A Vision for Browser-Assisted Web Authentication

Siddharth Bajaj, Slawek Ligier
Symantec Corporation
April 29, 2011

For: W3C Identity in the Browser Workshop

Abstract

In this paper we present a vision for browser-assisted web authentication. Our concept focuses on moving towards a higher security authentication on the web than username and password, without compromising on the usability. We try and explore whether we can replicate the physical world metaphor, of authenticating using an ATM card and PIN, for web authentication.

Observations and Key Concepts

In this section we briefly describe some observations and concepts that we have used to develop our concept for web authentication.

#1 - Issues with Passwords

We all understand the security and usability issues with username/password authentication that is prevalent on the web today. Some of the important issues are –

- Each website has its own username and password, placing an unnecessary burden on their users.
- Users tend to use the same password across multiple websites.
- Password policies such as password quality or requiring the users to change their password periodically do not improve security - users write their complex passwords down on a piece of paper and tend to forget them more often.

#2 –ATM Authentication: Balancing Security & Usability

One example where relatively high-security is achieved without compromising usability is bank ATM network. Users authenticate using two factors –

- ATM card (something you have)
- 4 digit PIN (something you know)

Users use a combination of the ATM card and a 4-digit PIN to authenticate to the ATM network and access their bank account. The use of two-factors offers a high level of security. Usability is maintained

since the user only needs to remember a 4-digit PIN. And most users don't or very seldom change their ATM PINs.

#3 – OpenID and other SSO technologies are not the complete solution

Last few years have seen the development of OpenID and other web single-sign on technologies. Many folks have talked about these technologies coming together and solving the issue with username/password based web-authentication today.

While these technologies may help reduce the number of password that the user has to remember, they are not the complete solution. You still have the issue of authenticating to your OpenID provider using your browser. We believe that this fundamental problem still needs to be addressed.

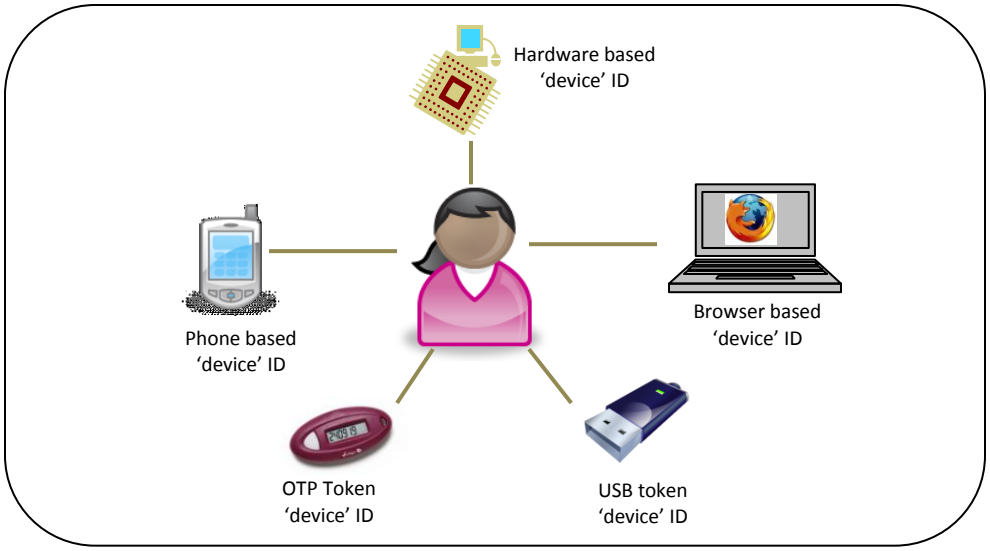
#4 – Using Device Credentials to authenticate the user

We would like to make the case for separating the credentials that are used to authenticate the user from the user's identity. This simple but powerful observation will allow us to use 'device' credentials that can be bound to user's identity in the application through a separate 'registration' or 'binding' process.

The credential could be associated with the user's access device (e.g. computer, mobile phone) and can be at the hardware, operating system, or application (browser) level. These credentials could be device-generated or provisioned from a third-party credential provider. To preserve the privacy and prevent collusion across sites, these credentials could be site specific.

Alternatively, these credentials could be distinct from the user's access device – for e.g. a standalone OTP generator device or a USB token.

Since the user will typically use more than one access device to access the web application, the application or the application's identity provider should allow more than one device credentials to be bound the user's identity as shown in the picture below.

