Erasure Coding Extension for the Bundle Protocol

Angela Hennessy

Laboratory for Telecommunications Sciences (LTS)

Outline

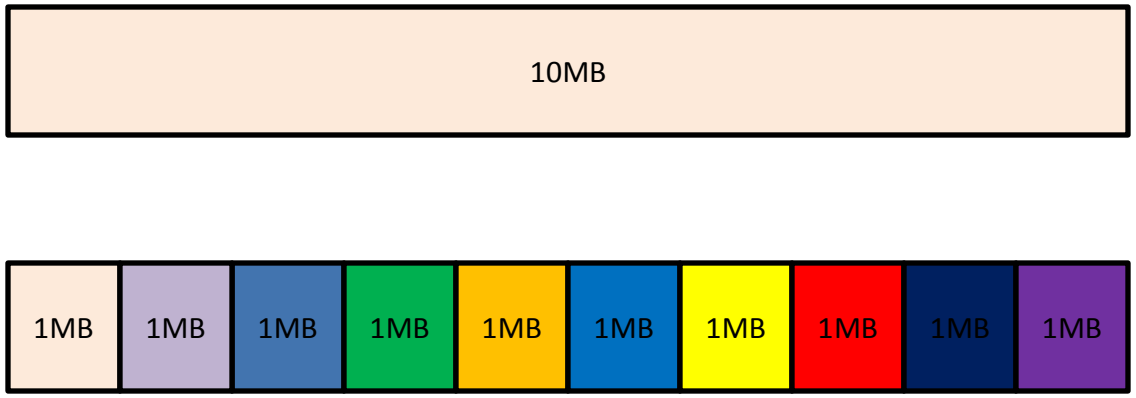
- Goals
- Background
- Use Cases
- Architectural Challenges
- Protocol Design
- Security Issues

Problem Statement

- Design a protocol to reliably send large amounts of data over disrupted networks.
  - Contact time may not be long enough to send the whole bundle, so we will rely on routing and bundle storage
  - The sender receives minimal feedback about the bundle transfer
  - Use forward error correction, no end-to-end acknowledgement expected.
  - Depend on existing DTN reliability and timeout mechanisms.

