Refresh Token and Consent Expiration

IETF 123 (Jul 25, 2025)

Nick Watson (Google LLC)

Recap: refresh token expiration

Two related but different concepts:

- Consent expiration: how long is the user’s consent good for (e.g. user opts for 30-day consent)
- Credential expiration: how long is the returned credential good for (e.g. when implementing RT rotation)

Recap: refresh token expiration

- For consent expiration, clients wish to remind the user of upcoming consent expiration while the user is active to avoid interruption of service if consent expires while the user is inactive.
- For RT rotation, clients need to know how often an RT must be used.

I-D Links

https://datatracker.ietf.org/doc/draft-watson-oauth-refresh-token-expiration/

https://github.com/njwatson32/rt-expiration

Spec tl;dr

Two new token endpoint response params.

```
refresh_token_expires_in

    The lifetime in seconds of the refresh token. This

    value SHALL NOT exceed the value in consent_expires_in.

consent_expires_in

    The lifetime in seconds of the user's consent. This

    value MAY exceed that of refresh_token_expires_in.
```

Questions for Group

- Use case for defining parameter(s) for clients to request time-limited tokens?
  - Authorization flow request
  - Token endpoint request
- *expires_in vs *expires_at?
- Worth adding any non-normative implementation notes about how consent expiration may be set/selected?
- Any reviewers? :)

(time permitting)

Example Error

DMA 5(2), which prevents different platforms under the same internet “gatekeeper” from sharing user data without consent.

1. AS issues token for YouTube.
2. Client calls YT API, triggers some feature that calls, say, Gmail.
3. An error is thrown, user doesn’t know why it failed or how to fix.
4. Even if the client resends the authz request, the AS doesn’t know what the issue was, happily reissues the YT AT.

Need a way to plumb error info through from RS to AS so AS can prompt the user to fix the error.

Generally Speaking

Some errors need HITL to resolve, typically issues with account state or underlying policy change, not credential itself.

Existing OAuth errors insufficiently expressive for the general case.

- From RS: Going to leverage `draft-lombardo-oauth-step-up-authz-challenge-proto` to pass an opaque “error handle” through RAR
- From token endpoint: Would like an additional “runtime” error code beyond invalid_grant, because the grant is not invalid, and clients often do not restart the authz flow upon receiving invalid_grant.

account_restricted

`account_restricted`

> **Indicates that some underlying state on the user’s account is preventing the current credential exchange. Clients SHOULD *automatically* begin a new authorization grant flow so the user can view or fix the error.**