RAIDAR

Rights & Asset Information in Decentralized, Authoritative Repositories

a joint project of
MIT and
Berklee College of Music

agenda

problems	• Background • Use cases
principles	• Key design constraints & objectives
architecture	• High-level functions & key APIs
roadmap	• Scope of v1 & beyond
next steps	

problem

To determine rights and rights-holders for music, one must employ cumbersome methods that often fail to return comprehensive, trustworthy results.

Music creators encounter inconvenient, time-consuming and costly processes when establishing rights to their work.

As a result rights and rights-holder data too often is:

- Not recorded, or not recorded comprehensively;
- Not discovered upon search, or the search yields incomplete or out of date results.

The situation impedes the ability to properly license music and to promptly and accurately compensate its creators for its use.

Background

Music (an asset) includes:

- Rights: publishing, performance, recording, synchronization, derivative…
- Rights-holders: songwriter/composer, lyricist, arranger, performers…
- Each right may be sold to a new right-holder or licensed. The sale or license terms may include collection & distribution of royalties.
- Metadata about both rights & licenses include:
  – provenance
  – veracity

Background

Problems with current methods

scale	• >20,000 new musical works uploaded each day
discovery	• Authoritative, comprehensive rights data hard to find • License data equally difficult; sometimes conflicting • Rights-holders cannot be found
trust	• Multiple licenses to each right • Conflicting claims & authorities • Lack of provenance • Opaque methods
expensive	• Rights registration can be relatively costly, especially for work-in-progress
slow	• Rights & licenses discovery, conflicting claims resolution, and royalty payments can take months/years

Use cases

artist	• Establish rights to a work • License those rights to a work • Identify, resolve conflicting rights/licenses
publisher, label, streaming service, etc	• Determine who holds which rights to a work • Contact rights-holder(s) • License or purchase those rights • Identify, resolve conflicting rights/licenses
royalty services	• Distribute collected royalties to correct recipients in timely manner
libraries	• Make available to others well-researched rights metadata

RAIDAR functional scope (v1):

- Record authoritative information about rights associated with music.
- Accept inquiries from the public.
- Return a comprehensive response to those inquiries; e.g., to allow interested parties to license use of the music.

v1 limits

- Controlled compositions: all rights held equally by same person(s)
- Do not contain derivative works; e.g., samples of other music
- Do not involve 3rd party licenses; e.g., publishers, record labels

Data scope (v1):

asset description	• Musical work, or • Link(s) to authoritative source(s) containing the work
asset metadata examples	Examples: • Title • Date/time of creation • Creator identity, pseudonym(s)
right description	Examples: mechanical right, performance right
right metadata	Examples: • Right-holder(s) identity • Geographical scope of right

Design principles

decentralized
- Do not rely on a single source for any essential function; e.g., data attestation, authentication, identity.
- Example: multiple methods may identify a recorded composition (ISRC, ISWC, HFA #, self-sovereign GUID)

authoritative, compre-hensive
- Includes provenance, source expertise & reliability
- Triangulate; include conflicting claims & resolution
- 1 query to any repo yields comprehensive reply from data held by all repos

future-tolerant
- technology-agnostic: for each function (e.g., database, immutable ledger, interface, proof of identity), allow implementation with a variety of technologies equivalent in ability and performance
- allow replacement or upgrade while maintaining compatibility with previous implementations.

