802.1CB improvements

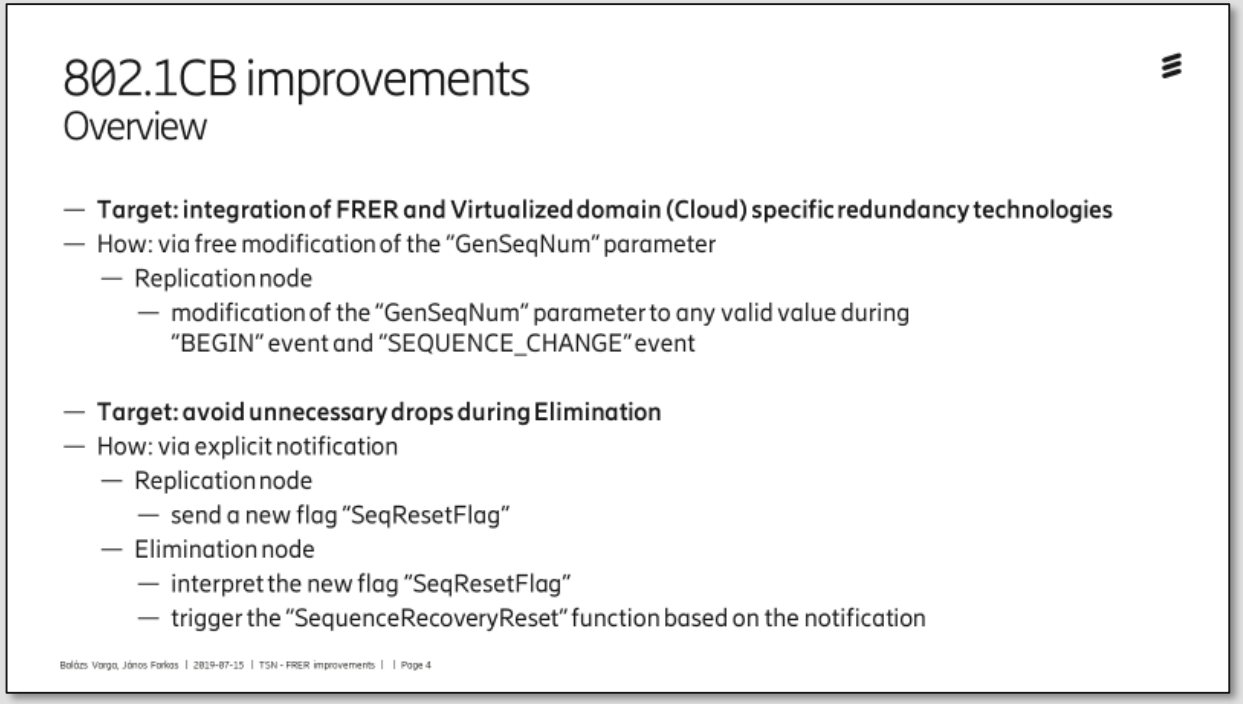
FRER
Improvements of Replication and Elimination functions (to achieve Seamless Reset)

IEEE 802.1 TSN TG
March, 2020

Balázs Varga, János Farkas Ericsson Research 2020-03-23

History

- Former proposal:
  - 2019, Vienna meeting: http://www.ieee802.org/1/files/public/docs2019/new-varga-FRER-improvements-0719-v01.pdf
  - Allow modification of "GenSeqNum"
  - Explicit signaling of reset via "SeqResetFlag" (reset of sequence generation function)



802.1CB improvements
Overview

- **Target: integration of FRER and Virtualized domain (Cloud) specific redundancy technologies**
- How: via free modification of the "GenSeqNum" parameter
  - Replication node
    - modification of the "GenSeqNum" parameter to any valid value during "BEGIN" event and "SEQUENCE_CHANGE" event
- **Target: avoid unnecessary drops during Elimination**
- How: via explicit notification
  - Replication node
    - send a new flag "SeqResetFlag"
  - Elimination node
    - interpret the new flag "SeqResetFlag"
    - trigger the "SequenceRecoveryReset" function based on the notification

