Random Linear Network Coding (RLNC)-Based Symbol Representation

draft-heide-nwcrg-rlnc-background-00
draft-heide-nwcrg-rlnc-01

IETF/IRTF 104, NWCRG
March 2019, Prague

Janus Heide (Steinwurf)
Shirley Shi, **Kerim Fouli**, **Muriel Médard** (Code On Network Coding)
Vince Chook (Inmarsat)

Agenda

1. Current Version: Changes with respect to draft-heide-nwcrg-rlnc-00
   - Splitting draft-heide-nwcrg-rlnc-background-00 and draft-heide-nwcrg-rlnc-01
   - Scope
   - New Definitions
   - New Sections
2. Next version: Future modifications
   - Comments from the email list

Current Version: Overview of Changes

draft-heide-nwcrg-rlnc-background-00

- General background informational on RLNC
- Symbol Representation as a standardization target

draft-heide-nwcrg-rlnc-01

- Symbol Representation Specification
- Definition of “Symbol Representation”
- New figures (32-bit template)

#ThanksVincent

Current Version: Symbol Representation

Spelling Out Assumed Definitions

draft-heide-nwcrg-rlnc-background-00

- “Symbol representation specifies the format of the **symbol-carrying data unit** that is to be coded, recoded, and decoded. In other words, symbol representation defines the format of the coding-layer data unit, including header format and symbol concatenation.”

draft-heide-nwcrg-rlnc-01

- “Symbol representation specifies the format of the symbol-carrying data unit that is to be used in network coding operations, including header format and symbol concatenation.”

