RObust Header Compression WG (ROHC)

57th IETF
Vienna, July 15, 2003

Chairs:

Carsten Bormann <cabo@tzi.org>
Lars-Erik Jonsson <lars-erik.jonsson@ericsson.com>

Mailing List:

rohc@ietf.org

57th IETF: Pre-Agenda

- **WG chair admonishments**
- **Real agenda**

✓Blue sheets
✓Scribe(s)
✓Jabber?

Hello! This is an IETF Working Group

- **We are here to make the Internet work (Fred Baker)**
  - Together! (Harald Alvestrand)
- **Rough Consensus and Running Code (Dave Clark)**
- **Working Group is controlled by**
  - IETF Process (RFC2026, RFC2418) – *read it!*
  - Area Directors (ADs): Alison Mankin, Scott Bradner
  - Charter (http://www.ietf.org/html.charters/rohc-charter.html)
  - Working Group Chairs: Lars-Erik Jonsson, Carsten Bormann
  - Technical Advisor: Erik Nordmark
- **Work is done on email list: rohc@ietf.org**
  - And on IETF meetings, interim meetings, informal meetings
  - Mailing list is official channel, though

Purpose of WG meetings

- We assume people have read the drafts
- Meetings serve to advance difficult issues by making good use of face-to-face communications
- No technical presentations, but lead-ins to stimulate/elicit discussion
- Objective: generate technical consensus

RFC 2026: Internet Standards Process

- **Standards track RFCs:**
  - **WG consensus (as judged by WG chairs)**
  - **WG last call**
  - **IETF last call**
  - **IESG approval (based on AD recommendation)**
    - Quality control!
  - **Publication by the RFC Editor**
- **Informational RFCs**
- **BCP (best current practice) RFCs**

RFC 2026: IPR issues (1)

- (10.2) No contribution that is subject to any requirement of confidentiality or any restriction on its dissemination may be considered […]
- Where the IESG knows of rights or claimed rights […] the IETF Executive Director shall attempt to obtain from the claimant […] a written assurance that upon approval by the IESG of the relevant Internet standards track specification(s), any party will be able to obtain the right to implement, use and distribute the technology […] based upon the specific specification(s) under openly specified, reasonable, non-discriminatory terms.

RFC 2026: IPR issues (2)

- **Contributions (10.3.1(6)):**
  "The contributor represents that he has disclosed the existence of any proprietary or intellectual property rights in the contribution that are reasonably and personally known to the contributor."
- **I.e., if you know of a patent application for a technology you are contributing, you have to tell.**
  Or just shut up entirely!

57th IETF: ROHC WG Agenda, 1(2)

14:15 - Chair admonishments and agenda	**Bormann (5)**
14:20 - Formal HC Notation	**(55)**
15:15 - Coffee Break	**(30)**
15:45 - Resume (AD present)	**(60)**

ROHC Notation

Carsten Bormann, 2003-07-15

The 3095 lesson

- Packet formats are non-trivial
  - We overstressed the RFC box notation
- It is not always clear what goes where
  - Labeling a field in box notation does not mean you know what it **means**
- Interops take much time for debugging the more complex formats

The ROHC Notation

- Originally invented for EPIC
  - Has spun off on its own
  - “EPIC” now refers to Hierarchical Huffman
- ROHC-FN Inspirations:
  - BNF
  - ROHC packet classifications
  - Huffman probabilities

ROHC Notation: example

- ip_header --> ip_version++
  ip_hdrlen++
  ip_tos++ …
- ip_version --> value(4, 4).
- ip_hdrlen --> value(4, 5).
- ip_tos --> static // irregular(6)++
  label(2, ecn).

Issues in formal notations

- Way too easy to come up with wishy-washy semantics
- Top-down design?
  - Small set of tools
  - Inflexible for new uses
- Bottom-up design?
  - Long way up to application requirements

The ROHC-FN approach

- Started from “EPIC-evolved” notation
- Defined FN in terms of well-understood CS concepts
  - Use a (high-level) programming language as the basis
  - Notation instance becomes executable program in this language

What does the spec specify?

- Relationship between:
  - Uncompressed packet
  - Compressed packet
  - Old and new compression contexts
- Mathematically, this is a **relation**
  - one (uncompressed packet, old ctx) ⊆
    ≥ 1 (compressed packet, new ctx)

Why not box notation (BoxN)?

- BoxN specifies compressed packet only
- Relationship to uncompressed packet and context must be captured in English
- BoxN is inadequate even for its limited purpose
  - It nicely nudges the designer to think about alignment, though…

Compilers and Interpreters

- The notation is executable (by an interpreter)
  - so we know unambiguously what it means
  - so we can test and debug our specifications
  - **Not** as a shortcut to an implementation
- How do you implement an FN spec?
  - You can write a compiler
  - You can program from the spec
- There is no concern about efficiency **of the interpreter**

A Non-Requirement

- ROHC-FN can theoretically be used for downloading specs into decompressors
  - Would need to fix “English” parts (ctx mgmt)
- Outside scope of this WG
- We are **not** taking this as input for requirements
  - Good CS design sense might use it as a tie-breaker

Another Non-Requirement

- ROHC-FN can be used for other things than compressing TCP headers
- Outside scope of this WG
- We are **not** taking this as input for requirements
  - Good CS design sense **will** consider generality
  - Testing ROHC-FN on RTP is a useful exercise
- Programming language foundation of FN provides avenue to extensibility later

What does the program do?

- Not clear. Best case:
  - Compress
  - Decompress
  - Compute some interesting properties of the notation instance (e.g., consistency)
  - Generate code for C implementation
  - Make coffee
  - …

FN -- Implementation Status

- 2.5 attempts at rooting FN in CS concepts
  - TZI / Prolog / unpublished (Atlanta IETF)
  - Roke / Haskell / sent to WG
  - Roke / Prolog / sent to WG
- Spec writers currently use Roke/Prolog version (= most maintained)

Which programming language?

- Prolog vs. Haskell

- Try to maintain both
  - no need to decide on programming language
- Use them to think of notation concepts **in a clear, unambiguous way**

What does the program do?

- Given uncompressed header, yields all compressed headers
- Given compressed header, yields uncompressed header
- Enumerate all combinations

- Print out box notation…
- Cannot (easily) **reason** about itself -- yet?

