Mguard Repo Plans?

Tianxing Ma, Junxiao Shi, Adam Thieme, Varun Patil, Suravi Regmi

Motivation

Replace the TCP bulk insert mechanism in mGuard with a more scalable and cleaner approach for inserting data into a NDN Repo.

Initial Plan

Implemented the pubsub protocol in C++ as an alternative to TCP bulk insert.

Separate out the pubsub API from mGuard and create a C++ client for ndn-python-repo.

Discussion and Options

Use SVS protocol instead of PubSub

Update the mguard Producer to use SVS protocol in the ndn-repo instead of the PubSub Protocol

Create a PSync Repo

Implement a new Repo with PSync in C++

PSync Repo

WHY?

Instead of maintaining multiple C++ clients for different repo libraries, build a repo that works natively with mGuard.

WHY?

Already in use in MGuard
Lightweight integration: Reuse existing sync logic (currently used for manifest sync).
Fast to implement.

WHY?

Avoids overhead of supporting multiple sync protocols (e.g., SVS, pubsub).

WHY NOT?

Yet another repo: Adds maintenance burden.

Is this the right approach?

TASKS

New repo instance that subscribes to manifest streams. (already in use).

Repo internal storage system.

CHALLENGE

“**another repo to maintain**” unless performance and design goals justify it.

Moving from PubSub to SVS

WHY?

Follows standard architecture: Matches the existing ndn-repo approach, which already uses SVS

SEPARATION OF CONCERN

Producer just produces data; repo fetches as needed.

TASK

Update ndn **SVS C++ to version 3** (current support is only for version 2).

Challenges

SVS v3 is only implemented in **Go** and **TypeScript**.

Challenges

Requires adding support for **bootstrapping time, data validation, state vector merging** and other updates introduced in SVS v3.

Challenges

Ndn-repo uses SVS V3
Ndn-python-repo uses SVS V1
ndn-SVS in C++ uses SVS V2
for mGuard we need SVS V3

PRO

SVS v3 introduces bootstrap time, making it cleaner for initialization and sync consistency.
In mGuard for Producer crash case.

THING TO CONSIDER

Producer gets no feedback: Not notified if repo fails to store data.

Failure handling needed: If repo insertion fails, how is it communicated back?

Work Done

Started the update of ndn SVS V3 in C++

Testing and Debugging Ongoing

NDN-Play updates

- Use NDNts version published on NPM registry.
  - This avoids frequent breaking changes from NDNts-nightly.
  - https://github.com/pulsejet/ndn-play/pull/6
- Recognize SVS v3 TLV-TYPE numbers.
  - https://github.com/pulsejet/ndn-play/pull/5

Future Work

Implement repo-client at Mguard Producer

Use ndn-repo instead of ndn-python-repo

Thank you everyone here for their suggestions/discussion /help.

Questions?