Node protection for SR-TE Paths

Shraddha Hegde
Chris Bowers
IETF-95

Topics

- Need for node protection
- Explicit paths in segment routing
- Solution
    - Building context tables
    - Node-sids, Adj-sids, Binding-sids protection
    - Operation in a failure scenario

Need for node protection

- High network resiliency needed for services

- Node can go out of service
  - Software crashes
  - Catastrophic events
  - Power failure

Explicit Routing in SR

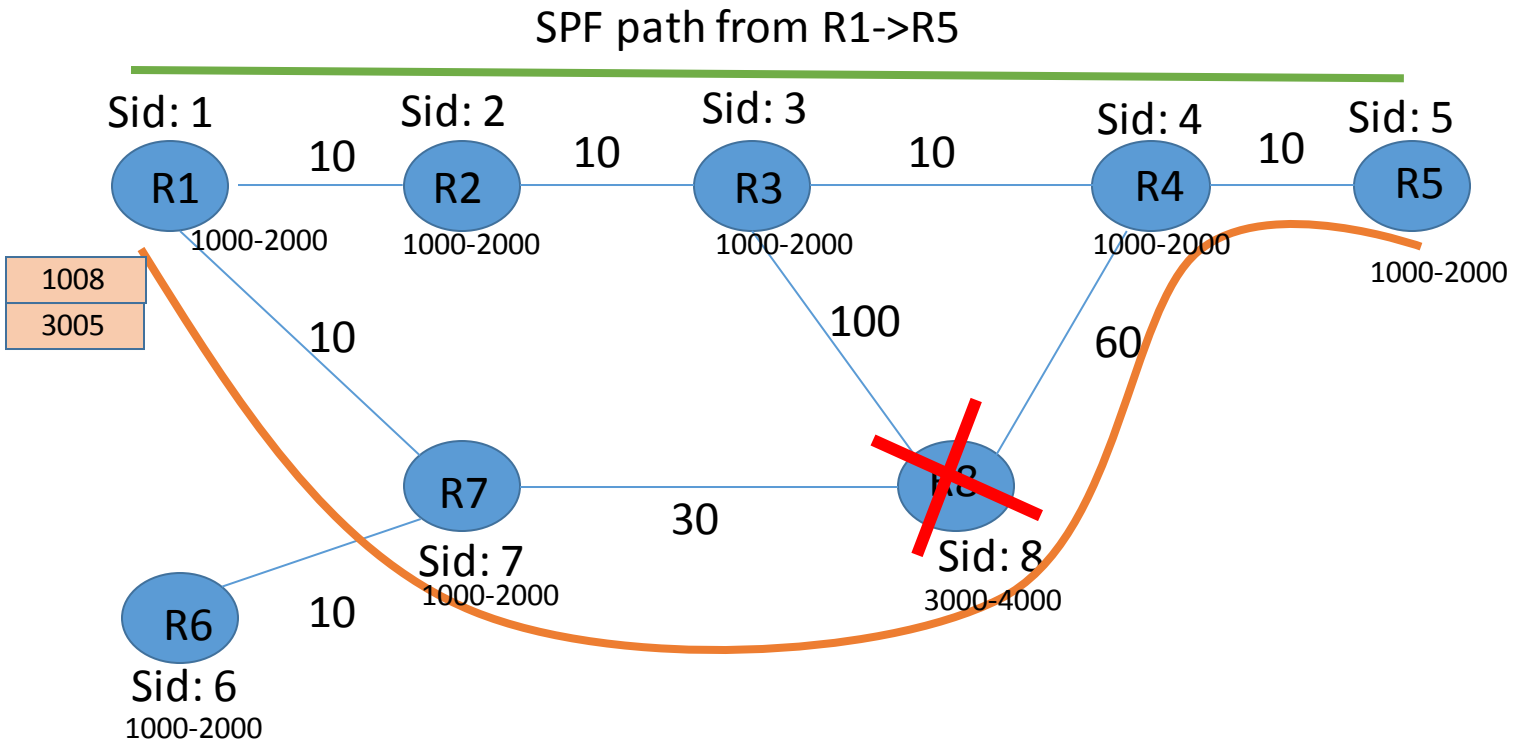
- Paths are described with a
  - list of adjacency segments
  - list of node segments
  - list of adjacency and node segments
  - list of binding sids and adjacency sids
- The midpoints do not maintain per-LSP state
- With explicitly routed LSPs using RSVP, a PLR can figure out the next-next-hop node by look at the RSVP signaling messages
- For SR explicit paths, the PLR needs to figure out the next-next-hop of a given LSP based on its label stack.