Elements of a notation

- ROHC FN = basic notation + library
- Basic notation:
  - Relations are defined by rules (like BNF productions), --> operator
  - Two implicit parameters: tuple, l/r environment
  - ++ operator (juxtaposition)
  - Binding operations (left, right, here)

The tuple for ROHC-FN

- Uncompressed packet
- Compressed packet
- Old Context
- New Context

- This is unlike DCGs…

Library

- Can unify with bit strings from packets
- Can relate to compression context
- Can manipulate bindings
  - Often need to hide left/right/here to avoid programming language variables
- Maths (= everything else)

Changes discussed: Bindings

- Replace “temporary variables” by bindings
  - left/right/here
  - replaces “map” and the special meaning of “nil”
- Issue:
  - require unique naming of bindings, or:
  - think about bindings in a stack

Changes discussed: Context

- Replace implicit references to context
  - Maintain some implicitness for legibility
  - Model this as explicit context element labels

- Issue:
  - Naming, again
  - (does not have to be the same as with bindings)

Resolution

- We haven't taken the final decision to base TCP work on FN
  - Pro: it seems to work
  - Con: error 33
- Let's decide that now
  - (and validate on the mailing list)

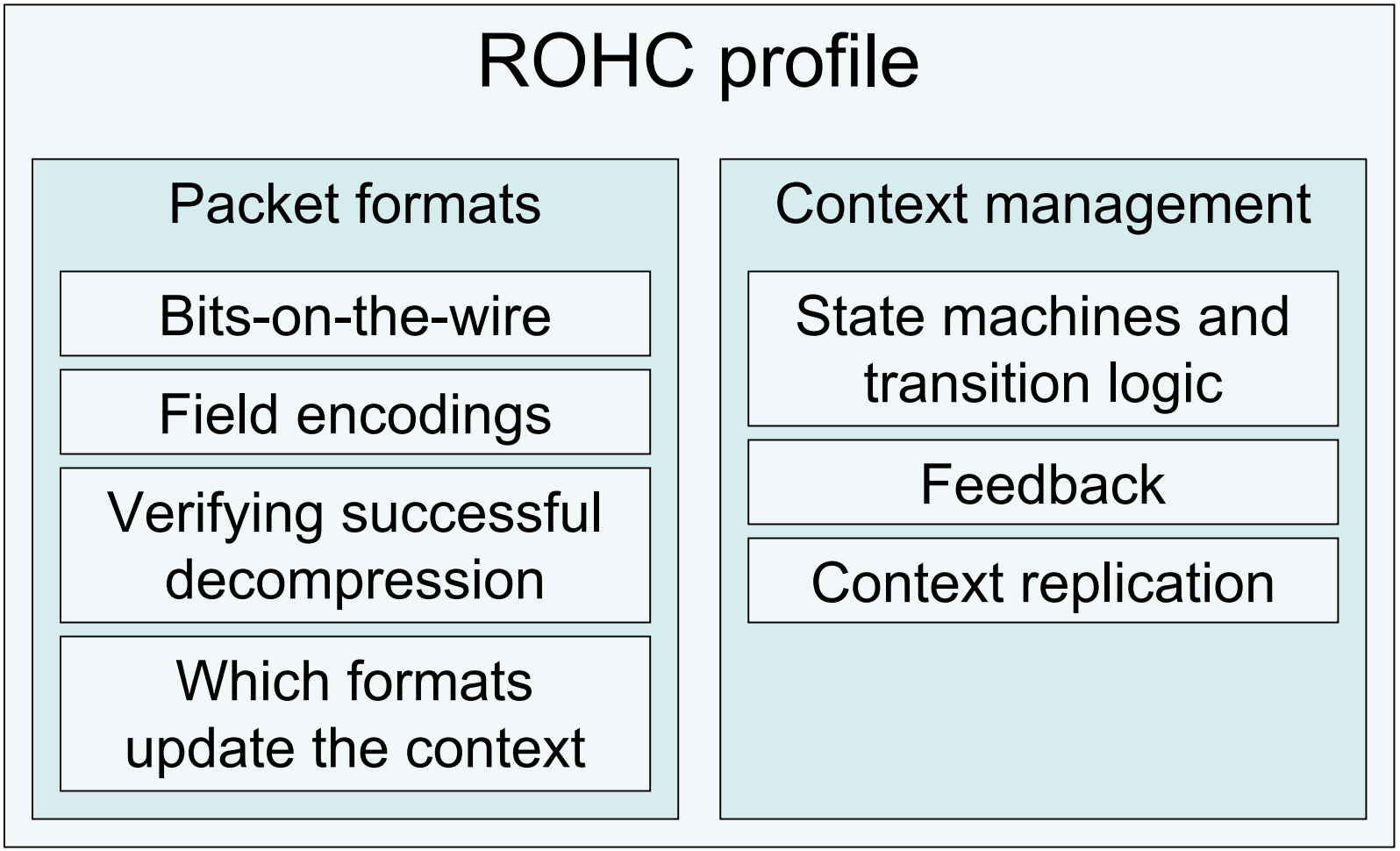
Roke Manor Research

Formal Notation Issues

Richard Price

