ForceAuthn or Not

ForceAuthn or Not, That is the Question

Depending on your point of view, SAML is either a very complicated security protocol or a remarkably expressive security language. Therein lies a problem. A precise interpretation of even a single XML attribute is often elusive. Even the experts will disagree (at length) about the finer points of the SAML protocol. I've watched this play out many times before.

In this particular case, I'm referring to the `ForceAuthn` XML attribute, a simple boolean attribute on the `<samlp:AuthnRequest>` element. On the surface, its interpretation is quite simple: If `ForceAuthn="true"`, the spec mandates that "the identity provider MUST authenticate the presenter directly rather than rely on a previous security context." (See section 3.4.1 of the SAML2 Core spec.) Seems simple enough, but a google search will quickly show there are wildly differing opinions on the matter. (See this recent discussion thread re `ForceAuthn` on the Shibboleth Users mailing list, for example.) I've even heard through the grapevine that some enterprise IdPs simply won't do it. By policy, they always return a SAML error instead (which is the IdP's prerogative, I suppose).

For enterprise SPs, such a policy is one thing, but if that policy extends outside the enterprise firewall...well, that's really too bad since it reduces the functionality of the SAML protocol unnecessarily. OTOH, an SP that *always* sets `ForceAuthn="true"` is certainly missing the point, right? There must be some middle ground.

Here's a use case for `ForceAuthn`. Suppose the SP tracks the user's location via IP address. When an anonymous user hits the SP, before the SP redirects the browser to the IdP, the IP address of the browser client is recorded in persistent storage and a secure cookie is updated on the browser. Using the IP address, the SP can approximate the geolocation of the user (via some REST-based API perhaps). Comparing the user's current location to their previous location, the SP can make an informed decision whether or not the user should be explicitly challenged for their password. If so, the SP sends a `<samlp:AuthnRequest>` to the IdP with `ForceAuthn="true"`.

Behaviorally, this leads to a user experience like the one I had recently when I traveled from Ann Arbor to San Francisco for Identity Week. When I arrived in San Francisco, Google prompted me for my password, something it rarely does (since I'm an infrequent traveler). Apparently, this risk-based authentication factor protects against someone stealing my laptop, defeating my screen saver, and accessing my gmail. This is clearly a Good Thing. Thank you, Google!

This use case avoids the `ForceAuthn` problem described above, that is, the SP now has some defensible basis for setting `ForceAuthn="true"` and the debate between SPs and IdPs goes away. Simultaneously, we raise the level of assurance associated with the SAML exchange **and** optimize the user experience. It's a win-win situation.

The downside of course is that this risk-based authentication factor is costly to implement and deploy, especially at scale. Moreover, if each SP in the Federation tracks geolocation on its own, the user experience is decidedly **less** than optimal. But wait…distributed computing to the rescue! Instead of deploying the risk-based factor directly at the SP, we can deploy a centralized service that passively performs the geolocation check described above. There are multiple ways to deploy such a centralized service: 1) deploy the service as an ordinary SAML IdP; 2) deploy the service in conjunction with IdP discovery; or 3) invent a completely new protocol for this purpose. There are probably other ways to do this as well.

Hey, SP owners out there! If such a service existed, would you use it?