Link-local Registration Concepts for TSN/DetNet

Rodney Cummings (Rodney.Cummings@ni.com), National Instruments

September 14, 2017

This white paper proposes architectural concepts for the evolution of link-local registration in TSN/DetNet, sometimes referred to as fully-distributed configuration.

In the 802.1 working group, link-local registration for AVB/TSN began with P802.1Qat, which created the Stream Reservation Protocol (SRP) of 802.1Q clause 35. SRP consists of three application protocols: MMRP for dynamic MAC addresses, MVRP for dynamic VLANs, and MSRP for stream reservation. SRP's application protocols are built on top of the Multiple Registration Protocol (MRP), the link-local registration protocol specified in 802.1Q clause 10.

The project P802.1Qcc enhanced SRP in many ways. One of the major enhancements was the transition of MSRP's data from a single monolithic PDU into a sequence of TLVs. The TLV design allows for optional features. If an optional feature is not applicable, its TLV is not used, thus saving data on-the-wire. Some of the TLVs related to the mixed centralized/distributed model using a central controller (CNC). Other TLVs related to fully-distributed, such as stream transformation (i.e. use of IP headers in talker/listener), and the maximum latency requirement.

Since the transition to TLVs required a new structure for the MSRP data, in P802.1Qcc we decided to specify two versions of MSRP. MSRPv0 is the original from P802.1Qat, using a single PDU. MSRPv1 is the new TLV-based design of P802.1Qcc. To support compatibility, translation between MSRP versions is specified normatively. An example of the MSRP version translation is shown in Figure 1.

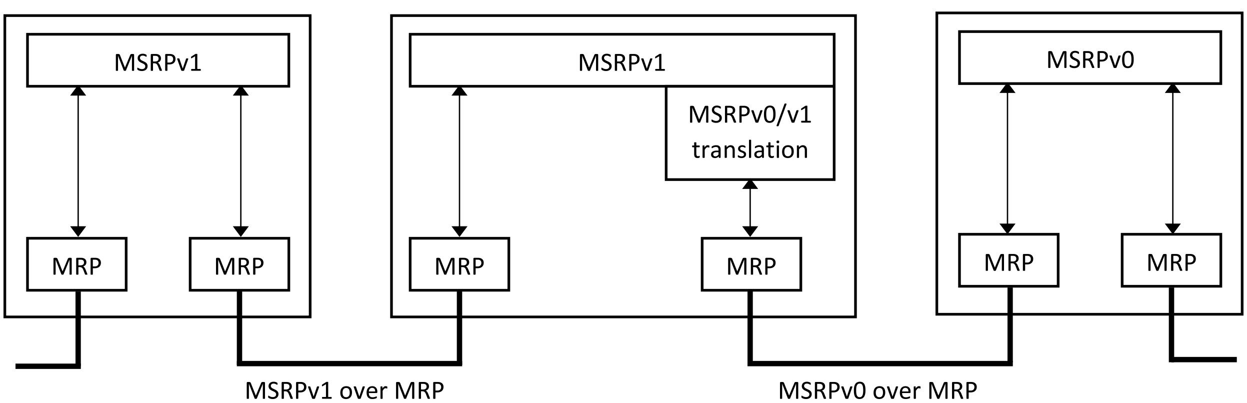

Figure 1: Example of MSRP version translation in P802.1Qcc

The upper block shows the version of MSRP implemented by the bridge (v0 or v1), including specifications for propagation of MSRP data from one port to another (i.e. MSRP's "MAP"). Implementation of MSRPv1 requires support for MSRPv0 for compatibility. An MSRPv1 bridge negotiates with its link-local neighbor to determine the highest MSRP version that is supported by both, and that MSRP version is used over that link. This means that if the neighbor of an MSRPv1 bridge implements MSRPv0 only, then MSRPv0 is used over that link. P802.1Qcc specifies normative rules for the translation of v0 data to/from v1, which is shown in the figure as a block below the bridge's MSRPv1 block.

The double-sided arrows represent an abstract interface that separates the application (e.g. MSRP, MMRP) from the link-local registration protocol (i.e. MRP). 802.1Q currently refers to this abstract interface as MRP Attribute

Declaration (MAD). Although MAD has some aspects that are specific to MRP, for the most part the application is abstracted from the details of MRP (such as its state machines). The MAD interface exchanges an "attribute", which is a unit of data in the application.

That is the past... what about the future?

P802.1CS will specify the Link-local Registration Protocol (LRP), a more efficient and scalable replacement of MRP. Since P802.1CS can transfer large amount of data over the link, it is well suited for TLV-based application data.

In addition, there is an informal assumption that as the IETF DetNet work progresses, that working group may want to integrate DetNet into RSVP-TE (RFC 3209). RSVP-TE uses TLVs in its design, much like MSRPv1. RSVP-TE can be viewed as another link-local registration protocol, where the link is IP-based.

Since one of the goals of DetNet is to support a mixed L2/L3 network, the working groups will presumably work toward on-the-wire cooperation for fully-distributed.