richard.price@roke.co.uk

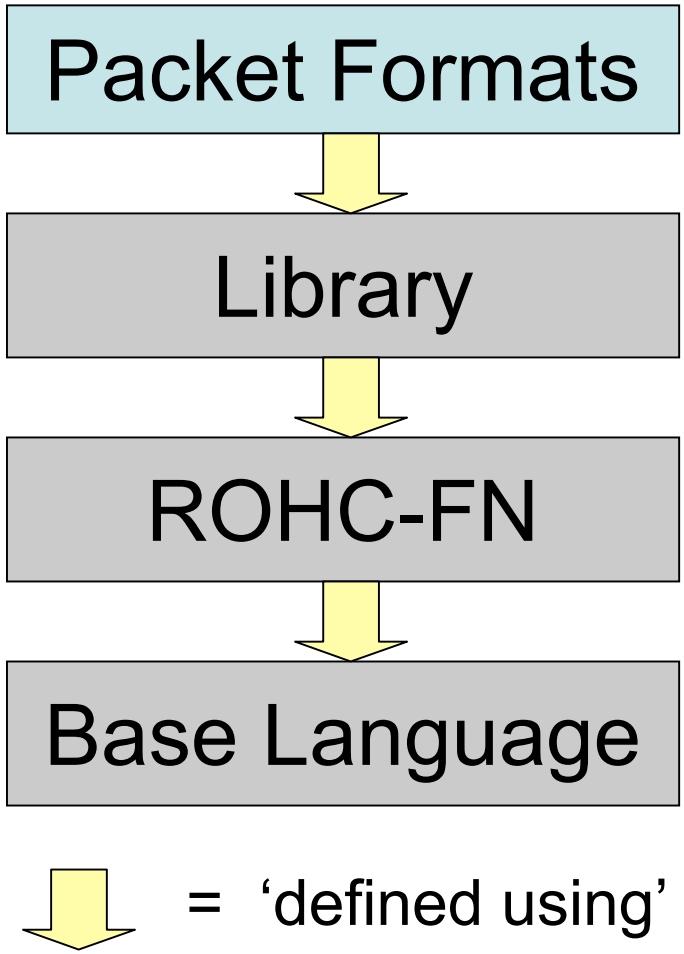
Structure of a ROHC Profile



What does ROHC-FN Provide?

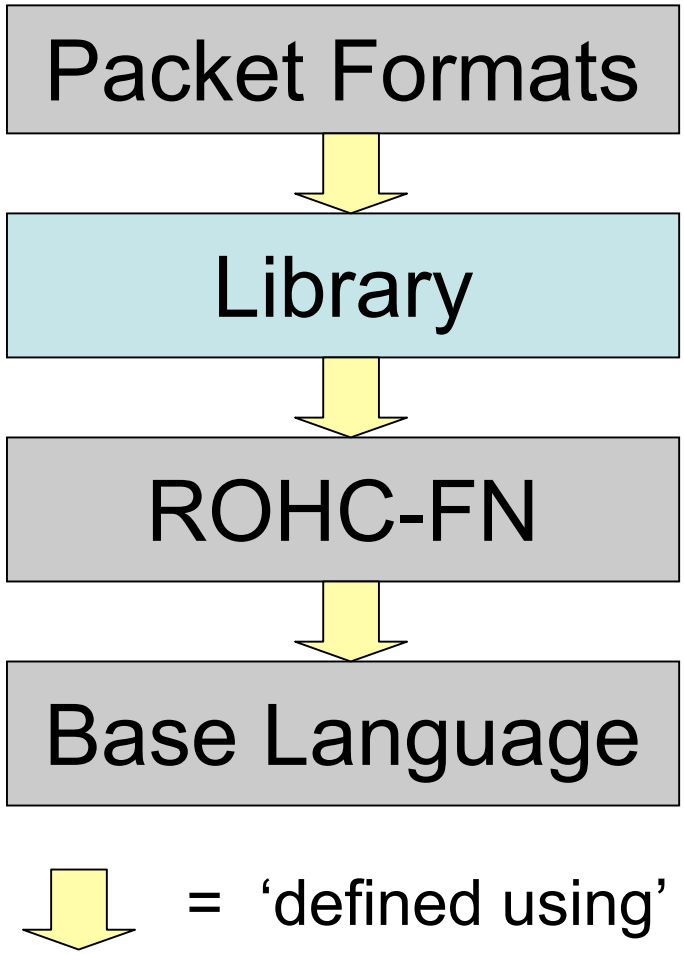
- ROHC-FN defines packet formats for a profile
  - Alternative to traditional ‘box notation’
- ROHC-FN does *not* define context management
  - State machines and feedback in ROHC-TCP draft
    - Context replication in separate ROHC-CR draft
  - Interface to ROHC-FN packet formats via the context
    - State machines run at compressor and decompressor
    - Packet formats invoked to (de)compress a header
    - Packet format selection based on context (including state)
  - Open issue: where to put the sliding window?

Layered Approach



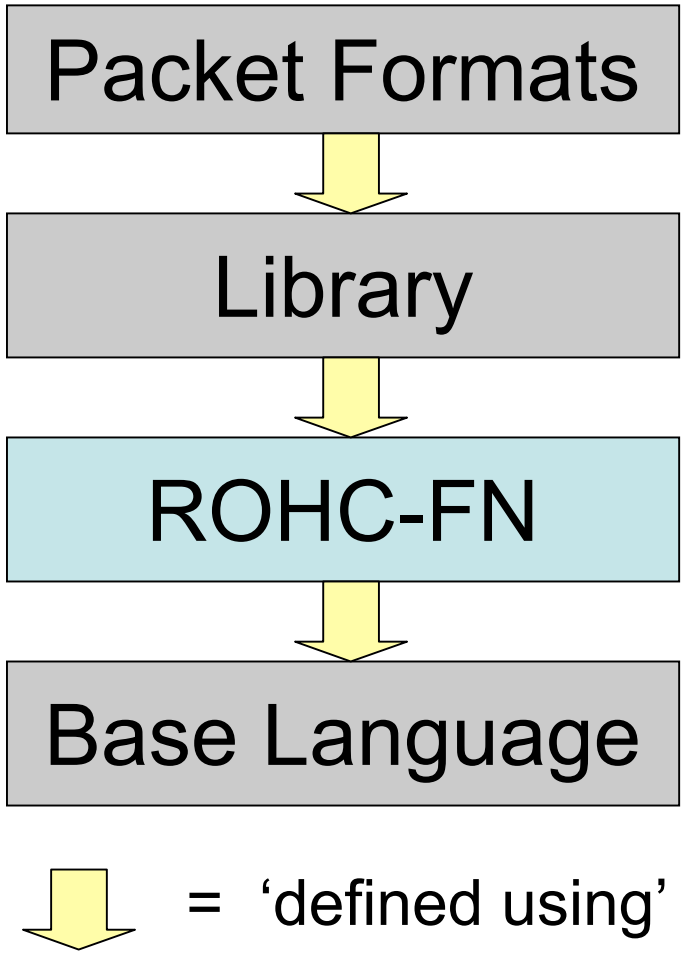
- ROHC-TCP packet formats
  - Compressed bits-on-the-wire
  - Encoding/decoding of fields
  - When to update the context
- Could define by hand
  - Leads to complex spec.
  - Hard to get interoperability
- Adopt layered approach
  - Minimises ambiguity
  - Easy to add new profiles

Layered Approach



- Library of encoding methods
  - Provides toolset for defining new packet formats
  - Not profile-specific
    - Motivated by ROHC-TCP …
    - … but reusable by other profiles
- Open issues
  - Which methods to include
  - How much of 3095 to reuse
    - Basic methods (Section 4.5)?
    - List encoding (Section 5.8)?