Explicit paths with node-sids

- SR explicit paths can be a set of node-sids describing the path.
- If a node described by one of the labels in the stack goes down, LFA procedures cannot protect the traffic
- The next label below the top label should be used to determine the protection path
- In case of different SRGBs across nodes, it's necessary for a PLR to

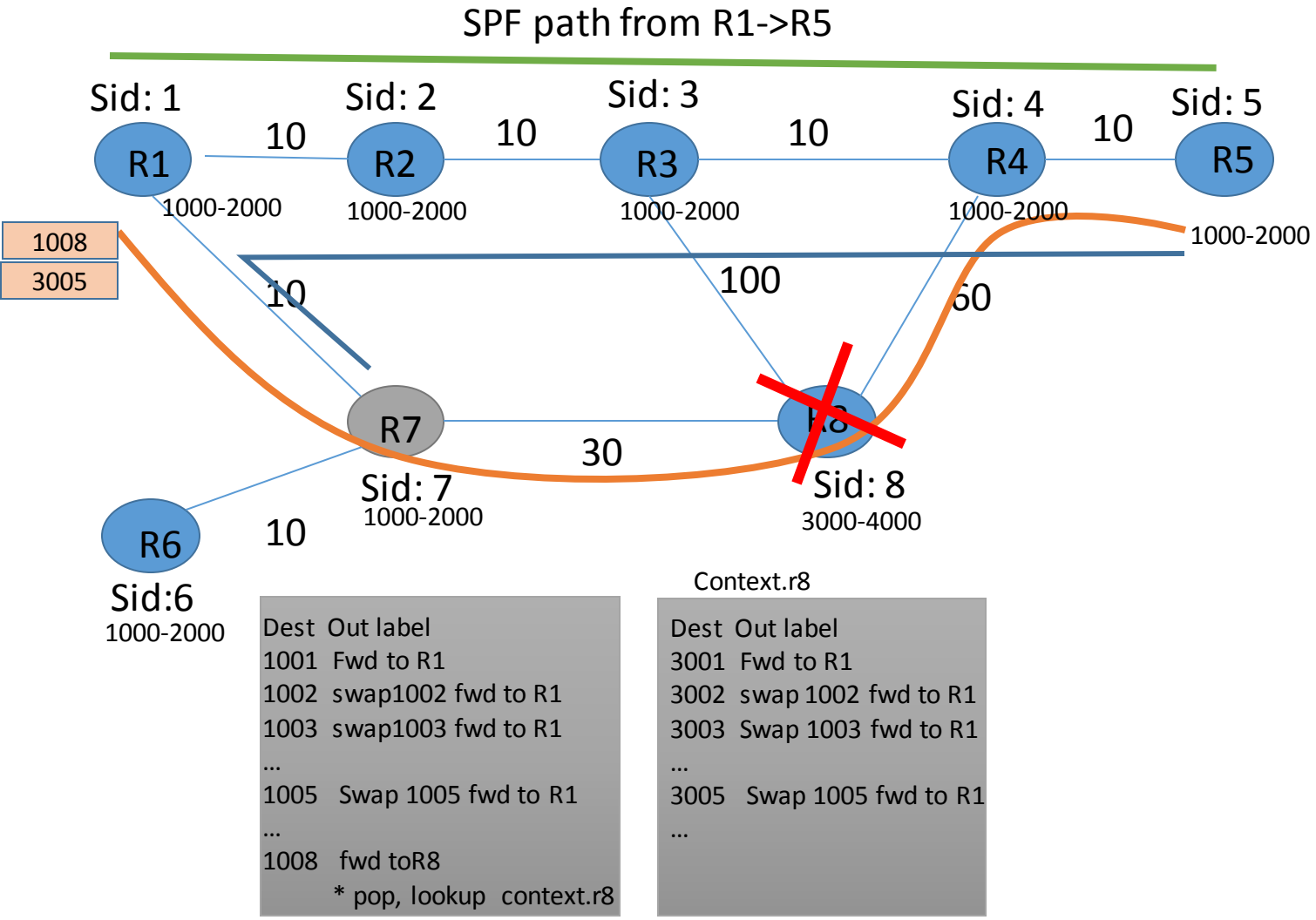
understand the next label in the stack, which must be interpreted in the context of the SRGB of the failed node

Explicit path using node-sid



- Explicit path from R1->R5 built using two label stack
- If R8 goes down, R7 drops the traffic since R7 cannot provide node protection for R8
- Need to look into the next label in the stack

