Security Header Ratings

Explanation

Security Header Ratings allow an objective assessment about the website's condition in terms of the security of the HTTP response headers. By adding and configuring security headers according to best practices, another layer of security will strengthen a website's protection. In Nimbusec Discovery, a new column within the report overview - consisting of grade A-F - evaluates the scanned security headers of a website.

The implementation is based on Mozilla Observatory. The detailed scoring is described in their Github repo: Scoring.md. That means we stick to Mozillas rating and scoring method.

Important update

Right now, Mozilla appears to be not that keen about keeping the project up-to-date with the current Security Header landscape. Meanwhile, several vendor have introduced new headers, altered the meaning of existing ones and even deprecated the use of some older headers which makes their consideration, evaluation and assessment of them in the Observatory project kinda obsolete. To adhere to the changes and hold our standards to the highest level, some important changes that can be seen below, were made.

Latest Changes

X-XSS Protection

Action

Removed from domain check

Reasoning

X-XSS Protection allows the user to activate the internal XSS Auditor / Filter which on behalf of the user conducts XSS checks for the given webcontent that served or presented. Back in the good old days of Internet Explorer this check was considered important to add another protection to the user. Nowadays, most major browser vendors (Chrome, Edge, Safari) have either retired or deprecated their XSS Auditors / Filters. Firefox, ironically, have never ever implemented such a check in the first place 1. The prevailing opinion is that this header has been completely replace by the superior Content Security Policy 2. Even OWASP does not recommend activating or enabling this header and advices turning it off 3. Currently, we do not grant any bonus points for having this header but rather penalize wrong or non - existant implementations. Given the current state, of major browser vendors, this header does not pose any usefulness any longer.

Sources

- https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-XSS-Protection
- https://scotthelme.co.uk/deprecating-xss-reports/
- https://github.com/mozilla/http-observatory/issues/432

HPKP = HTTP Public Key Pinning

Action

Removed from domain check

Reasoning

HPKP is an absolete header which is no longer supported by any web standards 1. It's original use was to pinpoint the public key of a website as a header when server via HTTPS. The reason was a protection against compromised CAs which would issue certificates for the same domain although it might belong to a completely different domain with malicious content behind. Although interesting when introduced in the 2010s, it became quickly a hot spot for misconfigurations and even counter-attacks. One immediate example is that browser were storing the pin of the public key for a pretty long time which meant that the certificate needn't to be changed. Which obviously contradicted short-lived certificates from Let's encrypt. As a result, many eager website maintainer found themselves very quickly in a position where the browsers stored and used primarily a pin for a certificate that had been obsolete. They were stuck. Other interesting examples are listed in 2. As of now, our implementation does not grant any bonus points while penalizing wrong configurations. It is listed as an active (but hardly addressed) issue in the Mozilla Github repository 3. Given the complete(!) deprecation of this header, it does not serve any purpose any longer.

Sources

- https://developer.mozilla.org/en-US/docs/Web/HTTP/Public_Key_Pinning
- https://scotthelme.co.uk/hpkp-is-no-more/
- https://github.com/mozilla/http-observatory/issues/422

Feature Policy / Permissions Policy

Action

Consider implementing this feature with only bonus points (+5 points) for correctly set headers (regardless of the actual feature that is restricted / allowed).

Reasoning

Feature Policy / Permission Policy is a header that was implemented around 2015 which allows the restriction of certain web features on a given page. Features include such things as notifications (like e.g.: when a website asks the user for permission to send notifications), camera, microphone, access to Payment Request API and the ability to make synchronous XML HTTP Requests among others. Crucially for the origin of this header is that the restriction can be extended to embedded pages as well, such as iFrames. This gives another layer of protection to the webpage developer as well as the visitor as malicious scripts that unintentionally happen to be embedded into the web page and attempt to access certain features are gonna be shut. Additionally, the web page developer can ensure that only those features are available that are truly needed. Given the current state-of-the-art browser landspace however, it is yet to be supported by Firefox and Safari. Chrome, Edge and Opera offer full support although the range of web features vary (please keep in mind that there are currently around 25ish web features that can be configured). Therefore, it is to be considered that we implemented bonus points (+5 points) for correctly setting this header, nothing more and nothing less.

