von der HTTP-Version auf die HTTPS-Version und ändern Sie die Sitemap-Referenz auf die neue Sitemap-Datei. Beispiel:

```
User-Agent: *
Disallow:
Sitemap: https://www.example.com/sitemap.xml
```

Google Search Console konfigurieren

Nachdem Sie den Inhalt auf die HTTPS-Version umgestellt , die Weiterleitungen auf der HTTP-Version eingerichtet und die XML-Sitemaps und die Datei robots.txt aktualisiert haben, ist jetzt der Zeitpunkt gekommen, zur Google Search Console zu gehen und Google über die Änderungen in Kenntnis zu setzen.

Site-Variationen hinzufügen

In der Google Search Console müssen mindestens die folgenden vier Variationen des Domain-Namens vorhanden sein:

http://example.com

http://www.example.com

https://example.com

https://www.example.com

Überprüfen Sie dies und fügen Sie die fehlenden Namen in der Google Search Console hinzu.

Verwendet die Website Subdomains oder getrennt zur Google Search Console hinzugefügte Unterverzeichnisse, müssen auch diese sowohl für die HTTP- als auch für die HTTPS-Versionen hinzugefügt werden. Beispiel:

http://m.example.com

https://m.example.com

Properties in Sätze gruppieren

Seit Mai 2016 bietet Google Search Console die Möglichkeit, die Daten von Properties in einen Satz zu gruppieren. Für die Umstellung auf HTTPS ist das äußerst hilfreich. Fügen Sie also in der Google Search Console einen Satz für jede relevante HTTP- und HTTPS-Property hinzu. Gruppieren Sie beispielsweise folgende Properties in einen Satz:

http://example.com/

https://example.com/

http://www.example.com/

https://www.example.com/

Wenn Sie Subdomains und/oder Unterverzeichnisse für eine bestimmte geografische Ausrichtung verwenden, fügen Sie für jede relevante HTTP- und HTTPS-Version weitere Sätze für jedes geografische Ziel hinzu, beispielsweise:

Satz 1:

http://www.example.com/nl/

https://www.example.com/nl/

Satz 2:

http://de.example.com/

https://de.example.com/

Überprüfung von Fetch and Render

Um sicherzustellen, dass für Googlebot alles wunschgemäß funktioniert, verwenden Sie das Tool „Fetch and Render“ in der Google Search Console zum Abrufen und Rendern:

» Gehen Sie zur Homepage der HTTP-Version und prüfen Sie, ob die Weiterleitung richtig funktioniert. Falls ja, klicken Sie auf „Submit to Index“.
» Prüfen Sie auf der Homepage der HTTPS-Version, ob die Weiterleitung richtig funktioniert. Falls ja, klicken Sie auf „Submit to Index“ und wählen „Crawl this URL and its direct links“, wenn Sie dazu aufgefordert werden.

Hinweis: Mit der Vorlage des Index wird außerdem Googlebot auf die HTTPS-Version hingewiesen und aufgefordert, diese zu crawlen.

Manuelle Aktionen prüfen

Prüfen Sie als allererstes sorgfältig, ob die Migration der Website möglicherweise durch manuelle Aktionen zurückgehalten wird. Gehen Sie dazu in der Google Search Console zu der Übersicht „Manual Actions“ für die alte, primäre HTTP-Version.

Liegen tatsächlich manuelle Aktionen vor, beauftragen Sie vertrauenswürdige Google-Penalty-Berater mit der möglichst zeitnahen Behebung dieser Google-Penalties. Sobald entsprechende Maßnahmen eingeleitet wurden, machen Sie mit dem nächsten Schritt weiter.

Einstellung der bevorzugten Domain

Um zu prüfen, dass die Einstellung für Ihre bevorzugte Domain richtig funktioniert, gehen Sie zu der primären HTTPS-Property in der Google Search Console und klicken auf das Zahnrad-Icon oben rechts; wählen Sie „Site Settings“. Vergewissern Sie sich, dass Ihre bevorzugte Domain auf die primäre Version eingestellt ist (siehe Abb. 6).