Solution



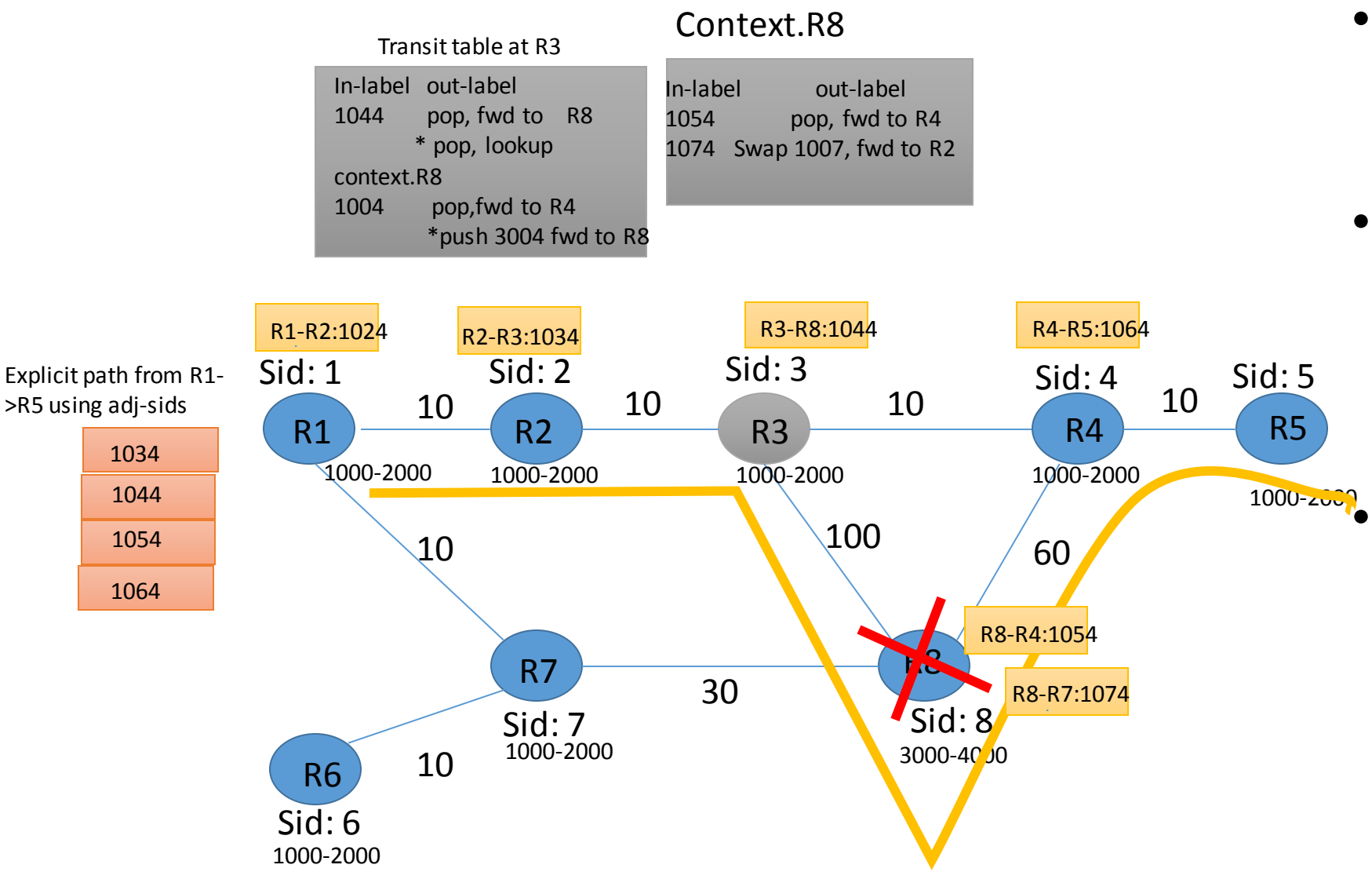
Context.r8

Dest	Out label
1001	Fwd to R1
1002	swap1002 fwd to R1
1003	swap1003 fwd to R1
…	
1005	Swap 1005 fwd to R1
…	
1008	fwd toR8
	* pop, lookup context.r8

Dest	Out label
3001	Fwd to R1
3002	swap 1002 fwd to R1
3003	Swap 1003 fwd to R1
…	
3005	Swap 1005 fwd to R1
…	