Design principles

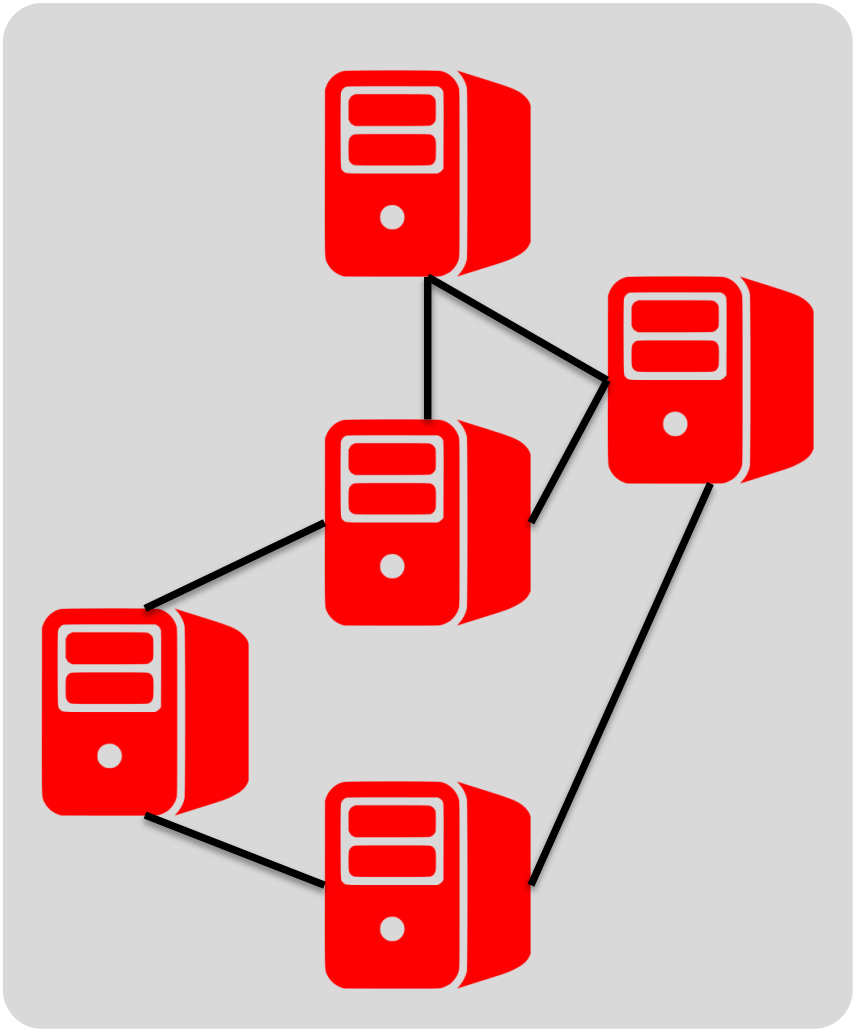
easy to use	• Inexpensive • Simple to integrate with other music tools; e.g., DAW, recording gig tracker apps
international	• support multilingual data and user interfaces • accommodate asset rights as defined in different regulatory/legal regimes.
open interface	• publish all interface specifications
open source software	• publish exemplary s/w for each side of every open interface (MIT license)
standards	• use relevant, published standards which: – meet functional requirements – have license terms equivalent to MIT license

Performance

availability, repeatability

- Similar to that of the internet, telephone network, and electric power grid in technically-advanced parts of the world.

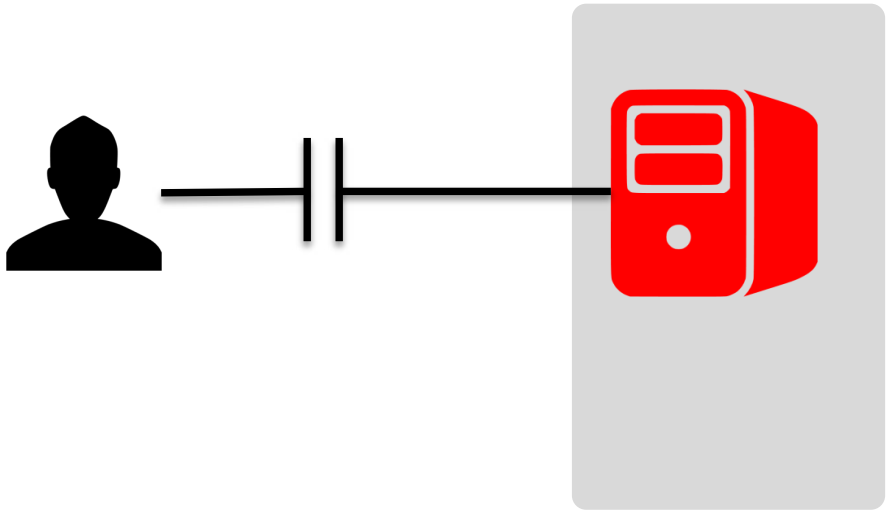
Architecture



Partial mesh of RAIDAR servers (“repo”)