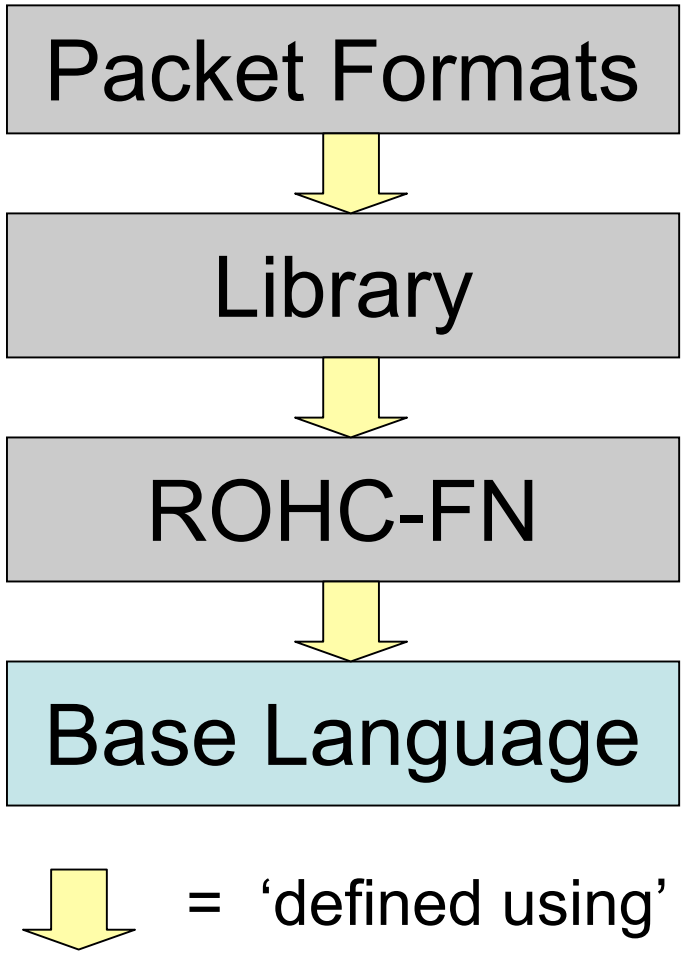
Layered Approach



= ‘defined using’

- Formal notation
  - Language for defining new encoding methods
- Three main components
  - Syntax
  - Data structures
  - Base encoding methods
- Open issues
  - How much of 3095 to reuse
    - Data structures (Section 6.5)?

Layered Approach



= ‘defined using’

- How to define ROHC-FN?
  - Currently just use English
  - Only 8 pages (so far!)
- Code also available
  - Prolog
  - Haskell
- Open issues
  - Is the draft well-defined?
  - Should we include any code?
    - Normative or informative?

Writing ROHC Profiles

- ROHC-FN *does not* generate packet formats
  - This is the job of profile writers!
  - Notation just captures the results
- Encoding method library is another matter …
  - In theory – new methods can be defined as needed
  - In practice – generally stick with library methods only
  - Assertion – this is a good thing!
    - Promotes code reuse between profiles
    - Resulting packet formats are simpler and more structured

Implementation Choices

- Compiler-based approach (minimum effort)
  - Implement the ROHC-FN layer only
    - Parser for ROHC-FN syntax
    - Data structures
    - Base encoding methods
  - Everything else is free …
- Hand-crafted approach (best performance)
  - Implement the packet formats by hand
    - Handwritten code for every method in the library
  - Similar to implementing ‘box notation’ formats

Issues for Discussion

Roke Manor Research

- Data structures rather implementation-specific
  - Field values referenced using pointers
  - Context values referenced using integer indices
- Possible solution: name every field explicitly
  - Reuse naming conventions from RFC 3095
    - hdr(X) – value of field X in uncompressed header
    - context(X) – value of field X in context
  - Often useful to generate field names automatically
    - CSRC list – generate names CSRC(0) … CSRC(15)

Outstanding issues

Computing the final encoding

- What does “or” (“/”) do exactly?
  - Completely specified in Notation vs. Additional process (such as EPIC) referenced
  - Body/Discriminator terminology useful?
  - Nesting/cross-products?
- Current approach: use a library method for a specific final encoding

Context management

- FN spec defines how **receiver** updates context
- Maintain “context families” in the notation?
  - Truncation of the family is the fun part
- (Context labeling change -- details TBD)

Alignment

- Can do byte alignment “by hand”
  - Not supported by notation ➡ tedious

- Related issue: using “unused” space in the last byte of a header
- Probably related to final encoding issue

IPR

- Can we keep the notation itself unencumbered?

- A “language” should not have IPR issues (no influence on technical systems unless you **say** something in that language)

ROHC Formal Notation Requirements

Julije Ozegovic

University of Split

Split, Croatia

E-mail: julije.ozegovic@fesb.hr

Scope of ROHC-FN

- ROHC-FN: to ease completion of tasks and goals
  - Profile writing
  - Profile refinement
  - Human readability
  - Compactness
  - Machine readability
  - Profile validation
  - Profile clarity
  - Hand code generation
  - Machine code generation
  - Profile interpretation

Scope of ROHC-FN - cont.

- Human requirements:
  - Benefit from: human readability, clarity and compactness
  - To complete: profile writing and refinement, and code generation
- Machine requirements:
  - Benefit from: machine readability
  - To complete: validation, code generation, and interpretation
- Profile interpretation: a future issue
  - Profile download: execute ROHC-FN profiles directly
  - Compr/Decompr SW: FN profile interpreter
  - Phases: offline (preprocessing) and online

ROHC-FN requirements

- ROHC-FN to be suitable for specification of:
  - Packet classification parameters
  - Uncompressed packet structure
  - header format
  - compressed header structure
  - context management
  - notation extensions

ROHC-FN requirements - cont.

