



Root initiated routing state in RPL

draft-ietf-roll-dao-projection

Pascal Thubert, Rahul Arvind Jadhav, Michael Richardson

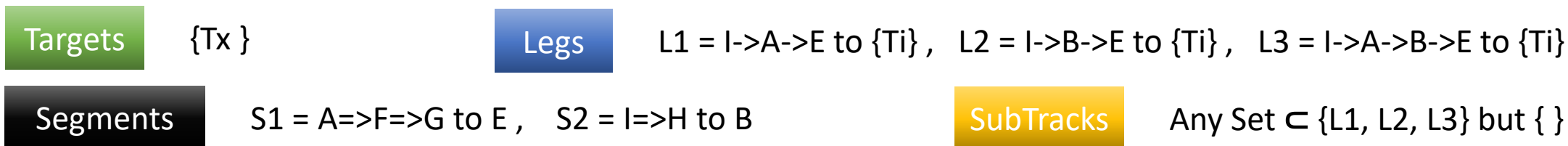
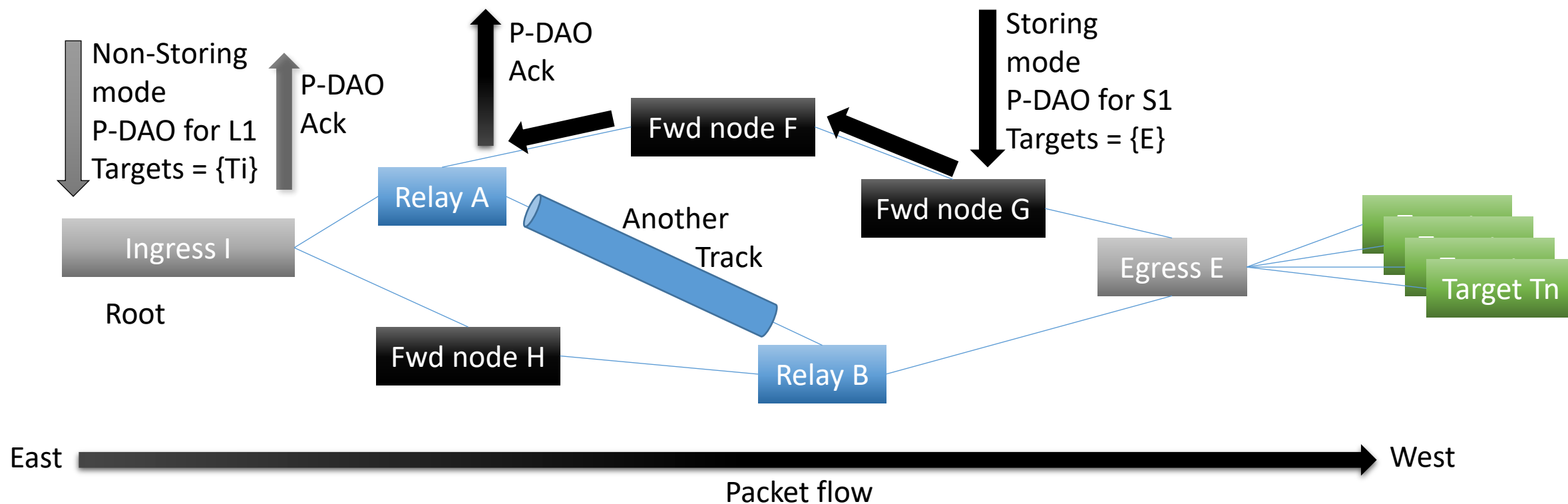
IETF 115, London

Presenter: Pascal Thubert

DAO Projection (Centralized RPL)

- Root connected-to or acting-as controller
 - Uses topological info from main DODAG
 - New Sibling Information Option (and P-DAO request)
 - Uses Projected DAO to install paths in the network
- Builds Segments to compress SHR
 - Compresses selected long paths in main DODAG
 - Uses Storing Mode Projected DAO to install strict (serial) paths
- Builds new DODAGs called Tracks
 - Enables optimized P2P (east – west) routing
 - Uses Non-Storing Mode Projected DAO to install loose (dotted-line) graphs
 - Leveraging Segments to complete the graph

The RPL Track: A DODAG rooted at Ingress



Draft Status

- WGLC at -26
- All known issues addressed at current (-29)
- Ready for publication

Status of the draft (cont.)

- 28 (as promised at IETF 114) refine on WGLC issues:
 - Clarify that each instance implies a RIB
 - Multi-topology routing loop avoidance rules:
 - **neighbor > indirect via common neighbor > Segment > Track**
 - **Partial order between RPL instances to allow jumping**
 - Crossing Segments discussion
 - Clarifying mcast DAO exposes neighbors in SIOs

Status of the draft (cont.)

-> 29 Clean up:

- Remove duplicated text in intro
- Lower case “main” in “main DODAG”

Next

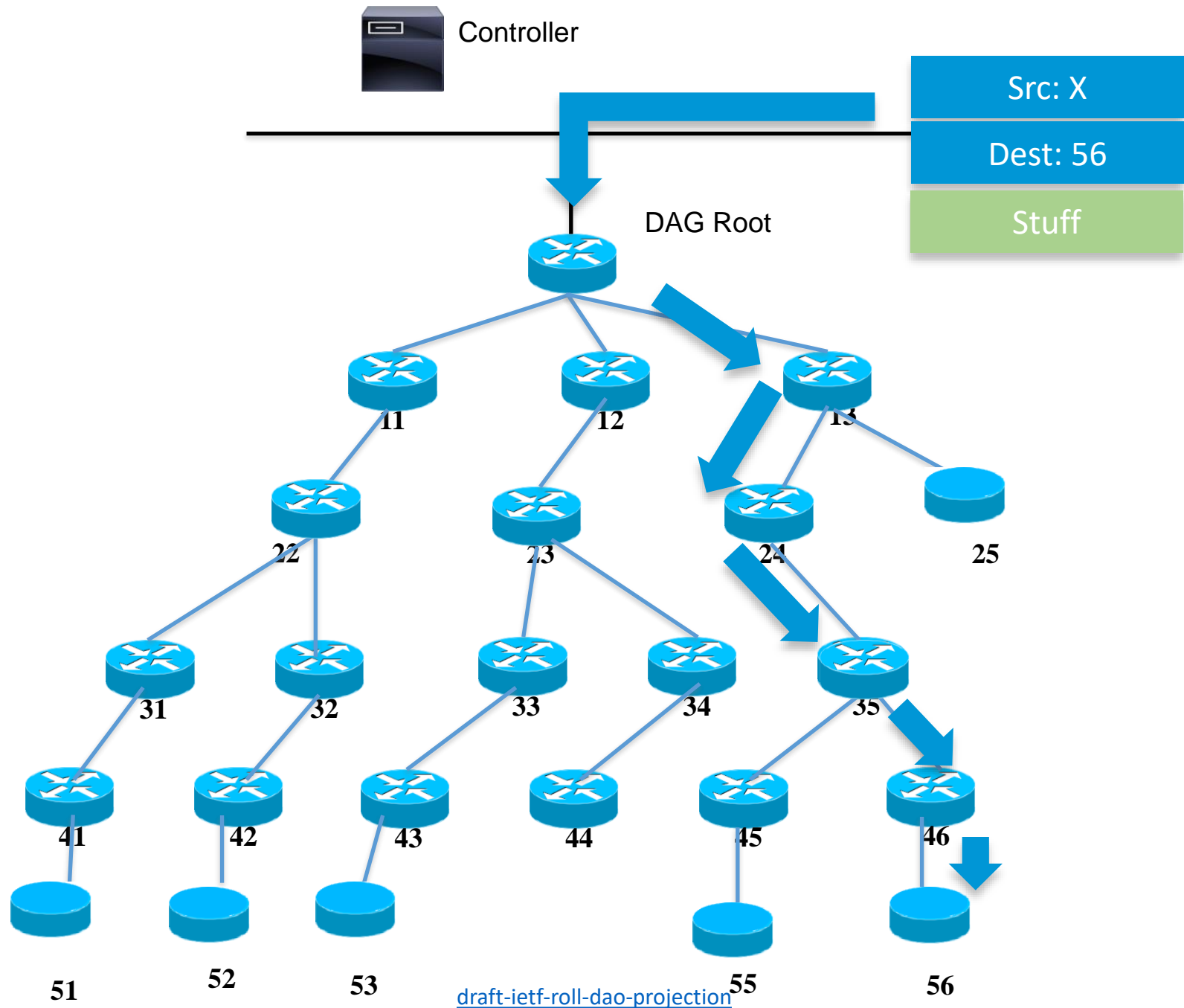
- Publication request?