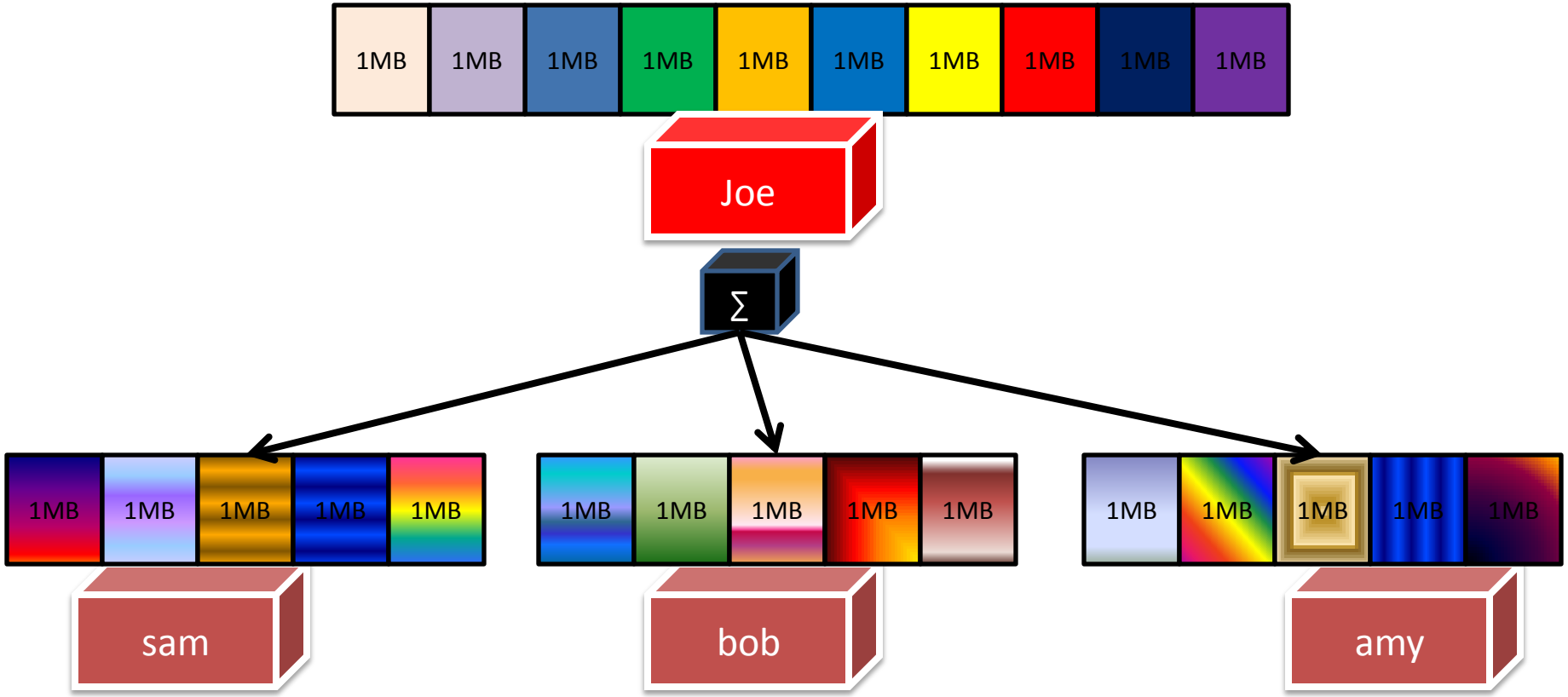
Background - Erasure Coding in DTNs

- Imagine trying to distribute a 10MB bundle in a DTN
- Idea: fragment into 1MB pieces



