UMA F2F 2010-11-01

UMA F2F 2010-11-01

Date and Time

- WG F2F on Monday, 1 Nov 2010, colocated with IIW XI at 11am-5pm PT (time chart)
    - No dial-in
    - No telecon this week
    - Boole room in the Computer History Museum

Agenda

Let's meet 11am-1pm, then do a working lunch (order in pizza?), then finish with a long afternoon stretch.

- Let's use #umaf2f along with #iiw as hashtags for the day
- "IPR benediction"
- Identify note-takers
- Roll call
- Agenda-bashing
- Approve minutes of 2010-10-28 meeting
- Reminder of next week's telecon and time change (see Next Meetings below)
- Action item review
- Bounty program status
- Other updates from the wider UMA world
- Brief "UMA 101" session as necessary for newbies in the room
- User stories: add, review, prioritize, decide
- Resource/scope registration: go through latest proposal and issues and decide
- AOB

Attendees

1. Fletcher, George
2. Hardjono, Thomas
3. Machulak, Maciej
4. Maler, Eve
5. Moren, Lukasz

Non-voting participants:

- Charles Andres
- Alan Karp
- Henrik Biering
- Joseph Holsten
- Jeff Stollman

Minutes

Thomas offered to serve as a notes-taker. Thank you!

New AI summary

2010-11-01-1	Alan	Open	Write up backup service/copy service use case, with reference to requester delegate scenario.	
2010-11-01-2	George	Open	Write up "problem B" as a user experience description that can be turned into a user story.	
2010-11-01-2	Eve	Open	Put the public-private continuum language and diagram into the Lexicon.	

Roll call

Quorum not reached.

Approve minutes of 2010-10-28 meeting

Deferred due to lack of quorum.

Action item review

- 2010-08-26-6 Eve Open Ping Denise Tayloe of Privo to see if she has interest in taking custodian scenario forward.
- 2010-09-02-1 Thomas Open Categorize all existing scenarios by their distinctive aspects. We'll discuss this today and decide whether to change this one.
- 2010-10-07-2 Sal, Domenico Open Propose the next version of the trusted claims solution, making appropriate simplifying assumptions.
- 2010-10-28-1 Eve Closed Work with Sal and George to put together a set of flow options/user stories for review at the IIW F2F.

Bounty program status

Two indications of submission interest have been sent in so far. The deadline for submission interest is 11:59pm Pacific time tonight.

Other updates from the wider UMA world

Maciej will speak on UMA at Devoxx in Antwerp soon. His paper for the MW4SOC conference will be published shortly.

Alan has worked on "transitive access" research, and will submit a scenario focusing on this. Our "requester delegate scenario" is related to this.

Over lunch, Alan demonstrated Tyler Close's work on "the Introducer" with some others (including Mike Hanson), which may offer one solution for the resource discovery problem in cases where the authorizing user wants to send the requesting party a link to a protected resource.

UMA for newbies

We reviewed the UMA entities, basic lexicon, trust a token/get a token/use a token protocol flow, the spec module map, and trusted claims (which Eve suggested should layer on top of the core protocol as an extension). The "Bob at Gmail" problem involves Alice wanting to grant access to Bob in his "bob@gmail.com" embodiment, perhaps through an access control list (ACL) that lists this email address. How can Bob show up, using some requester app, in a way that proves he is bob@gmail? This is illustrative of the trusted-claim problem.

In UMA terminology, we need to distinguish a "requester" (a software tool that implements an UMA protocol endpoint) and a "requesting party" (an individual or company that may assume legal liability for gaining authorized access).

The Liberty Alliance's Identity Web Services Framework (ID-WSF) solved for patterns like this, but it was considered by some as having too many components, and many web developers today consider it as being too complex.

There are many ways for Alice to provision knowledge of the resource to the requesting party. If the latter is Bob, for example, she can email the resource URL to him, hand him a business card with the resource location printed on it, or publish information about it openly on her blog. (Information cards and discovery services could also be employed.)

User stories: add, review, prioritize, decide

We reviewed the new User Stories page. Eve is trying to use some analogues of Agile techniquies to develop UMA user stories.

The page isn't very complete yet; Eve put in a sampling of user stories to test the columnal design, column sorting, and general "feel". Epics are tightly associated collections of stories; she has made epics all be in the "UX" category so that they focus on a human being's desire for some benefit. Some stories are "negative", in that they express some (typically malicious) entity's desired outcome that we want to avoid if at all possible.

It was observed that the stories zoom from high-level to low-level, and the "so that..." (motivation) portion of the descriptions could useful become more motivational and less technical/mechanistic even for the lower-level stories. Apparently the Ruby community likes to move the "so that..." portion earlier, in which case it would make sense to rephrase it as "in order to...". But since this is a fairly invasive change without obvious huge benefit we're likely to keep it in the current order.

Many new stories were suggested over the course of the rest of the meeting.

Resource/scope registration: go through latest proposal and issues and decide

The "share" pattern and other patterns: At first through discussing the "Register resources and available scopes" story and the pending "Request registration of resources and available scopes" story, and then through general discussion and examination of the latest proposal, we drilled down into the likely tasks an authorizing user might want to perform to try and figure out the flows that need to be supported. The proposal supports the first of these stories but not the second.