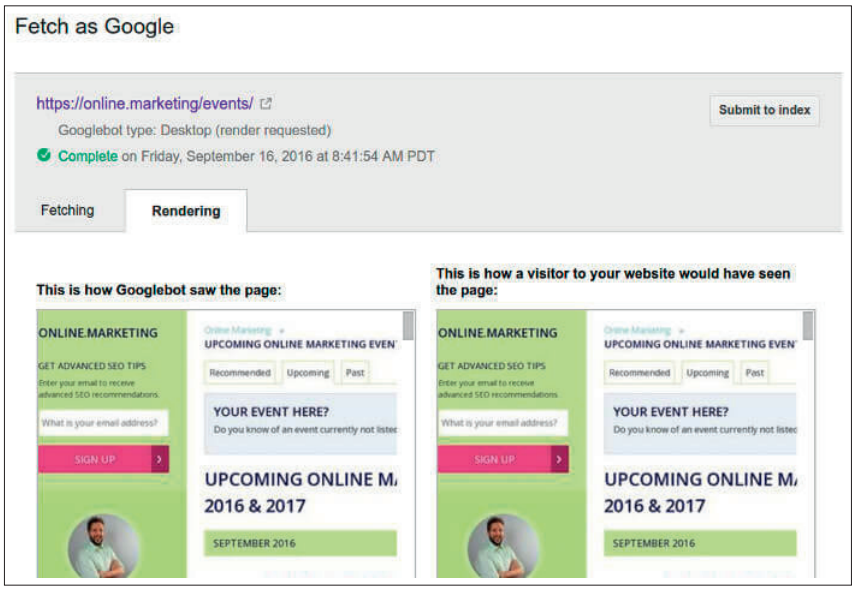

Abb. 4: Beispiel für das Tool „Fetch and Render“ in der Google Search Console

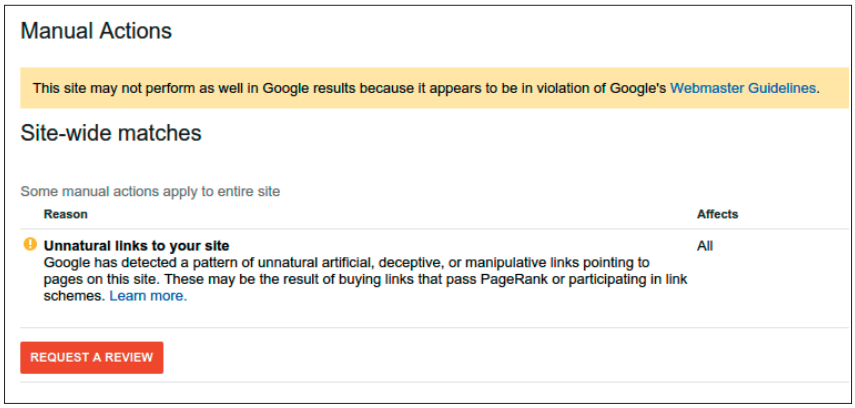

Abb. 5: Beispiel für eine Benachrichtigung über manuelle Aktionen in der Google Search Console

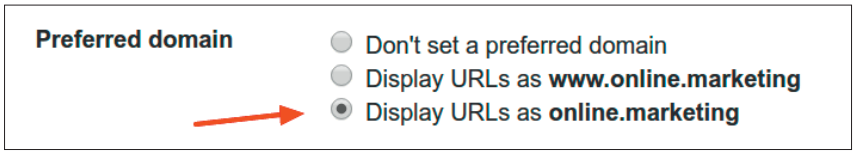

Abb. 6: Beispiel für die Einstellung einer bevorzugten Domain in der Google Search Console

Wurde noch nichts ausgewählt, aktualisieren Sie die Einstellung mit der primären Version.

Crawl-Geschwindigkeit

Diese Einstellung in der Google Search Console wird von den meisten Suchmaschinenoptimierern nicht geändert, aber falls jemand mit Zugangsberechtigung dies in der Vergangenheit für die HTTP-Property doch getan hat, müssen Sie jetzt diese Einstellung auch in der HTTPS-Property ändern.

