GPII Preferences Edge Proxy Refinement I

Meeting 18/1/16 to take forward task point 1 for prefs editor integration

Present: Alan, Antranig, Cindy, Dana, Giovanni, Justin, Michelle, Simon

List of actors

We would like to draw up a LIST OF ACTORS (agents) in the system to orient ourselves.

1. End user (ultimate client) sitting at a web browser, wanting to use a GPII-enabled web site.

2. Preferences consumer - owner of some resources of a web site which is to be personalised by means of GPII preferences - e.g. the ILDH.

- May also require to edit these preferences

- This may consist purely of static resource and this actor may not provide any code

3. GPII OAuth 2 authorization server - hosts

- A logon UI which accepts the end user's credentials, after announcing the name of a preferences consumer on whose behalf they are being requested
- A database associating access tokens and authorization codes with a particular end user's use of a preferences consumer

We expect that the person (agent) who wrote the Login UI exposed by the /authorize endpoint of the authorization server is the same person who implemented all the rest of the server's logic (e.g. the one managing access tokens and authorization codes), since we do not expect to share access to the user's credentials more widely.

HOWEVER, we nonetheless expect these endpoints (in the particular variant of OAuth 2 which we are planning to deploy here, the AUTHORIZATION CODE workflow) to be hosted on DISTINCT ORIGINS (in terms of HTTP's Same Origin Policy) in order to prevent leakage of credentials into the preferences consumer's origin. (Recheck and refine this point in next meeting)

Refined list of actors

2. a) The PREFERENCES EDITOR contains content which is embedded onto the preferences consumer's site (showing, for example, the panel-based UI at the top of the user wireframes)

b) The EDGE PROXY SERVER which receives requests issued by the end user targetted at the authorization server - in particular all those listed on the GPII OAuth 2 Guide -

/authorize - QUESTION - why does the demo not visibly use this endpoint?

/access_token

/add-preferences - QUESTION - presumably there is also a get-preferences endpoint now

c) The MULTI-PERSONALITY PROXY SERVER which is the real physical server to which the edge proxy passes HTTP requests, after adding information to the HTTP headers identifying on which preferences consumer's behalf the request was received - this proxy server unpacks the requests produced by the edge proxy server into the equivalent full requests as listed in the guide. QUESTION - is the nginx configuration language already sufficiently good that the edge proxy server could do ALL of the work of this server? If not, how much better would it need to be/what is the hardest part of this work? (e.g. unpacking a formenc request or JSON payload and inserting an extra field in it)

Workflow

Draft WORKFLOW for end user interacting with the combined system

1. User is on the site belonging to the preferences consumer. Embedded on it is a prefs editor which they have interacted with. They select an action which requires interaction with the prefs server (in the wireframes, Import or Export) Step 0
2. We enter Step 1 of the OAuth flow - the user's browser is redirected to GET <authorization-server>/authorize - Step 1a - they see a LOGIN UI (implemented by the GPII authorization server)
    a. This endpoint is NOT exposed at the same origin as the preferences consumer to avoid leakage of these credentials
    b. In the current authorization server endpoint this is exposed at the URL /login
    c. The authorization server's endpoint /login will need to be exposed by the overall cluster's proxy configuration at a distinct origin
3. The login UI redirects the user back to the consumer's space, now including an AUTHORIZATION CODE in the URL - Step 2. This can be read from the consumer's origin by client or server-side code.
4. Step 3 - this is then separately provided to the /access_token endpoint to exchange for an access token which can be used in further requests for preferences

   a. The /access_token endpoint *is* proxied by the edge server.
   b. Part of this POST body is the client secret. It appears that providing this is the key responsibility of the multi-personality server. It will be maintained securely in its configuration and fished out when it receives a matching request from the edge server.

5. The access token can be used in preferences endpoints.
   a. Question - is identication purely by access token sufficient for all of these endpoints? it appears that the authorization server can always decode the preferences consumer’s id by fishing this out of its tables indexed by the token - does this mean that the edge server’s actions for all these endpoints can just be a no-op proxying?
   b. Question - i) do we still need two different grant types (authorization code, and client credentials) - ii) does the presence of these different grant types have any implications at all for the implementation of the edge and proxy servers?