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| T |SYMBOLS|    ENCODER RANK      |  SEED or CODING COEFFICIENTS  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                       SYMBOL(S) DATA                          |
|                           ...                                 |
|                                                               |
+---------------------------------------------------------------+
```

Figure 1: A general symbol representation design.

Standardizing Symbol Representation

Flexibility as an argument for standardization

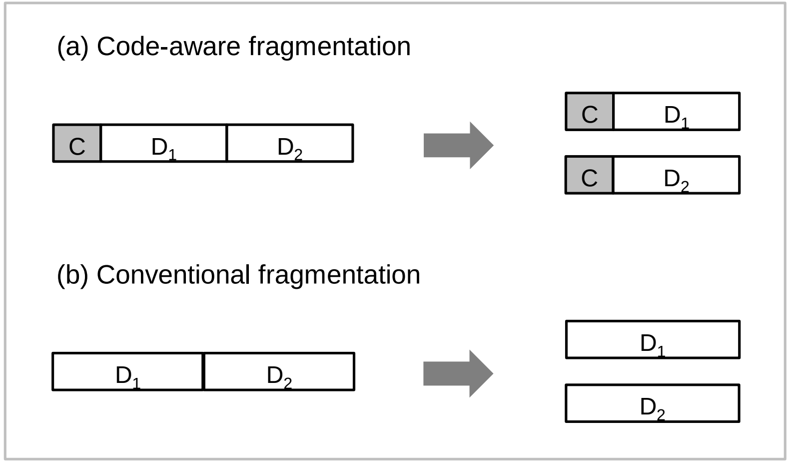
- Standardization is needed due to the flexibility of RLNC
- RLNC: dynamic structure, highly reconfigurable
    - Flexible **coefficient location** (Clustered, Indexed)
    - Dynamic **number of coefficients / symbols**
    - Flexible **symbol size** (Fragmentation, Padding, Encapsulation)
    - Flexible **field** (Coding complexity, Device capabilities)

Standardizing Symbol Representation

Important for Network Operations

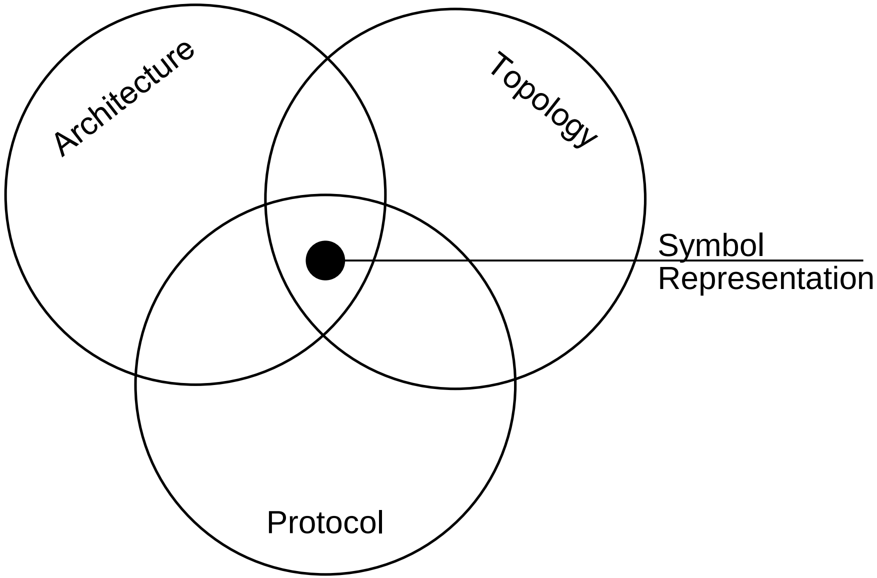
Network operations may be affected by symbol representation

- Example: Fragmentation
- Known coefficients
  - Can recode fragments
- Unknown coefficients
  (e.g., pre-coded or hidden)
  - Must use new coding layer



Standardizing Symbol Representation

Important Standardization Target



- Architecture:
  Layered architecture, Coding architecture
  (e.g., Encapsulation, Routing)
- Topology:
  Logical (coding) topology
  (e.g., no recoding if coefficients are not explicit)
- Protocol
  (e.g., Generation vs sliding window)

Next Version: Overview of Suggested Changes

draft-heide-nwcrg-rlnc-background-00

- More definitions
- Correcting networking terminology
- Trade-offs related to coding parameters
