Regular readers will know how fond I am of the existing security headers so it's great to hear that we're getting another! Referrer Policy will allow a site to control the value of the referer header in links away from their pages.

**What's a referrer?**

When a user clicks a link on one site, the origin, that takes them to another site, the destination, the destination site receives information about the origin the user came from. This is how we get metrics like those provided by Google Analytics on where our traffic came from. I know that 4,000 users came from Twitter this week because when they visit my site they set the referer[sic] header in their request.

Request Headers:

...

Host: scotthelme.co.uk

Referer: https://twitter.com/Scott\_Helme/status/760790725776830464

Upgrade-Insecure-Requests: 1

....

This referer header lets me know where the inbound visitor came from, and is really handy, but there are cases where we may want to control or restrict the amount of information present in this header like the path or even whether the header is sent at all.

**The Referrer Policy header**

The spec for Referrer Policy has been a W3C Candidate Recommendation since 26 January 2017 and can be found [here](https://www.w3.org/TR/referrer-policy/) but I'm going to cover everything in this blog to save you the trouble. The Referrer Policy is issued via a HTTP response header with the same name, Referrer-Policy, and can contain one of the following values as defined in the spec:

enum ReferrerPolicy {

"",

"no-referrer",

"no-referrer-when-downgrade",

"same-origin",

"origin",

"strict-origin",

"origin-when-cross-origin",

"strict-origin-when-cross-origin",

"unsafe-url"

};

I will break down each value and explain what the effects of issuing it would be.

Empty String

An empty string value in the Referrer Policy header indicates that the site doesn't want to set a Referrer Policy here and the browser should fallback to a Referrer Policy defined via other mechanisms elsewhere. This can include a HTML <meta> element, a referrerpolicy attribute on elements like <a> and <link> or the rel="noreferrer" keyword on <a> tags too. Issuing this policy will effectively have no impact but just confirms that the site has intentionally omitted it. You can even set your [Referrer Policy via the Content Security Policy](https://scotthelme.co.uk/csp-cheat-sheet/#referrer) header if you like.

no-referrer

The no-referrer value instructs the browser to never send the referer header with requests that are made from your site. This also include links to pages on your own site.

Referrer-Policy: no-referrer

| **SOURCE** | **DESTINATION** | **REFERRER** |
| --- | --- | --- |
| https://scotthelme.co.uk/blog1/ | http://scotthelme.co.uk/blog2/ | NULL |
| https://scotthelme.co.uk/blog1/ | https://scotthelme.co.uk/blog2/ | NULL |
| http://scotthelme.co.uk/blog1/ | http://scotthelme.co.uk/blog2/ | NULL |
| http://scotthelme.co.uk/blog1/ | http://example.com | NULL |
| http://scotthelme.co.uk/blog1/ | https://example.com | NULL |
| https://scotthelme.co.uk/blog1/ | http://example.com | NULL |

no-referrer-when-downgrade

The browser will not send the referrer header when navigating from HTTPS to HTTP, but will always send the full URL in the referrer header when navigating from HTTP to any origin. It doesn't matter whether the source and destination are the same site or not, only the scheme.

Referrer-Policy: no-referrer-when-downgrade

| **SOURCE** | **DESTINATION** | **REFERRER** |
| --- | --- | --- |
| https://scotthelme.co.uk/blog1/ | http://scotthelme.co.uk/blog2/ | NULL |
| https://scotthelme.co.uk/blog1/ | https://scotthelme.co.uk/blog2/ | https://scotthelme.co.uk/blog1/ |
| http://scotthelme.co.uk/blog1/ | http://scotthelme.co.uk/blog2/ | http://scotthelme.co.uk/blog1/ |
| http://scotthelme.co.uk/blog1/ | http://example.com | http://scotthelme.co.uk/blog1/ |
| http://scotthelme.co.uk/blog1/ | https://example.com | http://scotthelme.co.uk/blog1/ |
| https://scotthelme.co.uk/blog1/ | http://example.com | NULL |

same-origin

The browser will only set the referrer header on requests to the same origin. If the destination is another origin then no referrer information will be sent.

