Features needed for seamless redundancy (P802.1CB)

Norman Finn
Rev 1

July 9, 2013

Summary

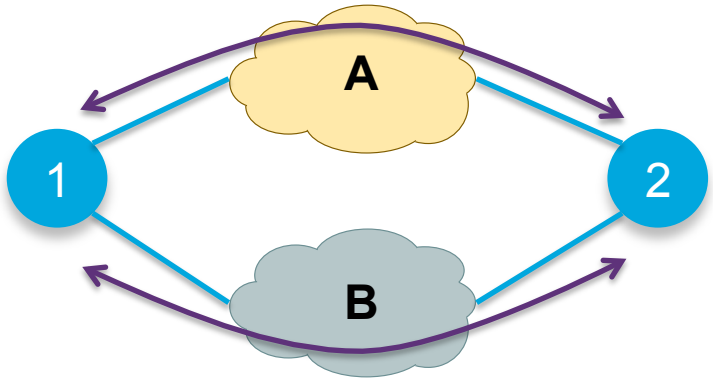
- In my opinion, we are writing a standard, in P802.1CB, that competes with ISO/IEC 62439-3. It is not clear who is stepping on who's toes, and arguments on this matter are unlikely to be fruitful. However, a liaison letter to ISO and/or IEC seems to be in order.
- There is a conflict between (my perception of) what the goals of P802.1CB should be, and interoperability with PRP, but these goals can be reconciled.
  - Specifically, the sequence number for P802.1CB should be per-source per-destination per-VLAN, rather than the per-source only sequence number of PRP.
- Interoperability between P802.1CB and ISO/IEC 62439-3 (edition 2) Parallel Redundancy Protocol will assist in the rapid adoption of IEEE Std 802.1CB, when complete.

ISO/IEC 62439-3 (Ed 2.0) PRP and HSR

ISO/IEC 62439-3 PRP and HSR

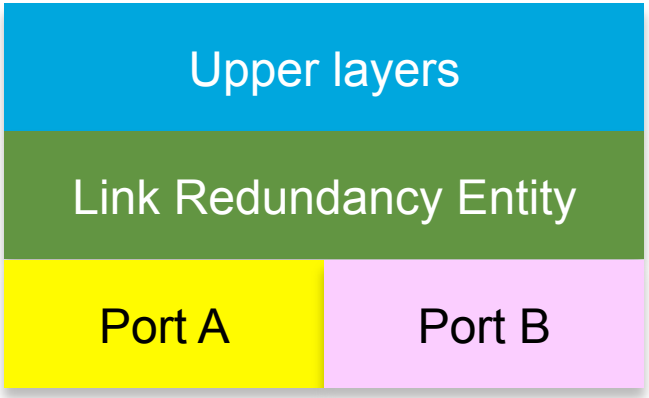
- IEEE 802.1CB does, in some sense, tread on the toes of ISO/IEC 62439-3, in that this standard defines two ways (HSR and PRP) to accomplish some of the same goals.
- On the other hand, one can argue that:
  - HSR (High-availability Seamless Redundancy) is more than “seamless redundancy,” since it defines forwarding operations around a ring.
  - PRP (Parallel Redundancy Protocol) is fundamentally flawed in that it defines a “tag” that is appended, rather than prepended, to an Ethernet frame. It is not always possible for a device to determine unambiguously whether or not the PFP suffix is present in a given frame.
- However, PRP has a certain amount of market traction. This may increase before P802.1CB is complete. (This author is less sure whether or how fast HSR will progress.)
- I conclude that good interoperability between P802.1CB and PRP will aid the adoption of IEEE Std 802.1CB.

Parallel Redundancy Protocol



- PRP assumes any number of “Doubly-Attached Nodes obeying PRP” (DANPs, 1 and 2 above).
- Each DANP has exactly two connections, one to each of two Bridged Networks (A and B).
- There an also be “Singly-Attached Nodes” (SANs, not shown, above) that connect either to one or the other network A or B, or to a “redundancy box” (RedBox, not shown) that has dual connections to networks A and B.