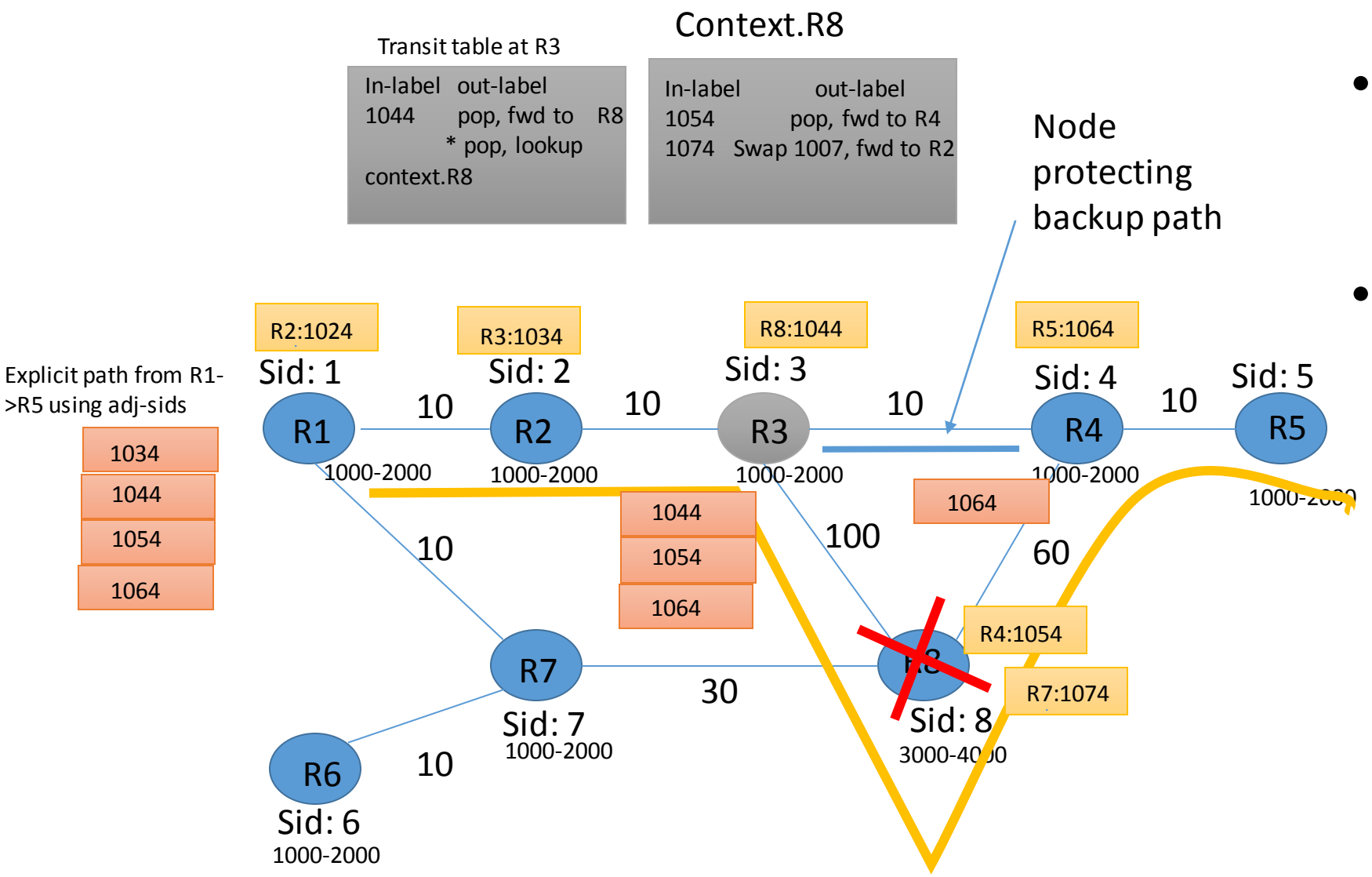
- Every node builds a context table for its neighbors
- The context table contains in-labels as per the SRGB of the neighbor
- The next-hop is built by looking into the SPF and Backup SPF computations of R7 for R5
- All the loop free paths (including primary & backup) are examined and the path that avoids protected neighbor is picked and installed in context table.