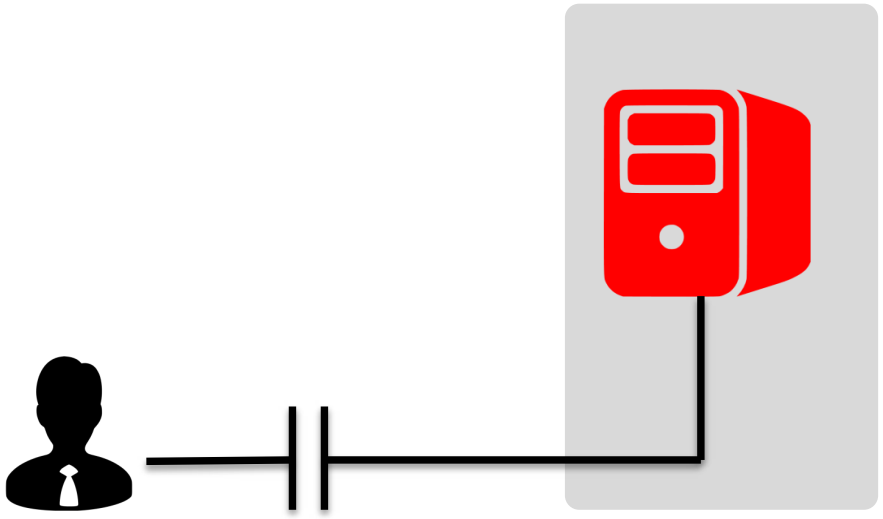
Architecture

User (client) app ↔ subscribed repo (server) interface (RESTful).



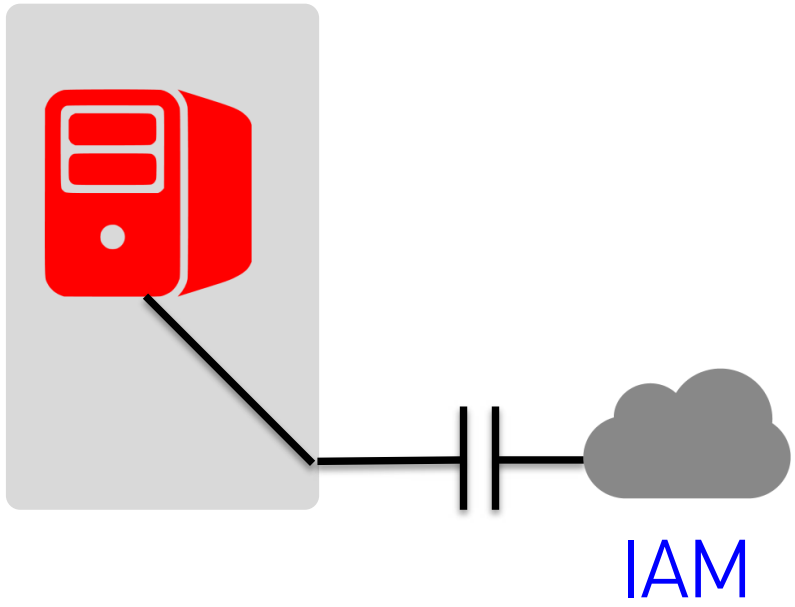
Architecture

Admin client app to manage subscriptions & authorizations.



