Introduction to SF1

- Objectives
  - Reserve a track to a destination multiple hops away
  - Fulfill bandwidth and QoS (e.g., priority, time-critical) requirements
  - Hop-by-hop, distributed

- Combination of RSVP-TE and 6top protocol (6P)
  - RSVP-TE: end-to-end resource reservation
  - 6P: cell negotiation between two neighbors

- When to trigger SF1?
  - The Sender has an outgoing bandwidth requirement for a new instance to transmit data to Receiver.
  - The Sender has a new outgoing bandwidth requirement for an existing instance to transmit data to Receiver.
Step 1: The sender sends a PATH message downstream hop-by-hop to the receiver.
Step 2: The receiver initiates a 6P transaction to verify if there are enough cells to fulfill the requirements. If so, the receiver sends the RESV message to reserve these cells.
Step 3: Upon receiving a RESV message, an intermediate node reserves the cells between its upstream neighbor in the same way as in Step 2.
Step 4: When RESV arrives at the sender before end-to-end timeout, a track from sender to receiver is built.
PATH and RESV messages

\[\text{PATH Message} ::= \text{Common Header} \ [ \ \ldots \ ] \]
\[\text{SESSION} \ \text{<RSVP_HOP> }\]
\[\text{<TIME_VALUES>} \]
\[\ldots \]
\[\text{<LABEL_REQUEST> }\]
\[\ [ \text{<SF1 OPERATION REQUEST>} \ ] \]
\[\ [ \text{<6P OPERATION REQUEST>} \ ] \]
\[\ldots \]
\[\text{<sender descriptor> }\]
\[\text{<sender descriptor} ::= \text{<SENDER TEMPLATE>} \text{<SENDER_TSPEC> }\]
\[\ [ \text{<ADSPEC>} \ ]\]
\[\ [ \text{<RECORD_ROUTE>} \ ] \]

\[\text{RESV Message} ::= \text{Common Header} \ [ \ \ldots \ ] \]
\[\text{SESSION} \ \text{<RSVP_HOP> }\]
\[\text{<TIME_VALUES> }\]
\[\ldots \]
\[\ [ \text{<6P OPERATION>} \ ]\]
\[\ldots \]
\[\text{<STYLE>} \text{<flow descriptor list>}\]
\[\text{<flow descriptor list} ::= \text{<FF flow descriptor>} \]
\[\text{<FF flow descriptor} ::= \ [ \text{<FLOWSPEC>} \ ] \text{<FILTER_SPEC>}\]
\[\ [ \text{<LABEL>} \ [ \text{<RECORD_ROUTE>} \ ] \]

- PATH: describe the flow (SENDER_TSPEC) and collect path properties (ADSPEC).
- The three request objects: used to verify if the nodes along the route have the requested capabilities.
- The LABEL_REQUEST is set to TSC (timeslot switching capability).

- RESV: describe the bandwidth and QoS requirements (FLOWSPEC), assign label to the upstream node (LABEL).
- Parameters in FLOWSPEC are calculated according to SENDER_TSPEC and ADSPEC.
- In case of 3-step transaction, 6P confirmation is encapsulated in 6P OPERATION.
One-hop operation using 3-step trans. (updated)

Note that the 2-step trans. can also be used.

Node A: upstream node

- 6P Request with an empty CellList
  - Metadata: slotFrame_ID

Node B: downstream node

- 6P Response with candidate CellList
- RESV carrying 6P Confirmation with selected CellList

Map bandwidth to number of cells, QoS to constraints on cells, e.g., slotFrame, slotOffset.

Timeout

- Multiple 6P transactions can be attempted in sequence

Arrival of PATH (if Node B = receiver) / Arrival of RESV (otherwise)

Cells reserved

Label assigned

Cells reserved

Timeout

×
Node B State Transition Diagram

Node A: upstream node
Node B: downstream node

- **Current reservation**
- **Requirement Mapping**
- **Reservation fails**
- **Reservation succeeds**
- **Updated reservation**
- **Send ResvErr to Receiver**
- **Send RESV including 6P confirmation to Node A**
- **Send RESV including 6P confirmation to Node A**
- **Qualified slotframes found**
- **Qualified cells found**
- **Select cells from candidate CellList**
- **Initiate 6P transaction on a slotframe**
- **Check another qualified slotframe**
- **Backup slotframe exists**
- **No qualified slotframe**
- **No more slotframe**
- **Timeout**
- **Wait 6P response**
- **Send 6P request**

Arrival of **PATH** (if Node B = Receiver) / Arrival of **RESV** (otherwise)
TrackID, Label and Bundle

- **TrackID in SF1** (updated definition)
  - 16 bits identifier, assigned by the sender
  - Mapped from (source & destination IP address, RPLInstance)
  - Encapsulated in SENDERTEMPLATE of PATH and FILTER_SPEC of RESV

- **Label** (updated definition)
  - 32 bits, mapped from a bundle between two neighbors
  - Encapsulated in the LABEL object of RESV
  - Locally valid between two neighbors, assigned by the downstream node
  - Associated to a track

- **Bundle between two neighbors**
  - A group of equivalent scheduled cells (slotFrame_ID, CellList)

TrackID  Association  Label  Mapping  Bundle
Next steps

- Complete the definition
  - 6P request, SF1 request, 6P operation, Teardown message, etc.
- Mapping the traffic requirements to cells
- As suggested in 6top, the following requirements need to be covered
  - Error Handling (more detailed error code)
  - Specify the SF behavior of a node when it boots
  - Security considerations
  - Examples
- Implementation: simulation and hardware deployment
Thank you!
Track Forwarding

- Not in the scope of SF1, just for the completeness of the story
- A track can be seen as an LSP using bundles as implicit labels
- The sender identifies which track a packet should follow based on “sender/receiver IP address, RPLInstance”.
- Then G-MPLS is used
  - The sender pushes the first label to the packet.
  - The label is swapped at each intermediate node
  - The label is popped out at the penultimate Hop or at the receiver
  - At each hop, the packet is forwarded using the bundle associated to the label.

<table>
<thead>
<tr>
<th>Label 1</th>
<th>Label 2</th>
<th>Label 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>PUSH</td>
<td>SWAP</td>
<td>SWAP</td>
</tr>
<tr>
<td>Sender</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>