Balázs Varga, János Farkas | 2019-07-15 | TSN - FRER improvements | | Page 4

Topics

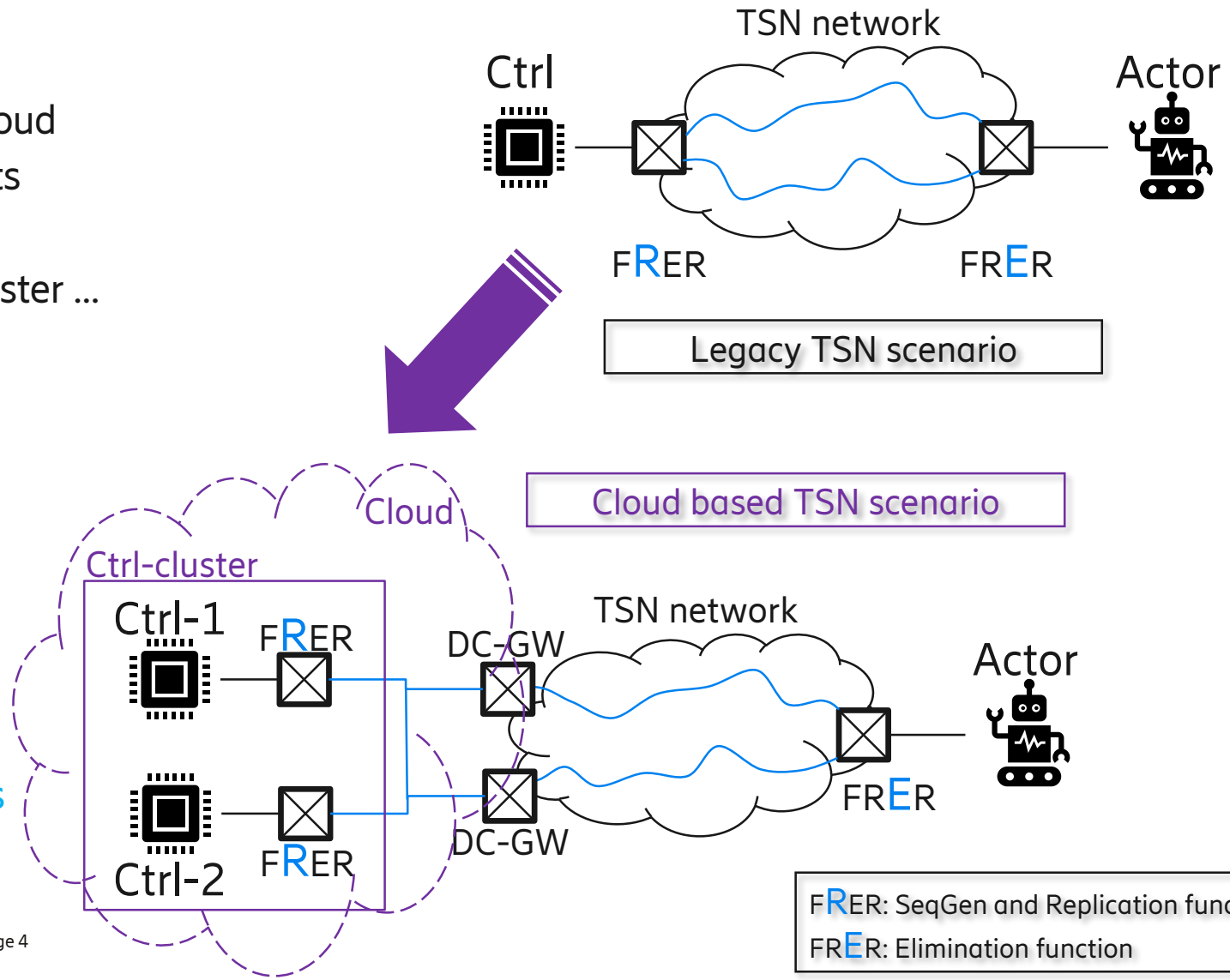
- Proposal
  - Improvements of “Sequence Number Space” in order to achieve seamless reset of FRER functions

- Background
  - Replication node specific changes
  - Elimination node specific changes
  - Backward compatibility

Moving towards Virtualized environments
Using FRER in a Cloud based scenario

Scenario: Moving a Talker/Listener to the Cloud
- TSN functions must go with the endpoints
  - FRER must work inside Cloud …
  - FRER can be an instance in a Ctrl-cluster …
- Typical Cloud actions
  - Run multiple VMs/instances
  - Create a VM/instance
  - Move a VM/instance
  - Reset a function
  - Remove a VM/instance
  - Etc.

Main motivation for proposed improvements is to allow cloudification, i.e., "cloud native" implementation of FRER functions.



802.1CB improvements
Overview

- **Problem: high probability of packet loss due to reset of Sequence Generation function**

- **Target: seamless reset of Sequence Generation function**
- How: change Sequence Number Space characteristics
    1. explicit notification of the reset event, and
    2. extending the sequence number space (additional new linear initial sequence number space)

    - Explicit notification of the reset event is based on the flag included in the R-TAG, namely the "SeqResetFlag" (see former contribution on history slide). The usage of the new linear initial sequence number space ("InitSeqNumSpace") is noted via a new flag included in the R-TAG, namely the "InitSeqFlag". Sequence values of the new number space ("InitSeqNumSpace") are also included in the R-TAG (using the existing sequence number field).