Parallel Redundancy Protocol

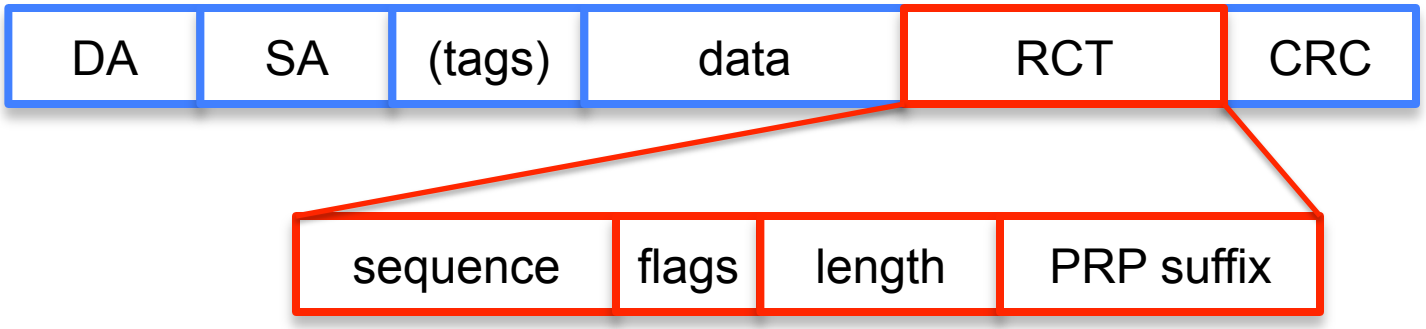


- A Doubly-Attached Node obeying PRP" (DANP) looks rather similar to an end station with Link Aggregation.
- Of course, in the case of PRP, the Link Redundancy Entity:
  - On output, adds a trailer to every frame and outputs the frame on **both** ports, each with a slightly different trailer, rather than LinkAgg's one port.
  - On input, discards frames receive on the wrong port, or frames whose sequence number duplicates a previously-received frame. The trailer can be removed for traffic configured to be PRP, but is otherwise **passed up the stack**, just in case it is not, in fact, present.

Parallel Redundancy Protocol

- When a DANP transmits a frame, it transmits two copies, one to each network.
- Each frame has a “Redundancy Control Trailer” attached to it.
- The “data” field is the usual Type/Length and data. It is the mac_service_data_unit before the RCT is appended by the Link Redundancy Entity.

Parallel Redundancy Protocol



- The flags mark whether the frame was sent on the A or B network, and can indicate other things.
- The length includes the RCT and data, but not the Length/Type at the beginning of the data.
- The sequence number is used to discard duplicate frames.
- **The "PRP suffix" is intended as a post-fix EtherType, to distinguish PRP from other suffixed tags.** (It was added in Edition 2.0.)

Why a trailer (!!!???!?!?!?!)

- The goals of PRP are clearly:
  - Work with existing bridges, including those that look beyond the MAC addresses and VLAN tag for various reasons.
  - Require software changes only, and only to end stations, not to bridges.
- Given those goals, a trailer is likely to be ignored by bridges, whereas a front-end tag might disable desirable features such as Link Aggregation distribution by IP 5-tuple.
- Given that the IP header includes a length, and that the primary use case for PRP is to carry IP packets, the trailer can be discarded by the IP layer, even if passed up the stack by the Link Redundancy Layer.

PRP/HSR use of Sequence Number

- PRP and HSR require that a DANP or DANH (Doubly-Attached Node obeying HSR) use a single sequence number variable. (This is per source MAC address used, but of course, the typical number of source MAC addresses is 1.)
- At the receiving side, a duplicate frame is detected and discarded by the pair, {source MAC address, sequence number}.
- **No algorithm is defined for duplicate deletion by ISO/IEC 62439 Edition 2.0.** The algorithm defined in the first edition was removed.
  - That algorithm remembered only the last-received sequence number on each port.

One-slide summary of HSR

- HSR supports DANHs (each of which has two ports) that are connected in a ring.
- In HSR, the redundancy entity modifies a frame from the higher layers by:
  - Adds a tag with flags, length, and sequence number rather like PRP, though more extensive, and in the usual position at the beginning of the frame.
  - Moves the destination MAC address to a position following the HSR tag.
  - Adds an HSR-specific multicast destination MAC address.
- HSR defines a “QuadBox” with four ports that can interconnect two rings.
- Two PRP RedBoxes on a ring can each connect to a Bridged LAN. They can send/receive PRP frames on the LANs, translate between PRP trailers and HSR tags, and thus create complex networks with more-or-less similar capabilities to those envisioned for P802.1CB.

