Porting GaiaCookieManagerService to IdentityManager

Author: lowell@chromium.org
Status: Draft (2019-01-11)

Overview

We are undertaking a servicification of Identity in Chrome.

The first step in the serviciation of identity is to gather the set of disparate classes that make up Identity today into a single API (called IdentityManager).

GaiaCookieManagerService is one of these classes that should become a part of IdentityManager, this document covers its migration.

Context

What is GaiaCookieManagerService?

GaiaCookieManagerService exposes the ability to merge Google accounts known to Chrome into the cookie responsible for tracking the list of logged in Google sessions across the web (the APISID cookie).

In addition to allowing cookie modification, GaiaCookieManagerService allows callers to be notified when the list of sessions has changed (due to log in / log out, or other invalidation such as a password change).

Challenges

GaiaCookieManagerService is stateful / not thread safe

Examples of GaiaCookieManagerService State

- There is an InitCookieListener function, which will begin listening for cookie changes, notifying observers whenever there is a change.
- There is logic to make network requests to the Google Accounts APIs using GaiaAuthFetcher, including the ability to have 'pending' log in / log outs (see is_running).

Due to GaiaCookieManager's statefulness, and sensitivity to order of cookie changes, there can be concurrency issues if there are multiple instances / concurrent access.

Proposed Approach

We will fully encapsulate the GaiaCookieManagerService within IdentityManager, so that it is not directly exposed outside of internal identity code.

Benefits of fully encapsulating GaiaCookieManagerService within IdentityManager

Thread Safety

By encapsulating GaiaCookieManagerService within IdentityManager, we can ensure there will be a single GaiaCookieManagerService instance, running on a single thread (the same thread as IdentityManager), protecting against race conditions / concurrency issues.

Minimal Public API surface

By exposing only the minimal necessary subset of GaiaCookieManagerService functionality that is needed externally, we can reduce the public API surface area.

Significant parts of GaiaCookieManagerService API are only need to be used by internal identity code, and we can make this explicit.

Cleaner interface for callers

This migration creates an opportunity to redesign the interface based on how it's used.

The P0 is a direct conversion, which is already an improvement, but API improvements can be made opportunistically as identify opportunities for improvements during the migration.

Proposed API

Observer Methods

The following observer callback methods will be exposed via IdentityManager::Observer:

- OnAddAccountToCookieCompleted
  - Observer with same behavior as OnAddAccountToCookieCompleted.
- OnAccountsInCookieUpdated
  - Observer with same behavior as OnGaiaAccountsInCookieUpdated.
- OnAccountsCookieDeletedByUserAction
  - Observer with same behavior as OnGaiaCookieDeletedByUserAction.
- OnSetAccountsInCookieCompleted
  - Observer with same behavior as GaiaCookieManagerService::Observer::OnSetAccountsInCookieCompleted
  - Note: we should document that this is called only related to SetAccountsInCookie, and for general cookie changes callers should use other observers.

InitCookieListener

Usages of GaiaCookieManagerService::InitCookieListener will be replaced with IdentityManager::InitCookieListener.

GetAccountsInCookieJar

IdentityService::GetAccountsInCookieJar will replace GaiaCookieManagerService::ListAccounts.

GetCookieURLLoaderFactory

IdentityManager::GetCookieURLLoaderFactory will replace GaiaCookieManagerService::GetURLLoaderFactory.

AccountsCookieMutator

The following methods will be exposed by a new AccountCookieMutator class

- AccountsCookieMutator::AddAccountToCookie
  - Wraps GaiaCookieManagerService::AddAccountToCookie
- AccountsCookieMutator::AddAccountToCookieWithToken
  - Wraps GaiaCookieManagerService::AddAccountToCookieWithToken
- AccountsCookieMutator::LogOutAllAccounts
  - Wraps GaiaCookieManagerService::LogOutAllAccounts
- AccountsCookieMutator::SetAccountsInCookie
  - Wraps GaiaCookieManagerService::SetAccountsInCookie
- AccountsCookieMutator::TriggerCookieJarUpdate
  - Wraps GaiaCookieManagerService::TriggerListAccounts
- AccountsCookieMutator::ForceOnAddAccountToCookieCompleted
  - Wraps GaiaCookieManagerService::SignalComplete

These 'write' methods are intentionally segregated from the read portion of the API, and will not be widely used (currently the only non-test consumers are oauth2_login_verifier.cc and account_reconcilor.cc and account_consistency_service.mm)

GetCookieManagerBackoffEntry

Usages of GaiaCookieManagerService::GetBackoffEntry will be replaced with IdentityManager::GetCookieManagerBackoffEntry.

CancelAllLoginRequests (exposed via identity_test_utils.h)

Some integration tests call GaiaCookieManagerService::CancelAll to avoid race conditions.

Since this functionality isn't used outside of tests, it will be exposed via identity_test_utils rather than directly on the IdentityManager interface.

Note: this test-only API method can be removed if workarounds are found that make these calls unnecessary, but this is out of scope of the initial API conversion.

Example Conversion CLs

OnAddAccountToCookieCompleted

Usages of GaiaCookieManagerService::Observer::OnAddAccountToCookieCompleted should be replaced with IdentityManager::Observer::OnAddAccountToCookieCompleted.

Example CL.

OnGaiaAccountsInCookieUpdated

Usages of GaiaCookieManagerService::Observer::OnGaiaAccountsInCookieUpdated should be replaced with IdentityManager::Observer::OnAccountsInCookieUpdated.

Example CL.

ListAccounts

Usages of GaiaCookieManagerService::ListAccounts should be replaced with IdentityManager::GetAccountsInCookieJar.

Example CL.
See also the follow-up CL related to ListedAccount<->AccountInfo conversion