Let's consider the following assumptions:

- MSRPv1 will be enhanced, retaining its TLV structure. This paper refers to this as "MSRPv2".
- IETF DetNet will specify TLVs for use with RSVP-TE. This paper refers to this as "RSVP-DetNet".
- A standard will specify translation between RSVP-DetNet and MSRP (v0, v1, and v2).
- RSVP-DetNet TLVs cannot "pass through" bridges without interpretation, as RSVP-TE does. TSN/DetNet only works when configured hop-by-hop in a contiguous set of bridges/routers.

To get these pieces to work together, we need some architectural concepts, shown in Figure 2 for a single port.

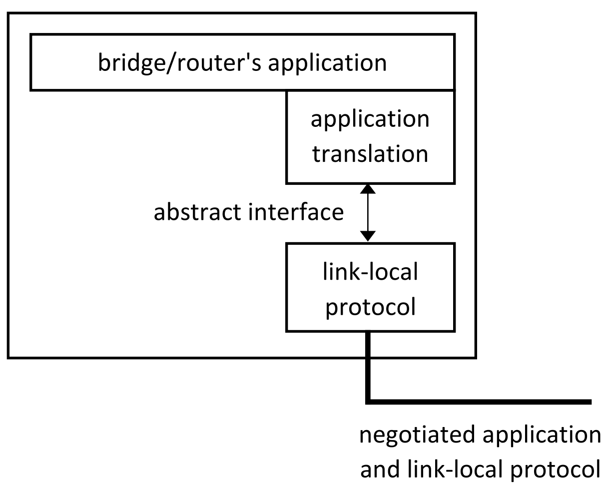

Figure 2: Architectural concepts

The "bridge/router's application" and "application translation" concepts are analogous to the concepts specified in P802.1Qcc. The bridge or router implements a single TSN/DetNet application, which can be one of MSRPv0, MSRPv1, MSRPv2, RSVP-DetNet, or a future application. The neighboring bridge/router on the link might not implement the same application as this bridge/router. To provide for maximum interoperability and compatibility, the neighbors negotiate to find the "best" application that is supported in both, and translation is performed as needed. For example, if one bridge supports MSRPv2, and its neighbor supports MSRPv1, the MSRPv2 bridge will translate to/from MSRPv1 for use on the link.

The "link-local protocol" concept is enhanced to support multiple options for the link-local protocol, including MRP, LRP, and RSVP-TE. Much like the application, two neighbors on the link will negotiate to find the "best" link-local protocol to use. For example, in the 802.1 working group, we seem to have consensus that a bridge that supports LRP shall also support MRP. If two neighboring bridges support LRP, then LRP is used on the link. If one of the neighbors supports MRP only, then MRP is used.

If the preceding assumptions are correct, we need to ensure that a router exchanges the DetNet TLVs with a neighboring TSN bridge. This can be accomplished with one of the following options:

- Bridge supports RSVP-TE as the link-local protocol. RSVP-TE carries DetNet TLVs. The bridge uses translation between DetNet TLVs and its MSRP application.
- Router supports LRP as the link-local protocol. LRP carries MSRPv2 TLVs. The router uses translation between MSRPv2 and its DetNet application. RSVP-TE packets continue to "pass through" bridges to the neighboring router without DetNet TLVs.

The author of this white paper has no strong preference, and this paper is not intended to include a discussion of the pro's and con's of each option. In the context of this paper, the important point is that whichever option we chose, it requires the concept of negotiation of the link-local protocol between bridge and router.

Now that we have multiple options for the application, and multiple options for the link-local protocol, the purpose of the "abstract interface" is obvious. The "abstract interface" provides a clear interface for <u>any</u> application to declare (i.e. write) and register (i.e. read) attributes using <u>any</u> link-local protocol that communicates with its neighbor.

Since the "abstract interface" is not represented on-the-wire, it has no compatibility issues. Therefore, statements like the following are misguided:

> "LRP requires an LRP-specific abstract interface, because MRP's MAD doesn't do everything we want."

To use a specific example, LRP will have the concept of a "record", where a record can represent a compact encoding of multiple "attributes" (data values). MRP's abstract interface (MAD) only supports exchange of a single attribute.

> If we specify a new abstract interface that uses a record, can that be applied to MRP?

The answer is Yes. Anyone who has implemented MRP knows that it already has the "record" concept (i.e. NumberOfValues, FirstValue). It is true that MRP's "record" is not explicitly represented in MAD. Nevertheless, MAD is simply an abstract interface, so MAD can evolve without breaking compatibility. Whatever "new" interface is specified for LRP, that interface can be applied to MRP as well.

In designing the abstract interface, we need to be careful to ask the following for each parameter:

> Is this parameter needed for the interface, or is it a detail of the underlying link-local protocol?

The answer must always be the former, and not the latter. We need the abstract interface to serve multiple link-local protocols... not just one. Parameters like checksums, counters, sequencing IDs, and so on have the tendency to be closely tied to on-the-wire usage in frames/packets.