Solution for adj-sids



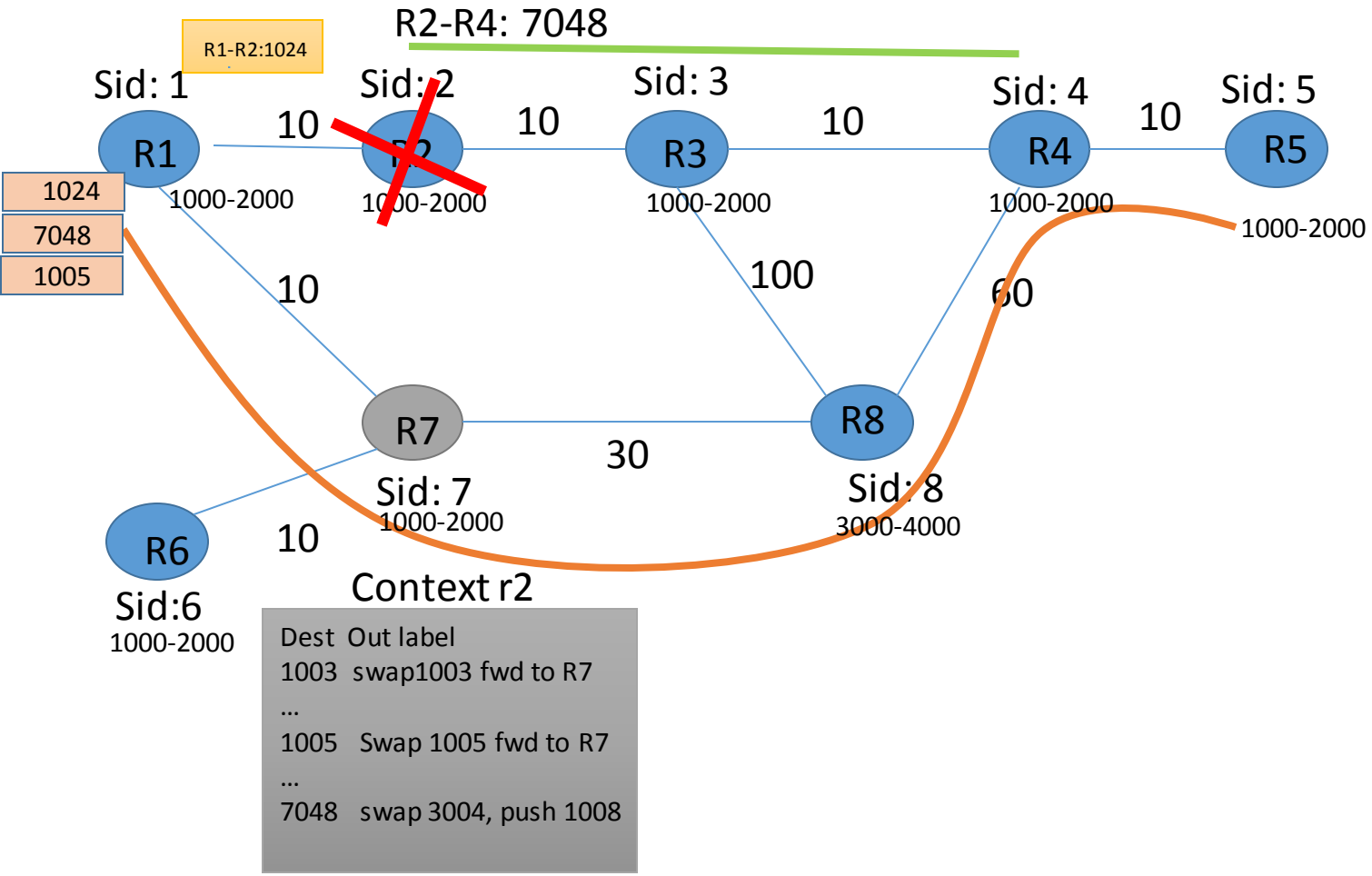
- Every node builds a context tables for its neighbors
- The context table contains adj-sids as advertised by the protected neighbor
- The next-hop in the context table is built by looking into SPF and Backup SPF computations for the end point represented by the label
- All the loop free paths (including primary & backup) are examined and the path that avoids protected neighbor is picked and installed in context table.

Operation on failure



- The backup path for label 1054 results into a context table lookup
- The context table of R8 is looked up for the next label in the stack
- The actions specified for the in-label 1054 are performed

Binding Sids



- Binding SIDs may be used to represent sub paths
- The backup nexthop for the remote end point represented by binding sid is built.
- All the loop free paths (including primary & backup) are examined and the path that avoids protected neighbor is picked and installed as backup nexthop.

Questions & Comments

THANKS