DAO Projection

Backup Slides

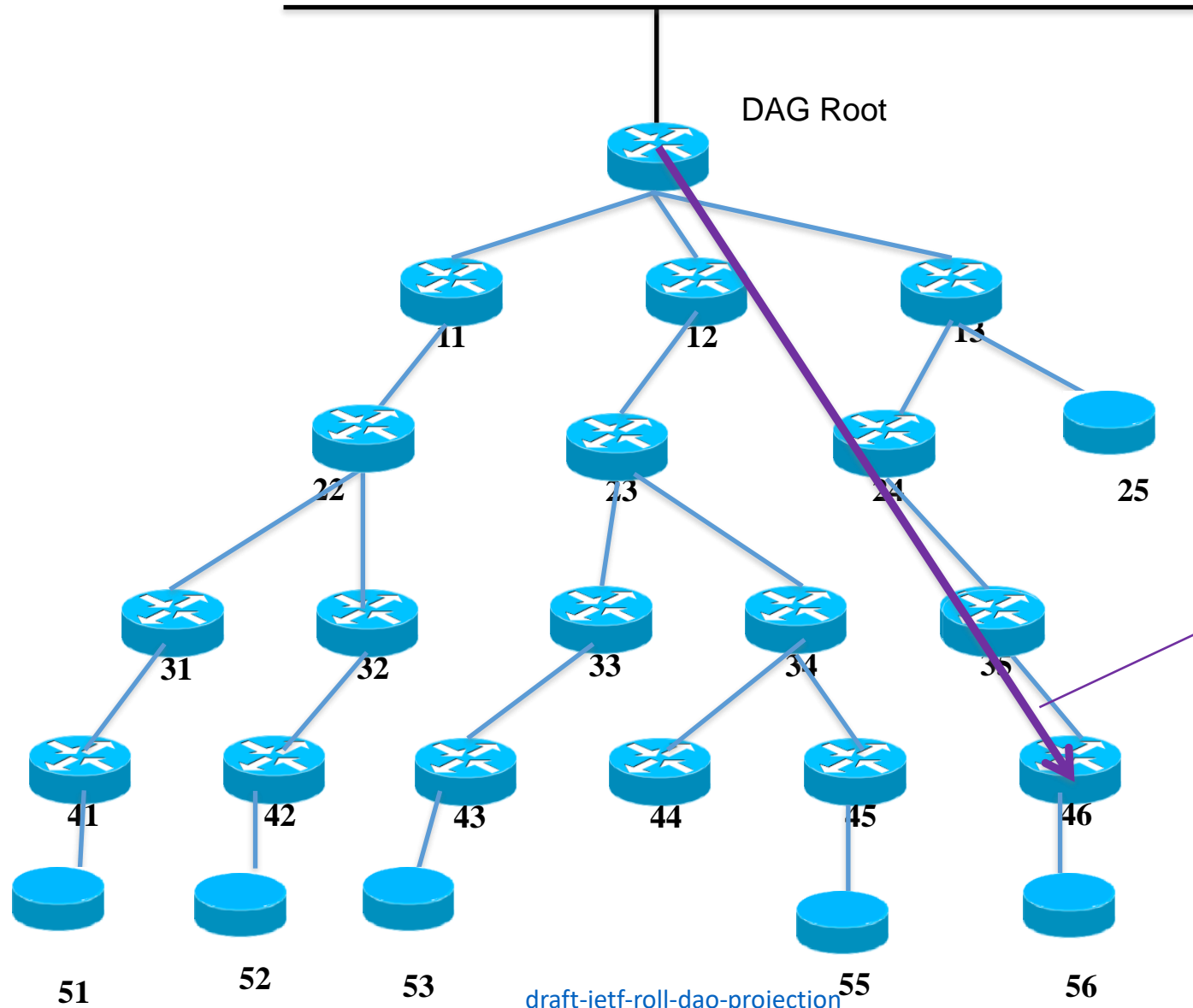
DAO Projection (Centralized RPL)

- Root connected-to or acting-as controller
 - Uses topological info from main DODAG
 - New Sibling Information Option (and P-DAO request)
 - Uses Projected DAO to install paths in the network
- Builds Segments to compress SHR
 - Compresses selected long paths in main DODAG
 - Uses Storing Mode Projected DAO to install strict (serial) paths
- Builds new DODAGs called Tracks
 - Enables optimized P2P (east – west) routing
 - Uses Non-Storing Mode Projected DAO to install loose (dotted-line) graphs
 - Leveraging Segments to complete the graph





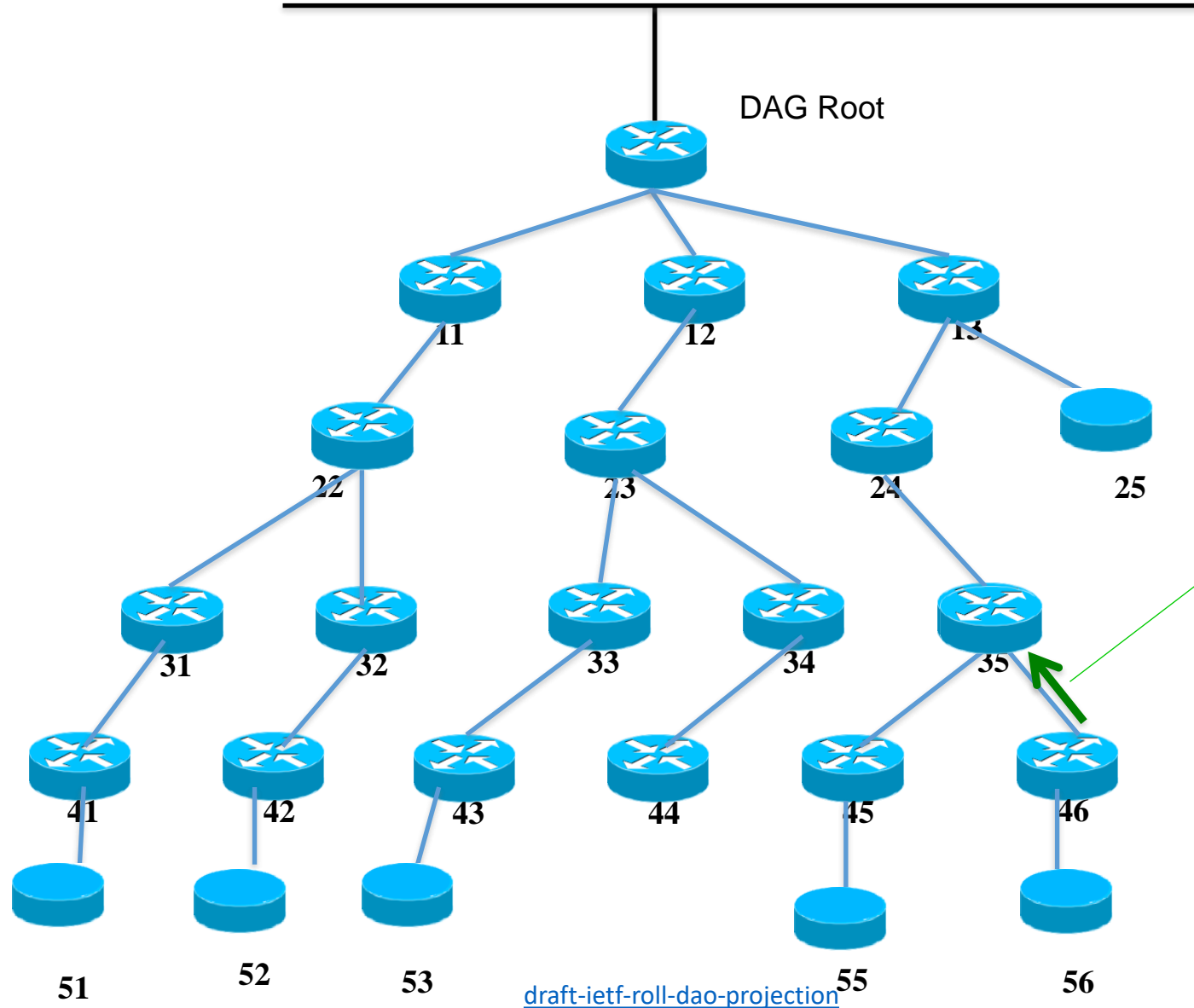
Controller



Projected-DAO to target 56 with path segment via 24 (ingress), 35, and then 46 (egress)