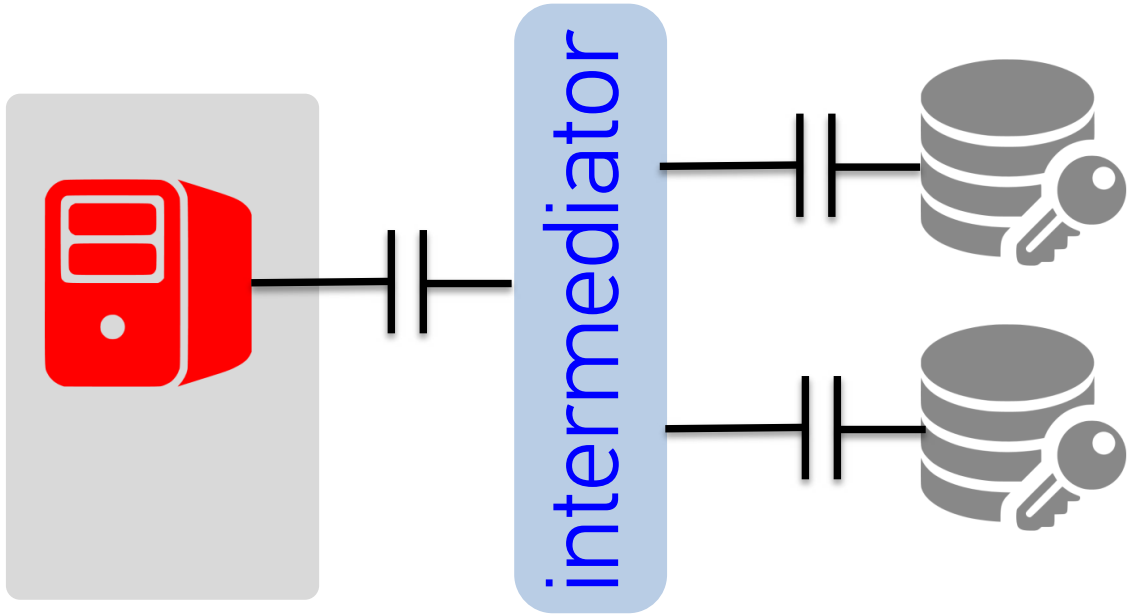
Architecture

Repo employs external identity & access management (IAM) service; e.g., OIDC.



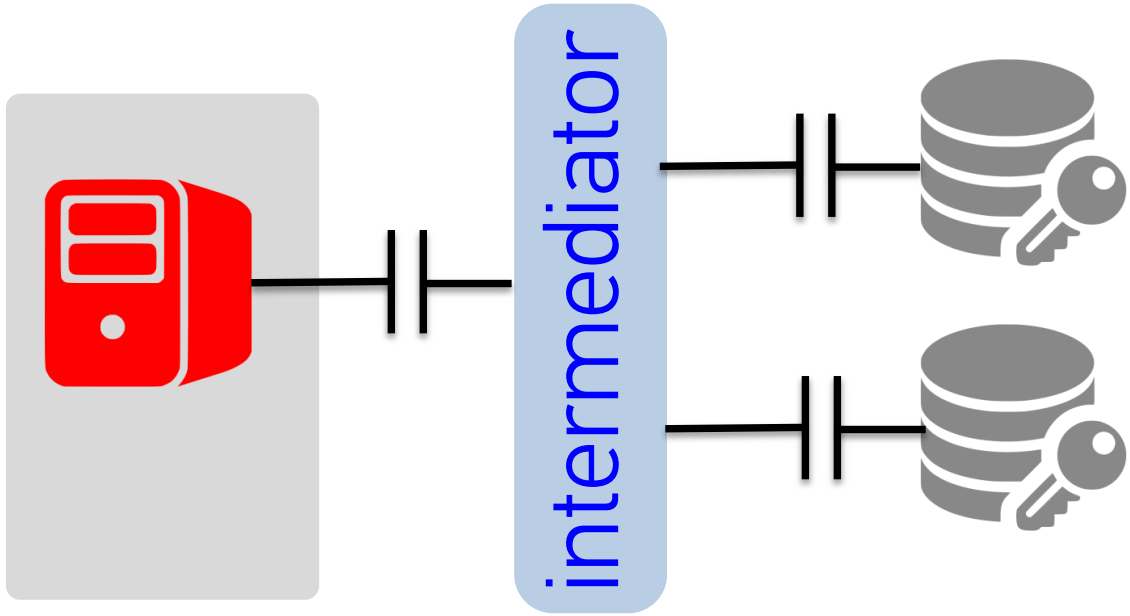
Architecture

Repo records data
signed by authorized sources
to redundant secure data storage service(s).