- Packet classification parameters: a future task
  - Recognize profile: use information from data link

```
MEDIA = Ethernet
   Protocol_ID = <link layer protocol ID for actual protocol>

MEDIA = PPP
   Protocol_ID = <link layer protocol ID for actual protocol>
```

  - Recognize context: use information from header fields

```
Context_check  = <sadr dadr sport dport>
```

  - Granularity control: fine or coarse, flow mixing, context reuse

ROHC-FN requirements - cont.

- Profile specification:
  - Uncompressed header structure: declare independently
  - Header format: how to compress particular fields
  - Compressed header structure: indicator bits positioning
- Profile specification degrees of freedom:
  - No freedom: compress fields as ordered, not optimal
  - Half on input: compress as ordered, but postpone - OPTIMAL
  - Full on input, read when needed: random access, complex
  - Full on input with parsing: too complex
  - Full on output: compressed fields reordering not necessary

ROHC-FN requirements - cont.

- Field postponement:
  - Possible source of unreadability and redundancy:

```
header = function1
         function2
    -------------
function1 = method11(length) | method12(length)
```

- Field postponement with separate header structure specification:
  - Improved readability (inline) and reduced redundancy

```
some_field1 = FIELD(length1)
some_field2 = FIELD(length2)
    -------------
header = some_field1 method11 | method12
         POSTPONE some_field2
    -------------
         method(some_field2)
    -------------
         ACTIVATE some_field2 method21 | method22
```

ROHC-FN requirements - cont.

- Compressed header structure:
  - Order of compressed fields: natural to be in order of compression
  - Order of indicator bits: different models
  - Profile model: monolithic or chained or generic

- Monolithic profile:
  - Single profile per stack: needs profile for every combination

```
TCP_IP_PACKET = ROHC3095(TCP_IP_function)
```

- Chained profile:
  - Single profile per protocol: needs one profile for every protocol

```
TCP_IP_PACKET = ROHC3095(IP_function)
                Ordinary_Huffman(TCP_function)
```

ROHC-FN requirements - cont.

- Generic profile:
  - Profile selection: id bits problem
  - Context memory: optimized

```
PACKET =
   Ordinary_Huffman(Ipv4_function | Ipv6_function)
   Ordinary_Huffman(TCP_function | UDP_function | SCTP_function)
   Ordinary_Huffman(RTP_function | Null)
```

- Profile preprocessing:
  - ID bits encoding: lot of computation to be done only once
  - Small terminal problem: download preprocessed ID encoding

ROHC-FN requirements - cont.

- Context manipulation: complex behavior is possible
  - New context: first copy values from previous one
  - Context update: most methods normally update context
  - Context update avoided: some methods do not update context
  - Context update skip: sometimes methods are prevented to update context by profile writer
  - Context update not executed: when field is not existent
  - Existing field reuse: different methods update the same context field (e.g. DCCP ACK, TCP SACK)

Notation extensions

- Levels of extension: needed to cope with future issues
  - Profile level: the profile writing itself
  - Method level: methods to build new methods, part of toolbox
  - Programming language level: build new basic method
- Decompressability: not guaranteed by notation itself
  - Extension validation: pass full standardization process
  - Profile validation: pass full standardization process

References

[1] Bormann, C., Burmeister, C., Degermark, M., Fukushima, H., Hannu, H., Jonsson, L-E., Hakenberg, R., Koren, T., Le, K., Liu, Z., Martensson, A., Miyazaki, A., Svanbro, K., Wiebke, T., Yoshimura, T. and Zheng, H., "RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed", RFC 3095, July 2001.

[2] Price, R., Hancock, R., Ollis, P., Surtees, A. and West, M., "Framework for EPIC-lite", draft-ietf-rohc-epic-lite-01.txt (work in progress), February 2002.

[3] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

[4] R. Price, A. Surtees, M. West "A Formal Notation for Header Compression" draft-west-rohc-formal-notation-00.txt (work in progress), June 2002.

References cont.

[5] Hongbin Liao, Qian Zhang, Wenwu Zhu: "Generic Header Compression Notation for ROHC" <draft-liao-rohc-notation-00.txt> (work in progress), June 2002.

[6] Richard Price, Abigail Surtees, Mark A West: "Protocol-Enabled BNF-Based LanguagE (PEBBLE)" daft-ietf-rohc-formal-notation-00.txt>, October 2002.

[7] Richard Price, Abigail Surtees, Mark A West: "Formal Notation for Robust Header Compression (ROHC-FN)" <draft-ietf-rohc-formal-notation-01.txt>, March 2003.

[8] Carsten Borman et. al.: "The slides from IETF-56 ROHC meeting" http://www.dmn.tzi.org/ietf/rohc/rohc-56.pdf, March 2003.

RTP profile

in ROHC formal notation

Julije Ozegovic

University of Split

Split, Croatia

E-mail: julije.ozegovic@fesb.hr

RTP profile in ROHC-FN

- RTP/UDP/IP: in focus for header compression
  - Previous work: ROHC RFC3095, RTP for EPIC
  - Current proposal: In ROHC-FN adopted for Prolog program
  - Prolog program: operational on RTP, posted to ROHC list
- RTP/UDP/IP profile for Prolog: minor drawbacks
  - UDP-length: not inferred, should be moved to the beginning of IP
  - NBO: not yet implemented in Prolog program
  - CRC: not yet implemented in Prolog program
  - NBO: not yet implemented in Prolog program
  - Scale: uses fixed scale factor defined explicitly
  - List: shortened to 8 entries due to Prolog constrains
- Conclusion: ROHC-FN is applicable to RTP/UDP/IP

References

[1] Bormann, C., Burmeister, C., Degermark, M., Fukushima, H., Hannu, H., Jonsson, L-E., Hakenberg, R., Koren, T., Le, K., Liu, Z., Martensson, A., Miyazaki, A., Svanbro, K., Wiebke, T., Yoshimura, T. and H. Zheng, "RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed", RFC 3095, July 2001.

[2] Price, R., Surtees, A. and M. West, "Formal Notation for Robust Header Compression ROHC-FN", draft-ietf-rohc-formal-notation-01.txt (work in progress), March 2003.

