OpenFeature - SIG - Remote Evaluation Protocol

Disclaimer

Some terms in this document are used, that need to be defined or renamed:

- **Wire protocol:** A special REST *(or similar)* API that a vendor/rules engine might implement, at which point they'd have a baked-in client for all the major open-feature clients.
- **Remote flag evaluation:** calculating a flag's value on a remote machine, given an input evaluation context.
- **Generic provider:** an OpenFeature provider ready to communicate through the wire protocol.

All those terms need to be changed/validated by the community, so feel free to propose a better alternative.

Description

The ~~wire protocol or~~ OpenFeature Remote Flag Evaluation Protocol, is a mechanism that allows the use of generic providers to connect to any feature flag service that supports the protocol.

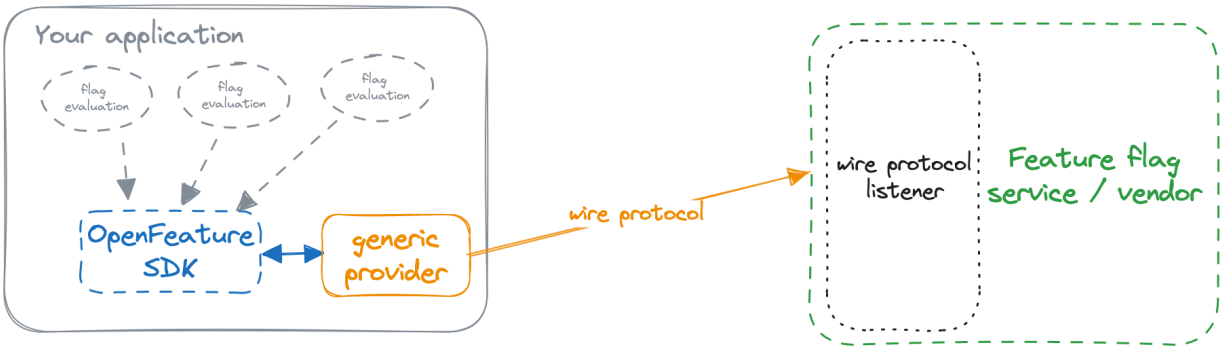
Currently feature flag vendors are expected to write OpenFeature providers for every language they support. With the wire protocol, they would only be required to implement a single set of endpoints *(technology needs to be defined)*, that could be consumed directly by all of the OpenFeature SDKs.

In the event that the wire protocol does not provide the required capabilities for the vendor, they could write language-specific providers if they wished.

This has 2 major objectives:

1. Ease the adoption of OpenFeature for the different feature flag solutions, by providing a simple way to integrate with the OpenFeature standard without developing OpenFeature providers in each language.
2. Ease the integration of OpenFeature for open-source projects, by avoiding them to package all the existing providers or by adding a plugin system to support multiple providers.

Those projects will be able to integrate only with the generic provider and be compatible with all the feature flag solutions implementing the protocol.



In Scope

- Define the remote flag evaluation protocol.
    - Schema
    - Technology (Connect, Buf, OpenAPI, gRPC, … ?)
        -
    - …
- Client and server SDK paradigms are part of the scope.
- Define what a generic provider should be.
    - Do we need a handshake?
    - How do we retrieve feature flag service metadata?
    - Should we do caching?
    - Capture telemetry data/event reporting?
    - …
- Define how a providers should act with the wire protocol.