Um dies zu prüfen, gehen Sie zu der alten primären HTTP-Property in der Google Search Console und klicken auf das Zahnrad-Icon oben rechts; wählen Sie „Site Settings“. Wenn hier nicht „Let Google optimize for my site“ angewählt ist, merken Sie sich die aktuelle Einstellung und gehen zu der primären HTTPS-Property in der Google Search Console. Klicken Sie erneut auf das Zahnrad-Icon oben rechts und wählen „Site Settings“. Ändern Sie die Crawl-Geschwindigkeit auf den für die HTTP-Property eingestellten Wert.

Sind Sie sich nicht sicher, ob Sie etwas ändern müssen, halten Sie mit dem IT-Team Ihrer Organisation Rück-

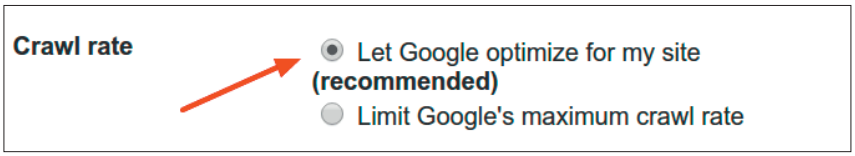

Abb. 7: Beispiel für die Crawl-Geschwindigkeit-Einstellungen in der Google Search Console

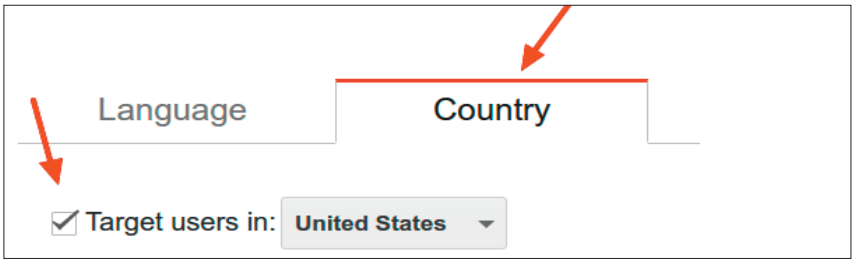

Abb. 8: Beispiele für die Einstellung der internationalen Ausrichtung in der Google Search Console

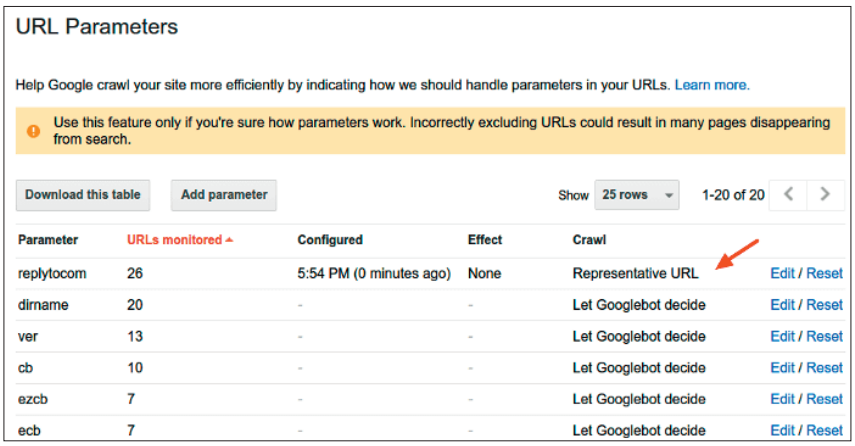

Abb. 9: Beispiel für die Einstellungen der URL-Parameter in der Google Search Console

sprache. Wie gesagt: Falls es nicht unbedingt notwendig ist, sollte diese Einstellung nicht geändert werden – das hört jeder SEO am liebsten.

Geotargeting

Wenn die Website nicht eine Domain oberster Stufe eines weitverbreiteten Ländernamens ist, wurde die internationale Ausrichtung in der Google Search Console vermutlich von Hand eingestellt bzw. muss diese Einstellung nun vorgenommen werden. Prüfen Sie die alte primäre HTTP-Property in der Google Search Console, um zu sehen, ob eine internationale Ausrichtung eingestellt wurde und mit einem Pull-Down-Fenster geändert werden kann; wenn ja, gehen Sie zur primären HTTPS-Version und stellen die internationale Ausrichtung auf die gleiche Region ein.

