UMA telecon 2017-08-23

UMA telecon 2017-08-23

Date and Time

- Special Wednesday time, 8-9am PT
- Screenshare and dial-in: http://join.me/findthomas
- UMA calendar: http://kantarainitiative.org/confluence/display/uma/Calendar

Agenda

- Roll call
- Approve minutes of UMA telecon 2017-08-17
- Schedule for upcoming weeks
  - No meeting Aug 30: use this time to review forthcoming specs preparatory to voting in a WG e-ballot for Draft Recommendations
    - Candidate motion when we’re ready: "Approve UMA 2.0 Grant rev nn and FedAuthz rev nn as amended by the instructions of UMA telecon 2017-xx-xx as Draft Recommendations and forward them to the Kantara Leadership Council to request immediate certification for an All-Member Ballot as Recommendations."
  - UMA V2.0 work:
    - Disposition of Comments/GitHub issues for V2.0/Grant swimlane/FedAuthz swimlane/Release Notes/UIG/Wikipedia/Grant rev 06/FedAuthz rev 06
    - Issue #348: No means no: see issue thread and also new email: decide today (related to issue #349 regarding invalid/expired RPTs – okay to upgrade them?)
    - Issue #347: We say it’s not an error when client doesn’t pre-register for a scope it requests, but should this be left flexible or made a definite error? See issue thread
    - Optional on issue #355d: add new swimlanes to spec or UIG?
- UMA2 logo discussion (defer till we’re in All-Member Ballot)
- AOB

Minutes

Roll call

Quorum was not reached.

Approve minutes

Approve minutes of UMA telecon 2017-08-17: Deferred.

Schedule for upcoming weeks

No meeting next week; everyone should review the final drafts for consideration in the e-ballot instead. (Justin and Mike are attending the MyData conference, and Mike is talking about UMA (and OIDC) there!) Eve discussed the question of repeating Public Comment review with the LC, and the conclusion was that it wasn’t necessary; we are just handling the comments that came in, and there’s no need to repeat that long review period. We just need to publish our DoC document properly and proceed.

UMA 2.0 work

#348 and #349:

Part of Justin's argument about refresh tokens is that they really shouldn't contain claims; their entire purpose is just to be exchanged for access tokens directly, with whatever existing scopes (permissions in our case) are there. However, the wording we picked to reflect this simplicity ended up being somewhat ambiguous: we said "MUST NOT treat...", not "MUST NOT do...".

Isn’t it the case that the AS should be invalidating tokens -- both access tokens (RPTs) and refresh tokens -- on the occasion of an RO's policy change, not necessarily a client's resource request? Because some policies could have partial effects on RPTs, then the proper effect might be to downscope a token. A "clever" AS (note that policy condition setting is outside the scope of the spec as stated in FedAuthz Sec 1.4.1, and we already advise documenting capabilities) has choices about how to coordinate policy setting and execute policy decision-making. There's a range of capabilities possible.

Note that GDPR has a "right to withdraw consent at any time" and its Recitals talk about "withdrawing consent without detriment". Any implementation that is trying to accede to a data subject's wishes as closely as possible would align with the resource owner's policy changes in terms of timing vs the client resource request, if possible. The ICO draft consent guidance (see page 37) says: "If someone withdraws consent, you should stop the processing as soon as possible. In some cases it will be possible to stop immediately, particularly in an online automated environment. However, in other cases you may be able to justify a short delay while you process the withdrawal."

For an expired (but otherwise okay) submitted RPT for upgrade, if the AS is willing to upgrade it, we think it's reasonable for that to be upgradable. We need to say this. For an invalid RPT or any other value for a submitted RPT that is not understandable, that is properly an invalid_grant error or the AS could ignore it. This is just normal OAuth error handling, and we shouldn't say anything about it. Same for an invalid PCT parameter, right? That's another optional parameter. There's already an error for invalid scopes. We decided not to say anything precisely because these two are optional parameters.
James proposes that the permission ticket could be **optional** when you're upgrading an RPT, because you're just asking for the same packet of permissions again, just extending the expiry time. Is this a new grant, a new sub-grant of the UMA grant, or what? Is this a replacement for the refresh flow? The use case for RPT upgrading is to allow the client to not have a million RPTs and be able to have more of a big RPT with all the permissions in it. The idea of no-ticket upgrading, we guess, is to pre-emptively refresh the RPT. And since the client is unaware of the fine-grained permissions inside an RPT, it won't necessarily get the benefit of "refreshing" any permissions inside its token without going through other gyrations.

**Decisions:**

- We choose this option: Clarify the current wording to explicitly **prohibit** the AS from re-evaluating policy: "The authorization server MUST NOT perform an authorization assessment calculation on receiving the client's request to refresh an RPT."
- We want to move 99% of FedAuthz Sec 1.4.1 to Grant Privacy Considerations (except for the PAT part). The PAT part should remain in FedAuthz, and it should be part of the Privacy Considerations.
- Currently it says "Supplying an existing RPT gives the authorization server the option of upgrading that RPT instead of issuing a new one (see Section 3.3.5.1 for more about this option)." We should add "Supplying an existing RPT (**which MAY be expired**) …" This is sufficient for indicating that this isn't an error condition.
- Create a new issue for James's idea of making the permission ticket optional when upgrading the RPT, and put the "extension" label on it.

**#347:**

At issue: Whether to allow the AS to decide, on whatever occasions, to reject the client's dynamic request for some scope(s) because the client hadn't pre-registered for that/those scope(s), and generate an error. We think that it's not a terrible overloading of **invalid_scope** to define/refine it (from OAuth's original definition) to let the AS use it for this purpose along with the other purposes we were already using it for.

The Note reads like this: "Note: It is not an error for the client to have requested a scope for which it did not pre-register because RequestedScopes might include that same scope, requested on the client's behalf by the resource server when obtaining the permission ticket. However, if such a scope were requested only by the client at the token endpoint, that scope would not be included in RequestedScopes." Can we simply edit the optional error condition into the error explanation and remove the Note? Yes.

**Decisions:**

- Remove the "Note" text.
- Add to the error conditions as follows: "At least one of the scopes included in the request does not match an available scope for any of the resources associated with requested permissions for the permission ticket provided by the client. **The authorization server MAY also return this error when at least one of the scopes included in the request does not match a scope for which the client is pre-registered with the authorization server.**"
- Check and try to change from "client pre-registers/ed" to "client is pre-registered" throughout where possible.

**#342:**

Cigdem has sent a fresh PDF to Eve with comments for inclusion.

**UMA logo**

Deferred.

**Attendees**

As of 7 Mar 2017, quorum is 4 of 7. (Domenico, Sal, Andi, Maciej, Eve, Mike, Cigdem)

1. Eve
2. Mike
3. Cigdem

Non-voting participants:

- James

**Regrets:**

- Domenico