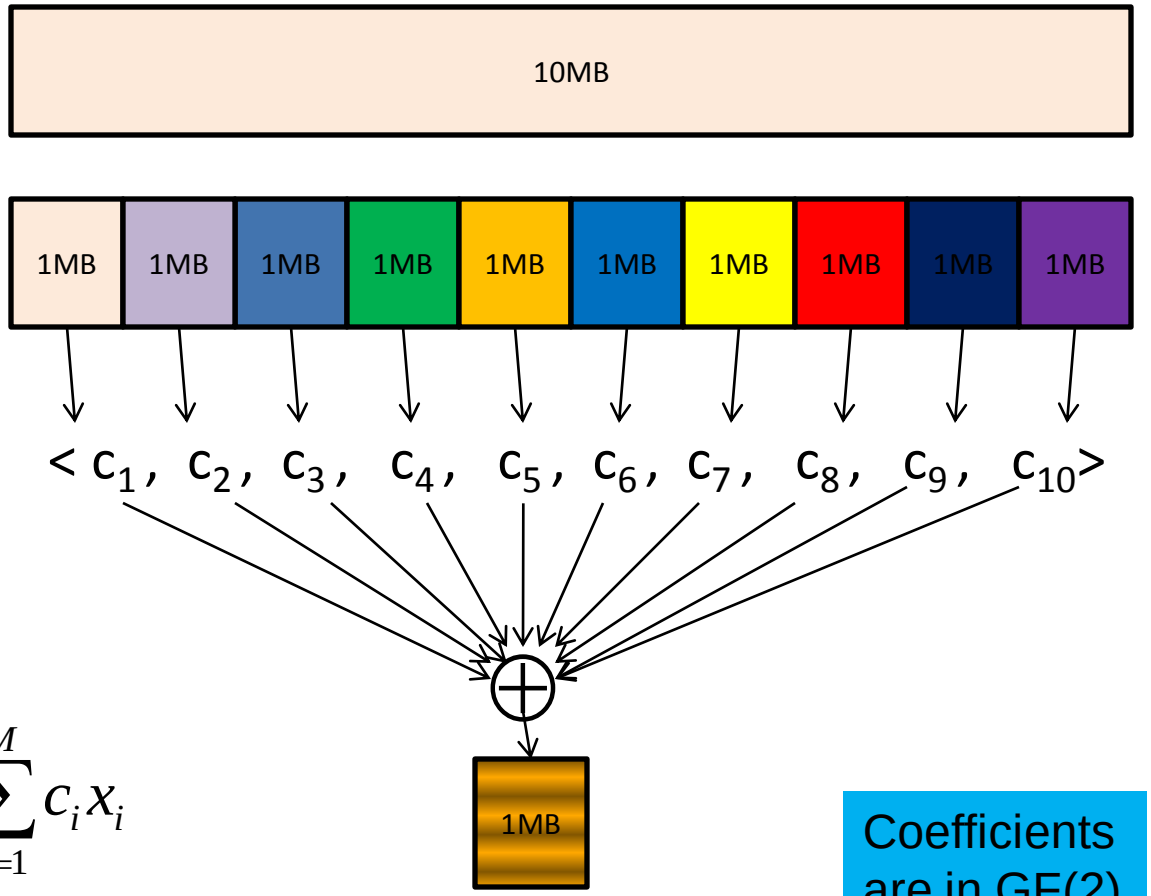
Erasure Coding in DTNs

- Send linear combinations of fragments
- A receiver can collect **any** ten pieces and recover data



Erasure Coding in DTNs

- Data Object
  (bundle) assigned a universally unique identifier (UUID)
- Source Symbols, x_i
  (fragments)
- Encoding Vector
  (vector of coefficients)
- Encoding Data, $w_c = \sum_{i=1}^{M} c_i x_i$
  (linear combination of fragments)
- Encoding = (Encoding Vector, Encoding Data)



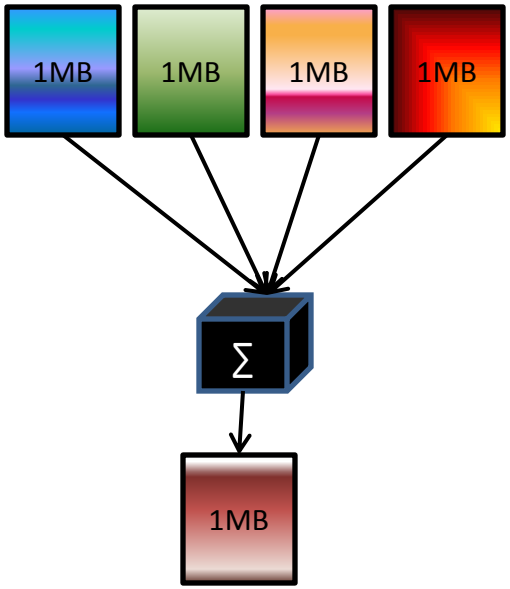
Coefficients are in GF(2)

Erasure Coding in DTNs

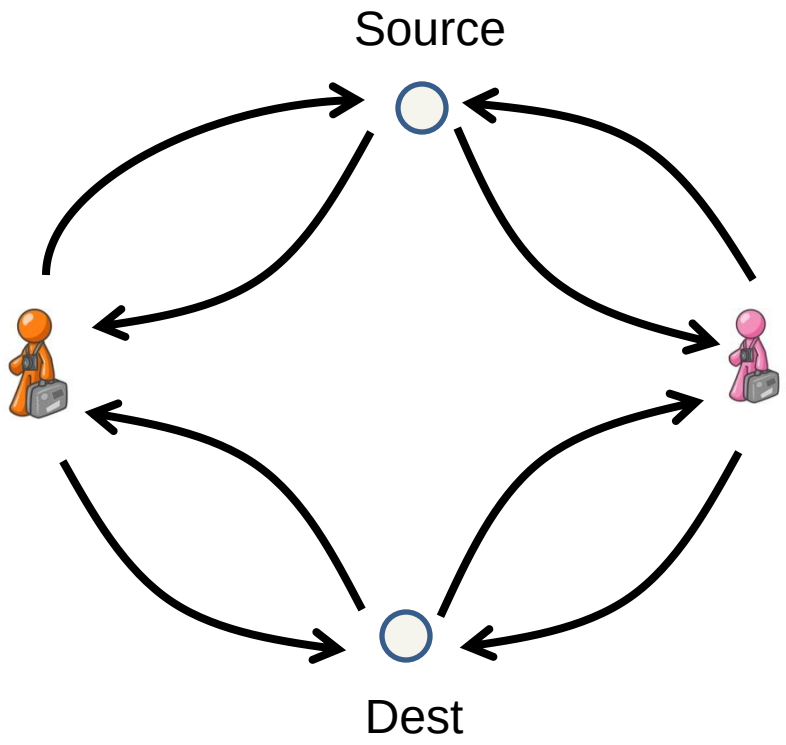
Given Encoding Data $w_{c^1}, w_{c^2}, \ldots, w_{c^r}$

one can re-combine them to obtain a new Encoding (re-encoding)