The goal in the SMART project was to enable a user to click on a "share" button while visiting the host, and then be redirected to the AM to map a policy to that resource. The project also has the goal for the host to accurately display to the user the status of a resource (whether it has been registered at the AM or not). These generally fall under what we started calling "problem A", acknowledged to be likely in-scope for UMA 1.0 based on the SMART UX study experiences.

We also considered the possibility that the user might want to **remain at the host while associating a policy with a resource** for lowest possible UX friction, which we called "problem B", and to **cancel protection for a particular resource while visiting the AM**, which we called "problem C".

Managed vs. unmanaged: We came up with some new terminology and assumptions to help us figure out the connection between protocol features and desirable user experiences. We believe it's likely that a host will want to offer to protect some or all of a user's resources there with the same AM, rather than offering to split the protection of resources among multiple AMs.

(After the meeting, Joseph observed that migration to outsourced authorization could be a reason why a host would offer to protect only some resources with that single AM.)

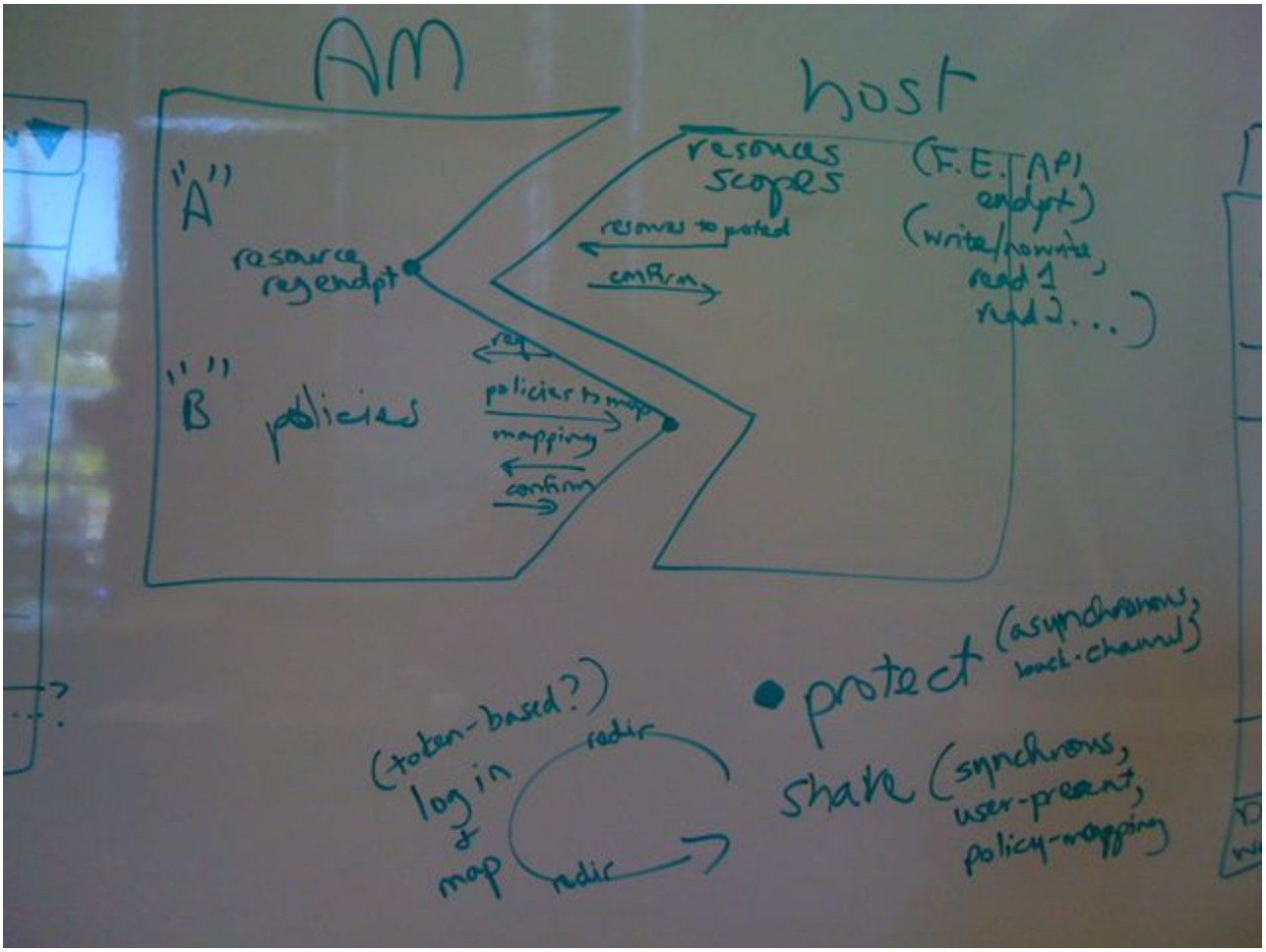
Any resources that a host ends up *not* registering with an AM are what we decided to call **unmanaged**. An unmanaged resource could be totally public or totally private, for example, and in both cases its existence is entirely unknown to the AM. It's expected that whatever a host does when access is attempted to such a resource, it does not engage in the step-2 "UMA dance" of issuing a 401 and sending the requester to the AM for an access token.