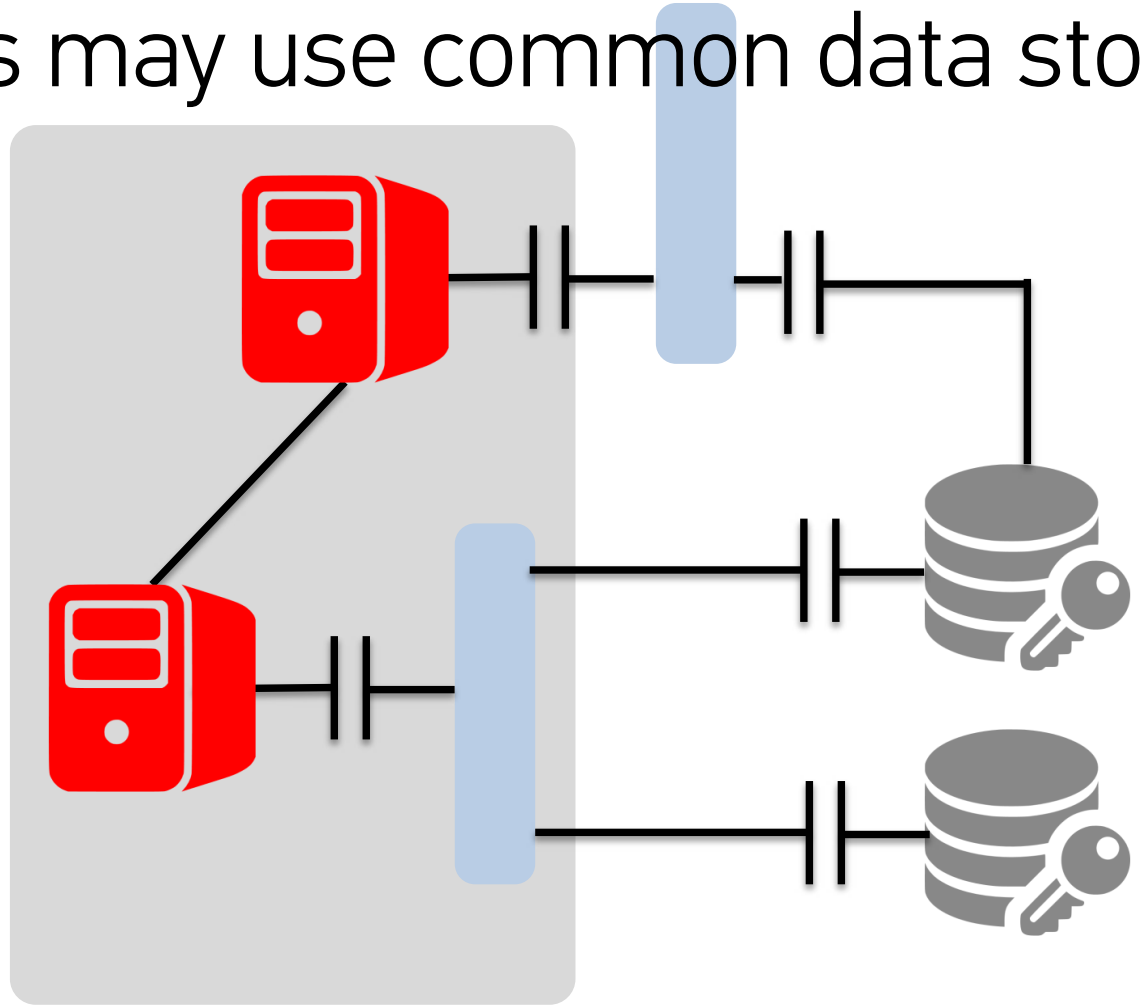
Architecture

Intermediation presents uniform API to repo.