URL-Parameter

Gehen Sie zu dem Tool „URL Parameters“ in der alten primären HTTP-Property und prüfen Sie, ob URL-Parameter gecrawlt werden. Fall ja, laden Sie die Tabelle der URL-Parameter mitsamt den Einstellungen herunter.

Gehen Sie jetzt zu dem „URL Parameters“-Tool in der primären HTTPS-Version, fügen Sie nacheinander die heruntergeladenen URL-Parameter der alten primären HTTP-Version ein und kategorisieren sie einzeln. Dieser Schritt kann Googlebot dabei helfen, sein beschränktes Crawl-Budget auf die wichtigen URLs zu konzentrieren und diesen beim Crawlen der neuen HTTPS-Version der Website Priorität einzuräumen.

Gelöschte URLs

Damit sensible URLs der HTTP-Version nicht erneut indexiert und als Suchergebnis in Google Search präsentiert werden, gehen Sie zu dem Tool „URL Removal“ in der Google Search Console für die alte primäre HTTP-Version und prüfen Sie, ob URLs zum temporären Löschen vorgesehen sind. Falls ja, notieren Sie diese; gehen Sie zum Tool „URL Removal“ für die primäre HTTPS-Version und fügen Sie jedes Pattern einzeln hinzu.

Hinweis: Dieses Tool löscht URLs nur vorübergehend. Wollen Sie bestimmte URLs permanent aus dem Google-Index entfernen, löschen Sie die markierten URLs aus der HTTPS-Version und senden Sie eine 404-Seite oder fügen Sie einen „Noindex“ Eintrag zu den Meta-Tags für die markierten URLs hinzu, damit der Googlebot sie nicht in den Index aufnimmt, wenn er die neue HTTPS-Version crawlt.

Disavow-Datei

Um für die neue HTTPS-Version Probleme mit Rückverlinkungen (z. B. Google Penguin oder eine anzuwendende manuelle Aktion) auszuschließen, gehen Sie für die alte primäre HTTP-Version zum „Disavow“-Tool in der Google Search Console und prüfen Sie, ob eine Disavow-Datei vorliegt. Falls ja, laden Sie sie herunter und ändern Sie das Suffix von .CSV in .TXT. Gehen Sie dann zum „Disavow“-Tool für die primäre HTTPS-Version und laden die umbenannte TXT-Datei auf die Google Search Console hoch.

Ist keine Disavow-Datei vorhanden oder wurde die Datei schon länger nicht mehr aktualisiert, sollten Sie Google SEO Consultants mit ins Boot holen, die Sie bei einer vollständigen Analyse der manuellen Rückverlinkungen und der Erstellung einer neuen Disavow-Datei unterstützen; so vermeiden Sie, dass die Website wegen eines riskanten Rückverlinkungsprofils nicht ihr volles Potenzial beim Google Ranking ausschöpft.

Crawl-Fehler

Um Probleme mit der Glaubwürdigkeit von Antwortcodes des Servers zu vermeiden, prüfen Sie in der Google Search Console die Crawl-Fehler-Übersicht für

die alte primäre HTTP-Version. Prüfen Sie insbesondere, ob Google Soft-404-Fehler gemeldet hat. Falls ja, müssen Sie diese unbedingt beheben, indem Sie die 404-Statuscodes für diese URLs senden. Da Googlebot die Soft-404-Fehler nicht mag, achten Sie sorgfältig darauf, keine möglichen Soft-404-Fehler von der HTTP-Version auf der primären HTTPS-Version zu duplizieren. Um zu erfahren, was Googlebot gegen Soft- 404-Fehler hat, klicken Sie hier.

XML-Sitemaps einreichen

Zu Beginn dieser Anleitung haben Sie eine Reihe von XML-Sitemaps erstellt, aktualisiert und in die Wurzelverzeichnisse der relevanten Subdomains und/oder Unterverzeichnisse und/oder des primären Host-Namens eingestellt, und zwar sowohl für die HTTP- als auch für die HTTPS-Version. Diese XML-Sitemaps enthalten ausschließlich indexierbare und crawlbare URLs (vorzugsweise kanonische), die aus den alten XML-Sitemaps, den Server-Log-Dateien und aus den Crawlingvorgängen der HTTP- und HTTPS-Versionen der Website extrahiert wurden. Beispiel:

http://www.example.com/sitemap.xml
http://de.example.com/sitemap.xml
http://www.example.com/nl/sitemap.xml

und

https://www.example.com/sitemap.xml
https://de.example.com/sitemap.xml
https://www.example.com/nl/sitemap.xml

Prüfen Sie jede XML-Sitemap und machen Sie sie auf der Grundlage von Protokoll und/oder Unterverzeichnissen und/oder Subdomains für die relevanten Google Search Console Properties zugänglich.