A **managed** resource could be said to be **protected** by the AM (in that a 401 with AM info is now in place no matter what), and it is prepared to be **selectiv ely sharable**, governed by which policy ultimately gets attached to it.

The public-private continuum: There are useful things an AM can offer in managing a resource, ranging from the "public" end – access tokens could be issued to all comers, but the user could track, monitor, audit, and observe analytics of access in various ways (see, e.g., the null scenario) – to the "hidden" end – either the user might not have mapped a policy to particular resource at all, so that access tokens are always denied to all comers, or the user might have mapped a very draconian policy to the resource so that it turns out no one is worthy of getting an access token.

It should be noted that AMs could offer user preference settings that allow for policies to be mapped to registered resources in some default pattern so that there are no resources that remain unmapped. Also, to the extent that a host *does not* need to resort to outsourcing token validation at the AM, and to the extent that an access token is long-lived, the AM is not exposed to every single successful access of a managed resource and so its analytics and audit logs might be more coarse-grained; perhaps this is amenable to an AM user preference setting too.

Problem B: Currently, the SMART host sample apps allow a user to click on a "Share" button, which (much like the Picasa-to-Piknik "Edit" flow) redirects the user to a different site, namely the AM site, so that they can associate a policy with that resource before getting redirected back. (The host actually registers the resource with the AM "silently" from the user's perspective before they get redirected.)



It was observed that the portion of this flow that might require the user to log in at the AM (if there's currently no active session there) before doing policy stuff could add to the user's burden. So we went down a path of exploring what it would look like for the AM to do something like "policy registration" at the host, to give it the names of policies that could be associated with that host's resources.

One idea we discussed was somehow using the host's already-existing host access token to help log the user in silently, vs. having the user log in as themselves. But this got hairier the more we thought about it; first of all, the host would be allowing the user to impersonate the host...which after all was delegated its authorization from the user. We found the semantics of this to be ugly. Secondly, the host likely has a subset of the user's rights at the AM, with the user's rights normally totally disjoint from the host's rights. So we gave up on this.

Ultimately we gained rough consensus not to target a solution for problem B in UMA 1.0. Although reducing friction is nice, it felt like premature optimization, which perhaps lots of other efforts (such as various "identity in the browser" efforts) would be trying to solve in a more elegant fashion. Also, this would add a lot of protocol overhead and state management and give hosts lots more responsibility for authorization jobs than we had intended to give them.

Problem C: The SMART project folks had assumed that it would be possible for a user, while visiting the AM, to "delete" a resource there (or, more accurately stated, canceling the AM's management of that resource's protection). Since the SMART host sample apps have found it useful to display to the user whether a resource is currently under protection management, the host would have to have a way to "pull" the current state of registrations from the AM in order to offer an accurate display.

(If the registration state also included some notion of which policies were mapped to which resources, then we'd partly be back in problem B territory, but we concluded that only resource registration information could be played back to the host if we were to decide to solve problem C.)

Other reasons why it would be useful for a host to pull resource registration state include various kinds of failure recovery where the host and AM have gotten out of sync. There are actually other, admittedly more clunky, ways to solve for the use case of letting users cancel protection "while visiting the AM", such as having the AM offer to open a new window to let the user visit the host and cancel protection management from there.

(After the meeting, in further discussions over beers, Eve suggested a more RESTful approach that could kill more birds with one stone, since we do have a design principle for RESTfulness and resource orientation. The host could POST a new resource registration construct at the AM representing the managed resources of a particular user, could add to it with PATCH, could GET its state at any time, and so on. We suspect Paul Bryan would approve! Maciej intends to take this idea forward.)

Proposal draft issues: We collected the following comments and issues on the proposal draft (in addition to the RESTful idea above, which may supersede some of these):

- Should it be possible to "boxcar" the registration of multiple resources? The current "uri" parameter could have a comma-separated list as its value, as long as the same available scopes applied to all of them.
- Is it okay not to explicitly label scopes in the request message (e.g. with a "scopes" parameter)? Maciej and Lukasz felt that it would be possible to accommodate extension parameters by requiring them to be named "x-something".
- Is it okay to only have single-language display names for the scopes? We felt yes, since the host would already know the user's preferred language (if it even offered its interface in multiple languages) and so would have the opportunity to pick the most suitable display names for that user.
- Can we solve a use case where the user just indicates to "protect" (manage) resources at the host without going through all the steps of actually selectively sharing them? (For example, this might come into play when the user wants to ensure that some or all of their resources are "non-public", as you can do when uploading a bunch of photos to Flickr in batch.) We believe so, since the resource registration piece involves only host-AM communication and doesn't have to have the redirect-the-user-to-the-AM piece on the end.

Next Meetings

- WG telecon on Thursday, 11 Nov 2010, at 9-10:30am PT (time chart) - Maciej to chair