Rolling out Public Key Pinning with HPKP Reporting

By Emily Stark

Emily is a contributor to Web**Fundamentals**

Using SSL on your site is an important way to preserve security and privacy for your users. But enabling SSL isn't the end of the story: there are many steps you can take to further enhance the security that your site provides, from setting the Secure attribute on your cookies to turning on HTTP Strict Transport Security to using Content Security Policy to lock down your site's privileges. Deploying these powerful features can sometimes be tricky, though. To help you roll out a stricter form of SSL, Chrome 46 ships with a feature called HPKP reporting.

What do all these acronyms mean?

Security on the web today relies on SSL certificates: cryptographic signatures proving that a website is who it says who it is. When your browser sends a request to a URL such as https://developers.google.com, the server provides an SSL certificate, and if the certificate is valid, the browser allows the request to proceed and shows the website URL with a green padlock in the address bar.

What is a valid certificate, though? To be considered valid, a certificate must be signed by a certificate authority (CA), or by another certificate that was signed by a CA (known as an intermediate CA). Browsers and operating systems ship with a list of several hundred CAs that are trusted to issue certificates. The problem, though, is that by default, any of these CAs can issue certificates for *any* website. If any one of them is compromised or misbehaving, that could be devastating for the entire web.

Enter HTTP Public Key Pinning, or HPKP. This standard allows websites to send an HTTP header instructing the browser to remember (or "pin") parts of its SSL certificate chain. The browser will then refuse subsequent connections that don't match the pins that it has previously received. Here's an example of an HPKP header:

```
Public-Key-Pins:
       pin-sha256="d6qzRu9zOECb90Uez27xWltNsj0e1Md7GkYYkVoZWmM=";
       pin-sha256="LPJNul+wow4m6DsqxbninhsWHlwfp0JecwQzYpOLmCQ=";
       max-age=259200
```

This header specifies two certificate hashes as pins. One is a hash of a certificate in the site's certificate chain, and the other is a backup pin, or a hash of a certificate that the site can use in the event that it needs to rotate its certificate. The header also includes a **`max-age`** value. After that number of seconds elapses, the browser will forget about the pin.

For more about HPKP in general, check out the spec or fellow Chrome developer Chris Palmer's excellent blog post.

Should I go turn on HPKP right now?

Not necessarily. When you deploy HPKP, it's quite easy to make a mistake and accidentally DoS your site. If you pin your site to one set of certificates and then have to deploy a new one, users who have seen the pin will be unable to access your site until the pin expires (based on the **`max-age`** value in the header).

Because it's tricky to get right, HPKP is mostly used by a handful of high-profile, security-sensitive sites right now. If you decide to turn on HPKP, you should start with a very short max-age value and gradually increase it if you don't have any problems.

What's HPKP reporting and how does it help?

HPKP reporting, shipping in Chrome 46, is a feature that you can use to detect misconfigurations as you're rolling out HPKP.

First, you can start by sending the **`Public-Key-Pins-Report-Only`** header instead of the Public-Key-Pins header:

```
Public-Key-Pins-Report-Only:
       max-age=2592000;
       pin-sha256="E9CZ9INDbd+2eRQozYqqbQ2yXLVKB9+xcprMF+44U1g=";
       pin-sha256="LPJNul+wow4m6DsqxbninhsWHlwfp0JecwQzYpOLmCQ=";
report-uri="https://example.net/pkp-report"
```

When your site sends such a header, Chrome will verify if the current connection matches the pins, and sends a report to the **`report-uri`** if not. Chrome will never block requests based on the pins in a Report-Only header, so this is a safe way to try out HPKP and see if it causes problems for your users without running the risk of DoSing your site.

Note that a Report-Only header only applies to the request on which it is received. The browser doesn't remember Report-Only pins as it does for real pins. This allows you test your configuration without worrying about caching bad values in your users' browsers, and you can roll out incrementally (for example, just on a single resource) to avoid flooding your server with reports.

When you roll out the real `Public-Key-Pins` header to start enforcing your pins, you can include a report-uri value in that header as well, so that you will continue to get reports if any problems occur.

What goes in a HPKP violation report?

An HPKP violation report is a JSON message sent in an HTTP POST request to your configured `report-uri`. The list of fields can be found in the spec, but I'll highlight two of them here: `served-certificate-chain` and `validated-certificate-chain`. The `served-certificate-chain` is the certificate exactly as Chrome received it when setting up the SSL connection for the request. The `validated-certificate-chain`, on the other hand, is the chain that Chrome rebuilt when trying to validate the server's certificate, which, surprisingly, can be different than the `served-certificate-chain`. Different clients perform certificate validation in different ways, and this can be a common cause of HPKP misconfigurations. Be sure to check this field if you are receiving unexpected reports.

One last "gotcha"

If you're deploying HPKP reporting, remember that Chrome does pin validation for all requests—including report-sending requests. So if you have deployed HPKP for your site, you probably want to send HPKP reports to a different domain that you have not pinned. Otherwise, a pin violation on your site will trigger a report to the same domain, which will also fail pin violation, and thus you will not receive the report.

If you don't have another domain handy, you could instead try a service such as report-uri.io, which handles violation reports for you.