Sources

- https://scotthelme.co.uk/a-new-security-header-feature-policy/
- https://w3c.github.io/webappsec-permissions-policy/
- https://developers.google.com/web/updates/2018/06/feature-policy
- https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Feature-Policy#browser_compatibility
- https://scotthelme.co.uk/goodbye-feature-policy-and-hello-permissions-policy/
- https://github.com/mozilla/http-observatory/issues/361
- https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Feature-Policy

Server

Action

Consider penalizing 5 points (-5) if the Server header in a HTTP response shows the version number.

Reasoning

The version number grants attackers that desire to hack the website additional information about the Server that is in use (e.g.: Nginx or Apache). Vulnerabilites targeted for specific version numbers can be search for directly on platforms like ExploitDB (not to mention that NIST or Mitre list the specific numbers for CVEs as well). Even the RFC describing the **Server** header mentions that web page revealing that version number are more likely to be targeted by attackers. As a result, it might be helpful to discourage the usage of the version number by penalizing such configurations.

Sources

- https://github.com/mozilla/http-observatory/issues/443
- https://scotthelme.co.uk/hardening-your-http-response-headers/#server
- https://securitytrails.com/blog/top-tips-harden-http-headers
- https://pretty-rfc.herokuapp.com/RFC2616#header.server

Other Changes

Action

Consider penalizing 5 points (-5) if the Server header in a HTTP response shows the version number.

Sources

- https://github.com/mozilla/http-observatory/pull/368
- https://github.com/mozilla/http-observatory/pull/437
- https://github.com/mozilla/http-observatory/pull/444

Scoring

Every scanned domain starts with a base score of 100. According to Mozilla's scoring method points are added for special configurations or subtracted if they are missing or insecure.

This generates a score which then can be mapped to a grade according to the following table:

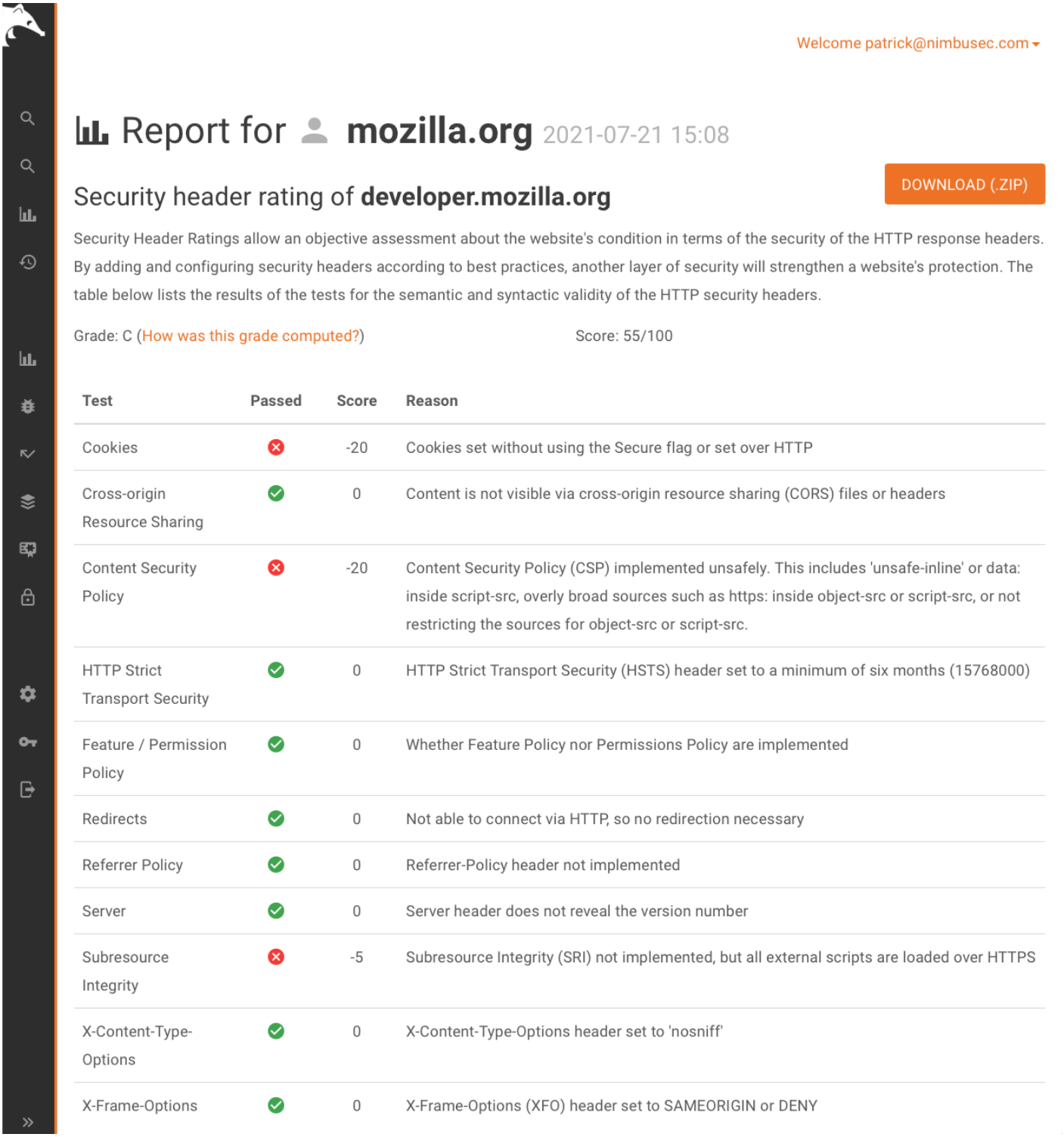
Scoring Range	Grade
100+	A+
90-99	A
85-89	A-
80-84	B+
70-79	B
65-69	B-
60-64	C+
50-59	C
45-49	C-
40-44	D+
30-39	D
25-29	D-
0-24	F