[3] Schulzrinne, H., Casner, S., Frederick, R. and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 1889, January 1996.

[4] A. Surtees, M. West, "RTP Profile for EPIC", draft-surtees-rtp-epic-00.txt, February 2002.

[5] Price, R., Hancock, R., Ollis, P., Surtees, A. and M. West, "Framework for EPIC-lite", draft-ietf-rohc-epic-lite-01.txt, February 2002.

57th IETF: ROHC WG Agenda, 2(2)

15:45 - WG status update	**Bormann (5)**
15:50 - MIB Doctor review issues	**Quittek (10+)**
16:00 - ROHC RTP	
16:00 - "IP-only" profile	**(5)**
16:05 - UDP-lite profiles	**(5)**
16:10 - SigComp doc status update	**Bormann (10)**
16:20 - TCP Issues	
16:20 - Context Replication	**Pelletier (5)**
16:25 - TCP Issues	**West (20)**

Document status update, 1(3)

- **Old**
  - **RFC 3095: Framework and four profiles** (was: draft-ietf-rohc-rtp-09.txt)
  - **RFC 3096: RTP requirements** (was: draft-ietf-rohc-rtp-requirements-05.txt)
  - **RFC 3241: ROHC over PPP** (was: draft-ietf-rohc-over-ppp-04.txt)
  - **RFC 3242: LLA RTP** (was: draft-ietf-rohc-rtp-lla-03.txt)
  - **RFC 3243: 0-byte RTP req's** (was: draft-ietf-rohc-rtp-0-byte-requirements-02.txt)

  - **RFC 3320: SigComp** (was: draft-ietf-rohc-sigcomp-07.txt)
  - **RFC 3321: SigComp extended** (was: draft-ietf-rohc-sigcomp-extended-04.txt)
  - **RFC 3322: SigComp Req.** (was: draft-ietf-rohc-signaling-req-assump-06.txt)
  - **RFC 3408: LLA R-mode** (was: draft-ietf-rohc-rtp-lla-r-mode-03.txt)
  - **RFC 3409: ROHC RTP LLG** (was: draft-ietf-rohc-rtp-lower-guidelines-03.txt)

Document status update, 2(3)

- Old related
  - RFC3485: SigComp SIP/SDP static dictionary
  - RFC3486: Compressing SIP with SigComp
- In RFC editor queue
  - NONE!
- Submitted to IESG -- see next segment
  - draft-ietf-rohc-mib-rtp-06.txt (PS)
  - draft-ietf-rohc-terminology-and-examples-02.txt (Info.)

Document status update, 3(3)

- **Current WG documents**
  - **RTP/Framework – 4 drafts**
  - **TCP profile – 5 drafts**
  - **SCTP profile – 1 draft (requirements)** **Work on hold**
  - **Notation – 1 draft**
  - **SigComp – 3 drafts**

Review of MIB Doctor issues

- 1550–1600
- Jürgen Quittek

ROHC-MIB-RTP

<draft-ietf-rohc-mib-rtp-06.txt>

Juergen Quittek <quittek@ccrle.nec.de>
Hannes Hartenstein <hartenst@ccrle.nec.de>
Martin Stiemerling <stiemerling@ccrle.nec.de>
NEC Europe Ltd.

Recent History

- Version -05 in Dec 02
- MIB review by SNMP community requested in Dec 02
- WG last call in Jan 03
- Version -06 in Mar 03
  - implementing changes during last call
- Doc submitted to AD review
- Extensive comments from MIB review received in Jun 03
  - very good, very detailed comments

Issues Raised by the Reviewer

- Vendor information of instances and profiles
  - Do we need independent vendor information for ROHC instances and ROHC profiles?
- Indexing scheme
  - What would a management system typically be looking for when retrieving information from the MIB?
- Patent issues
  - We have to determine for sure, if or if not a known patent is required in order to implement the MIB module.

Patents to be considered

- [ERICSSON-ROCCO] * ... a proposal "RObust Checksum-based header COmpression (ROCCO)" dated June 22, 1999 has been made by Lars-Erik Jonsson at Ericsson, Mikael Degermark at Lulea University
- [ERICSSON-SIGCOMP.txt] * ... that might possibly have technical relations to the Internet Draft draft-ietf-rohc-sigcomp-extended-**.txt.
- [NOKIA-RFC3095.txt] * "Variable length context identifier for real time header compression": FI "Variable length encoding of compressed data“
- [ERICSSON-RFC3095-ROBUST-HEADER-COMPRESSION.txt] * Framework and four profiles: RTP, UDP, ESP, and uncompressed". indexing scheme

Is anyone planning to implement the MIB?

ROHC IP Profile

- draft-ietf-rohc-ip-only-02.txt
- Compress packets that can't be compressed by 3095
  - 3095 compresses UDP, RTP, ESP with certain outer headers
- Similar to IP/UDP profile, but sans UDP
  - Chain terminates for unsupported header
- More than two IP headers
  - Carry around rest in dynamic chains
  - Less efficient than 3095, but still better than not compressing
- Constant IP-ID

Last-Call now? CDRs!

UDP-lite Profiles

- draft-ietf-rohc-udp-lite-00.txt
- Like 3095, but allows for UDP-Lite usage
  - draft-ietf-tsvwg-udp-lite-01.txt
- UDP-Lite itself is waiting to emerge out of IESG
  - https://www1.ietf.org/IESG/EVALUATIONS/draft-ietf-tsvwg-udp-lite.bal (Revised ID needed)
- Our profile should be ready then
  - Was due in Apr-03

Last-Call now? CDRs!

SigComp NACK

- draft-roach-sigcomp-nack-01.txt
- Some outstanding issues (from Abbie)
- –02 needed

CDRs!

SigComp User Guide

- draft-ietf-rohc-sigcomp-user-guide-00.txt
- Check that all examples have been run recently
  - Obtain reports during last-call comments
  - Is shared-mode a problem here?

Last-call now? CDRs!

SigComp Implementers' Guide

- draft-ietf-rohc-sigcomp-impl-guide-01.txt
- draft-price-rohc-sigcomp-torture-tests-02.txt

- Will stay live I-Ds while Interop work continues

CDRs!