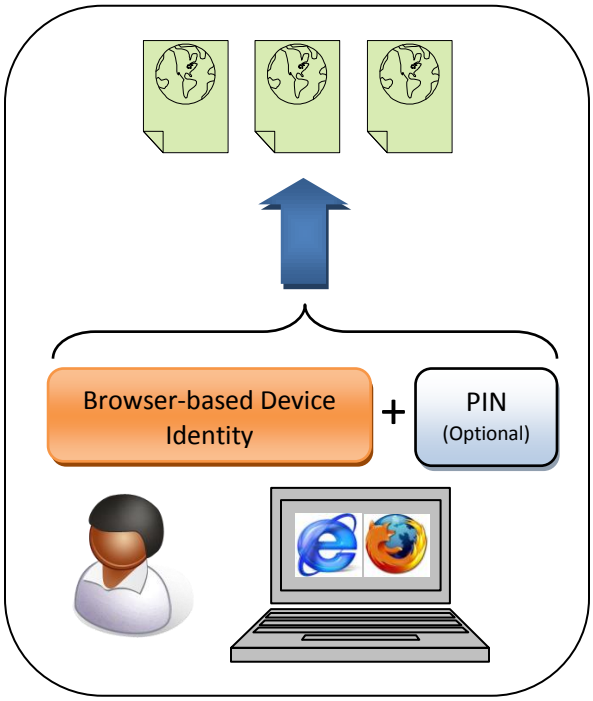


Browser-Assisted Web Authentication

We feel that when we are looking for a way to solve the authentication problem on the web, we can learn from the physical world. Building on top of observations #2 and #4 above, a potential solution could try and re-create the ATM card analogy for the web.

The picture below shows a conceptual diagram of the ATM card analogy applied to web authentication. The user can authenticate to the website using the following two factors -

- Browser-based Device Identity (Web-ID) - this could be provisioned by the application into the browser or generated within the browser. The only requirement is that the Web-ID is distinct from the user's identity, and has a unique identifier that can be used by the application to 'register' the user's browser/device.
- PIN (Optional: 4 digits) – this could be optional based on user policy for low value websites, and could be used for high value websites based on site policy. Further, this PIN could be local or remote.



Since the user may use multiple machines to access the web application, the application should allow the user to 'register' multiple browser-based device identities. We feel that for majority of users, the access to web applications is from a fixed set of devices and this approach would cover a majority of scenarios.

We also acknowledge that we need to avoid the scenario where the user needs to remember too-many PINs. Browsers should provide an enforceable way for applications to specify and monitor policies

around use of 'Local' PIN. A Local PIN is something that the user sets locally in the browser, and can be used to authenticate locally to the browser, across multiple web applications. For example, the browser could generate an assertion that indicates that the user's local PIN was verified successfully and send it to the application for compliance-tracking.

To enable access from 'kiosk' machines, the web application can either allow users to register non-browser based devices or allow temporary access based on some out-of-band authentication.

Key Requirements & Gaps

Some of the key requirements towards accomplishing the above are –

- A way for the web application to specify authentication policies – credential requirements, credential provisioning end-points, authentication end-points, etc.
- A way for the web application to seamlessly provision a new 'Web-ID' to the user's browser. Externally provisioned credential case.
- A way for the web application to query and discover existing device identifiers from the user's browser.
- A secure authentication protocol between the application and the browser using the browser-based device identifier (Web-ID).
- An enforceable way for application to specify and monitor policies around use of a 'local' PIN.
- A way to securely transmit the PIN to the application in case the application uses a 'remote' PIN.

Re-using PKI as a foundation for the browser-based 'Web-ID'

The discussion thus far has been technology agnostic. When we look at the requirements to accomplish the above scenario and look at the available technologies, one technology that has several desirable characteristics viz. PKI comes to mind.

Desirable characteristics of PKI:

- High security: TLS with mutual authentication is resistant to phishing and MITM attacks
- Large deployed infrastructure: Browsers, web & application servers
- Based on open standards defined by IETF and other SDOs.
- Low fulfillment costs: No hardware needs to be shipped to user, but does not preclude the use of optional hardware.
- Once you get it working, it works and is seamless

Admittedly, PKI also has some drawbacks as well, but we feel that they can be overcome.

- Inconsistent experience across browsers and CAs
- Too much PKI-specific information that does not make sense to an average user
- Usability issues with significant lifecycle events, such as expiration and renewal
- Too many clicks & dialogs
- Relying parties have no control/ knowledge of user events

We feel that we can specify and implement a 'profile' of PKI specifically for the purpose of browser-based 'web-id'.

- By removing any user-specific information from the certificate, we can deploy longer-lived certificates that do not need to be renewed every year.
- Since the certificate just has a unique identifier and no user-specific attributes, the verification process for issuing certificates should be greatly simplified. We could completely automate the provisioning of the certificates into the user's browser from a remote provisioning end-point specified by application's authentication policies.
- We feel that by enabling web applications to specify PKI-related policies, the browser can make the experience completely seamless and transparent to the user.

There are other examples where PKI has been used 'under-the-hood' for device identification in large scale deployments. We have also prototyped some of the concepts in this paper for a browser-based environment. The authors feel that there are enough positive characteristics to justify the use of PKI and certificates for the purpose of browser-based 'web-id' rather than starting from scratch.

Conclusion

We feel that the ATM metaphor for authentication is deployed and successful in the physical world, and we should try and move the web-authentication infrastructure in that direction.

The browser can be a key enabler in addressing the challenges around web-authentication that we face today. In this paper the authors have presented one possible approach that enables browser-based web-authentication.

Further, the authors make a case for the use of PKI standards and technologies as a starting point towards developing a browser-based web-id technology.

Lastly, the authors would like to acknowledge that they have not attempted to answer all questions in this short paper, but rather present one possible approach towards moving to a more secure and usable technology for web authentication than what is prevalent today.