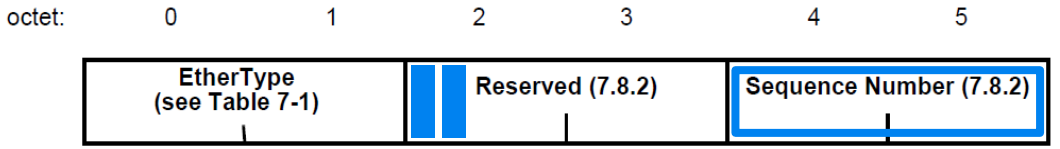
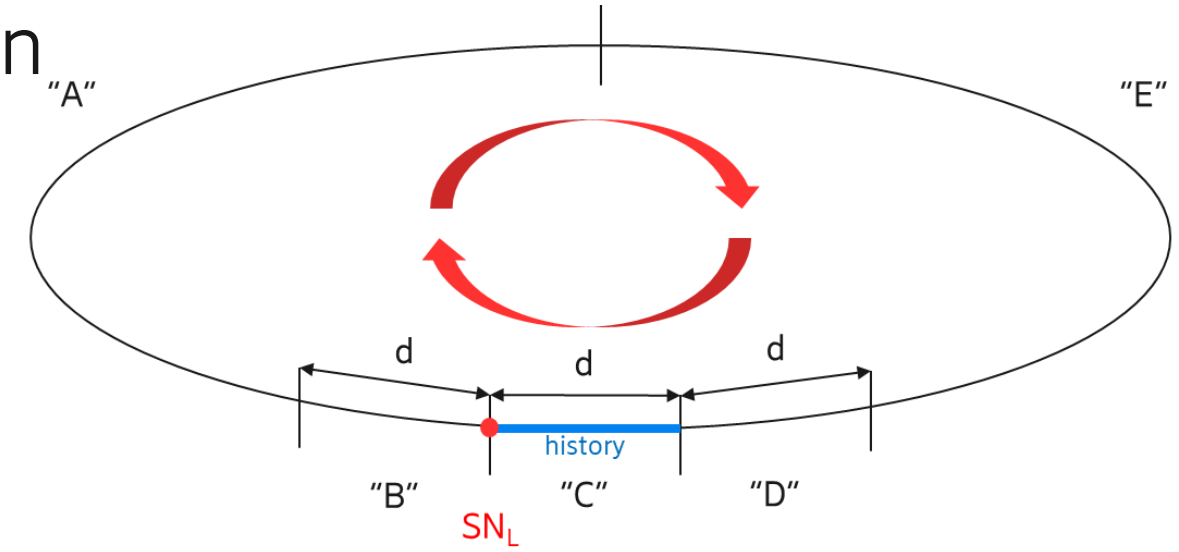


Figure 7-4 in IEEE 802.1CB

Evaluation of existing reset mechanism

Analysis of 802.1CB
Reset of Sequence Generation Function

Ranges of Sequence Number Space used for analysis

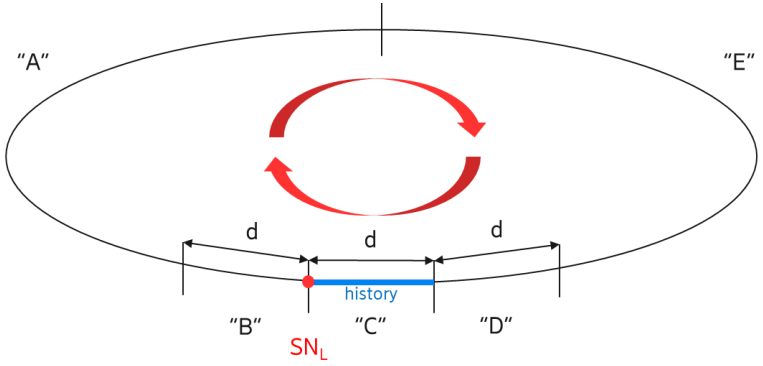


- The impact of the reset of the sequence number generation function depends on the actual value of the sequence_number used for the last packet sent before the reset (noted here as SN_L) and the value of the sequence_number used for the first packet sent after the reset (noted here as SN_R).
- 802.1CB-2017: SN_R is always 0 and SN_L is a value in the range of {0; ... ; GenSeqSpace - 1}.
- We can define the following ranges to analyze the impact of the reset for the solutions
  - A: $SN_R > SN_L + d$
  - B: $SN_L + d >= SN_R > SN_L$
  - C: $SN_L >= SN_R > SN_L - d$
  - D: $SN_L - d >= SN_R > SN_L - 2d$
  - E: $SN_L - 2d >= SN_R$

  where d = "frerSeqRcvyHistoryLength"

Note that due to the cyclic characteristic of the original sequence number space "A" and "E" are adjacent ranges, the border between them is be defined here for the analysis as $modulo_{GenSeqSpace}(SN_L + GenSeqSpace/2)$