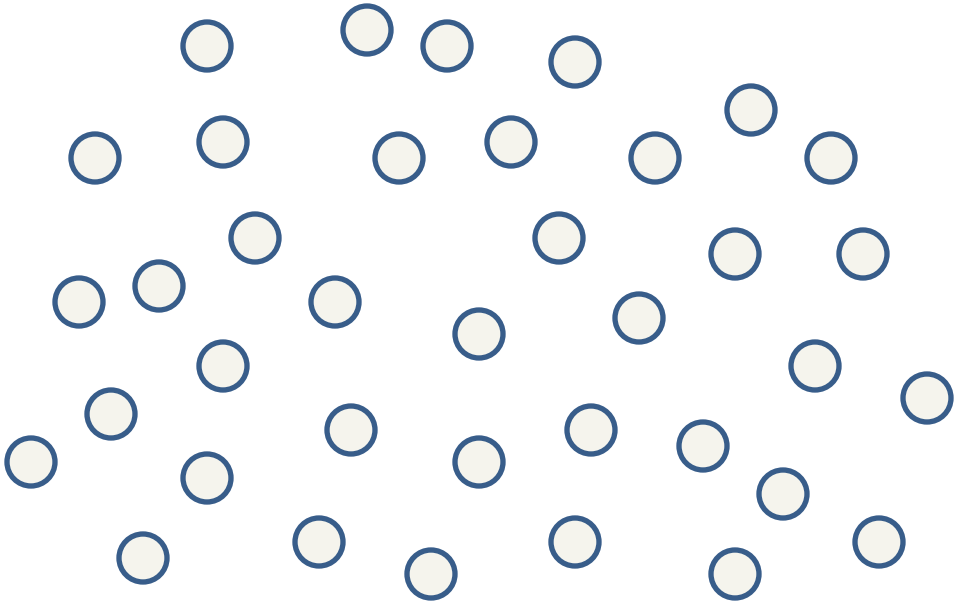
$$w_{c'} = \sum_{k=1}^{r} d_k w_{c^k} = \sum_{k=1}^{r} d_k \left(\sum_{i=1}^{M} c_i^k x_i \right) = \sum_{i=1}^{M} \left(\sum_{k=1}^{r} d_k c_i^k \right) x_i$$