SigComp SIP mapping

- draft-ietf-rohc-sigcomp-sip-00.txt
- Reflects San Francisco discussion
  - Want to last-call real soon
- **Needs Review**

CDRs!

TCP document roadmap

- draft-ietf-rohc-tcp-requirements-06.txt
- draft-ietf-rohc-tcp-field-behavior-02.txt

- draft-ietf-rohc-context-replication-00.txt
- draft-ietf-rohc-tcp-04.txt
- draft-price-rohc-tcp-packet-formats-00.txt

- draft-ietf-rohc-tcp-enhancement-01.txt

Context Replication

- 1620–1625
- Ghyslain Pelletier

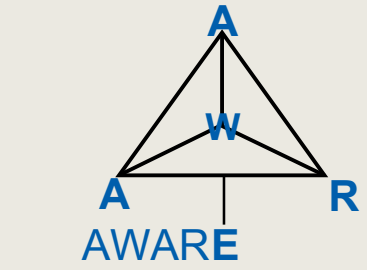
ERICSSON

57th IETF – ROHC WG

Context replication for ROHC profiles

(draft-ietf-rohc-context-replication-00.txt)

Ghyslain.Pelletier@epl.ericsson.se
+46 920 20 24 32



ERICSSON

Context Replication for ROHC Profiles

- ***compression from the very first packet*** of a new flow, using information from an another (previously acknowledged) context.
- ***motivated by the TCP profile*** (short-lived TCP flows).
- written as a ***generic ROHC mechanism***
- ***defined in its own draft***:

  draft-ietf-rohc-context-replication-00.txt.

  - compressor logic (selection of base context, action on feedback)
  - decompressor logic (actions upon failure, feedback)
  - general format and requirements for the IR-Replicate

ERICSSON

Recent discussions on the mailing list

- **ROHC-CR as a generic mechanism** (consensus)
  - reusable for other profiles
  - defined using assumptions coherent to the TCP profile
  - profile specific part defined in supporting profile itself
- **ROHC-CR and modes of operation** (consensus)
  - no mode specific efforts
  - a U/O-mode type of operation based on the TCP profile is assumed.
- **Others agreements**
  - 8-bits CRC coverage of IR-Replicate is same for all profiles
  - actions upon CRC-8 failure vs section 5.2.6 of rfc-3095 to be clarified

ERICSSON

Open issues

- **Should a new compressor state be introduced?**

  Three alternatives to document the logic

  – Additional logic on top of IR state for using the IR-Replicate (current approach)

  – Defined as a sub-state of the IR state

  – Introduce a new state for context replication

- **Modes of operation and context replication**

  – How do specific context replication as a generic mechanism with consideration to profiles with multiple modes?

ERICSSON

Next update (1)

ROHC-CR and modes of operation:

- Assumptions of the type of operation will be clarified (I.e. U/O-mode)
- Mode considerations will be clarified along the following lines:

"A profile supporting context replication and defining other modes than U/O-mode cannot replicate a context if a mode other than U/O is active at replication time, unless proper additional logic to address this is specified as part of this profile."

ERICSSON

Next update (2)

8-bits CRC within IR-Replicate:

- Coverage not profile-dependant - entire packet, excluding payload
- Actions upon CRC-8 failure vs section 5.2.6 of 3095 to be clarified.

Logic for context replication and compressor states:

- ? New state for replication ?
- ? Complete example of the resulting state machine in appendix ?

Next step:

final review and send it to WG last call at the same time as ROHC-TCP.

TCP issues

- 1625–1645++
- Mark West

TCP Compression Issues

'From the trivial to the ridiculous…'

draft-price-rohc-tcp-packet-formats-00.txt

mark.a.west@roke.co.uk

A few easy pieces…

- IP version will be ‘4’ or ‘6’
  - Should we send it in full or as a single bit?
- Should we account for NBO when calculating IP-ID offset values?
  - Yes or No
- Should we treat ‘random’ IP-ID behaviour as a special case?
  - Yes or No
- Should we treat ‘always 0’ IP-ID behaviour as a special case?
  - Yes or No
- Should we try and compress the TCP ‘PSH’ flag?
  - Yes or No

A few easy pieces…

- IP version will be ‘4’ or ‘6’
  - Should we send it in full or as **a single bit**
- Should we account for NBO when calculating IP-ID offset values?
  - **Yes** or No
- Should we treat ‘random’ IP-ID behaviour as a special case?
  - **Yes** or No
- Should we treat ‘always 0’ IP-ID behaviour as a special case?
  - **Yes** or No
- Should we try and compress the TCP ‘PSH’ flag?
  - Yes or **No**

Some slightly harder pieces…

- Should it be possible to ‘replicate’ DSCP?
  - Not guaranteed to be shared
  - Would have to allow ‘STATIC’ or ‘IRREGULAR’
  - Yes or No
- Should it be possible to replicate TTL?
  - Yes or No
- Should it be possible to replicate IP-ID?
  - Yes or No

Some slightly harder pieces…

- Should it be possible to ‘replicate’ DSCP?
  - Not guaranteed to be shared
  - Would have to allow ‘STATIC’ or ‘IRREGULAR’
  - **Yes** or No
- Should it be possible to replicate TTL?
  - **Yes** or No
- Should it be possible to replicate IP-ID?
  - Yes or **No**
  - *Is this a well-formed question?*

Some slightly harder pieces…

- Should it be possible to ‘replicate’ DSCP?
  - Not guaranteed to be shared
  - Would have to allow ‘STATIC’ or ‘IRREGULAR’
  - **Yes** or No
- Should it be possible to replicate TTL?
  - **Yes** or No
- Should it be possible to replicate IP-ID?
  - Yes or **No**
- Should it be possible to replicate IP-ID *behaviour*?
  - **Yes** or No

‘Reserved’ bits

- Reserved flags (IP and TCP)…
  - Should we do anything about them?
  - Yes or No

‘Reserved’ bits

- Reserved flags (IP and TCP)…
  - Should we do anything about them?
  - **Yes** or no
- Reserved flags should either not be included or always included
  - We cannot predict future behaviour
  - What about granularity of including reserved bits?
    - They will probably be allocated in (pardon the pun) bits