- Security section

draft-heide-nwcrg-rlnc-01

- Clarify definitions

#ThanksDave
#ThanksSalvatore

Next Version: Definitions

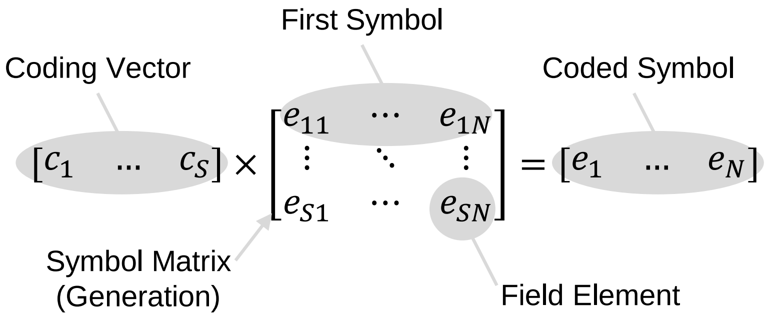
Clarifying Assumed Definitions

- Correcting networking terminology
  - “Connection”
- Clarify a number of terms:
  - “Field elements”: communication/information theoretic “symbols”
  - “Symbol”: array of field elements, “coding data unit”
  - “Raw data”: application data, “uncoded” / “systematic” / “raw” symbols
  - “Representation”: what goes on the wire
  - “Coding Layer”: new vs. current coding layer
  - “Coding Vector” (see next slide)
  - “Hidden” Coefficients (see Security notes below)
- Link / refer to taxonomy draft

Next Version: Coding Vector

Spelling Out Assumed Definitions

- "Raw" Vector
  - Mathematical/full vector of coefficients
  - "Yields coded symbol when multiplied with symbol matrix"
- Different from representation
  (i.e., "what is sent on the wire")
- Representation requires
  - Coefficient values
  - Symbol mapping
- Examples of representations
  - Raw vector (useful in dense coding)
  - Coefficient values + symbol indices (sparser codes)
  - Seed



Next Version: Protocol Trade-offs

Emphasize fundamental trade-offs

- Multiple trade-offs related to coding parameters
- Fundamental trade-offs
  - Field size: coding complexity, code diversity (linear dependence), required redundancy
  - Symbol size
  - Generation / window / block size: latency, throughput, redundancy granularity
- Application-related trade-offs (optional)
  - Block code vs. sliding window
  - Systematic vs. full coding
  - Sparse vs. dense coding
  - Feedback vs. no feedback

Next Version: Security

Updating Security Section

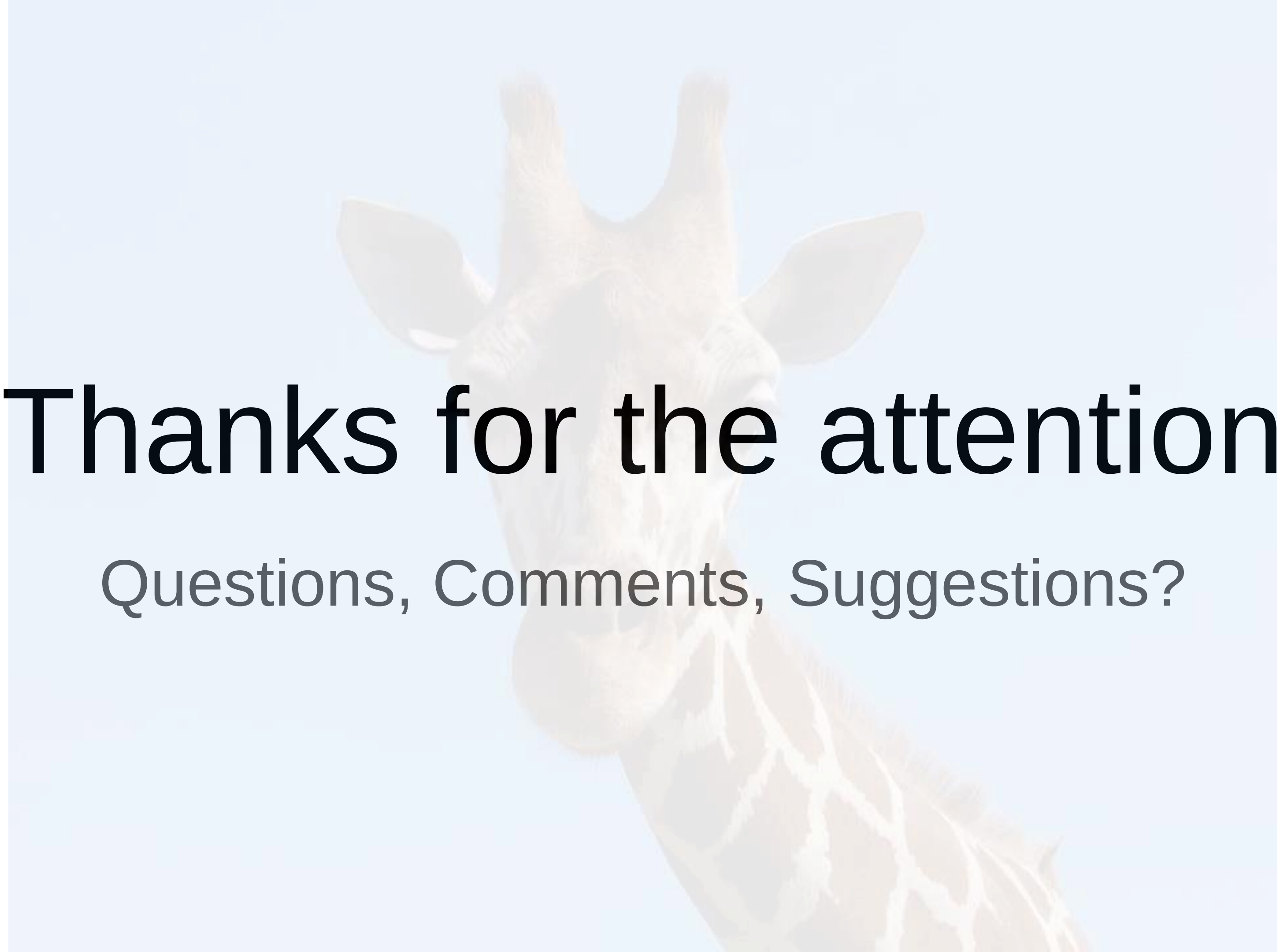
```
3.   Security Considerations