Looks in Discovery

The new column in Discovery shows the Grade from A+ to F for every discovered domain, where a rating was possible.

Sync	Rank	Domain	Inbound Redirects	Malware	Defacement	Reputation	Application	TLS	SHR
	2998	send.firefox.com	0	✓	✓	✓	✓	!	A+

Clicking on the grade will show the details how we measured the rating. On the details page you will see the reached score and all factors used to calculate it.



Welcome patrick@nimbusec.com

Report for mozilla.org 2021-07-21 15:08

Security header rating of **developer.mozilla.org**

DOWNLOAD (.ZIP)

Security Header Ratings allow an objective assessment about the website's condition in terms of the security of the HTTP response headers. By adding and configuring security headers according to best practices, another layer of security will strengthen a website's protection. The table below lists the results of the tests for the semantic and syntactic validity of the HTTP security headers.

Grade: C (How was this grade computed?) Score: 55/100

Test	Passed	Score	Reason
Cookies	✗	-20	Cookies set without using the Secure flag or set over HTTP
Cross-origin Resource Sharing	✓	0	Content is not visible via cross-origin resource sharing (CORS) files or headers
Content Security Policy	✗	-20	Content Security Policy (CSP) implemented unsafely. This includes 'unsafe-inline' or data: inside script-src, overly broad sources such as https: inside object-src or script-src, or not restricting the sources for object-src or script-src.
HTTP Strict Transport Security	✓	0	HTTP Strict Transport Security (HSTS) header set to a minimum of six months (15768000)
Feature / Permission Policy	✓	0	Whether Feature Policy nor Permissions Policy are implemented
Redirects	✓	0	Not able to connect via HTTP, so no redirection necessary
Referrer Policy	✓	0	Referrer-Policy header not implemented
Server	✓	0	Server header does not reveal the version number
Subresource Integrity	✗	-5	Subresource Integrity (SRI) not implemented, but all external scripts are loaded over HTTPS
X-Content-Type-Options	✓	0	X-Content-Type-Options header set to 'nosniff'
X-Frame-Options	✓	0	X-Frame-Options (XFO) header set to SAMEORIGIN or DENY

On the new analysis page you can filter for websites passing and/or failing specific tests. That will help you to focus on the most important websites first.

Revision #1
Created 2021-08-02 08:58:49 UTC by Patrick Wall
Updated 2021-08-02 09:08:12 UTC by Patrick Wall