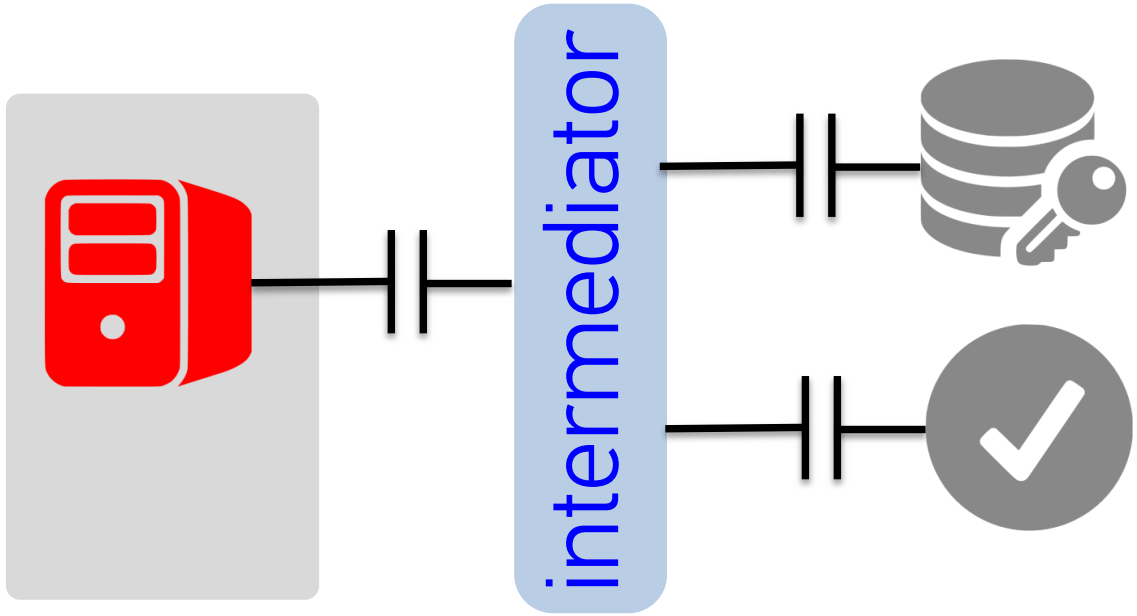
Architecture

Multiple repos may use common data stores.



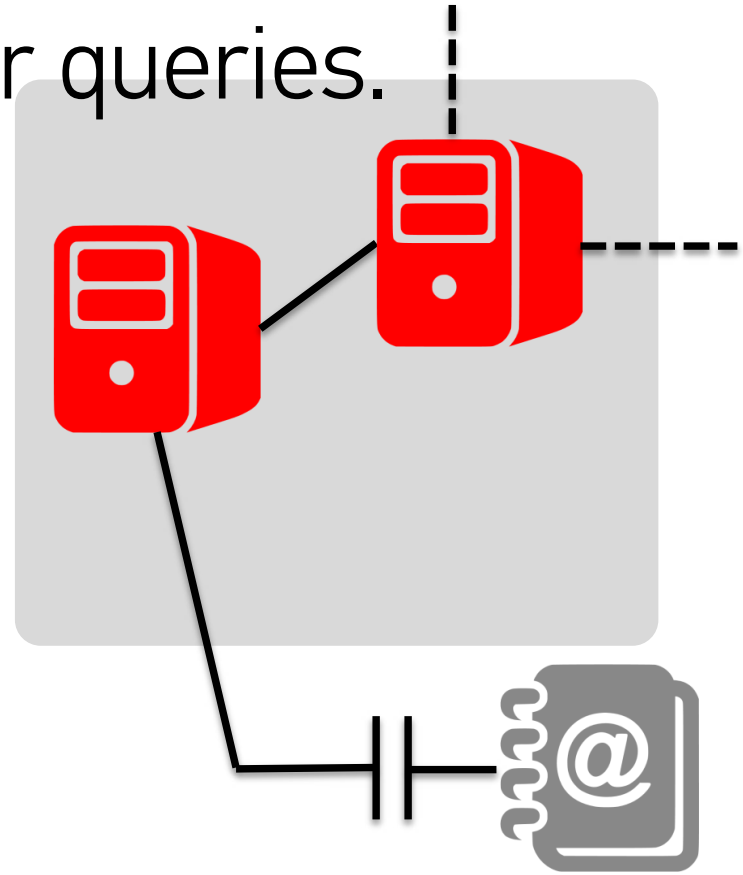
Architecture

Intermediator delivers proof of correctness with retrieved data; e.g., ledger verification.