Out of scope

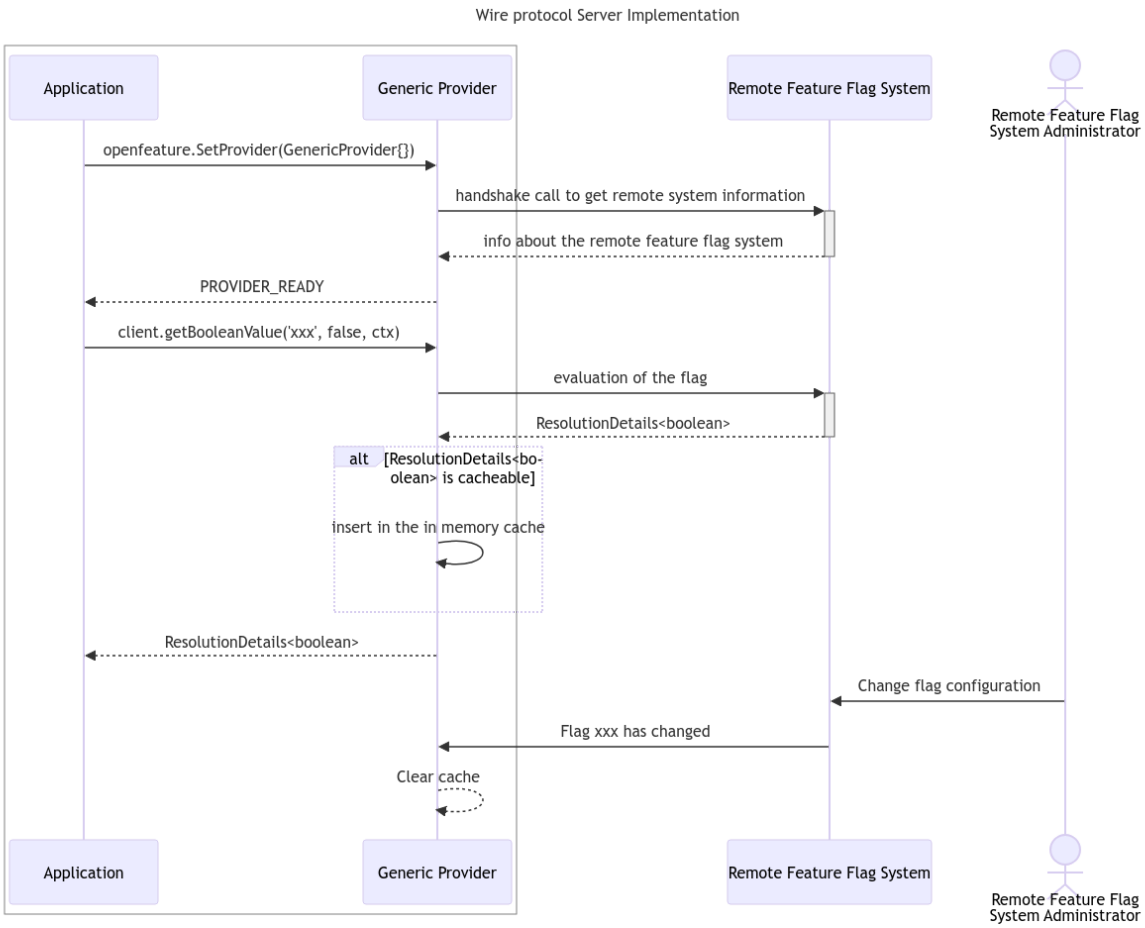
- In-process flag evaluation.
- Standard Feature Flag format definition (DSL).

Non-Functional Requirements

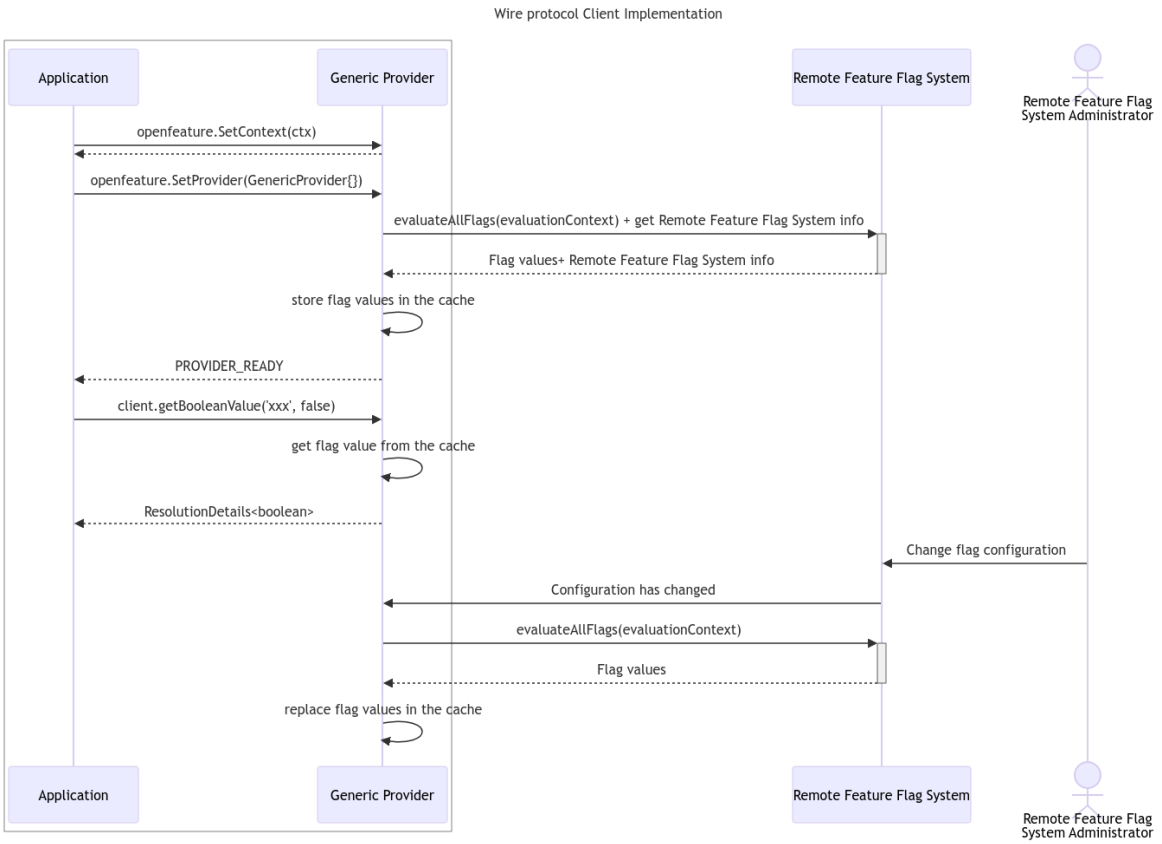
- Consideration needs to be made for latency and the minimization of requests being made
- Thought needs to be given for privacy/security (i.e. there is no privacy by design? Should requesting flags just require some sort of Environment/Project key and nothing else? AFAIUI that's how most providers work)

