802.1Qcc 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) 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to wait for an attribute to be defended, as a result of receiving a LeaveAllEvent or a Leave, before the attribute is withdrawn.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: <the reason for my Sponsor Ballot comment #i-22>
  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.

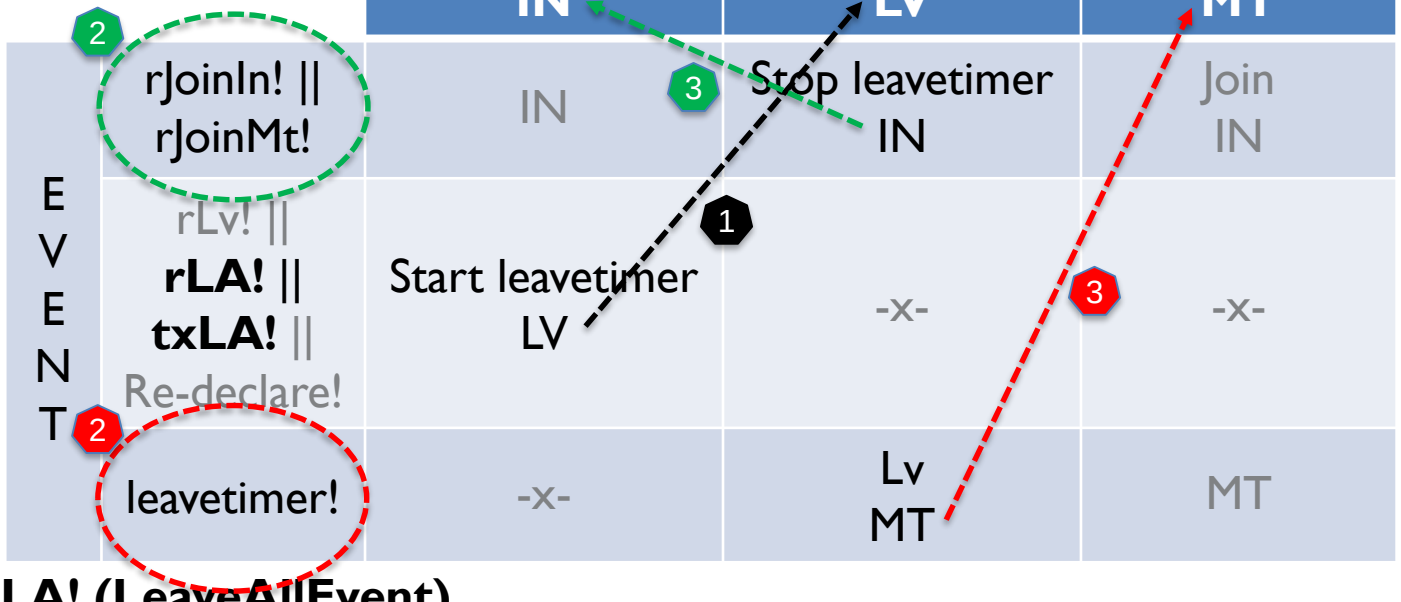
Effects of LeaveTime on LeaveAllEvent

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

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

Subset of Table 10-4 Registrar state table

EVENT	IN	LV	MT
2 rJoinIn! || rJoinMt!	IN	Stop leavetimer IN	Join IN
rLv! || **rLA!** || **txLA!** || Re-declare!	Start leavetimer LV	-x-	-x-
2 leavetimer!	-x-	Lv MT	MT



Legend
Green: Stream continues
Red: Stream stops

rLA! txLA! (LeaveAllEvent)

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!) 2 the attribute, it will return to the *IN* State 3; otherwise an *Lv* is propagated 3 to all associated ports and the attribute moves to the *MT* state. Lengthening **LeaveTime** 2 simply slows down garbage collection 3