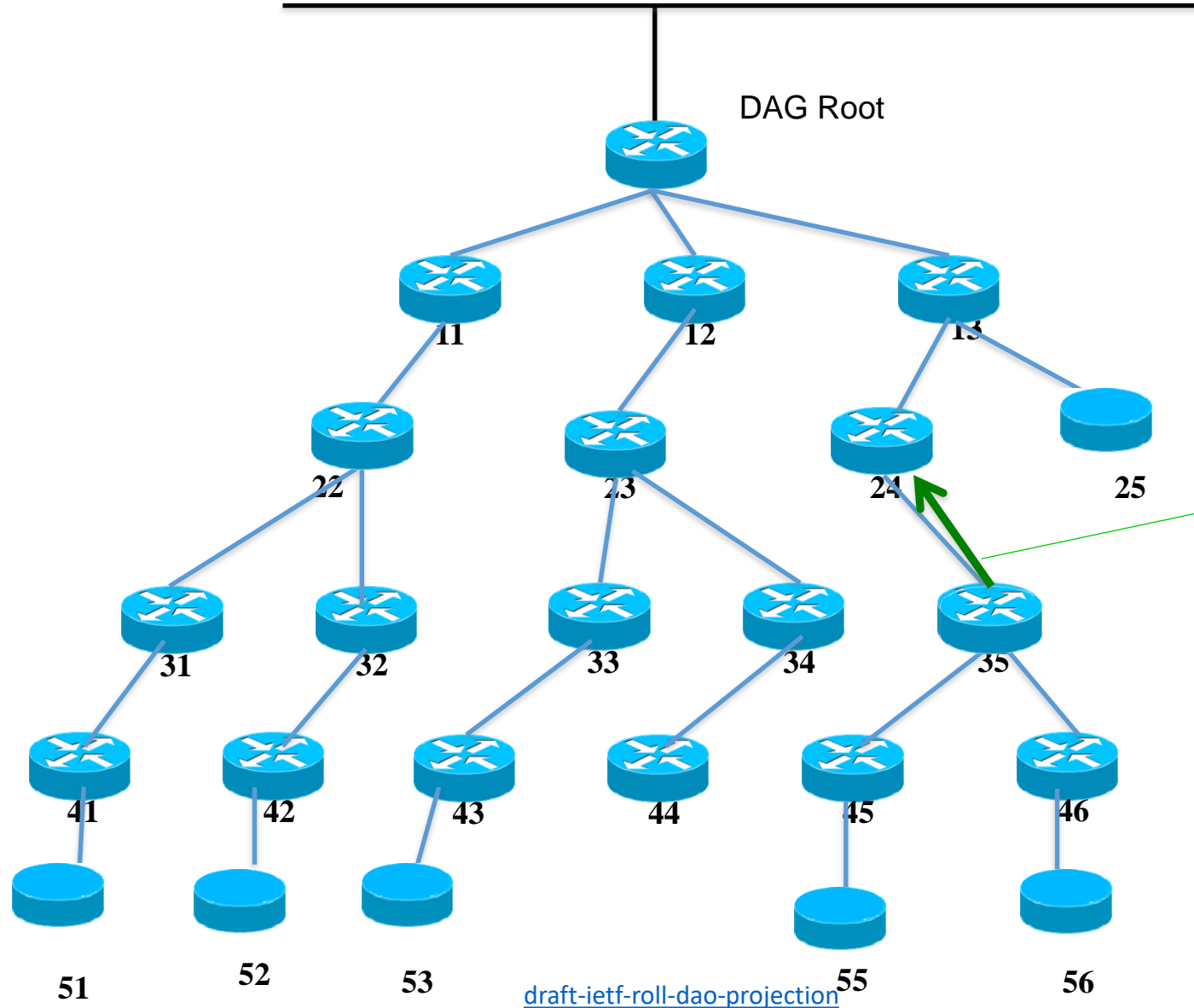
Controller



Storing mode DAO
to 56 upwards
segment
(24, 35, 46)



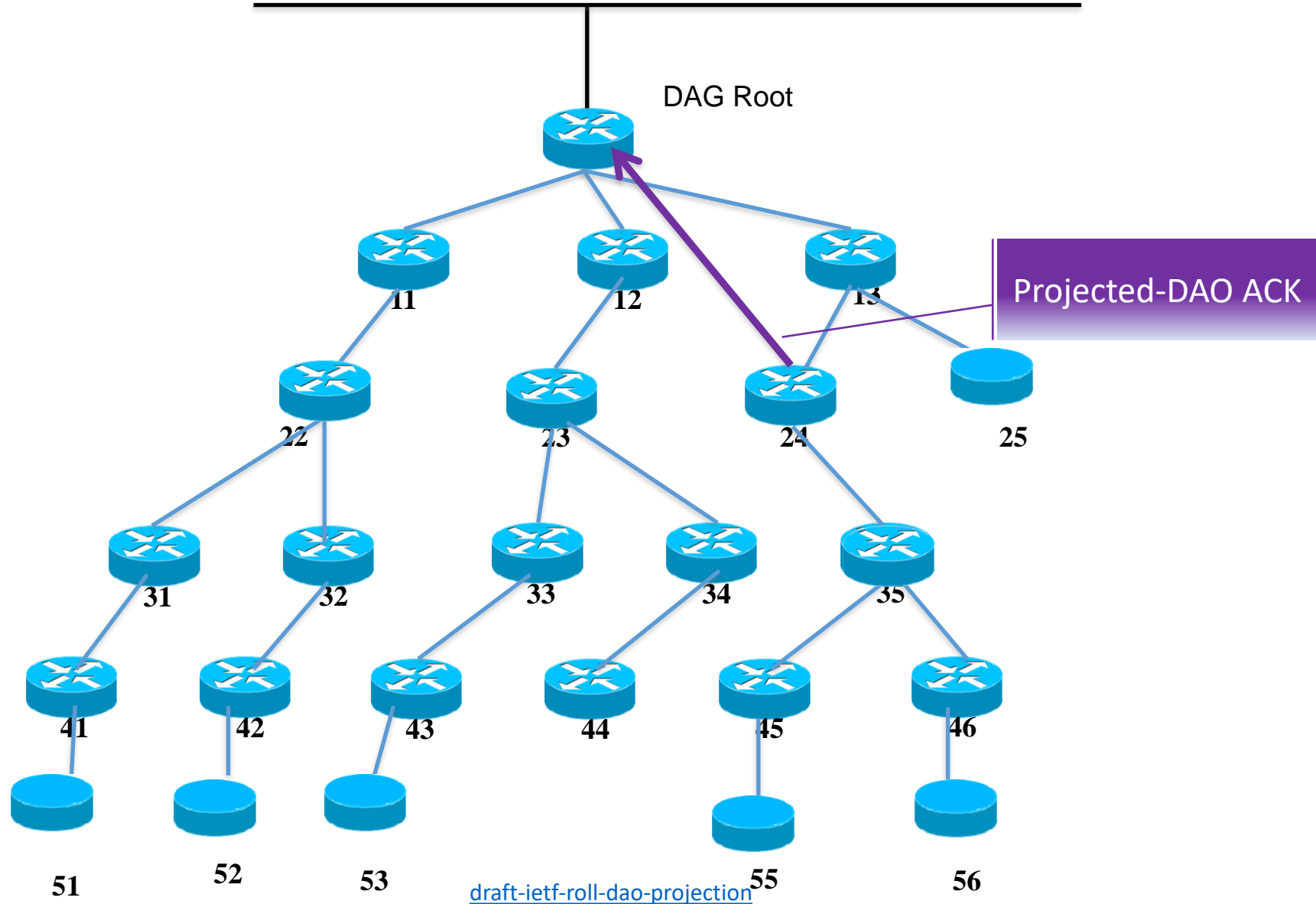
Controller



Storing mode DAO
to 56 upwards
segment
(24, 35, 46)