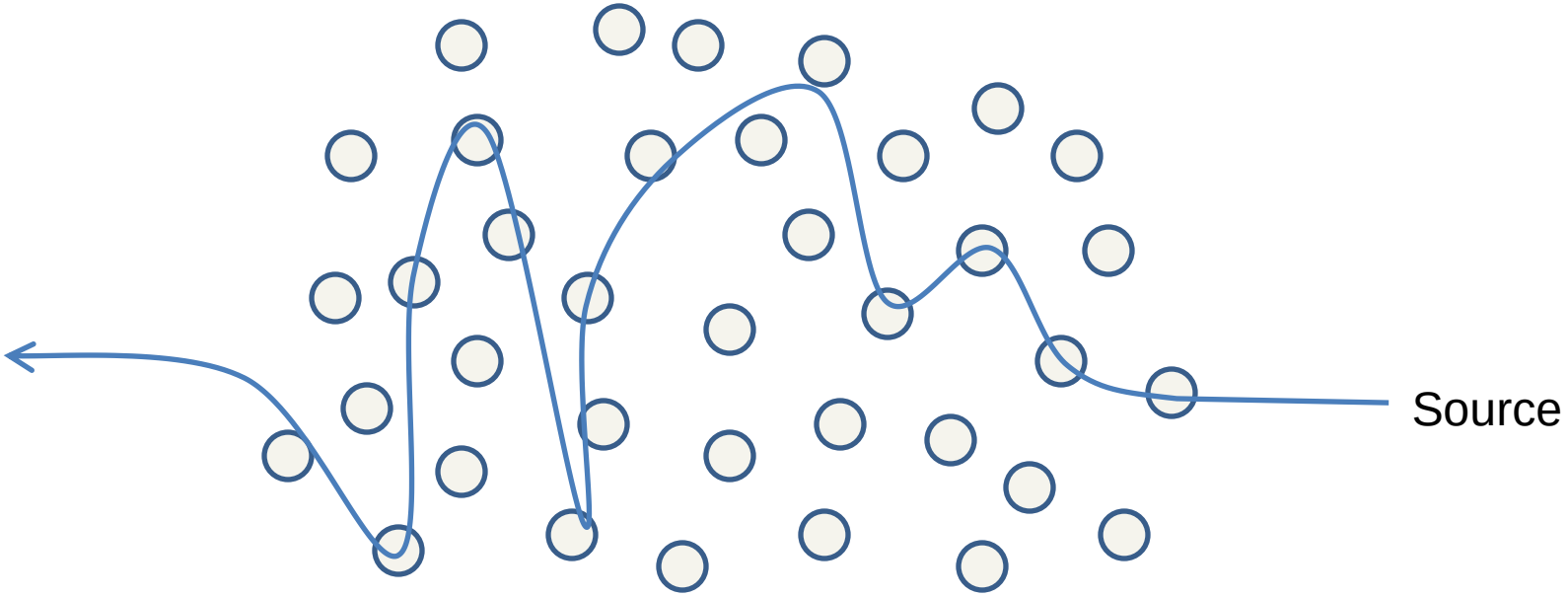
Use Case – Multiple Data Mules



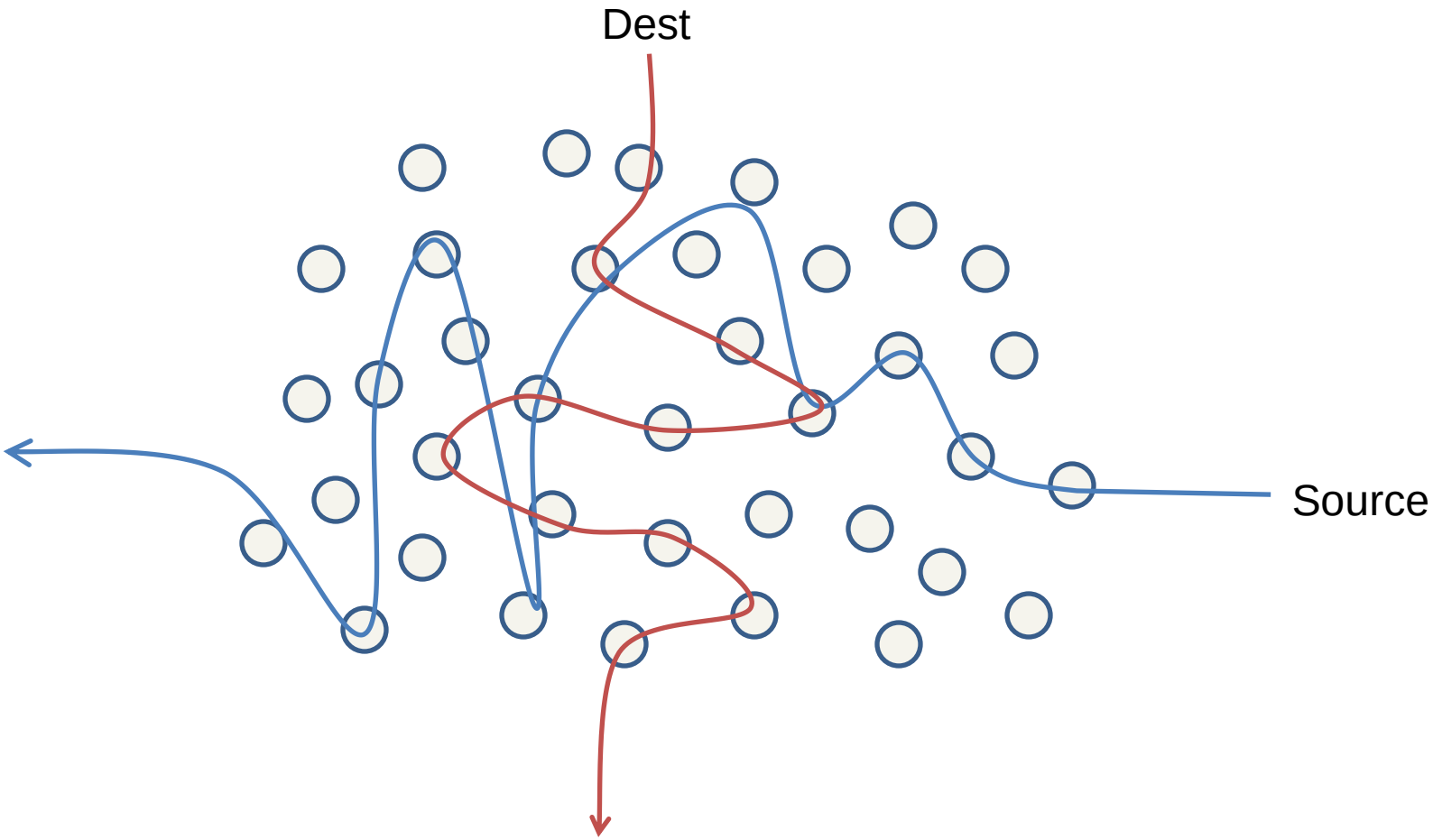
Use Case – Data Distribution



Use Case – Data Distribution



Use Case – Data Distribution



Design Assumptions

- The Erasure Coding protocol will use a bundle extension block and not modify RFC 5050.
- The Erasure Coding protocol will have one encoding per bundle.
- A bundle that contains an encoding can be further fragmented using the standard DTN bundle fragmentation.
  - Convergence Layer may limit max bundle size.
  - Encoding size should be adjusted to avoid fragmentation

Architectural Issues

- Should coding take place at Application layer, or in the BPA?
- Simpler at the Application layer:
  - Fragment input data (file, stream, etc.)
  - Generate fixed number of encodings
  - Each encoding in a separate bundle
  - Pass off to the BPA
- Advantages if the BPA is “coding-aware”
  - Generate more encodings if necessary
  - Intermediate nodes can generate new encodings from existing ones
  - Make intelligent routing decisions
  - Balance multiple encoding sets

Architectural Issues

- How should coding be implemented inside the BPA?
- As a Convergence Layer?
  - If the coding were along a single hop (between two BPAs), CL makes sense
  - Our use cases are between several BPAs, along several paths
- As a Router?
  - Allows the BPA to make intelligent routing decisions
  - Balance the generation and sending of encodings among several neighbors

Architectural Issues

- Generating encodings in the BPA:
  - Receives a large bundle to fragment
  - Encapsulate the bundle, then fragment and generate encodings
  - Modify fields in the primary block of the Encoding Bundles (Bundle Transfer Spec.)
    - Source EID of the new bundle is the node generating the encoding
    - Creation Timestamp is set to the time the Encoding Bundle was created