Referrer-Policy: same-origin

| **SOURCE** | **DESTINATION** | **REFERRER** |
| --- | --- | --- |
| https://scotthelme.co.uk/blog1/ | https://scotthelme.co.uk/blog2/ | https://scotthelme.co.uk/blog1/ |
| https://scotthelme.co.uk/blog1/ | http://scotthelme.co.uk/blog2/ | NULL |
| https://scotthelme.co.uk/blog1/ | http://example.com/ | NULL |
| https://scotthelme.co.uk/blog1/ | https://example.com/ | NULL |

origin

The browser will always set the referrer header to the origin from which the request was made. This will strip any path information from the referrer information.

Referrer-Policy: origin

| **SOURCE** | **DESTINATION** | **REFERRER** |
| --- | --- | --- |
| https://scotthelme.co.uk/blog1/ | https://scotthelme.co.uk/blog2/ | https://scotthelme.co.uk/ |
| https://scotthelme.co.uk/blog1/ | http://scotthelme.co.uk/blog2/ | https://scotthelme.co.uk/ |
| https://scotthelme.co.uk/blog1/ | http://example.com/ | https://scotthelme.co.uk/ |

*Warning:* Navigating from HTTPS to HTTP will disclose the secure origin in the HTTP request.

strict-origin

This value is similar to origin above but will not allow the secure origin to be sent on a HTTP request, only HTTPS.

Referrer-Policy: strict-origin

| **SOURCE** | **DESTINATION** | **REFERRER** |
| --- | --- | --- |
| https://scotthelme.co.uk/blog1/ | https://scotthelme.co.uk/blog2/ | https://scotthelme.co.uk/ |
| https://scotthelme.co.uk/blog1/ | http://scotthelme.co.uk/blog2/ | NULL |
| https://scotthelme.co.uk/blog1/ | http://example.com/ | NULL |
| http://scotthelme.co.uk/blog1/ | https://scotthelme.co.uk/blog2/ | http://scotthelme.co.uk/ |
| http://scotthelme.co.uk/blog1/ | http://scotthelme.co.uk/blog2/ | http://scotthelme.co.uk/ |
| http://scotthelme.co.uk/blog1/ | http://example.com/ | http://scotthelme.co.uk/ |

origin-when-cross-origin

The browser will send the full URL to requests to the same origin but only send the origin when requests are cross-origin.

Referrer-Policy: origin-when-cross-origin

| **SOURCE** | **DESTINATION** | **REFERRER** |
| --- | --- | --- |
| https://scotthelme.co.uk/blog1/ | https://scotthelme.co.uk/blog2/ | https://scotthelme.co.uk/blog1/ |
| https://scotthelme.co.uk/blog1/ | https://example.com/ | https://scotthelme.co.uk/ |
| https://scotthelme.co.uk/blog1/ | http://scotthelme.co.uk/blog2/ | https://scotthelme.co.uk/ |
| https://scotthelme.co.uk/blog1/ | http://example.com/ | https://scotthelme.co.uk/ |
| http://scotthelme.co.uk/blog1/ | https://scotthelme.co.uk/blog2/ | http://scotthelme.co.uk/ |

*Warning:* Navigating from HTTPS to HTTP will disclose the secure URL or origin in the HTTP request.

strict-origin-when-cross-origin

Similar to origin-when-cross-origin above but will not allow any information to be sent when a scheme downgrade happens (the user is navigating from HTTPS to HTTP).

Referrer-Policy: strict-origin-when-cross-origin

| **SOURCE** | **DESTINATION** | **REFERRER** |
| --- | --- | --- |
| https://scotthelme.co.uk/blog1/ | https://scotthelme.co.uk/blog2/ | https://scotthelme.co.uk/blog1/ |
| https://scotthelme.co.uk/blog1/ | https://example.com/ | https://scotthelme.co.uk/ |
| https://scotthelme.co.uk/blog1/ | http://scotthelme.co.uk/blog2/ | NULL |
| https://scotthelme.co.uk/blog1/ | http://example.com/ | NULL |

unsafe-url

The browser will always send the full URL with any request to any origin.

