802.1 Qcc Sponsor Ballot Comment #i-22
“Declaring more MSRP streams”

CRAIG GUNThER
CRAIG.GUNThER@HARMAN.COM, CRAIGGUNThER@YAHOO.COM

January 2018
**Goal: Increase number of streams in MSRP**

According to the *MRP Timers – Maximum attribute registrations* presentation I gave in 2010 ([at-cgunther-mrp-timers-0310-v02.pdf](at-cgunther-mrp-timers-0310-v02.pdf)) you can expect a worst-case maximum of 530 streams. In that presentation I listed MRP configuration options that can be used to increase that number:

- **Lengthen LeaveTime** (default = 600-1000 msec)
  
  *Purpose:* Controls how long to wait for an attribute to be defended, as a result of receiving a LeaveAllEvent or a Leave, before the attribute is withdrawn.
  
  *Benefit:* Waiting longer allows more attributes to be declared.
  
  *Detriment:* Stale attributes take longer to be removed.
  
  *My current recommendation:* **Required to achieve goal!**

- **Lengthen LeaveAllEvent timer** (default = 10 seconds)
  
  *Purpose:* Controls how often “trash collection” occurs.
  
  *Benefit:* Waiting longer allows more attributes to be declared between events.
  
  *Detriment:* Stale attributes take longer to be removed.
  
  *My current recommendation:* Use if necessary.

- **Shorten JoinTime** (default = 200 msec)
  
  *Purpose:* Controls frequency of MSRPDU transmissions.
  
  *Benefit:* Allows attributes to be declared in less time.
  
  *Detriment:* Increases contention for SR Class interval bandwidth.
  
  *My current recommendation:* Don’t use.
EFFECTS OF LeaveTime ON LeaveAllEvent

There are two uses of LeaveTime that are of interest to this presentation:

- **LeaveAllEvent** (rLA! and txLA!)
- Attribute withdrawal (rLv!)

When garbage collection occurs the attribute must be re-declared in order for the registration to remain intact. This process utilizes LeaveTime (Start leavetimer) to delay the propagation of the withdrawal to all associated ports. If the peer device re-declares (rJoinIn! or rJoinMt!) the attribute, it will return to the IN State; otherwise an Lv is propagated to all associated ports and the attribute moves to the MT state. Lengthening LeaveTime simply slows down garbage collection.

So far, so good.
**Effects of LeaveTime on Shared Attribute Withdrawal**

There are two uses of **LeaveTime** that are of interest to this presentation:

- LeaveAllEvent (rLA! and txLA!)
- Attribute withdrawal (rLv!)

On **shared media** only the first Listener needs to register to receive a stream. Subsequent Listeners will have already “heard” that registration and know that the stream will be sent across the media. However, if the first Listener withdraws its attribute (rLv!) another Listener will need time to re-declare the Listener attribute or the stream will be torn down. Note that this behavior is identical to what happens in response to a LeaveAllEvent as shown on the previous slide.

Still good to go.

---

**Legend**

- **Green**: Stream continues
- **Red**: Stream stops

---

**Subset of Table 10-4 Registrar state table**

<table>
<thead>
<tr>
<th>Event</th>
<th>IN</th>
<th>LV</th>
<th>MT</th>
</tr>
</thead>
<tbody>
<tr>
<td>JoinIn!</td>
<td></td>
<td>IN</td>
<td></td>
</tr>
<tr>
<td>JoinMt!</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rLm!</td>
<td></td>
<td>Start leavetimer LV</td>
<td></td>
</tr>
<tr>
<td>rLA!</td>
<td></td>
<td>Re-declare!</td>
<td></td>
</tr>
<tr>
<td>txLA!</td>
<td></td>
<td>leavetimer!</td>
<td></td>
</tr>
<tr>
<td>RJoinIn!</td>
<td></td>
<td>rJoinMt!</td>
<td></td>
</tr>
<tr>
<td>Stop leavetimer IN</td>
<td></td>
<td>Join IN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-x-</td>
<td>-x-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Lv MT</td>
<td>MT</td>
<td></td>
</tr>
</tbody>
</table>
Effects of LeaveTime on P2P Attribute Withdrawal

There are two uses of LeaveTime that are of interest to this presentation:

- LeaveAllEvent (rLA! and txLA!)
- Attribute withdrawal (rLv!)

On point-to-point media there is only one Listener, and when that Listener no longer wants to receive the stream it should stop playing immediately.

Currently the observed behavior is the LeaveTime associated with rLv! is creating an unnecessary propagation delay throughout the network which increases linearly with the number of bridges on the path. The default 600 to 1000 msec LeaveTime (see Table 10-7) can result in 3.6 to 6.0 seconds of unnecessary stream tear-down delay on a seven hop network. Lengthening LeaveTime makes it worse.
SOLUTION FOR LEAVETIME ON P2P ATTRIBUTE WITHDRAWAL

There are two uses of LeaveTime that are of interest to this presentation:
• LeaveAllEvent (rLA! and txLA!)
• Attribute withdrawal (rLv!)

My suggestion is to ignore the leavetimer on attribute withdrawal on point-to-point links (operPointToPointMAC is TRUE). This will allow an instantaneous transition through the “Lv/MT” action in the LV state, thus allowing immediate propagation of attribute withdrawals.

This can be done by adding a footnote to Table 10-4 as mentioned in my Sponsor Ballot comment #i-22: "For point-to-point links attribute withdrawal propagation will be accelerated by avoiding the leavetimer and proceeding directly to the Lv/MT transition as defined in the LV State for the leavetimer! event. Non point-to-point links need to implement the state transitions as shown."
Questions? Discussion?

Thanks!