Analysis of 802.1CB
Performance

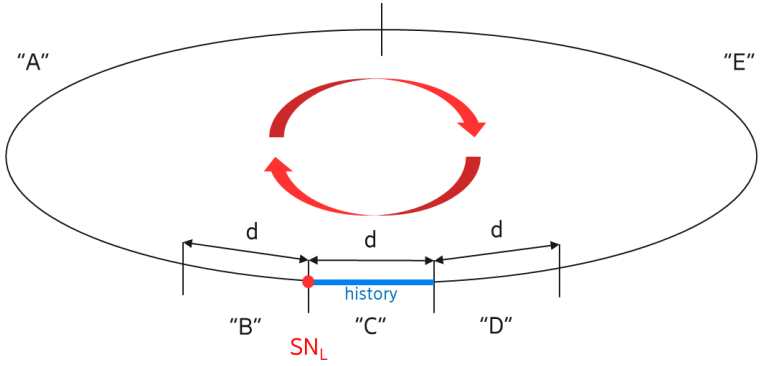


— **For the evaluation of 802.1CB-2017**, the following range is important:
  — history window (HSW) = { SN_L + d; .. ; SN_L – d +1 }, where d = "frerSeqRcvyHistoryLength"

  Note: HSW is practically the merge of range "B" and range "C" used for the evaluation.

— As per 802.1CB-2017, packets with a sequence_number out of the history window (HSW) are dropped and packets within the history window are evaluated against the "SequenceHistory" to decide whether or not they are duplicates. Therefore, 802.1CB-2017 operates as follows:
  — A: packets are dropped until timer expires or the sequence_number of packets reaches SN_L + 1,
  — B: no packet drop, SN_R is accepted,
  — C: packets are dropped until sequence_number of packets reaches SN_L + 1,
  — D: packets are dropped until sequence_number of packets reaches SN_L + 1,
  — E: packets are dropped until timer expires or the sequence_number of packets reaches SN_L + 1.

Analysis of 802.1CB
Performance …



- There is high probability of packet drop (99.8%)
  - in most cases until timeout triggers the accept of the next packet.
  - packet drop means packet loss for the application whose operation may be damaged by the lost packets.

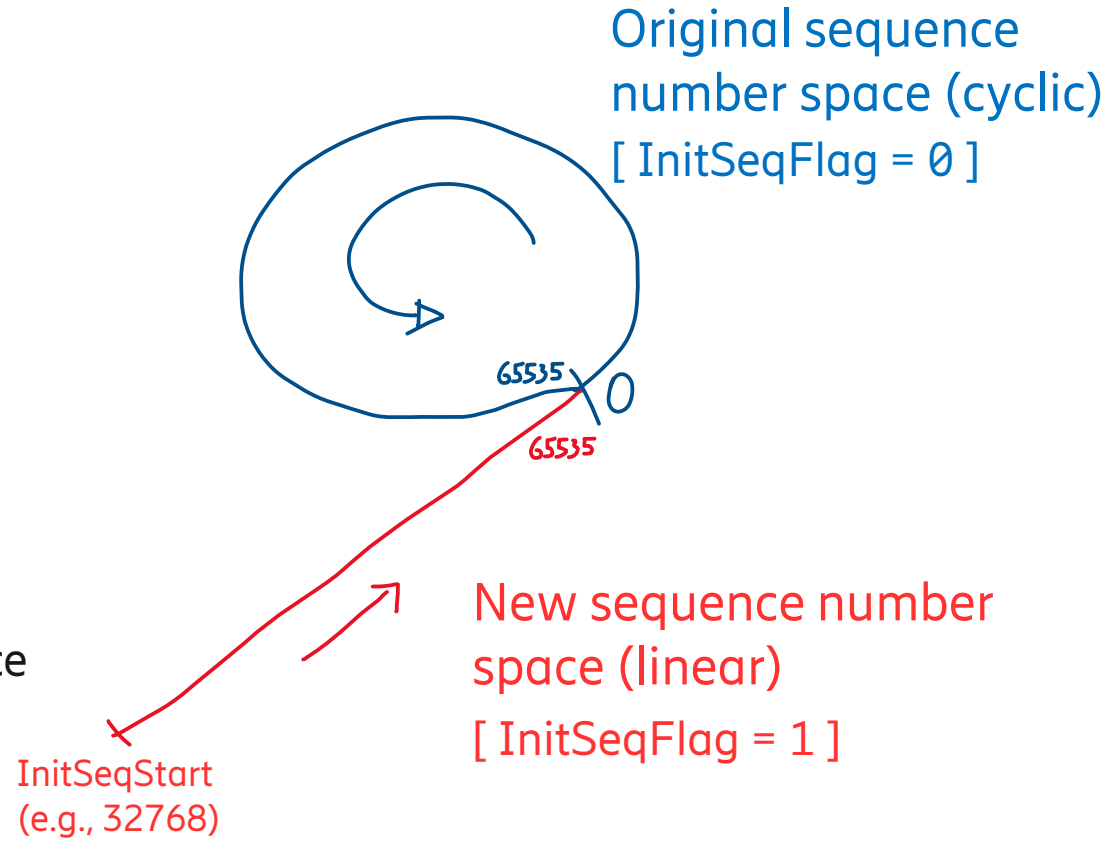
  Note: The assumption for the analysis is that (i) there are no other events in the network, only the sequence generation function is reset and (ii) the "frerSeqRcvyHistoryLength" (d) takes value of "100" for the probability calculations.