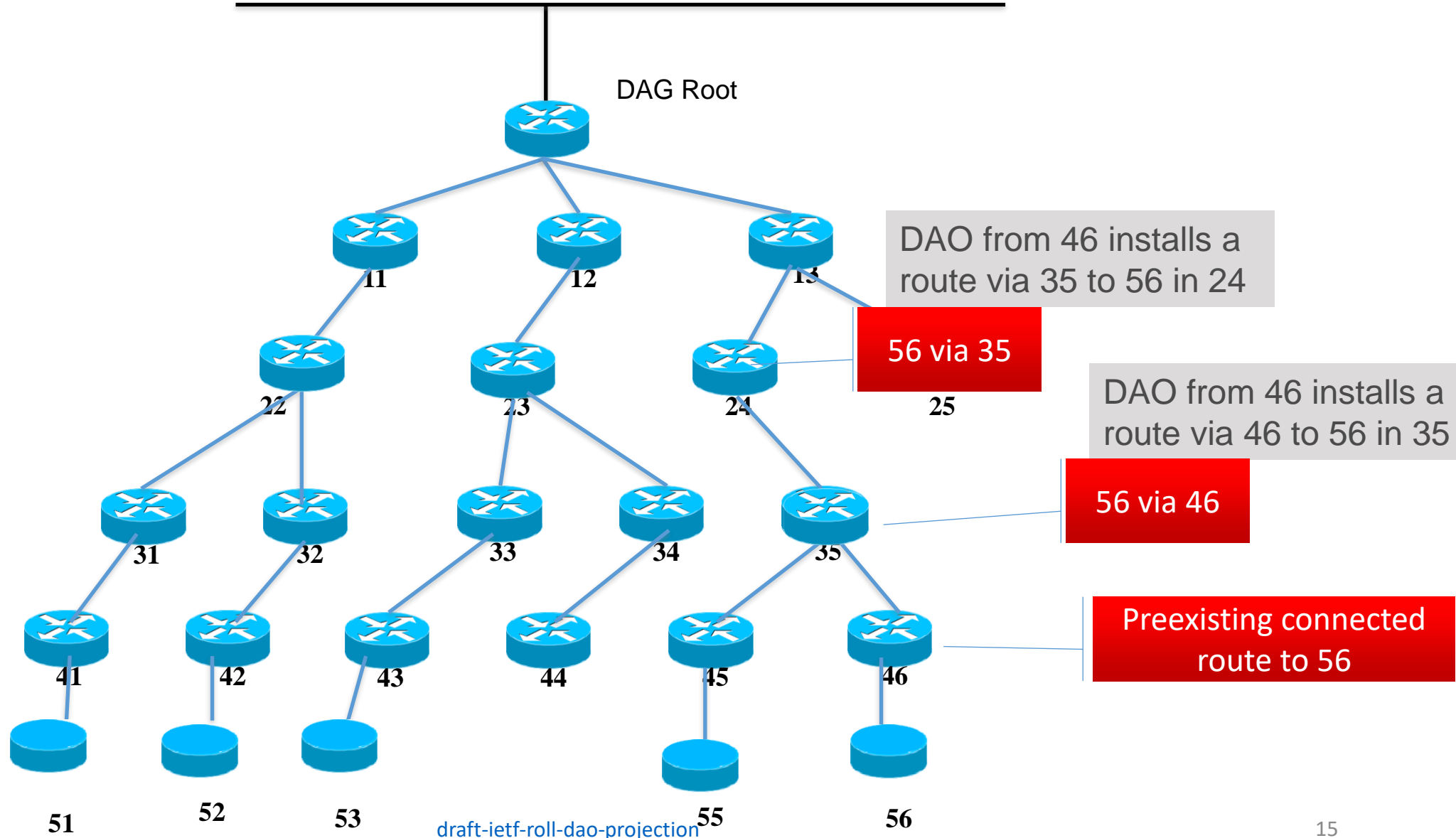


Controller



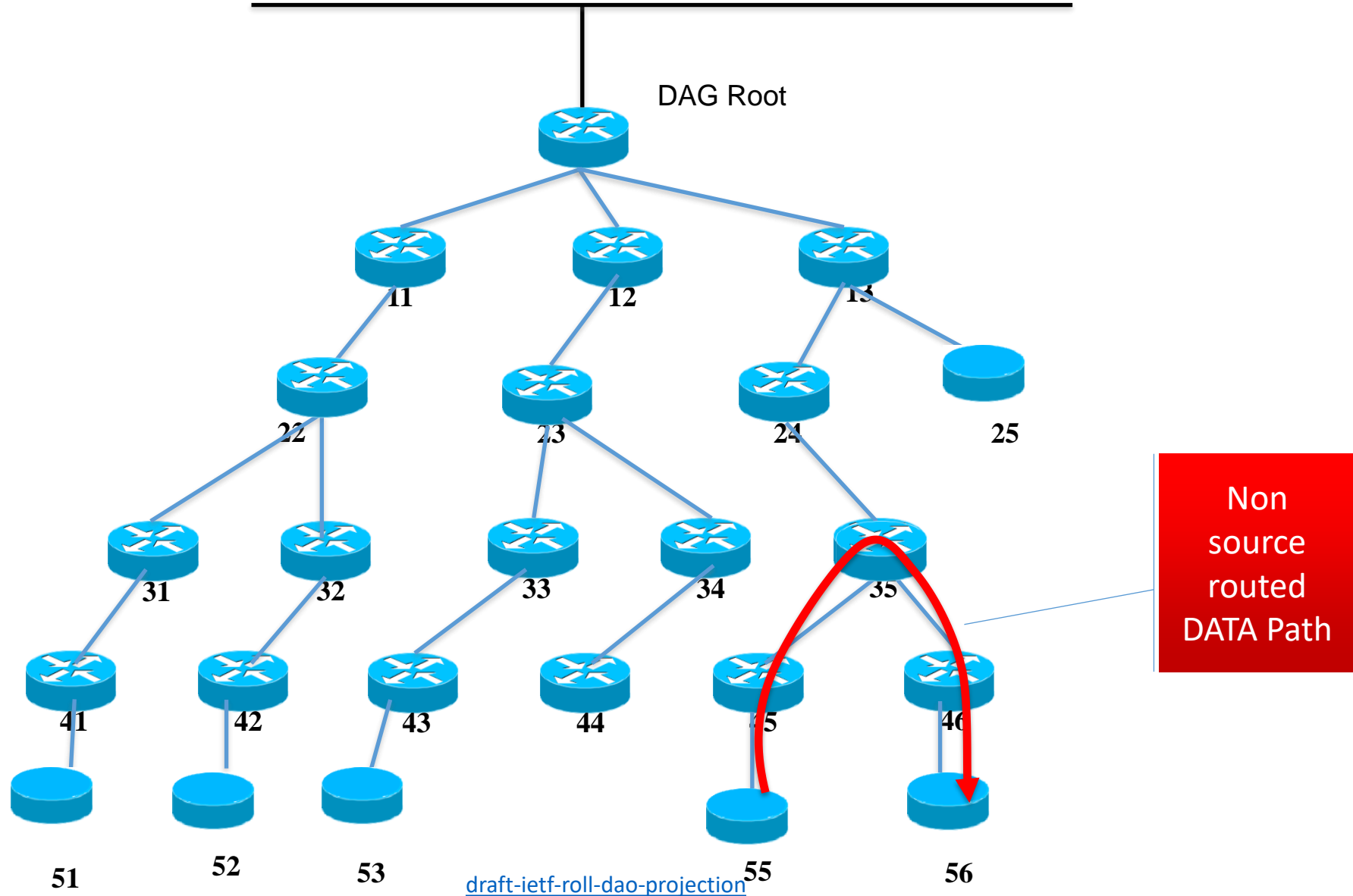


Application Server D



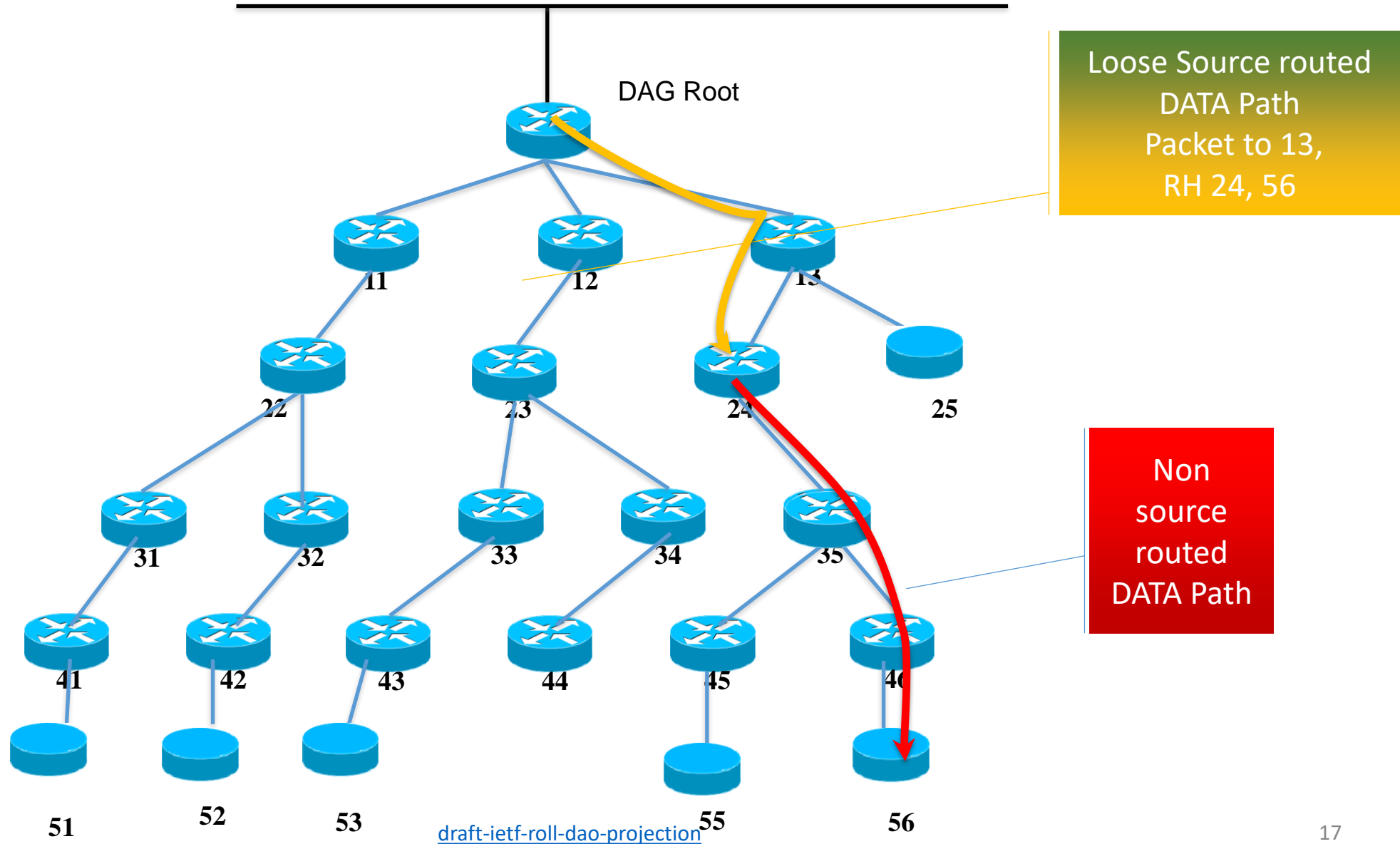


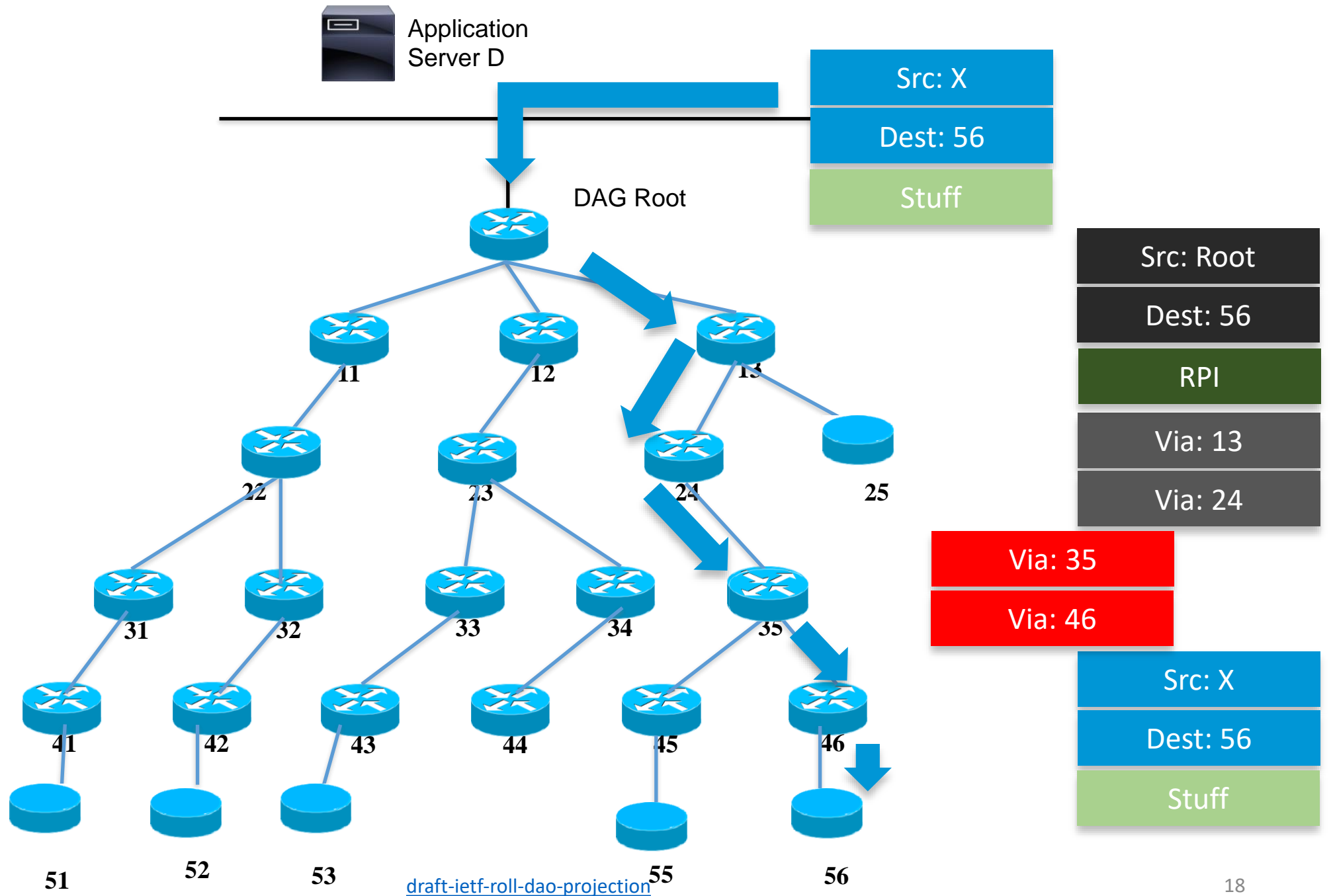
Controller





Application Server D

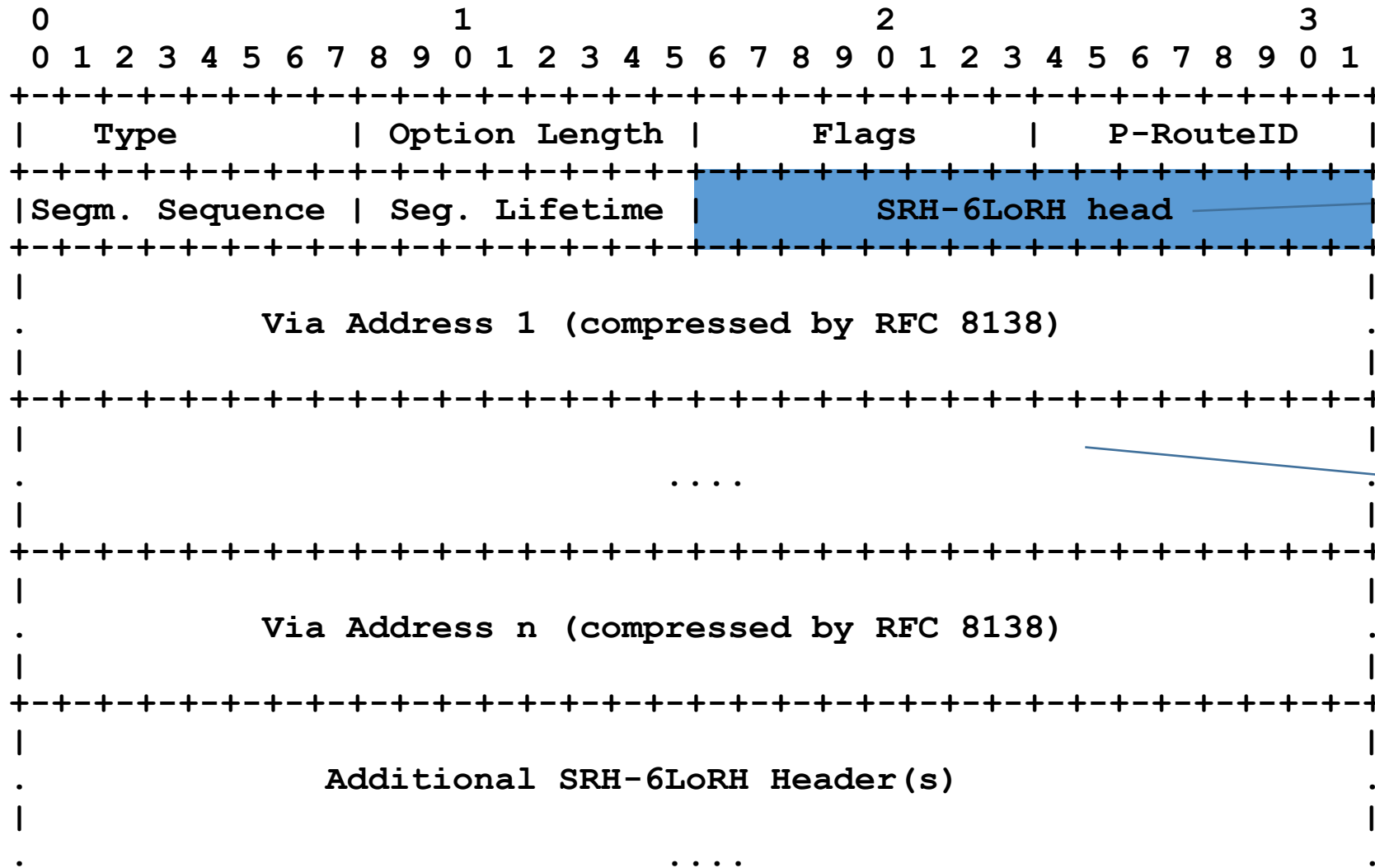




P-DAO construction

- RPL Target Options can be factorized
- But there is one and only one VIO (SF-VIO or SR-VIO)
- So the Ack management is easier
- VIO sent to egress; SR-VIO sent to ingress
- Track ID is a RPL local instance ID
- Taken from the Track Egress Name Space