Ausblick

In der nächsten Ausgabe behandeln wir den Abschluss der Umstellung auf HTTPS mithilfe der Google Search Console und der Log-Dateien sowie die Messung der Wirkung.¶

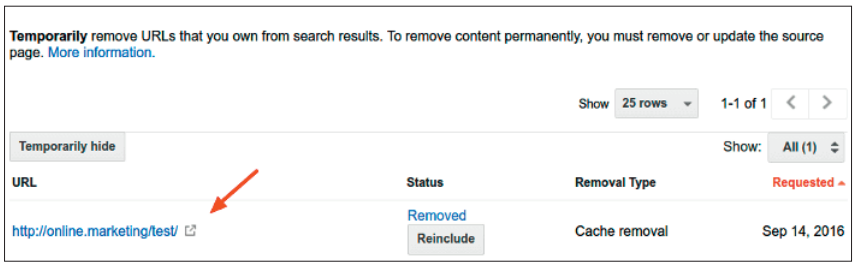

Abb. 10: Beispiel für das Tool „URL Removal“ in der Google Search Console

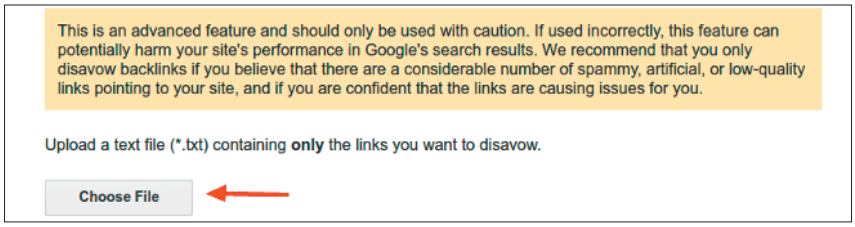

Abb. 11: Beispiel für das „Disavow“-Tool in der Google Search Console

Abb. 12: Beispiel für Fehlerberichte in der Google Search Console

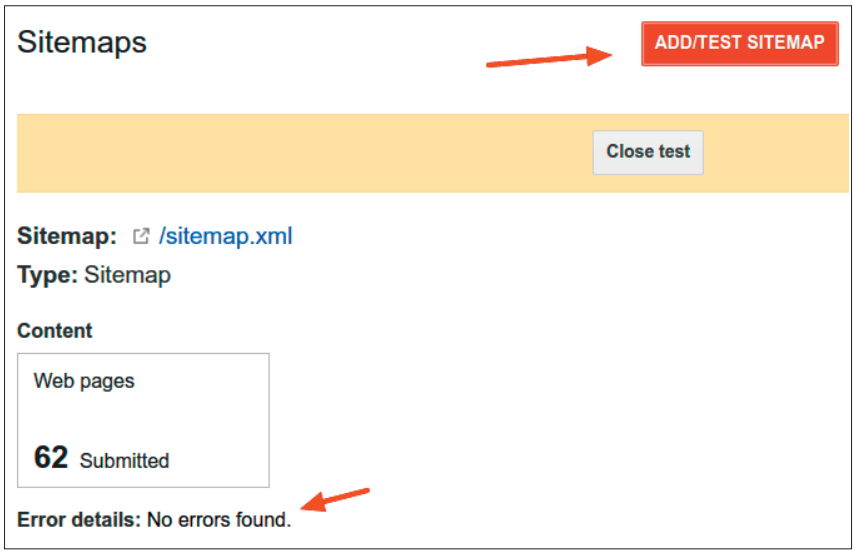

Abb. 13: Beispiel für das Prüfen und Einreichen einer XML-Sitemap in der Google Search Console