What is Web Cryptography API?

Andrew Ayer

Share

Recent Posts

- Post-Mortem of Cyber Crime Investigation
- The Art of Persuasive Reporting: A Comprehensive Guide
- Common cyber-attack vulnerabilities of web applications

Categories

- Advisory
- Cryptography
- Cyber Security Tips
- Exploits
- Methodology
- News
- Pen-Tester Blog Posts
- Uncategorized

Search

Search ...

Before getting down to today's topic, Web Cryptography API, note that all the Chrome demos are available here: 1-better-quality

Chrome developers have added a new set of cutting-edge browser cryptographic operations that having been implemented in Chrome 37 (Dev channel). Mozilla has also implemented some of the functionality in Firefox 34 (Aurora) behind a flag. It currently has a partial set of operations.

The library has been working well for years, so why should I care using it?

No trivial allowed here

The end of the blog post is not to say that Web Cryptography API is the best API or that you must replace your existing crypto implementation with Web Cryptography.

The point is just to remind you what Web Cryptography API is and what it offers.

What is The Web Cryptography API?

From Mozilla's FAQ:

> Operations of the cryptographic kind, **cryptographic operations** in web parlance, such as hashing, ciphering and signature generation and verification are ubiquitous. Application developers have had to rely on JavaScript implementations of the cryptographic algorithms, however, the large amount of JavaScript to be shipped over the wire, and the performance penalty associated with interpreted JavaScript, have been a deterrent to the use of cryptography in web applications. Hence, the implementation of cryptographic algorithms in native code.
>
> Fortunately, there are already, in all current-day, mainstream browsers, libraries that can, some what, offer:
>
> - implementations of cryptographic algorithms in native code,
> - Pseudo-random number implementations,
> - Secure storage for cryptographic keys, and
> - Interfaces over many, but different implementations of the algorithms supported,
> - A comprehensive by-pass mechanism for the operations implemented.
>
> Unfortunately, the situation is not ideal. Developers have had to develop for extensions across or the provider of the libraries, or, Google Crypto library for Chrome, Mozilla Network Security Services (NSS) for Firefox, Microsoft Crypto API for the IE's, Safari's Underlying Cryptographic Operationals[1] ? Behind the following standards:
>
> This specification proposes, two ideal ways, the underlying low-level primitives and an API that will be **offered** to web developers. The low-level primitives, will be based on the Cryptographic Extensions for Web Platform (Web Cryptography) specification, such as those used within IE11. **Including** APIs are available as part of the underlying operating system or third-party modules that are managed independently of the user agent.
>
> The IE11 recommended browsers alternative use the Mathematical Cryptographic Secure Primitives but instead of providing a low-level interface to the primitives, the Web Cryptography API is proposed to be used. This includes the advantages for handling of cryptographic operations in a similar fashion.
>
> In addition using Web Cryptography API brings the following benefits:
>
> - Ease of use for web developers, compared to low-level primitives,
> - Ability to leverage platform-provided or hardware-backed cryptographic operations.

The end of white rabbit syndrome[2]

From a security point of view, it is a bit a bum-a-sanitizers the implementations of an algorithm, you will not tolerate a bone-to-bone, a slightly a short of a bit and also will only the vulnerable versions of the cryptography library and the perfect number updates and bolstered-in. The planning will be actually on the implementation of the cryptographic algorithms, and the Web Cryptography API is a bit of a bum-a-sanitizer.

Chrome's source code, as a guide for what the Google Blink/WebKit team has done in the implementation of cryptographic transaction that uses a binary binary included. Firefox, since it has and it has stages of implementation, the Chrome's binary source code is a good example that could have a bum-a-sanitizer application.

Browser support level

Is Web API Ready for Web Crypto?

Is Web API Ready for Web Crypto?
Yes
No
Yes
No
Yes
No