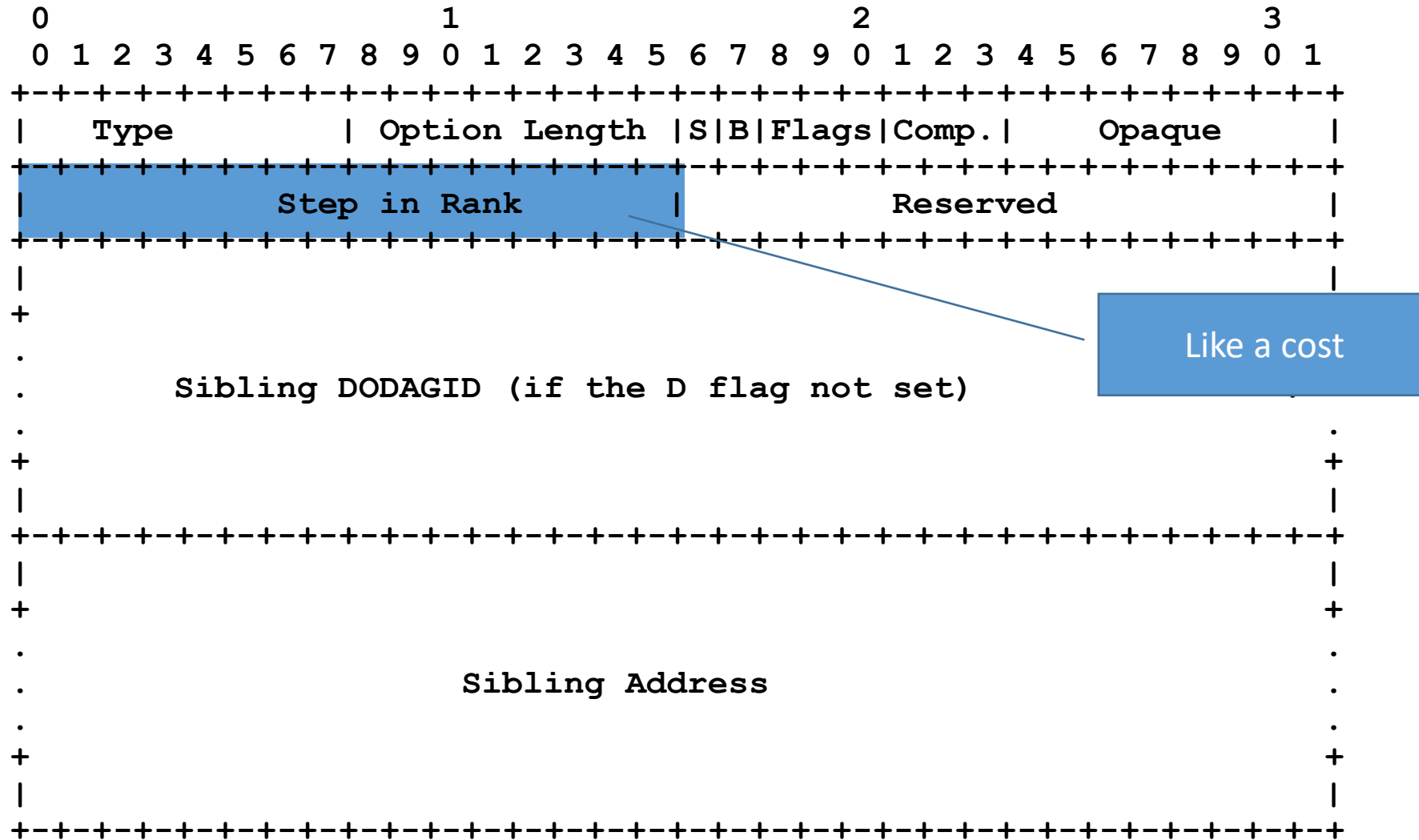
New Via Information Option Format



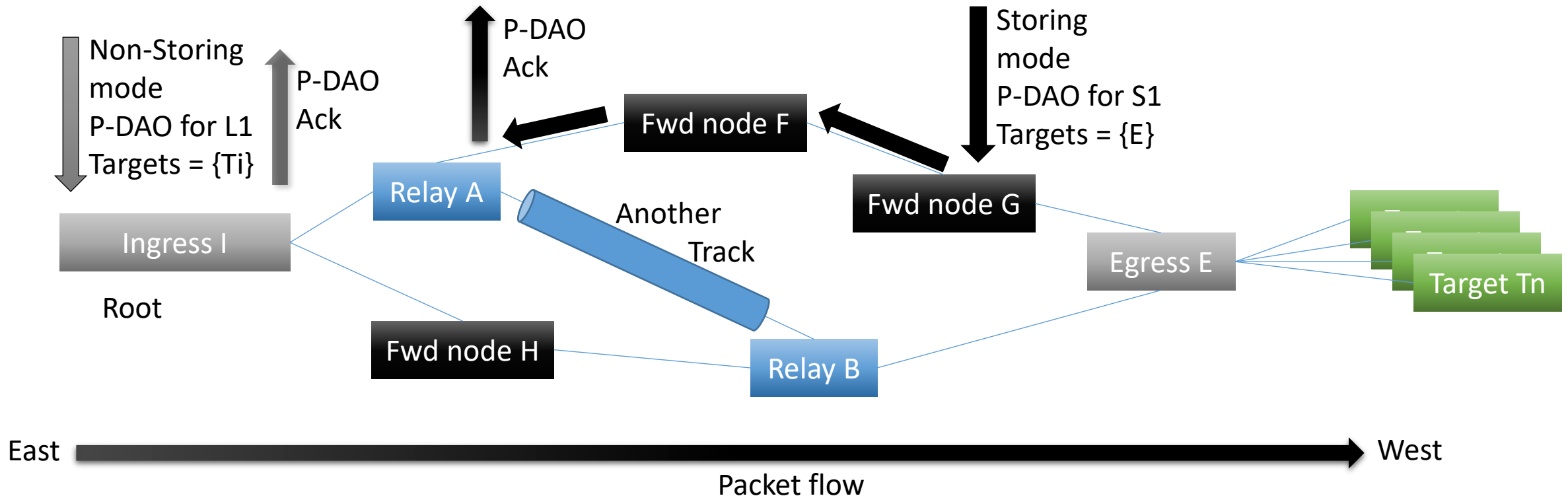
May be more than one in Non-storing Mode

Must be optimized in Non-storing Mode, to be used as is in packets

New Sibling Information Option Format



The RPL Track: A local DODAG rooted at Ingress



Targets {Tx }

Legs

L1 = I->A->E to {Ti}, L2 = I->B->E to {Ti}, L3 = I->A->B->E to {Ti}

Segments

S1 = A=>F=>G to E, S2 = I=>H to B

SubTracks

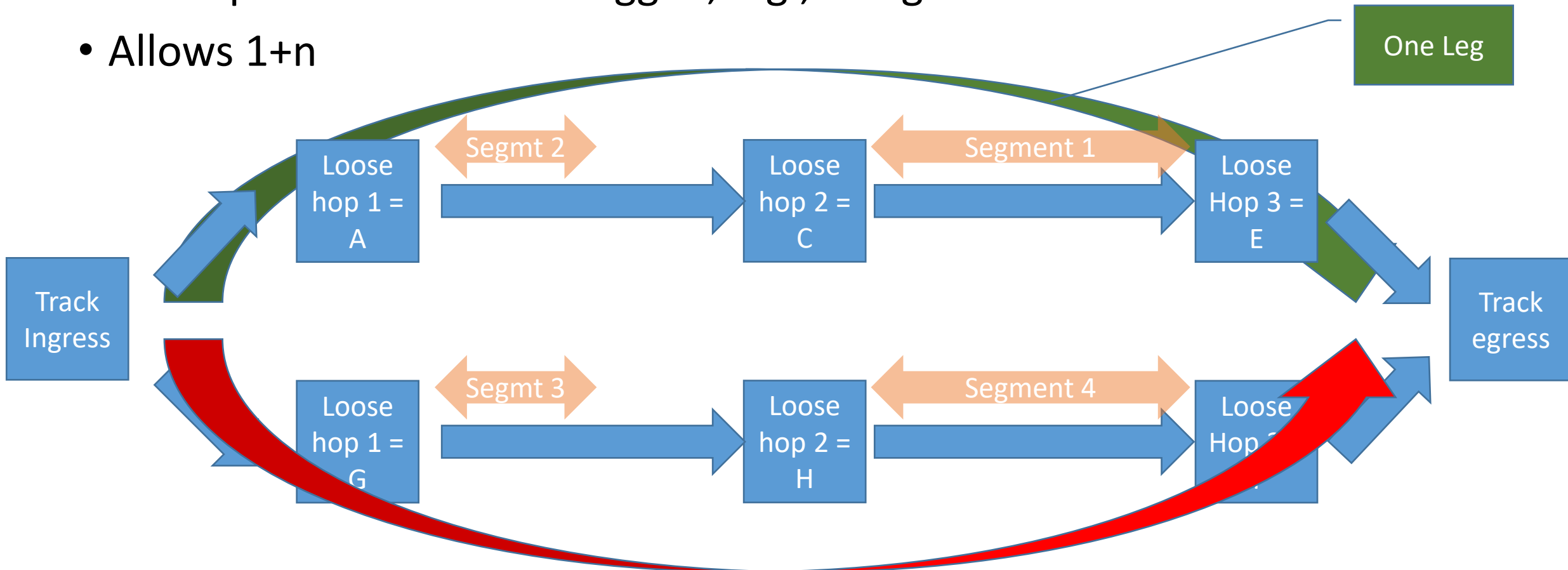
Any Set \subset {L1, L2, L3} but { }

Some rules

- Track is set up by installing Legs and Segment
 - with the same Track ID
- Non-Storing Mode P-DAO signals a Leg
- Storing Mode P-DAO signals a Segment
- Storing Mode P-DAO enables loose hops
 - in Non-Storing main DODAG (typically TrackId is Global instance ID)
 - in Tracks (typically TrackId is Local instance ID to track Ingress)
- Track Egress is implicit Target in Non-Storing Mode
- Leg hop is either a Segment of this Track or another Track

