Latest News from 100TB

Get it from the source

Configuring HSTS In Apache

Published Aug 18, 2016 by Hayden Smith

In recent years there has been a big push across the industry to encourage website operators to move towards not only supporting HTTPS on their websites, but also making it the default connection option. One very visible aspect of this has been the creation of the Let's Encrypt certificate authority, a service that provides free domain validation SSL certificates using an automated system, making the enabling of HTTPS on your website easy.

At around the same time as the Let's Encrypt project was started, a specification was being published for HTTP Strict Transport Security (HSTS). This is a system designed to make a number of man-in-the-middle attacks against websites much more difficult by preventing browsers from attempting to communicate with a website using HSTS over plain HTTP connections which can easily be read by an intercepting attacker.

How to Set Up HSTS on Apache

With this in mind we are going to look at how we can set up HSTS on the Apache web server. Note that we are assuming you already have an SSL certificate for your domain, and already have this configured in Apache, and working as normal. If not then that's a step you'll need to complete first. Whilst SSL certificates themselves can be expensive, unless you are running a web store and need the higher grades of security that come with an expensive certificate, then a simple domain validation certificate will be fine for your website. 100TB customers can get a domain validation certificate for free from us that lasts a year, or alternatively - as previously mentioned - the Let's Encrypt project automates acquiring 90 day domain validation certificates for free.

So with your server set up and already able to serve websites over HTTPS for your chosen domain, we now need to look at the changes required for HSTS support. The first thing we have to do is enable the modules that we'll need, which are rewrite and headers. For Debian and Ubuntu systems this can be done with the following commands:

```
sudo a2enmod rewrite
```

```
sudo a2enmod headers
```

For CentOS and Red Hat this is a touch more complicated as you'll need to create the module files. In this example I'll be using nano, but you can use whichever text editor you prefer. So first, let's create the file to enable the rewrite module:

```
sudo nano /etc/httpd/conf.modules.d/02-rewrite.conf
```

Then paste in the following line:

```
LoadModule rewrite_module modules/mod_rewrite.so
```

Save and exit that file. We now need to do a file for the headers module:

```
sudo nano /etc/httpd/conf.modules.d/02-headers.conf
```

Paste the following line into the file:

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

Save and exit the file - you now have the modules configured.

Redirect HTTP to HTTPS

Next, we need to configure the server to redirect HTTP connections to HTTPS and set the HSTS header. This can be done globally or at the individual virtualhost level. For this example I'm going to assume that you are running multiple domain websites on your server, and that you are using virtualhosts for them. So you'll need to open the configuration file for the virtualhost that defines the domain for which you wish to apply HSTS and place the following lines within the virtualhost definition:

```
<IfModule mod_rewrite.c>

  RewriteEngine On

  RewriteCond %{HTTPS} off

  RewriteRule (.*) https://%{HTTP_HOST}%{REQUEST_URI} [R=301,L]

</IfModule>

Header always set Strict-Transport-Security “max-age=31536000; includeSubDomains”
```

The first section informs Apache that it should redirect any connections to the virtualhost that come via HTTP to use HTTPS; you may or may not already have this configured from when you configured HTTPS for your server. The final line informs Apache that it should send the header for Strict-Transport-Security setting the max-age parameter to one year (in seconds), and that subdomains should be included when the browser records which domains to use HSTS with. If you don't wish to configure HTTPS for all your subdomains then you can remove the “includeSubDomains” setting from the header.

Once you've set your virtualhost configuration you can save and exit the file. The final task is to restart Apache for it to take on the new configuration. For Debian and Ubuntu systems use:

```
sudo service apache2 restart
```

For CentOS and Red Hat systems use:

```
sudo service httpd restart
```

Now if you attempt to connect to your website from a web browser you should be redirected to your HTTPS site, and the browser should no longer try to connect via HTTP.

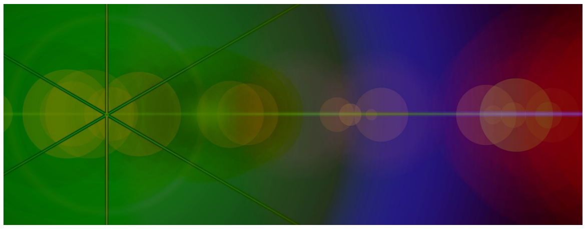
Give 100TB's servers a try. You will never look back.

☆ Categories: How to Guide

Tweet Share 0 Like 5 Share G+

Shining A Light On 'Dark (Big) Data'

‹ Previous Article



Yes QUESS, Let's All Get Entangled in Space!

Next Article ›

Receive The Latest 100TB News

▼

✉ Sign Up Now

Share This Post

▼

Navigate this blog

▼

How to Guide (122)
Technology News (60)
Gaming (54)
Media Streaming (54)
Start-ups (46)
Big Data (44)
Science & Technology (44)
Digital Transformation (43)
Ecommerce (14)
Technology (14)
Security Bulletin (11)
100TB News (6)
Marketing & Brand Management (6)
Security (5)

News (4)
Tips & Tutorials (4)
QoE (1)
Reselling Services (1)
Resources (1)
Xmas Special (1)

Latest News

The Ultimate App Development Checklist

MIT Researchers Have Found a Perfect Bio-Medical Tool — Lego

Distracted? Attention Management Might Be The Skill You're Looking For

Google Flights Introduces New Predictive Flight Information

Decoupling Schema Database Migrations From Code Deployments

Sign Up To Receive The Latest 100TB News

Enter your email...

Submit Email

YOUR INDUSTRY

OUR PLATFORM

SERVERS

OTHER SERVICES

PARTNER WITH US

LEGAL

SUPPORT

BLOG

powered by

UK2 GROUP