- This is not acceptable for cloud based scenarios !
- Remaining slides shows a solution via modifying the Sequence Number Space characteristics

NEW Sequence Number Space and the reset related FLAGs

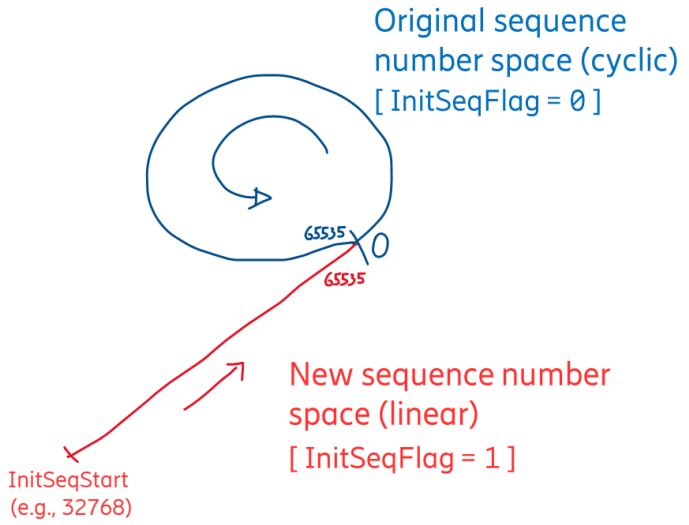
NEW Sequence Number Space
Basics of operation

- Solution: introducing a new additional sequence number space "*InitSeqNumSpace*", which is used after initialization/reset of the sequence generation function.
  - *InitSeqNumSpace* is illustrated by the red linear sequence number space.
  - The sequence_number of the next packet is stored in a new variable, namely the "*InitGenSeqNum*".
  - When this new number space is exhausted the sequence generation function starts to use the original sequence number space, which is illustrated by the blue cyclic sequence number space.
- New flags:
  - Usage of the new Sequence Number Space is marked by a new flag in the R-TAG, namely the "InitSeqFlag".
  - A set of packets are marked after reset with a new flag in the R-TAG, namely the "SeqResetFlag".



An implementation may use e.g., two variables to enable/disable the use of the reset flag ("*UsingResetFlag*" = 1/0 (enable/disable)) and the new initial sequence number space ("*UsingInitSpace*" = 1/0 (enable/disable)).

NEW Sequence Number Space
Details of operation (InitSeqNumSpace)

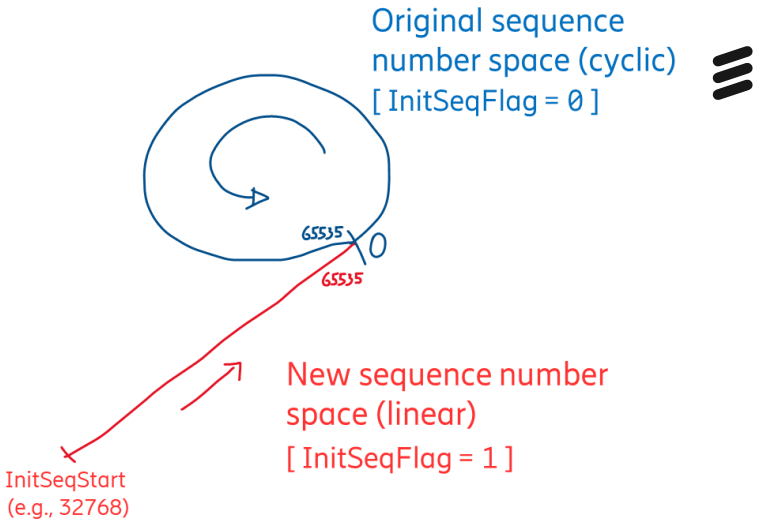


"*InitSeqNumSpace*" is used as follows:

- When sequence generation function was initialized/reset, the "*InitSeqNumSpace*" is used to generate the sequence number for the packets.
- "*InitSeqNumSpace*" is a linear sequence number space starting with "*InitSeqStart*" (configurable, e.g., 32768) and ends with "GenSeqSpace-1" (i.e., 65535 as per 802.1CB-2017).
- When this new number space is exhausted the sequence generation function starts to use the original sequence number space

- "*InitSeqNumSpace*" has its own set of variables that are used to process the sequence_number of a packet by the Replication function and Elimination function.
  Note: *InitSeqNumSpace* variable names are derived from the variable names of the original sequence number via appending them with an "*Init*" prefix, e.g., "*InitGenSeqNum*", "*InitRecovSeqNum*", "*InitSequenceHistory*", "*InitTakeAny*", "*InitRemainingTicks*", etc.