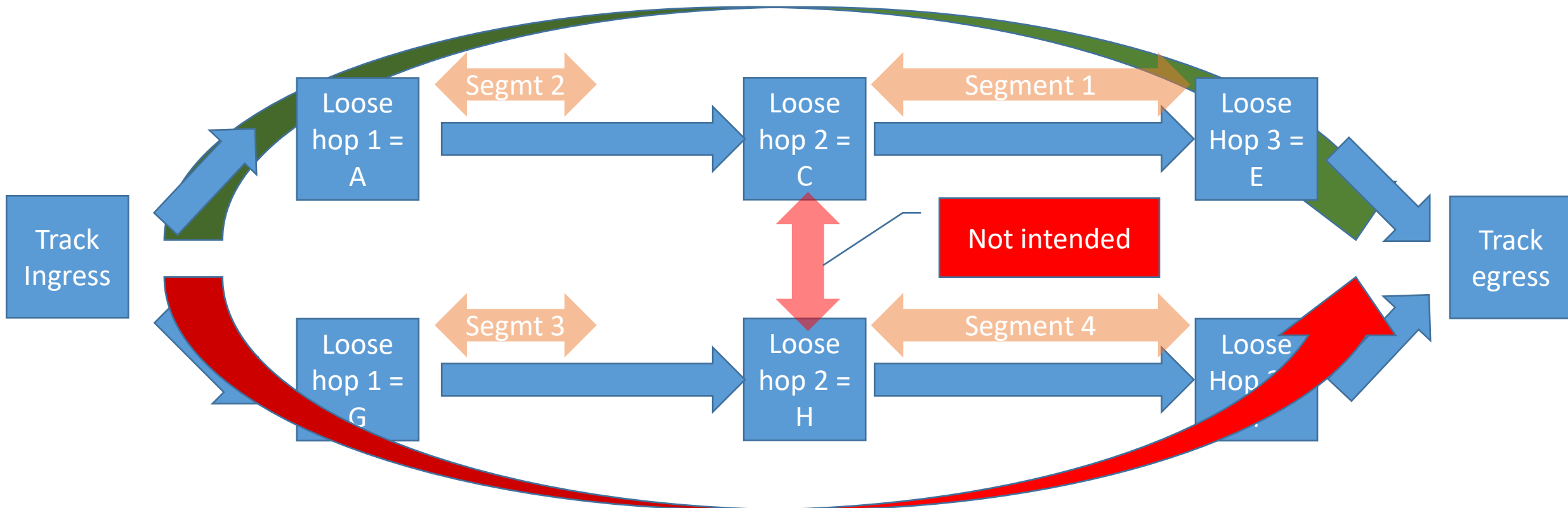
Complex track

- A complex track is multi-legged, e.g., 2 Legs below
- Allows 1+n



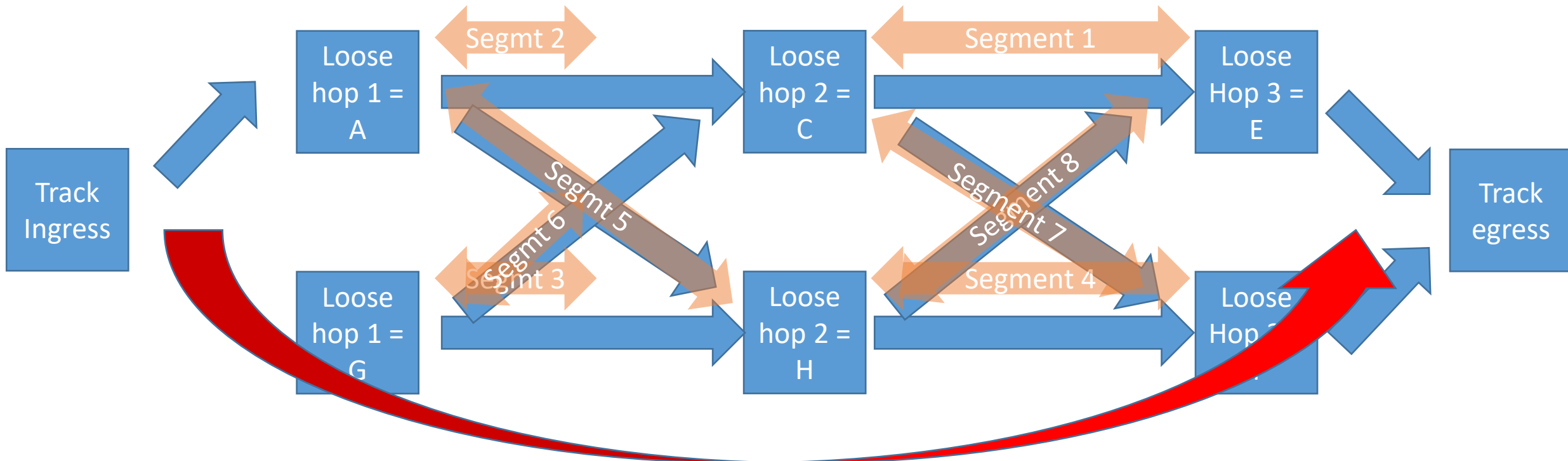
RPL vs RAW

- RPL has no North-South Segment



Inter Leg

- RFC 6550 non-storing Target and Transit to indicate loose parent child relationship, many of them in one P-DAO

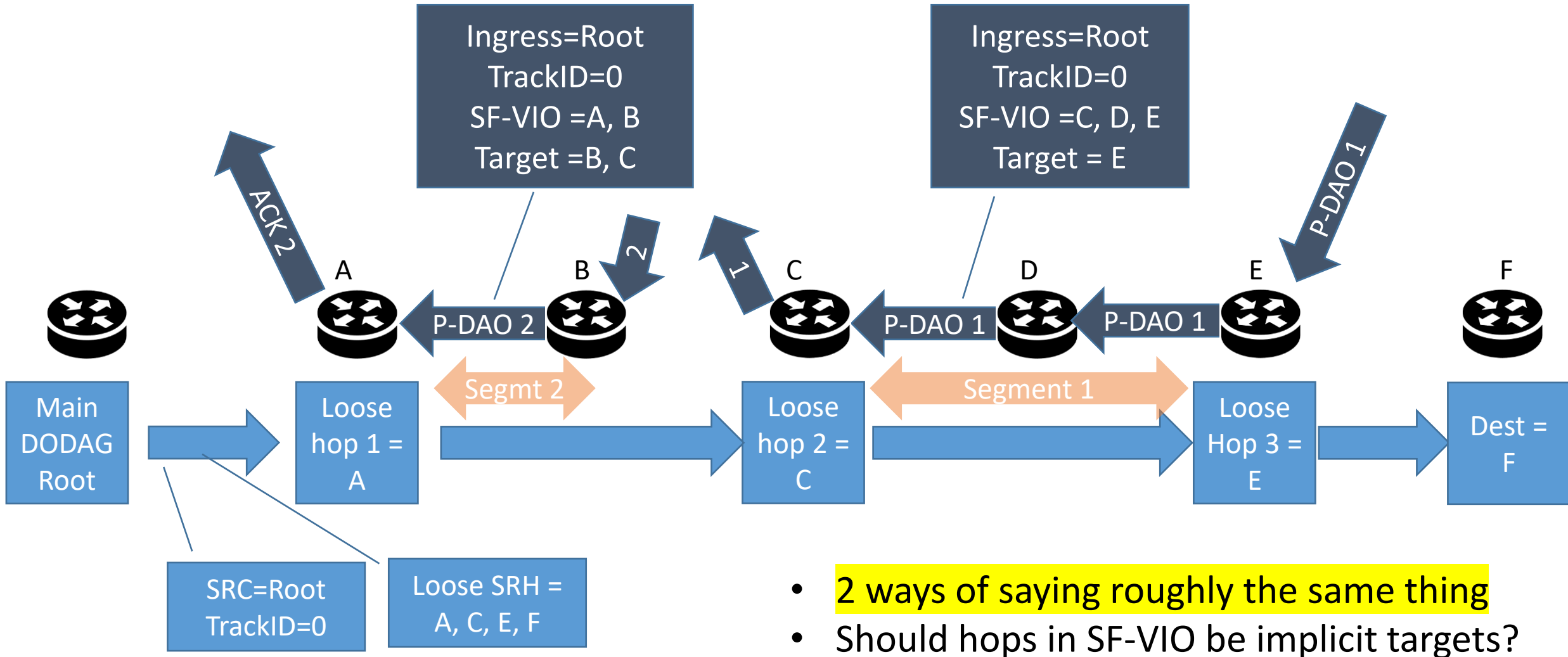


Encapsulation Details

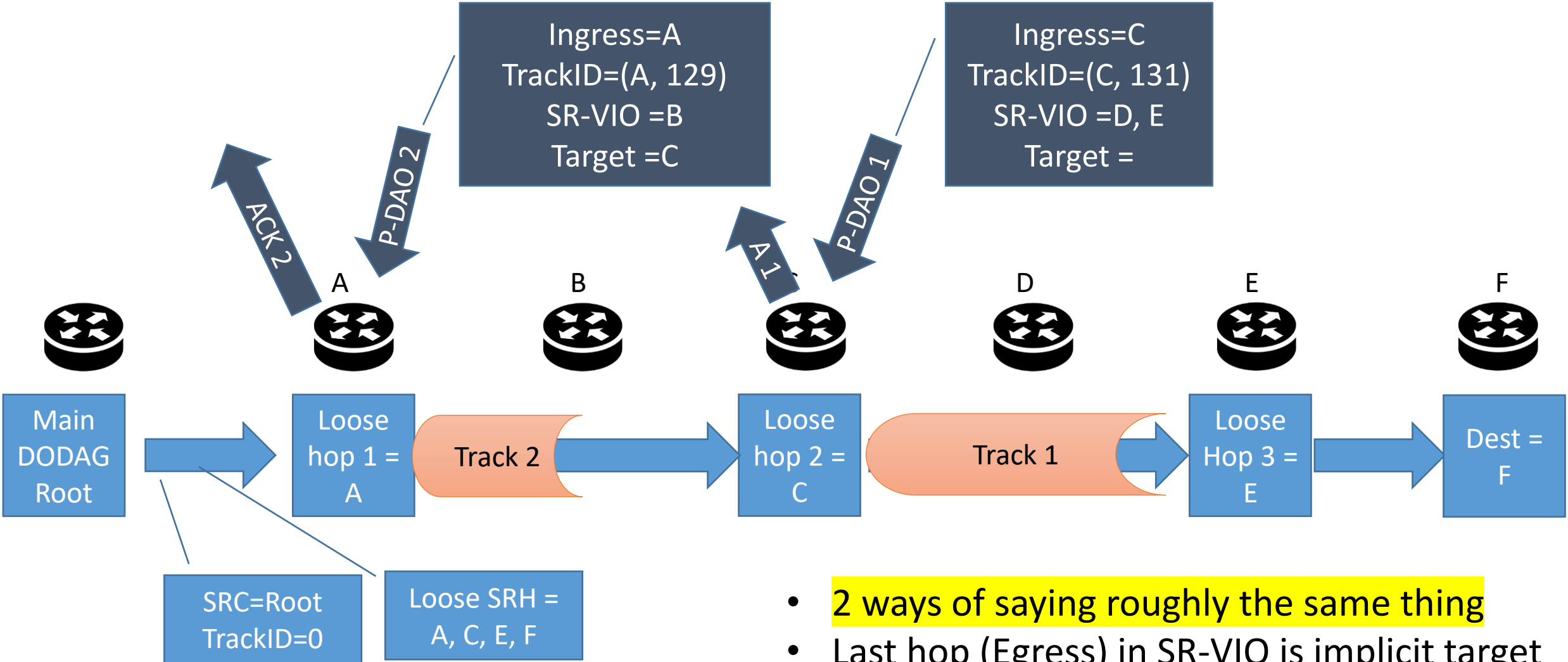
- Source of outer header MUST be Track Ingress- think DODAG Root
- RPL Instance ID in RPI MUST indicate TrackID (if not main DODAG)
- SR-VIO: Loose from Track Ingress, excluded, to Egress, included
 - Copied Verbatim in inserted SRH-6LoRH,
 - Requires encapsulation (can be recursive)
- SF-VIO: Strict from Segment Ingress to Egress, both included
 - No Encapsulation if Source and RPI both match Segment definition
 - A Segment is an Implicit Track if P-DAO Ingress == 1st SF-VIO entry
- TBD: matching rules, Flow Info option, when to tunnel?