Per-source versus per-more-things Sequence Numbers

What are the use cases for P802.1CB?

- While there a large number of use cases for 802.1CB, they seem to fall into two broad categories:

1. **Process control frames**, generally one per operational cycle, similar to the PRP use case.
2. Redundancy for **data streams**, e.g. video, over unreliable media.

- These two categories have different needs for discarding replicates.

Sequence numbering

There are (at least) three different sets of requirements one can place on a replicate elimination algorithm. All have a "… BUT!"

1. Hans Wiebel's dual sequence number algorithm satisfies:
   a. Packets are **never delayed**, but either delivered or dropped immediately.
   b. Packet delivery is **never out-of-sequence**.
2. The algorithm for P802.1CB mentioned by Oliver Klienberg satisfies:
   a. Packet are **never delayed**, but either delivered or dropped immediately.
   b. Packet delivery **can be out-of-sequence**.
3. Obvious extensions of our current work could satisfy:
   a. Packets **are delayed**, being buffered to keep them in sequence.
   b. Packet delivery is **never out-of-sequence**.
4. (Delayed but out-of-sequence doesn't seem very useful.)

Sequence numbering: PRP

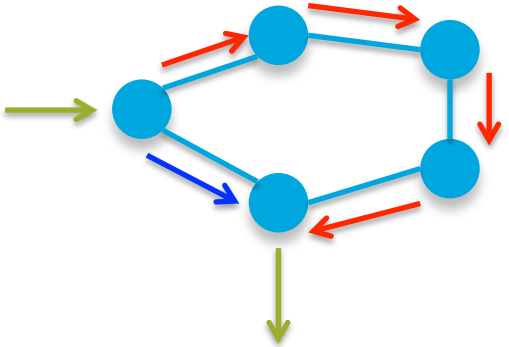
- Weibel's PRP algorithm can be implemented very simply, with one or two sequence number registers per port.
- Basically, you remember the last-received sequence number for each source MAC address on each port. If a new frame has a higher sequence number than any received so far from that source address, it is delivered, else it is discarded.
- It doesn't really matter whether the sequence number is tied to a source MAC address, or a {source address, destination address, VLAN} triple, since only one sequence number is remembered.

Sequence numbering: PRP

- On the bad side, if the difference in delivery time for the two paths is greater than the packet transmission interval for the stream, and if the faster path is flakey, then delivery is flakey.
  - A1 = sequence number 1 on path A. (B3) = dropped packet.
  - Arrival order at discard function: A2 B1 (A3) B2 A4 B3 (A5) B4 A6 B5 (A7)
  - Delivery to customer: A2 A4 A6, because A4 arrived before the missing B3.
- **So: "Never delay, never resequence" requires that the network delay difference is less than the inter-packet interval**.
  - This is assumption is **valid** for many process control applications, but not all.
  - This assumption is **invalid** for many streaming applications such as audio/ video.

Sequence numbering: .1CB to date

- It was suggested in e-mails by Kleineberg that a “Never delay, but allow resequencing” algorithm could handle high-speed data streams.
- This requires that the discard function remember some number of already-seen sequence numbers, up to the maximum number that can be received on one path before a potential duplicate of the first can be received on another path.
- Using the same example as before:
  - Arrival: A2 B1 (A3) B2 A4 B3 (A5) B4 A6 B5 (A7)
  - Delivery: A2 B1 A4 B3 A6 B5



Sequence numbering: .1CB to date

- But this requires that the replicate deletion function remember the sequence numbers of all recently-received frames.
- **If the source is using a single sequence number for all transmissions**, and if only a fraction of those frames are being sent to a given deletion function (the rest being sent to other deletion functions), then:
  - The deletion function must maintain a table, per source MAC address, of recently received sequence numbers.
  - **Every frame's sequence number must be looked up in the table (presumably at line rate), to see whether to discard and/or add the sequence number to the table.**