Example of flows

Server Implementation



Client Implementation



- Bulk evaluation needed for client-side, but we should consider in the client to ask for a set of flags.
- We need to have a cache invalidation.
    - Polling is a possibility
    - gRPC or other solution can help
- Check with vendors to get some feedback on how different players are doing
    - Unleash do polling
    - GO Feature Flag is doing websocket
    - Server set events
-

Project steps proposition

1. Agree on the scope.
2. Agree on the naming.
3. Define the protocol schema.
4. Find 1 or 2 feature flag services that are willing to implement a 1st proof of concept of the wire protocol.
5. Implement generic providers to test it with the available POC implementation.

Dec 1, 2023 - OpenFeature: wire protocol kickoff

Recording:

Notes:

- **Naming**
  - Wire protocol is confusing and we are looking for a better name.
  - Thomas Poignant will open a discussion in GitHub to find a better name.
  - We have a few options
    - Remote evaluation protocol
    - Remote Flag Evaluation Protocol
    - Standard protocol
    - OpenFeature protocol
    - Out of process evaluation protocol
- **Goals of the protocol**
  - Before defining the protocol we will create a doc containing the different high-level goals we want to follow for the protocol.
  - We have mentioned the following one during the meeting:
    - Reducing dependencies *(specially on the client side)*
    - Portability
    - Ease of utilization
  - We will not spec anything in the beginning to make the discussion more fluid.
  - We will have to spec it later but probably not in the actual spec
    - Dedicated spec
    - Or in the appendix
- **Technology to use**
  - Different options
    - REST API / Websocket / Server Sent Event
    - Connect
      - Can do REST API (but not in a restful way)
      - Can do gRPC
    - gRPC not mature enough for UI side
  - We need to check if the technology support the same coverage of languages that the one OpenFeature is covering
  - Reducing dependencies as minimum specially on the client side.
  - We should check with different flag management systems what they are using.
- **Flag management system survey**
  - In order to meet existing flag management systems where they are, we are willing to ask some questions of the existing solution.
  - Check with the vendors which systems they are using

        - polling vs push,
        - gRPC vs REST
        - Also check with which languages we should start.
    - Michael.Beemer@dynatrace.com and Pete will work on the survey and build a matrix for it.
- **Should we start with client-side or server-side?**
    - Client-side will fit more use-case but harder to build
        - For client-side we will need a flag management system->provider from day 1 to invalidate the cache.
    - Server-side should be more straightforward.
        - We don't need cache in the beginning.
- **Extension discussion**
    - **Telemetry and data collection**
        - How can we send telemetry data?
        - Hooks maybe not enough
    - **Dot notation getBooleanValue("myflag.myproperty")**
    - We need to give flexibility to people using the generic providers to build around it.
- **Interested parties to POC the wire protocol**
    - GO Feature Flag
    - flagD
    - Spotify *(no strong commitment)*
    - Unleash *(no strong commitment)*

- **Actions:**
    - Start discussion about naming **(Thomas)**
        - https://github.com/orgs/open-feature/discussions/288
    - Start a document about the goal of the wire protocol **(Todd & Thomas)**
    - Create a survey for vendors about what tech they are using **(Michael & Pete)**