File-Like ICN Collections (FLIC)

draft-irtf-icnrg-flic-02

Marc Mosko
PARC
Dave Oran -
Network Systems Research & Design

Background

- FLIC has been around for a long time; design was done back around the same time as CCNx 1.0
- Manifests are useful in NDN, but pretty much critical in CCNx:
  - "nameless objects" that just have a hash
  - segmentation for large objects
  - Collections, like directories of objects "lower" in a namespace
- It's been implemented and in use all along

Current State of Affairs

- -01 draft expired in 2018
- Original authors have mostly moved on to other stuff
- DaveO cajoled Marc and Christian to resurrect work with informal meeting at ICNRG in Montreal
- DaveO and Marc finally produced -02 a few weeks ago

IT'S ALIVE!!!

What hasn’t changed

- FLIC still uses the idea of a HashGroup
  - ordered list of content object
  - These point to other manifests or data objects
- FLIC still has metadata in the manifest
  - both at Node level (top level) and per-hash group
- FLIC encryption keys are unrelated to data encryption keys
  - so retrieval access does not imply data access.

What has changed: Namespaces

- Adds the concept of Namespaces
  - Defines the naming convention for manifest content objects and application data content objects. (Prior draft assumed CCNx nameless objects).
- Three defined namespaces:
  - Nameless operation
  - Single prefix
  - Segmented prefix (where each name is unique).
- Each HashGroup can use its own namespace
  - so manifest and application data namespaces could be different.

What has changed: Encryption

- Syntax has changed a little to better accommodate encryption.
  - No information leaks about the manifest,
    - in prior format some metadata leaked
  - Now supports in-place encrypt/decrypt.
- One encryption key per manifest
  - prior draft allowed keys to vary by Hash Group.
- Specifies a pre-shared key encryption and two group key methods
  - All three devolve to the same encoding in practice
- Both the encryption mechanism and key location mechanism are extensible.

What has changed: Metadata

- Manifest metadata refactored
  - Allows both direct and subtree sizes and direct and subtree hashes.
  - Regularized between the node level and hash group level.
- General extensibility mechanism added to allow defining new Manifest-level metadata
- Supports both *Plain* Pointers and *Annotated* Pointers inside a hash group.
  - Plain pointers are as before -- just an array of HashValues.
  - Annotated Pointers allow adding metadata and extensions to each pointer, such as object sizes, traversal order, video decoding hints or other information.

What has changed: Miscellaneous

- Locators can now be an array, not just a single locator
- Much more detail in this draft, including both NDN and CCNx encodings for all three namespaces.
- Python implementation
  - Slightly out of date - it does not support annotated pointers yet.

Still to be done

- Code:
  - bring the reference Python implementation up-to-date with the draft
  - update the CICN implementation
  - provide an NDN implementation.
- IANA considerations section.
- Security considerations section.
- Update the text for seeking to a byte location to exploit the new subtree size information, if present.