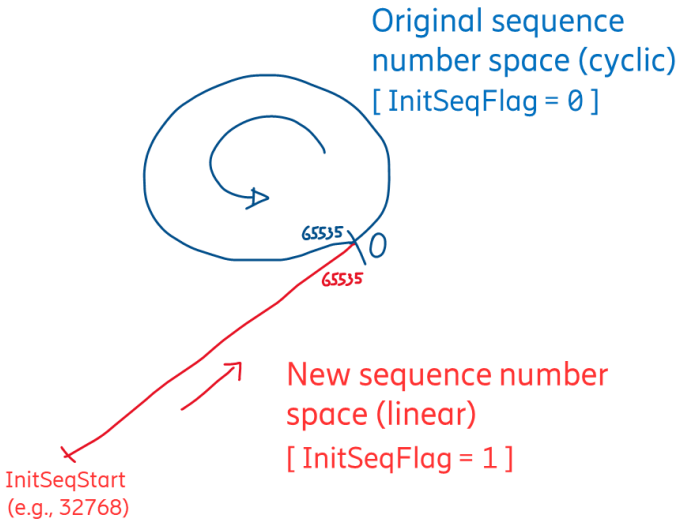
NEW Sequence Number Space
Details of operation (InitSeqFlag)



- Packets sent with a sequence_number from "*InitSeqNumSpace*" are marked with "*InitSeqFlag*" set (value=1)

- SeqGeneration function (e.g., the Talker)
  - does not change the "*InitSeqFlag*" value of the packet if R-TAG is already included in the header of the packet.
  - set (value=1), when sequence generation function was initialized/reset, and the "*InitSeqNumSpace*" is used to generate the sequence number for the packet.
  - clear (value=0) by sequence generation function in all other cases.
- Elimination function
  - does not change the "*InitSeqFlag*" value of the packet.

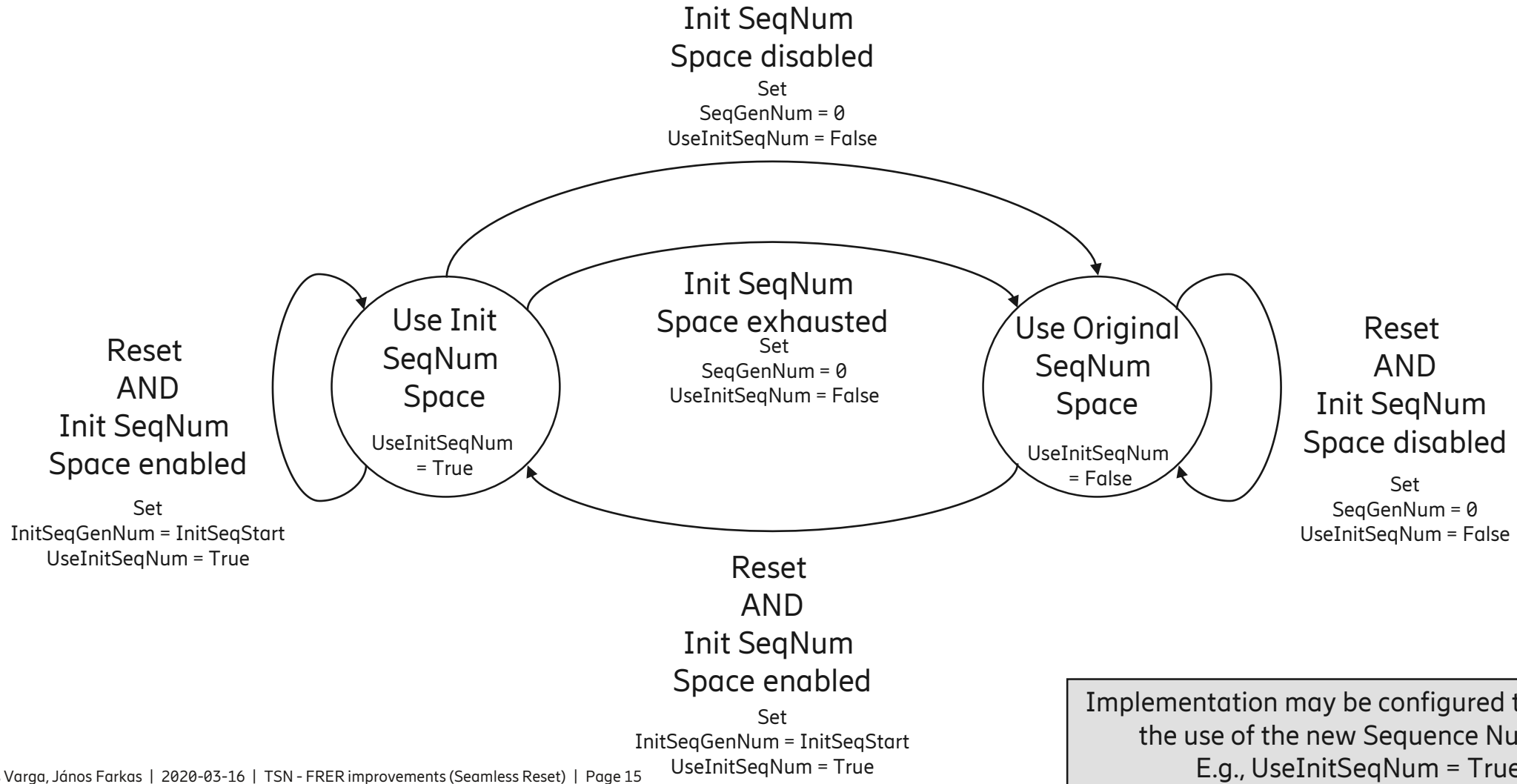
NEW Sequence Number Space
Details of operation (SeqResetFlag)