Some odds and ends

- Does ‘DF’ need to be included in CO packet formats?
  - Yes or No
- TTL has a number of ‘common’ values
    - Should we use this for optimising IR packets?
  - Yes or No

- ECN will either be used throughout a flow or not at all
    - Should this be accommodated?
  - Yes or No
    - Should we try and compress the ECN bits?
  - Yes or No

Some odds and ends

- Does ‘DF’ need to be included in CO packet formats?
  - Yes or **No**
- TTL has a number of ‘common’ values
    - Should we use this for optimising IR packets?
  - Yes or **No**
  - *Only works if compressor is co-located with end-system*
- ECN will either be used throughout a flow or not at all
    - Should this be accommodated?
  - **Yes** or No
    - Should we try and compress the ECN bits?
  - Yes or **No**

IP Headers

- How many IPv4 headers should be handled?
  - How many should be compressed?
  - How many should be passed through transparently?
- Is it worth having a separate (simpler) set of packet formats for IPv6?
  - Yes or No
- Does the use of TCP change the way that we treat any extension headers?
  - Yes or No

IP Headers

- How many IPv4 headers should be handled?
  - How many should be compressed?
  - How many should be passed through transparently?

'enough!?'

- Is it worth having a separate (simpler) set of packet formats for IPv6?
  - **Yes** or No
- Does the use of TCP change the way that we treat any extension headers?
  - Yes or **No**

Some TCP behaviour

- TCP Window behaviour
  - Should we assume changes to the window?
    - Yes or No
- TCP Urgent Pointer
  - What happens if URG is ‘0’?
    - Assume that URP is unchanging? 0?
- TCP Urgent Flag
  - Is URG used frequently enough to warrant special handling?
    -

Some TCP behaviour

- TCP Window behaviour
  - Should we assume changes to the window?
    - **Yes** or No
    - *May well be a small set of commonly occurring values*
- TCP Urgent Pointer
  - What happens if URG is ‘0’?
    - Assume that URP is unchanging? 0?
- TCP Urgent Flag
  - Is URG used frequently enough to warrant special handling?
    - *Obviously needs to be handled…*
    - *Can we have a show of hands from* telnet *users..?*

'yes!?'

Sequence and Ack numbers

- TCP sequence and acknowledgement behaviour may be quite structured
- To what extent should we try and exploit this?
  - Packets may frequently have the same size
    - However, there will always be exceptions
    - Can we exploit these patterns
  - How robust (in terms of packet loss) does ROHC-TCP need to be?
    - This will affect how useful repeated packet sizes are…
- What size should be assumed for the TCP MSS?
  - Affects choice of LSB values…
- Are there any other issues..?

TCP Options

- Is it worth treating ‘SYN only’ options specially to optimise initialisation packets?
  - Yes or No
- Is it worth having flags to handle options that come and go (e.g. SACK)?
  - Yes or No
- Is it worth having compression of the SACK block?
  - Yes or No
- Is it worth compressing the TimeStamp option?
  - Yes or No

- Is it worth linking TS to ack flag?
  - Yes or No

TCP Options

- Is it worth treating ‘SYN only’ options specially to optimise initialisation packets?
  - **Yes** or No
- Is it worth having flags to handle options that come and go (e.g. SACK)?
  - **Yes** or No
- Is it worth having compression of the SACK block?
  - **Yes** or No
- Is it worth compressing the TimeStamp option?
  - **Yes** or No
  - *What clock resolutions are actually used..?*
- Is it worth linking TS to ack flag?
  - Yes or **No**

TCP Options (Part 2)

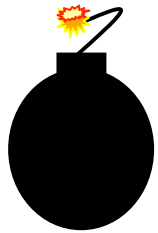
- Are there any ‘generic’ options that could be more aggressively compressed?
  - Planning to handle:
    - SACK
    - SACK Permitted
    - WScale
    - Timestamp
    - MSS
  - Any missing..?
  - Any useful treatments for ‘generic’ options?
    - e.g. remaining the same for successive headers

Other ROHC TCP Issues

- Master Sequence Number
  - Is this necessary in every packet?
    - Can only ‘ack’ packets with an MSN
    - What else is / might be dependent upon it?
    - What is the cost of *not* including it?
  - Can we assume that it will start at ‘0’
    - How much does this save?
    - Does this raise any concerns?
- Padding
  - What do we do with the ‘spare’ bits
    - More MSN; more protection; something else; reserved..?

Verification / Protection

- Use of CRC to protect TCP compression
  - Is 3 bits enough?
  - Should we protect packets that update large amounts of context more?

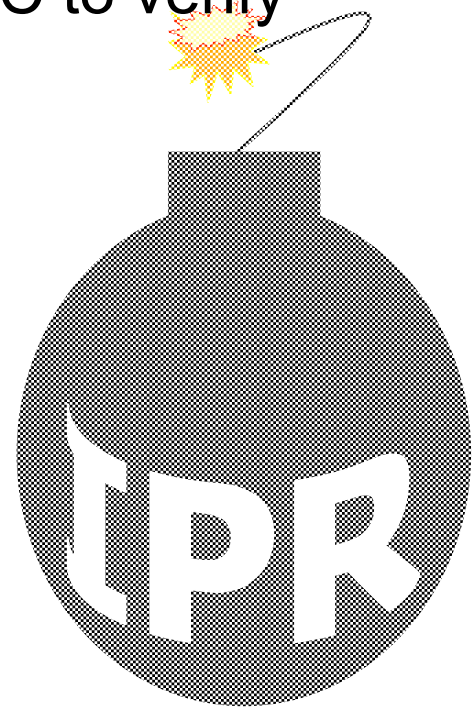


Verification / Protection

- Use of CRC to protect TCP compression
  - Is 3 bits enough?
  - Should we protect packets that update large amounts of context more?

Verification / Protection

- Use of CRC to protect TCP compression
  - Is 3 bits enough?
  - Should we protect packets that update large amounts of context more?
- What are the alternatives to using a CRC to verify decompression?
  - Can we ‘work around’ the IPR?
    - CRC over the *context*..?
  - Can we use the TCP checksum?
    - RFC1144/2507 highlights some weaknesses in this…
    - Will ROHC-TCP suffer similar problems?
- What about the use of LSB encoding?



And anything else that I've forgotten…