Strong Authentication In and Beyond the Browser

Lessons Learned from FIDO

Brad Hill, PayPal, brad.hill@paypal.com
Jeff Hodges, PayPal, jeff.hodges@paypal.com
Davit Baghdasaryan, NokNok Labs, davit@noknok.com
Christiaan Brand, Entersekt, christiaan@entersekt.com
Rob Philpott, RSA, robert.philpott@rsa.com

- Users and services want keys that are hardware-bound or isolated from malware and difficult to steal
- Consumer privacy demands that devices not be “pre-identified” or linkable
- Ergo: services make consumers complete a (costly) “registration ceremony” to associate themselves with a new key
- Once they’ve created a key, users expect to be able to use it everywhere they use that service with that device (whether embedded or externally connectable)

Services have many *facets*

On my one device I can access PayPal as:

- A website on Chrome for Android App
- A website on Chrome Beta for Android App
- A website on Firefox for Android App
- A website on “Internet” App (Samsung browser)
- The PayPal App
- The eBay App

And if my keystore is a USB or BT token, those facets may be in different computing environments, each trusted by the user but unknown to each other.

Just solving for a browser is not enough.

But not just any app should have access to the user's key, either

- "PayPa1" should not be able to use the same registered key

- Or, the malicious "Disgruntled Avians" app shouldn't be able to prompt me to "swipe my finger to save my game" and actually be taking over my bank account

Crypto keys need a new reference monitor

- Create keys and associate them with a scope
- Strongly identify apps attempting to use keys and determine their authorization for that scope

- We call this reference monitor the “FIDO Client” in our solution

The Pieces of the Puzzle:

- Key scoping mechanism ("AppID")
- Scope authorization mechanism ("Trusted Facet List")
- Platform-specific* reference monitor ("FIDO Client")
- Set of logically-equivalent but platform-appropriate APIs to access reference monitor functions from Web apps, Android, iOS, etc. (<- ***W3C goes here?!***)

*Full feature set requires a platform notion of "app identity" and security barriers between apps that are not present in, all systems (e.g. Win32, POSIX)

API functionality in FIDO

- Discovery: allow an app to determine what authentication capabilities/modalities are available in user agent / device
- Registration: create a new credential and register it with a scope
- Authentication: provide proof of control of (and optionally verify a transaction using) a registered credential
- Deregistration: allow a server to manage credential lifecycles transparently to the user

See also: https://fidoalliance.org/specifications/download