- Solution uses a new flag introduced to the R-TAG, namely the "*SeqResetFlag*", being used as follows:

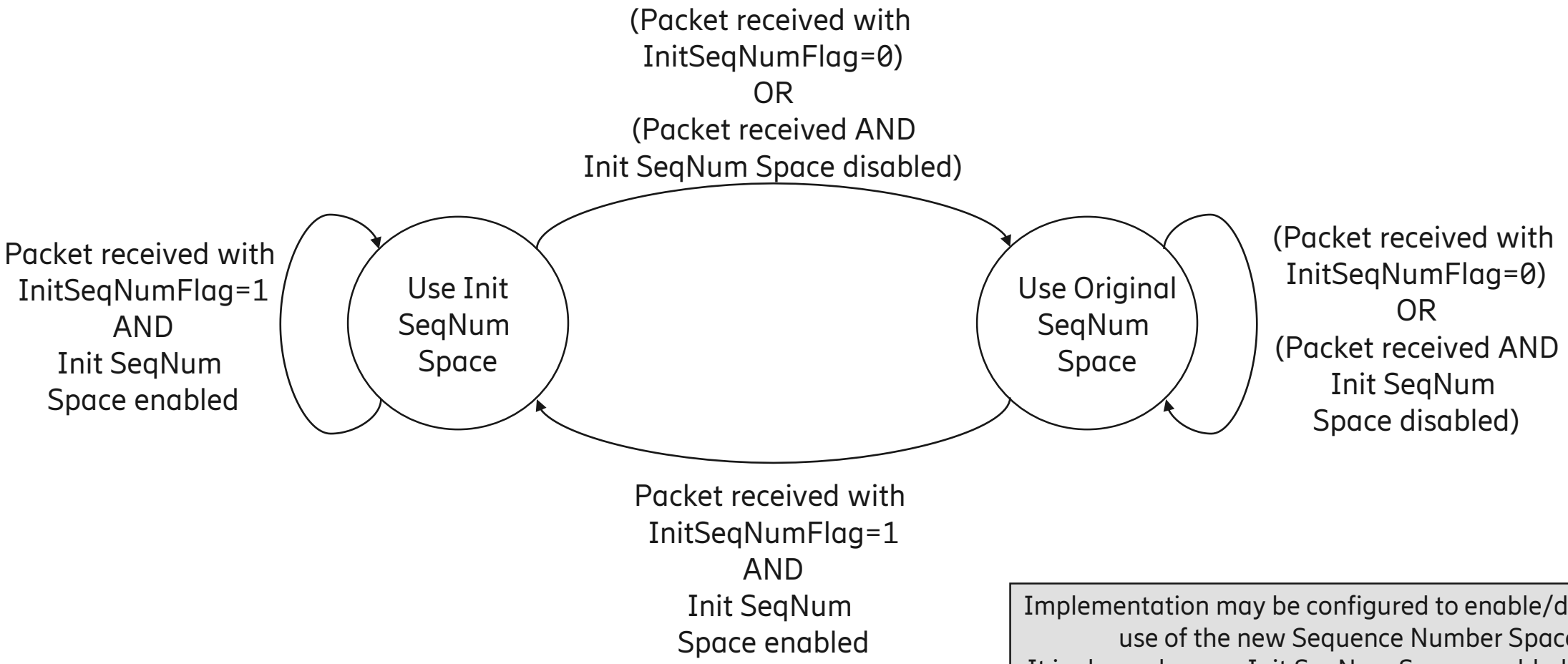
- SeqGeneration function (e.g., the Talker)
  - does not change the "*SeqResetFlag*" value of the packet if R-TAG is already included in the header of the packet.
  - set (value=1) for a given time period or for a number of sent packets when sequence generation function was reset.
  - clear (value=0) by sequence generation function in all other cases.
- Elimination function
  - does not change the "*SeqResetFlag*" value of the packet.

