The W3C Web Cryptography API: Payment, Authentication, and Next Steps

Harry Halpin (W3C/MIT)

In order for work of a universal payment API to proceed, it must develop an overall architecture that solves the authentication problem: How do we bind a particular identifier to a particular user so they can authorize transactions? In fact, solving the authentication problem is the very first problem that must be solved before any other system can continue. Given the fallibility of computational systems, authentication is not easily solved with perfect accuracy, but it can be solved with higher accuracy than it is currently. The W3C Web Cryptography API, created by the Web Cryptography Working Group, is a unified cross-browser API for cryptographic primitives, provides much-needed primitives, ranging to digital signatures to key derivation, to that can make real progress on the authentication problem.

Why is authentication the most important problem facing payments on the Web? Abstractly, systems of payment move coins from one purse to another. The primary problem is the binding of wallets with the agreed-upon coins to individuals who can then authorize transactions that move coins, discounting global per-currency problems such as controlling the creation of new coins and contracts for transactions. Any system of currency must have strong binding between wallets and coins - and so real human users and their wallets - if they are to avoid coins being stolen or created in an arbitrary manner, which would destroy the currency system. A few systems that aficionados claim to be anonymous such as Bitcoin do not avoid the authentication problem: A transparent and decentralized ledger by very definition can't let users be is anonymous; it can only attempt not to bind identifiers that aren't real names to wallets. Any problem that involves binding identifiers of any kind to humans so that they can authorize: whether that binding identification for a real user is to email addresses, real names, private key material, or something else entirely.

No system, much less standard, has yet solved the authentication problem. HTTP Authentication relies on broken cryptography at best rather than more sane SRP protocol. Every part of the Web is damaged by the massive attack surface which is the certificate authority system. Several systems created by Linked Data enthusiasts such as Web Payments ignore the authentication problem or punt it to solutions that have a number of critical flaws (such as lack of cross-device functionality and interrupting TLS handshake in media res) such as WebID, as well as using their own idiosyncratic JSON notation for keys. Other systems that attempted also to avoid W3C standardization and use private key material for improved authentication, such as Mozilla Personae, have suffered severe problems with uptake in the larger Web as vendors believe mistakenly that by branding the solution "Mozilla Personae" (previously known as BrowserID) it will only work with Mozilla. Furthermore, even these systems involve downloading untrusted Javascript for the cryptographic primitives, since at this time Mozilla Personae is not build on top of the W3C Web Cryptography API. Other specifications in this space such as OpenID Connect, a profile of the very popular OAuth standard, do not address authentication in any meaningful way, but instead focus on shared string-names for authorization. In this regard, OpenID Connect simply re-uses in most cases the same pattern as used by Google and Facebook Connect, as well as most banks, which is redirecting the user to another service to ask for authentication.

On the Web authentication is done via cookies, user-names, and passwords (or in the case of many banking systems, so-called "secret" questions) - i.e. shared secrets that can be often be stolen. This sad state of affairs has not changed much since the Web began. The problem, equally unsolved by 3-D Secure (Verified by Visa) and Facebook Connect, is what Ben Laurie has termed the "phishing heaven" that results from

accustoming users to redirection. Because of the higher risk in any financial transaction from this kind of authentication, transactions need more non-Web verification and take longer to clear. Thus, a number of companies have created the FIDO alliance to work on multi-factor authentication, and some parts of that API may end up being Web-facing. Equally so, many banks are interested in using hardware tokens with private key material for authentication, ranging from smartcards to SIM cards. This offers a much higher value of authentication as people generally notice when their phone is stolen and the private key material is strongly bound on the hardware layer to the device. However, currently both multi-factor authentication and hardware tokens can not be accessed by the Web in a consistent and reliable manner cross-browser.

The Web Cryptography API currently has all major browsers involved and most are implementing (Chrome, Safari, and Microsoft Internet Explorer). The Web Cryptography API includes constant-time primitives for encryption, decryption, key generation, hash functions, digital signing and verification, import and export of keys, key wrapping and unwrapping, as well as random number generation. The security model is the same-origin policy. The API should be at Last Call soon, and we hope many independent experts carefully review the API and ensure security properties such as key storage, and that many Javascript developers attempt to use the API and provide feedback on its usability. The creation of jQuery-style "high-level" APIs with appropriate defaults still need to be done, but such a more "beginner-friendly" API could not be created without the lower-level primitives revealed uniformly first. Once the Web Cryptography API is out of Last Call, there will be a re-chartering of the Working Group and a new workshop focusing on how uniform access to key material on hardware tokens by WebApps can enable high-value authentication. One practical result from the Payments workshop should be exactly what kinds of higher-value authentication, including the role of hardware tokens and other forms of multi-factor authentication, are needed to enable real-world payments on the Web.