Tetrys, a Patent-Free Network Coding Protocol Update

Jonathan Detchart, Jérôme Lacan, Emmanuel Lochin
ISAE
Vincent Roca
INRIA

IETF92
NWCRG
March 27th, 2015
Dallas

Note Well

- We, the authors, didn't try to patent any of the material included in this presentation
- We, the authors, are not reasonably aware of patents on the subject that may be applied for by our employers
- If you believe some aspects of this presentation and its corresponding draft may infringe IPR you are aware of, then please fill an IPR disclosure and let us know

http://irtf.org/ipr

Motivations and Goals

- At IETF 86 we introduced Tetrys:

  http://www.ietf.org/proceedings/86/slides/slides-86-nwcrg-1.pdf

- At IETF 91, we presented the first version of the Tetrys draft as a patent free protocol for network coding:

  draft-detchart-nwcrg-tetrys-00

- Recent updates are reported in this presentation (v01 of the draft)

- The goal of the work: a Tetrys *Protocol Instantiation* and a collection of architectural *Building Blocks*

  Some of these elements could become independent contributions

Generic Header Format

- Inspired from the LCT header (RFC 5651)
- Common header for all packets
- Per packet-type section
- Header extensions

  Compatible with existing ALC/NORM header extensions

Independence from the Coding Scheme

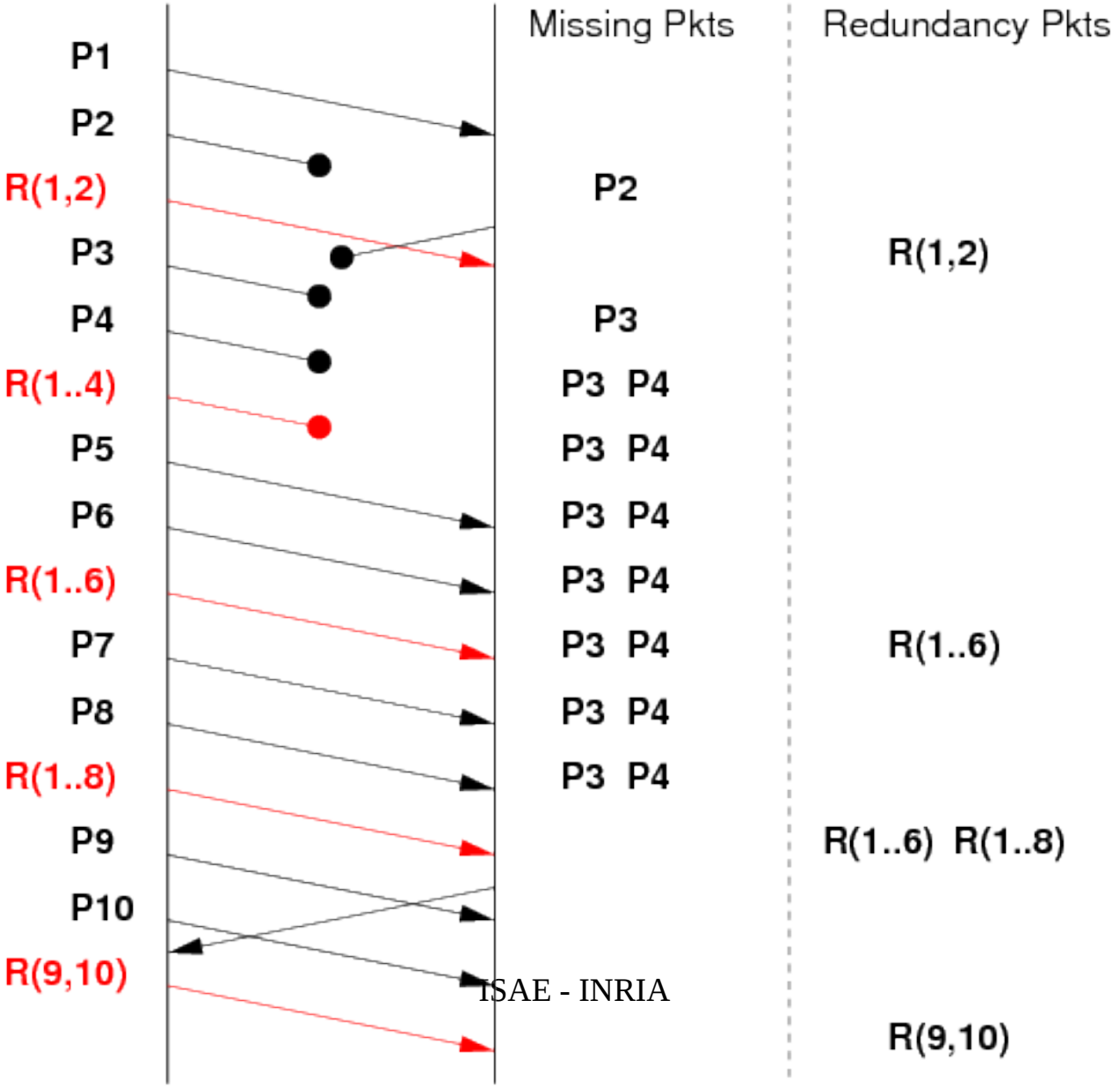
- The coding scheme is independent of the protocol and can change dynamically

  For example if the network conditions change

- Tetrys allows the use of both block codes and window-based codes

  The coding section could be moved to a different Building Block

Example: Elastic Encoding Window



Example: Elastic Encoding Window

- Each coded symbol is generated from a set of source symbols

  Depending on the coding scheme:

  - store the coefficients in the encoding vector
  - use the coding scheme's method to compute them

  Need to store the source symbol IDs

- Question: can we reduce the overhead of the symbol ID list?

An Efficient Symbol ID List Compression

- Using a delta compression (we keep the differential values)
- Using a bit vector field

- Depending on the situation, both are available

  To be moved to a different architectural Building Block

An Efficient Symbol ID List Compression (cont.)

- A symbol ID list is a set of (sorted) 32-bit words

  Rather than sending the full list (32-bit * NB_SYMB), we send the bounds.

  [1,2,3,4,7,8,9,10] => [1,4,7,10]

- But we can also compress this new list:

  Keep the first element of the list (1).

  Apply a differential transform to the others:

  [4,7,10] => [4-1, 7-4, 10-7] => [3,3,3]

  Compute the number of bits needed to store each element (here 2)

  Write the first symbol ID, and the NB_SYMB – 1 as 2-bit words.

Conclusion

- We proposed Tetrys, a *flexible* network coding *Protocol Instance*
- Different coding schemes (any block or window-based codes) are supported
- The code can change dynamically

- Many elements will be moved to different Building Blocks to align with the NC Architecture

Thank you !

Authors:
jonathan.detchart@isae.fr
jerome.lacan@isae.fr
emmanuel.lochin@isae.fr
vincent.roca@inria.fr

Draft:
http://tools.ietf.org/html/draft-detchart-nwcrg-tetrys-01