    - Life Time is changed to expire at the same time as the original bundle
- Decoding encoded bundles
  - Store encoded bundles as they arrive
  - When “enough” encodings have been collected, invert the matrix of Encoding Vectors

Architectural Issues

- The spec allows for any combination of “Erasure Coding architectural components”
  - Erasure Coding-aware DTN applications
  - Legacy DTN application
  - Erasure Coding-aware BPAs
  - Legacy BPAs
- Intermediate re-encoders (i.e. intermediate nodes generating new encodings) must be in the BPA

Architectural Issues

- Should the coding metadata (i.e. the Encoding Vector) be contained in a Metadata Block or an Extension Block?
  - If the coding is only at the application level (i.e. BPAs simply forward encoding bundles verbatim), then metadata block is fine
  - Since the BPA may modify the metadata (i.e. generate new encodings), extension blocks are more appropriate

Erasure Coding Specs

- Bundle Protocol Erasure Coding Extension
  - Defines the overall architecture
  - Describes coding at the application layer and in the BPA
  - Abstraction allows for different types of coding schemes
- Random Binary FEC Scheme for Bundle Protocol
  - Describes the specific encoding/decoding schemes
  - How to represent the Encoding Vectors
- Bundle Protocol Erasure Coding Basic Objects
  - Defines formats for transferring data between two applications ("File Data Object"), or between two BPAs ("Bundle Data Object")

Security Concerns

- Data can be encrypted/authenticated at the application layer or in the BPA (Bundle Security Protocol)
  - Using BSP, the payload and extension blocks can be encrypted separately
  - Interferes with re-encoding (intermediate nodes generating new encodings)
- If an Encoding Vector (in the extension block) is modified, decoding will result in scrambled data
  - Not possible to add end-to-end authentication for extension blocks in BSP

Questions?