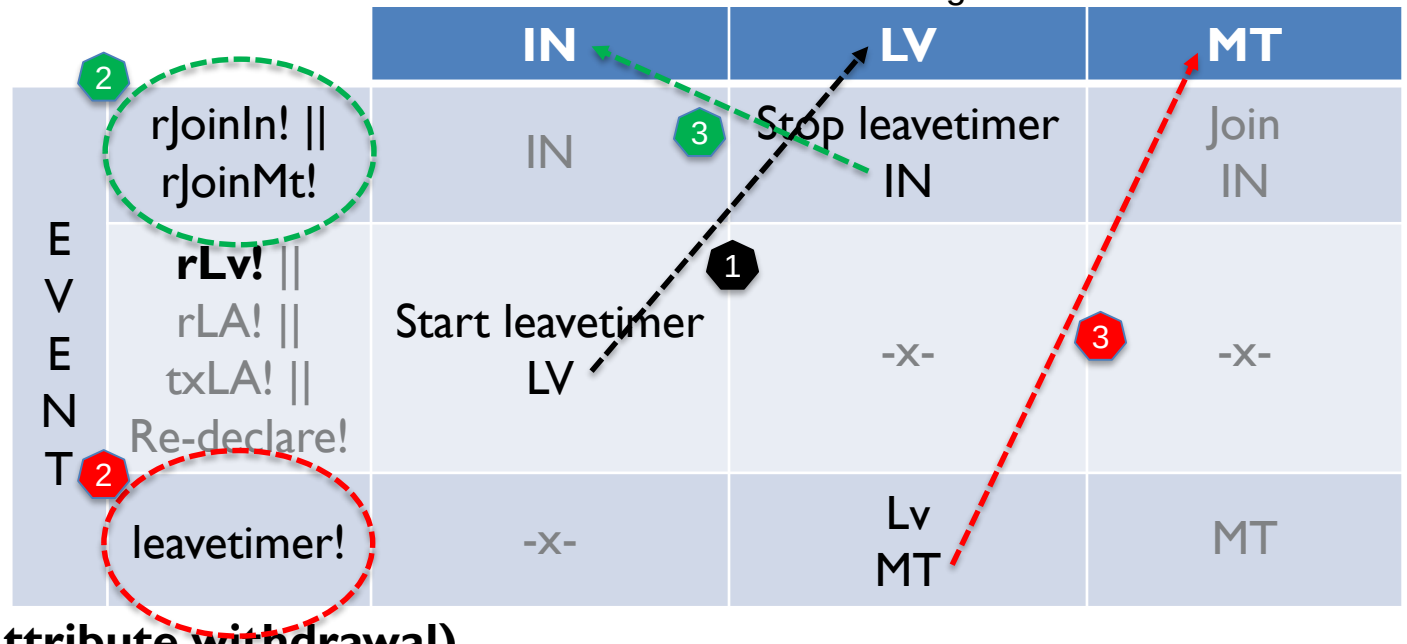
So far, so good.

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!)**

Subset of Table 10-4 Registrar state table



rLv! (Attribute withdrawal)

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 2 or the stream will be torn down 2 3. Note that this behavior is identical to what happens in response to a LeaveAllEvent as shown on the previous slide.

Still good to go.

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!)**

Subset of Table 10-4 Registrar state table

EVENT	IN	LV	MT
rJoinIn! || rJoinMt!	IN	Stop leavetimer IN	Join IN
rLv! || rLA! || txLA! || Re-declare!	Start leavetimer LV	-x-	-x-
leavetimer!	-x-	Lv MT	MT

1 2 3

Legend
Green: Stream continues
Red: Stream stops

rLv! (Attribute withdrawal)

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.