Profile 1:

Compress SRH in main DODAG with strict SM Segments

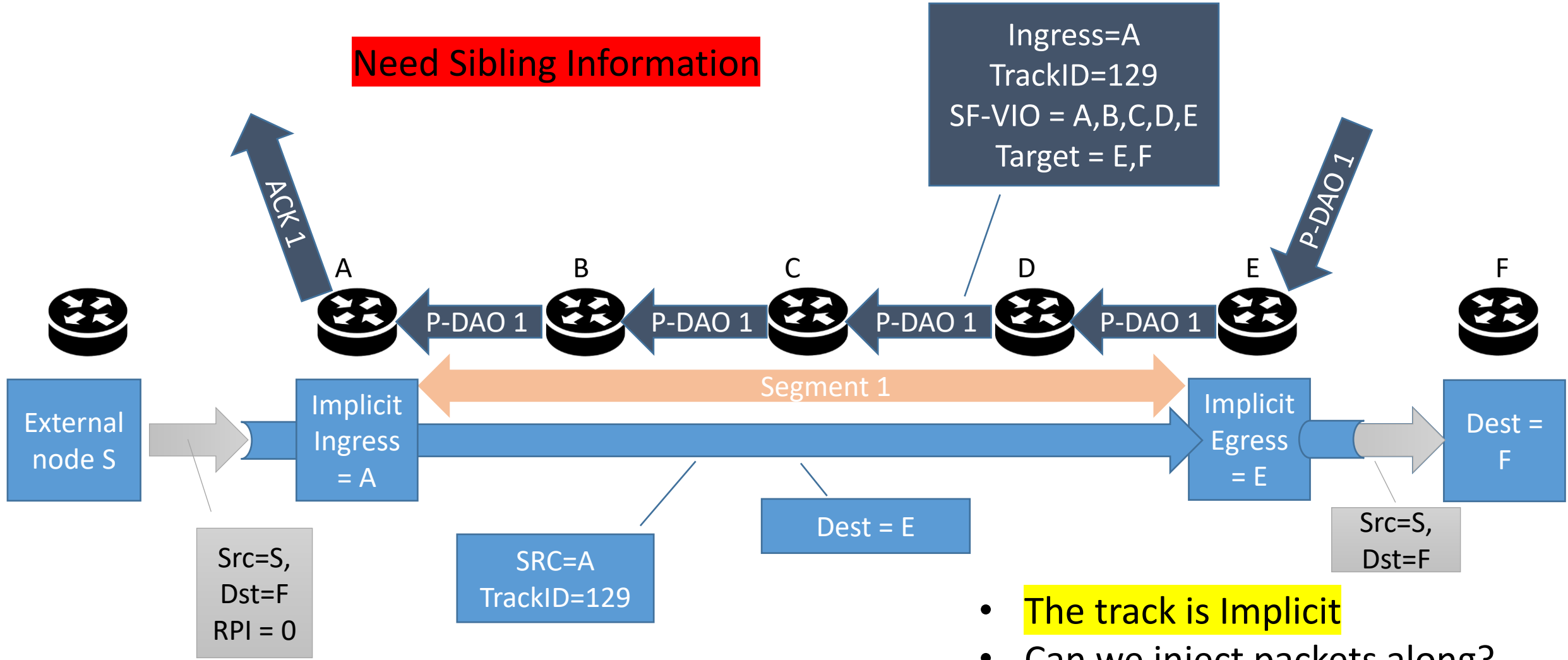


Profile 2: Compress SRH in main DODAG with Strict NSM Tracks



- 2 ways of saying roughly the same thing
- Last hop (Egress) in SR-VIO is implicit target

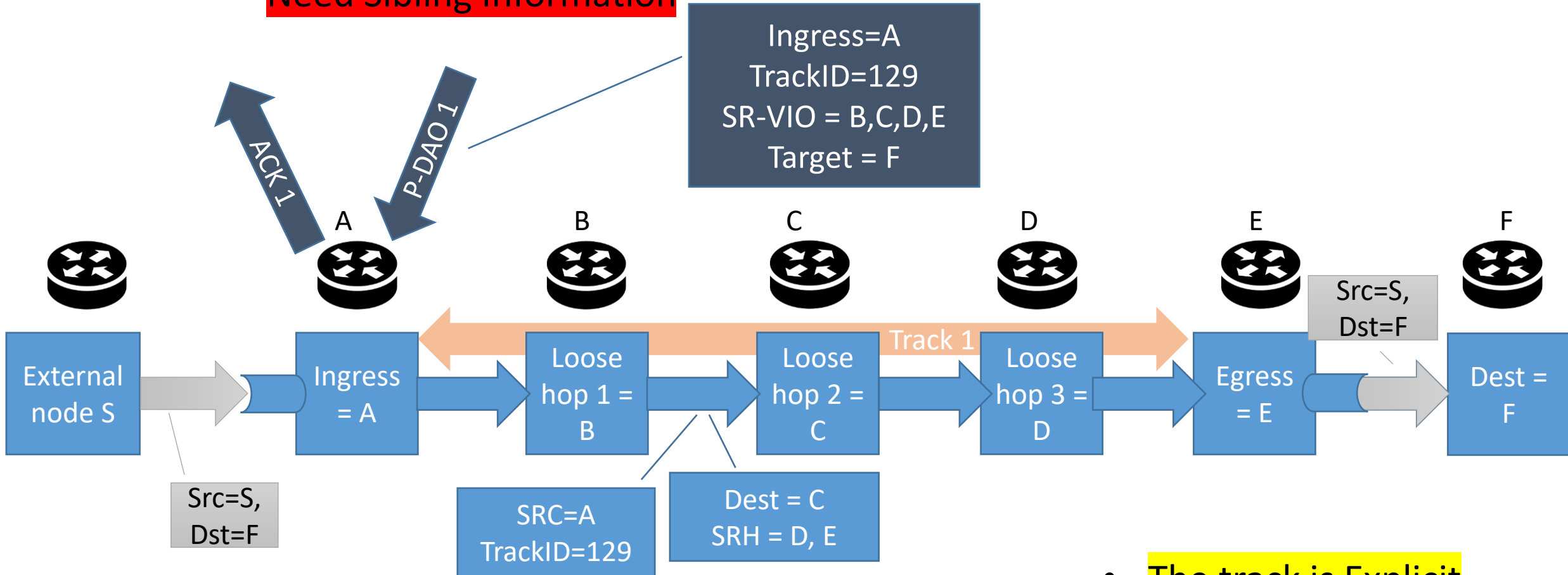
Profile 3: Implicit Track with Strict SM Segments,



- The track is Implicit
- Can we inject packets along?

Profile 4: Strict NSM Explicit Track

Need Sibling Information

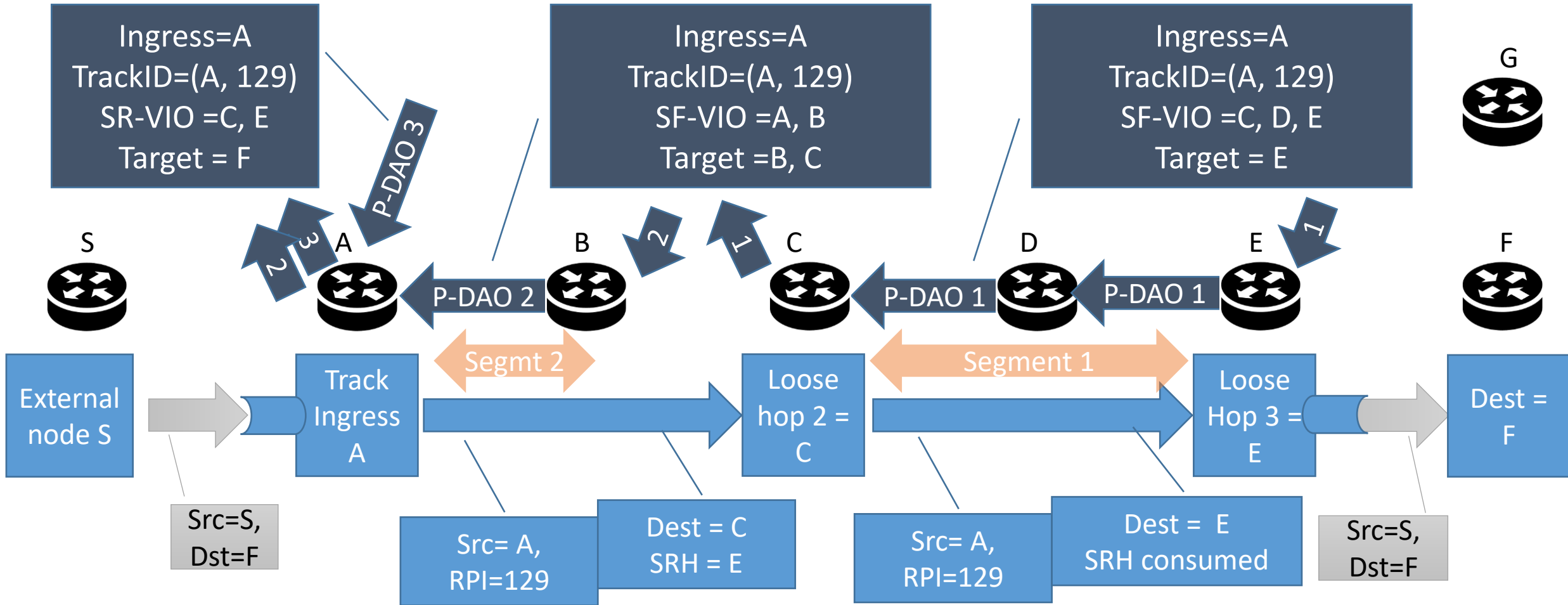


- The track is Explicit
- Same encap as profile 2

Profile 5:

Need Sibling Information

Compress SRH in Track with Strict SM Segments



- Same as Profile 1, but for Track

Profile 6:

Compress SRH in Track with NSM Tracks (Recursive?)