Referrer-Policy: unsafe-url

| **SOURCE** | **DESTINATION** | **REFERRER** |
| --- | --- | --- |
| https://scotthelme.co.uk/blog1/ | https://scotthelme.co.uk/blog2/ | https://scotthelme.co.uk/blog1/ |
| https://scotthelme.co.uk/blog1/ | https://example.com/ | https://scotthelme.co.uk/blog1/ |
| https://scotthelme.co.uk/blog1/ | http://scotthelme.co.uk/blog2/ | https://scotthelme.co.uk/blog1/ |
| https://scotthelme.co.uk/blog1/ | http://example.com/ | https://scotthelme.co.uk/blog1/ |

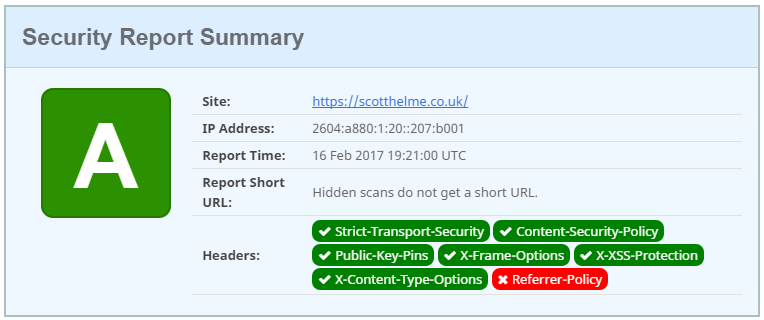
*Warning:* Navigating from HTTPS to HTTP will disclose the secure URL in the HTTP request.

**Recommendations**

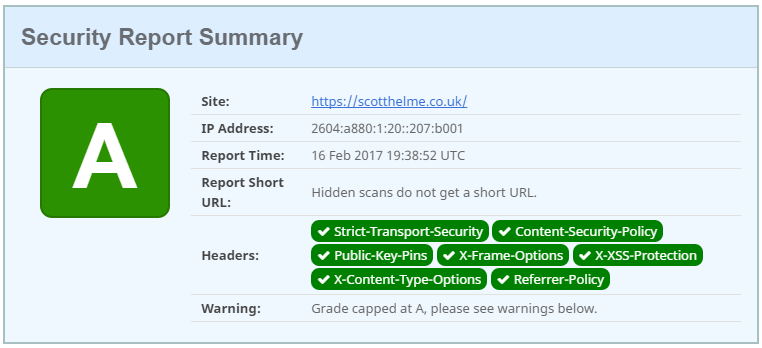
Which header you will want or need to use will depend on your requirements but there are some that you should probably stay away from. The unsafe-url value kind of gives you a hint in the name and I wouldn't really advise anyone use it. Likewise if you're thinking of using origin or origin-when-cross-origin then I'd recommend looking at strict-origin and strict-origin-when-cross-origin instead. This will at least plug the little hole of leaking referrer data over an insecure connection. I don't have anything sensitive in the URL for my site so I will probably look at a value like no-referrer-when-downgrade just to keep referrer data off HTTP connections.

**securityheaders.io**

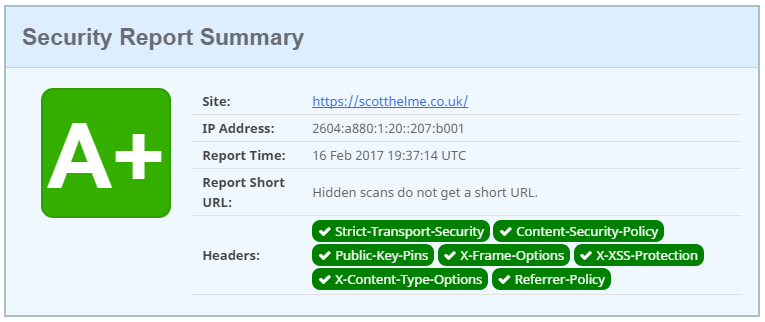
I've added this header to [securityheaders.io](https://securityheaders.io/) as it's now a W3C Candidate Recommendation and it does count towards your score. You can see the new results for my site here:



Of course, you can't achieve a grade A now without the new Referrer-Policy header properly configured. If you try and set it with no policy, or a bad policy, it's not going to help you.



You have to set the header and use a good policy to be awarded top marks!



It will be interesting to see how much of an impact this has on the grading criteria as it will drag grades down across the board. Hopefully sites will be fast to respond in deploying the new header and asserting more control over the information shared with referrer data.