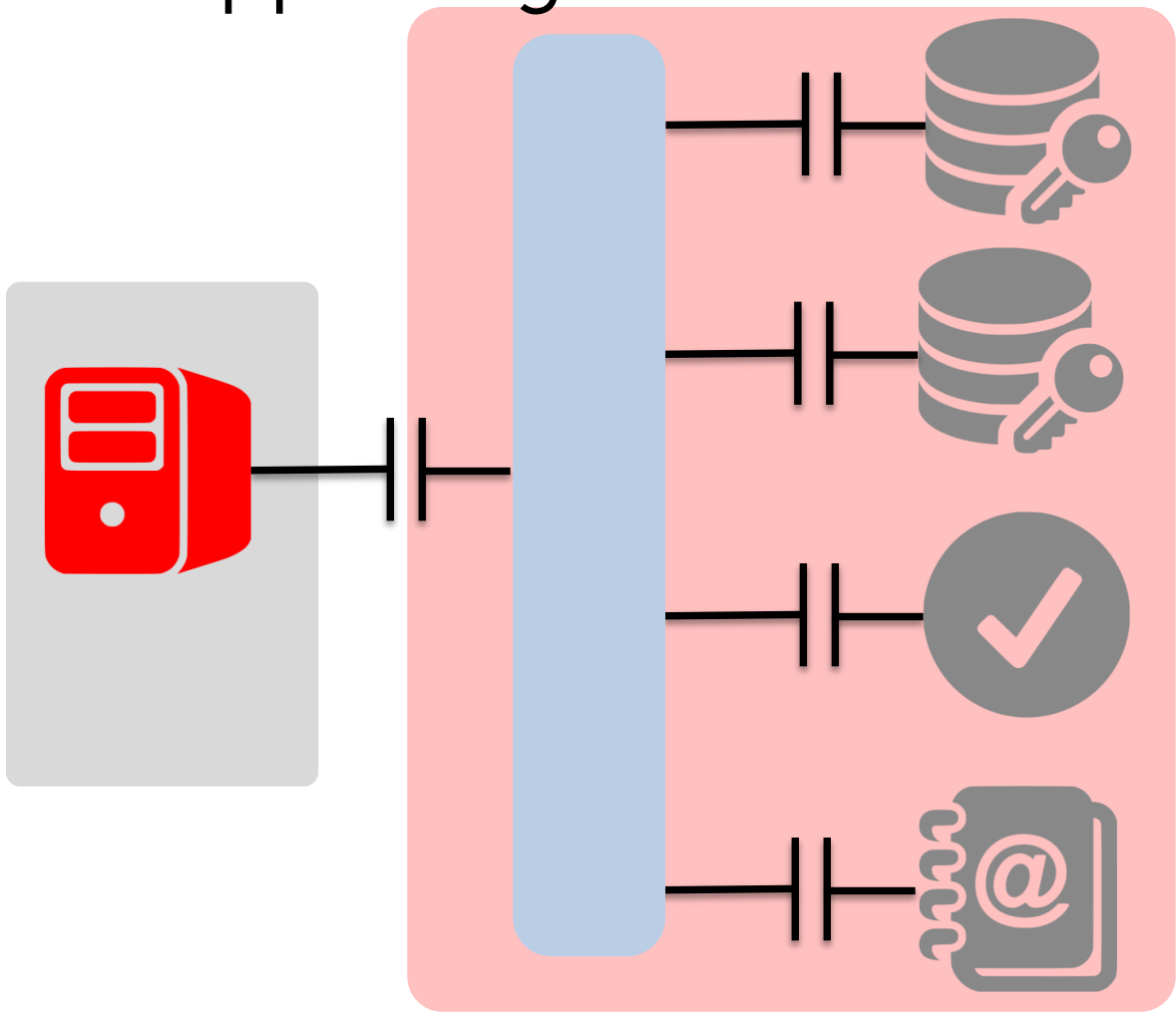
Architecture

No repo contains all data.
Look-up services (e.g., content addressing) & inter-hub queries support comprehensive replied to user queries.



Architecture

New open-source services (e.g., IPFS) may provide multiple supporting functions.



Architecture

Key design task for each API shown above:

- definition of message flow & contents.

Road map to v1

API design
- data structures, messages employed by multiple APIs
- user-repo
- intra-repo
- generic data store service mediation
- generic verification service mediation
- generic lookup for non-local content
- generic identity & access management service mediation

Road map to v1

implement
- artist user app
- searcher (e.g., publisher, streaming service) app
- 3-4 repo instances: not fully connected, partially-overlapped data
- with simple DB, dummy ledger, dummy IAM
- add 1, then 2 compliant DB alternatives (SQL, noSQL, IPFS)
- add 1, then 2 compliant ledger services
- add 1, then 2 compliant IAM services
- bulk-loader app

load
- subset of Berklee data (large, well-researched)
- full Riptide Music Group data set (50k works)

operate
- open for use by Berklee students

Beyond v1

open usage	• Open to other users
increase number of repos	• Open to additional repo instances • Add interfaces to other data sources
expand data scope	• Uncontrolled compositions: different rights held by different person(s) • Derivative works • Licenses • Licenses by 3rd parties • Royalty events
expand asset types	• Multi-media works

You can help

- Architects/designers
- Software developers
- Open source alternatives to key functions to:
  - prove technical agnosticism
  - provide redundancy
- Funding