Sequence numbering: .1CB to date

- On the other hand, if the source uses a separate sequence number for each destination MAC address (and VLAN), then the replicate deletion function expects to see contiguous sequence numbers, and instead of a lookup table, a single register and 1-bit-wide shift register can be used:
  - Receiving a packet with a higher than expected sequence number (but not too much higher) advances the shift register.
  - A packet with a sequence that is lower than what is covered by the shift register, or is too much higher, is an error/out-of-sequence/reset condition to be handled by software.
  - A packet within the shift register range is delivered or discarded and the bit is set.
- **So: "Never delay, but resequence" can handle streams, and can be implemented relatively easily IF sequence numbers are per-source-destination-VLAN, but delivers the frames out of order.**

Sequence numbering: possible future

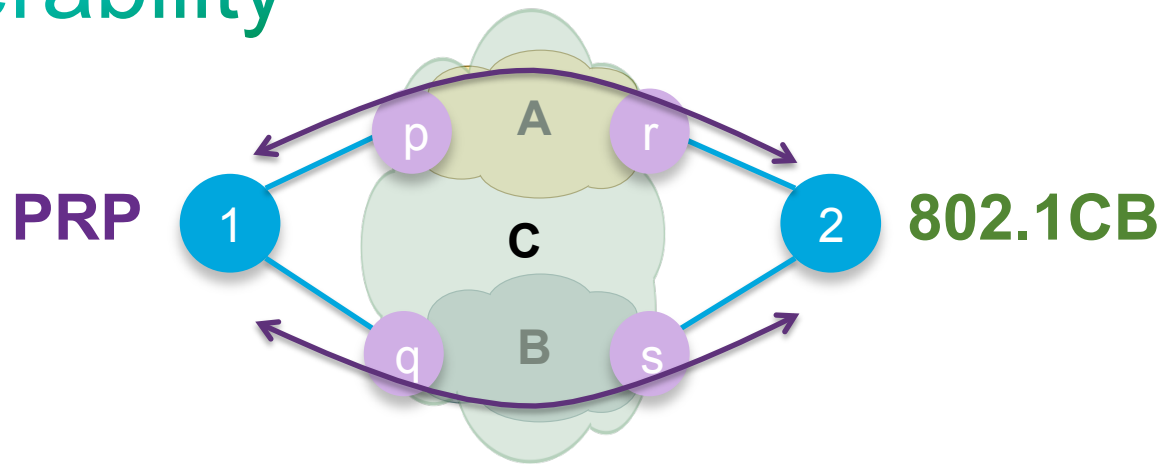
- Can we meet the “ideal” of delivering reliably and in-sequence?
- Yes, but it requires buffering the faster path until the slow path can deliver missing frame(s).
- It also requires timeouts, in case the missing frame never arrives.
- The timeouts make it difficult to deliver on latency guarantees.
- Given the complexity required, and given that stream data already has methods for discarding duplicates and/or dealing with out-of-order delivery, this idea seems to be a non-starter.

Sequence number summary

Capability	Use case	Sequence numbering
No delay, no reorder (PRP)	Process control only	Doesn't matter
No delay, reorder (.1CB)	Process control or streams	Per-source, per-destination, per-VLAN much easier to implement than per-source.
Delay, no reorder (future?)	Process control or streams	Per-source, per-destination, per-VLAN easier to implement, but buffers are the real problem.

Interoperability with PRP

Interoperability



- It will be easier to deploy 802.1CB if a PRP DANP can communicate with an 802.1CB bridge or end station. **But:**
  - As discussed, it would be best if 802.1CB sequence numbers are per-source-per-destination-per-VLAN.
  - As discussed, PRP sequence numbers are per-source only.
- Bridges p, q, r, and s **cannot reliably create/translate** between the two sequence number schemes at full speed, because of the possibility of lost frames on the 1-p, 1-q, 2-r, and 2-s links.

Suggestion for interoperability

- I would suggest, therefore, that our P802.1CB (and perhaps P802.1Qca and P802.1Qcc) protocols support the operation of 802.1CB using **either per-source sequence numbering or per-source, per-destination, per-VLAN sequence numbering, at the option of the sender,** depending on whether the data is slow, or is a stream.
- This may require a bit in the tag, assuming that there is a tag.

Thank you.