Sequence Generation node
Which SeqNum Space to use for the next packet to be sent?



Implementation may be configured to enable/disable the use of the new Sequence Number Space, E.g., UseInitSeqNum = True / False

Elimination node
Which SeqNum Space to use for a received packet?



Implementation may be configured to enable/disable the use of the new Sequence Number Space.
It is shown here as Init SeqNum Space enabled/disabled

Encoding the Flags and the SeqNum
R-TAG to be updated

— Explicit notification of the reset event is based on the flag included in the R-TAG, namely the "SeqResetFlag" (see former contribution on history slide).
— The usage of the new linear initial sequence number space ("InitSeqNumSpace") is noted via a new flag included in the R-TAG, namely the "InitSeqFlag".
— Sequence values of the new number space ("InitSeqNumSpace") are also included in the R-TAG (using the existing sequence number field).

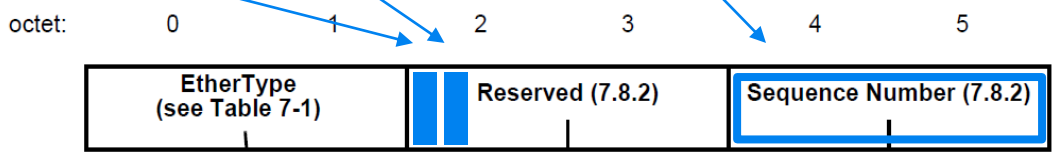
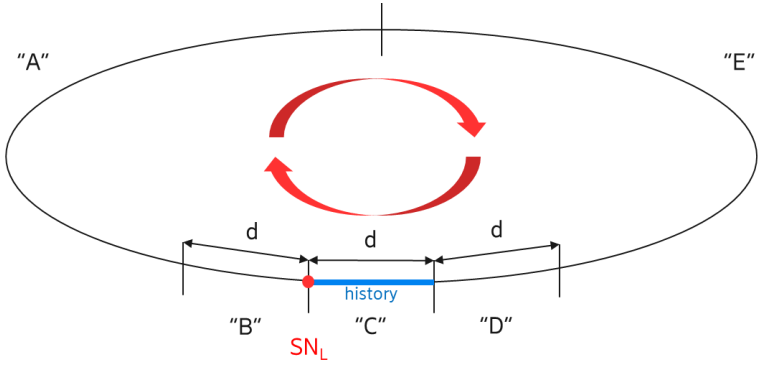


Figure 7-4 in IEEE 802.1CB

Evaluation of the new solution

Analysis of new method
Performance



- **Evaluation of the new method**:
- Using the "*SeqResetFlag*" and the "*InitSeqFlag*" flag in the received the Elimination node can use the appropriate Sequence Number Space and accept packets received after reset immediately:
  - A: no packet drop, SN_R is accepted,
  - B: no packet drop, SN_R is accepted,
  - C: no packet drop, SN_R is accepted,
  - D: no packet drop, SN_R is accepted,
  - E: no packet drop, SN_R is accepted.

- There is no packet drop, packets sent after the reset are immediately considered valid. TSN applications are not impacted at all and the implementation impact is moderate.
  Note: The assumption for the analysis is that there are no other events in the network, only the sequence generation function is reset once and no resets happen until the cyclic Sequence Number Space is reached.

Analysis of new method
Backward compatibility

— The new method is backward compatible.
— Implementation not prepared to deal with the new flags and the extended Sequence Number Space fallbacks to current operation and performance of 802.1CB-2017.

Impact on 802.1CB-2017

Target and Solutions at-a-glance
Avoid unnecessary drops due to reset of sequence generation

- Target: seamless reset of Sequence Generation function, i.e., avoid dropping frames unnecessarily due to replication function reset

- Solution: change Sequence Number Space characteristics
  - explicit notification of the reset event, and
  - extending the sequence number space (additional new linear initial sequence number space)

- **Impact: these improvement can ensure seamless adaptation to failure scenarios and protects again unnecessary packet drops.**

802.1CB impact
What needs to be changed/added …

- Sequence Generation function
  - send the new flag “SeqResetFlag”
  - send the new flag “InitSeqFlag”
  - use an extended Sequence Number Space

- Elimination function
  - interpret the new flag “SeqResetFlag”
  - interpret the new flag “InitSeqFlag”
  - use an extended Sequence Number Space

- New R-TAG fields to be defined

Questions …