   This document does not present new security considerations.
```

- Initial assumption: operating inside the “coding layer”
  - Focus on coding operations, erasure correction, performance enhancement
  - Security provided by other layers
- Network coding operates by allowing mixing of data
- What are the security consequences of such mixtures?
- Three aspects:
  - Data hiding
  - Byzantine or pollution attacks – detection and correction
  - Verification

Next Version: Other Suggestions

- Looking into incoming suggestions
  - Adding references
  - “Encoder Rank”



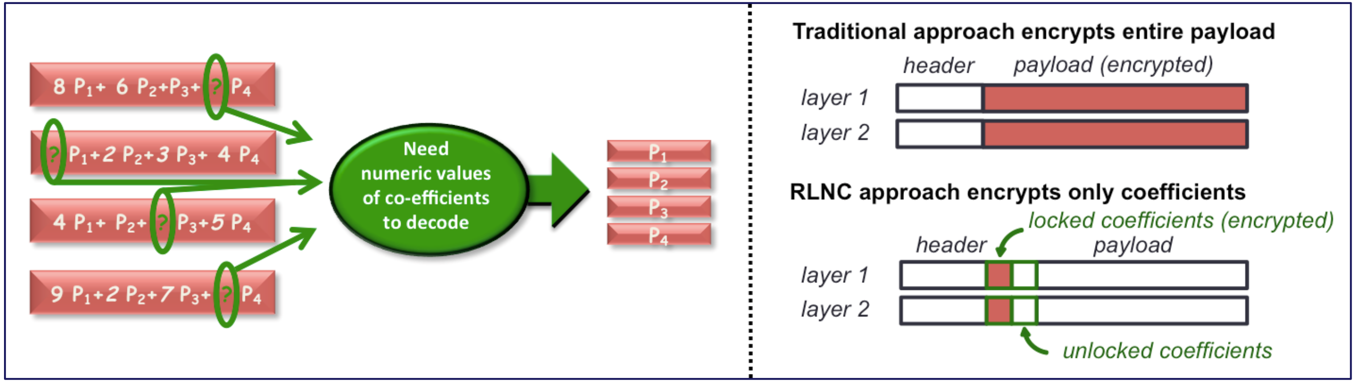
Thanks for the attention

Questions, Comments, Suggestions?

Content inaccessible without coefficients

⇩

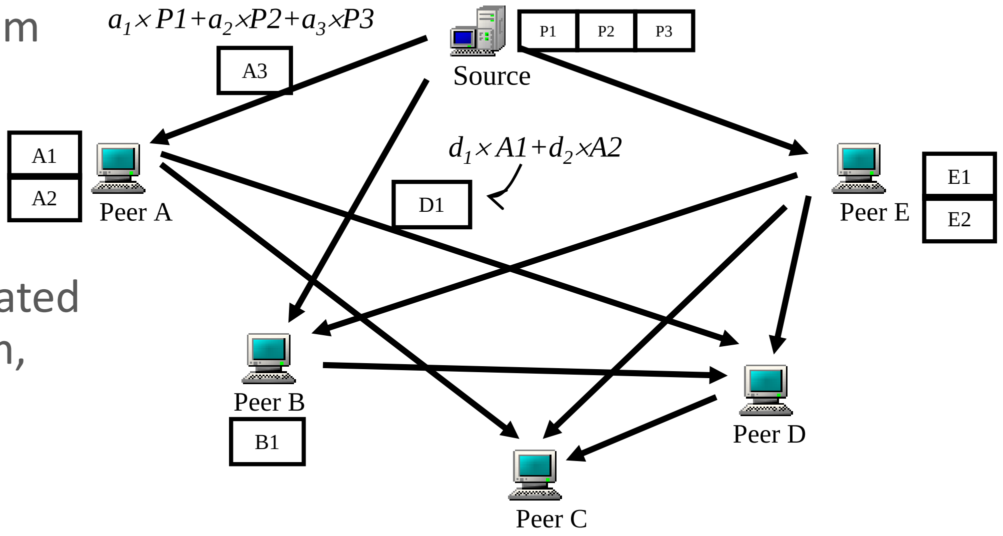
Encrypt coefficients instead of payload



- Enforce selective access to broadcast data
- *e.g.*, protect multi-resolution video layers

Content distribution of large files

- Use network coding to increase the efficiency of content distribution in a P2P cooperative architecture.
  - Instead of storing pieces on servers, store random linear combination of the pieces on servers.
  - Clients also generate random linear combination of the pieces they have received to send out.
  - When a client has accumulated enough degrees of freedom, decode to obtain the whole file.



Detecting and Eliminating Pollution Attacks

Problem

- A malicious user can send packets with valid linear combination in the header, but garbage in the payload.
- The pollution of packets spreads quickly.

Solution

- Use homomorphic signature scheme
  - Compute file signature at source
  - Include in packets (use public key)
  - Verify that packet is valid linear combination (polynomial hash function)
- Intermediate nodes drop contaminated packets

Features

- No need to decode
- No need to contact source
- No need to retransmit contaminated data
- Low overhead
- Finding packet satisfying hash function is hard (= discrete logarithm)
- Packet- vs. block-level detection of pollution attacks