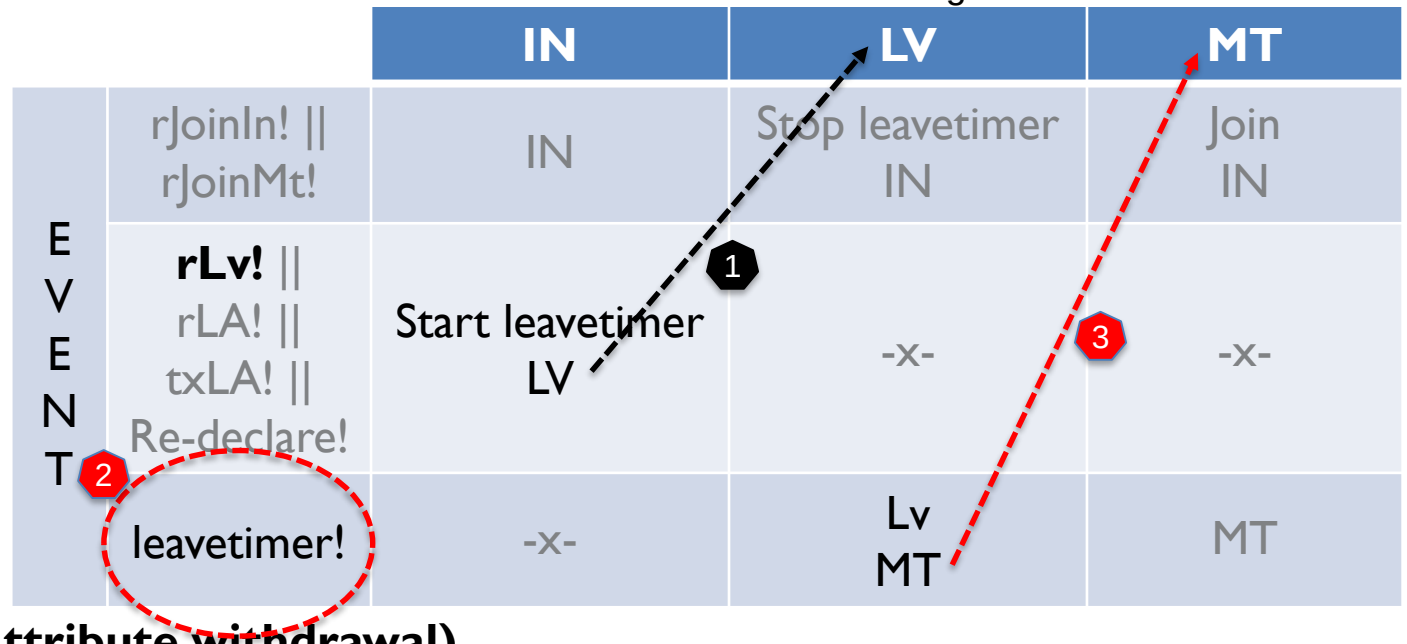
SOLUTION FOR LEAVETIME ON P2P ATTRIBUTE WITHDRAWAL

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

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

Subset of Table 10-4 Registrar state table

EVENT	IN	LV	MT
rJoinIn! || rJoinMt!	IN	Stop leavetimer IN	Join IN
rLv! || rLA! || txLA! || Re-declare!	Start leavetimer LV	-x-	-x-
leavetimer!	-x-	Lv MT	MT



Legend
Green: Stream continues
Red: Stream stops

rLv! (Attribute withdrawal)

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."

GARP (802.1D-1998) Already Had This

It has been pointed out* that this behavior was already specified in 802.1D-1998 as part of the GARP specification.

MRP was defined in 802.1ak-2007 as a replacement for GARP and therefore should have implemented this support.

What happened?

- 802.1D-1998, clause 13.2.2 *gid.h* contained this text:

  ** Standard GID operates as if the GID entity were connected to a shared*
  ** medium. GARP information may be transported more quickly if the link*
  ** is known to be point-to-point. Specifically received Leave messages*
  ** can give rise to immediate Leave indications, without the need to*
  ** solicit further Joins from other potential members attached to the shared*
  ** medium.*

- 802.1D-2004 removed all of clause 13, which removed the text
- When the 802.1ak project began in 2005 the functionality had disappeared from 802.1D, therefore this feature was not there to implement!

It's time to bring back the lost functionality from 802.1D-1998!

* Thanks to Mick Seaman for providing this research!

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

HARMAN

Questions? Discussion?

Thanks!