Monday 03/25 - 16:10-18:10

IETF 104 ROLL

Routing over Low-Power And Lossy Networks

Chairs:
Peter van der Stok
Ines Robles

Secretary:
Michael Richardson

Note Well

This is a reminder of IETF policies in effect on various topics such as patents or code of conduct. It is only meant to point you in the right direction. Exceptions may apply. The IETF's patent policy and the definition of an IETF "contribution" and "participation" are set forth in BCP 79; please read it carefully.

As a reminder:

- By participating in the IETF, you agree to follow IETF processes and policies.
- If you are aware that any IETF contribution is covered by patents or patent applications that are owned or controlled by you or your sponsor, you must disclose that fact, or not participate in the discussion.
- As a participant in or attendee to any IETF activity you acknowledge that written, audio, video, and photographic records of meetings may be made public.
- Personal information that you provide to IETF will be handled in accordance with the IETF Privacy Statement.
- As a participant or attendee, you agree to work respectfully with other participants; please contact the ombudsteam (https://www.ietf.org/contact/ombudsteam/) if you have questions or concerns about this.

Definitive information is in the documents listed below and other IETF BCPs. For advice, please talk to WG chairs or ADs:

BCP 9 (Internet Standards Process)
BCP 25 (Working Group processes)
BCP 25 (Anti-Harassment Procedures)
BCP 54 (Code of Conduct)
BCP 78 (Copyright)
BCP 79 (Patents, Participation)
https://www.ietf.org/privacy-policy/ (Privacy Policy)

Source: https://www.ietf.org/about/note-well/

Meeting Materials

- 16:10-18:10 Monday Afternoon session II
- Remote Participation
  - Jabber Room: xmpp:roll@jabber.ietf.org?join
  - Meetecho: https://www.meetecho.com/ietf104/roll
  - Etherpad: https://etherpad.tools.ietf.org/p/notes-ietf-104-roll
  - Minutes taker: **MILLION THANKS** to Dominique B.
- Jabber Scribe: **MILLION THANKS** to Rahul J.
- **Please sign blue sheets :-)**

Agenda

IETF 104 - ROLL Meeting

16:10-18:10 Monday Afternoon session II - 25th March

Material: https://datatracker.ietf.org/meeting/104/materials/

Time	Duration	Draft/Topic	Presenter
16:10 - 16:20	10 min	Introduction	Ines/Peter
16:20 - 16:30	10 min	draft-ietf-roll-aodv-rpl-06 Asymmetric AODV-P2P-RPL in Low-Power and Lossy Networks (LLNs)	Charlie
16:30 - 16:50	20 min	draft-rahul-roll-mop-ext-00 RPL Mode of Operation extension	Rahul
16:50 - 17:00	10 min	draft-ietf-roll-dao-projection-05 Root initiated routing state in RPL	Pascal
17:00 - 17:10	10 min	draft-thubert-roll-unaware-leaves-06 Routing for RPL Leaves	Pascal
17:10 - 17:20	10 min	draft-ietf-roll-nsa-extension-01 RPL DAG Metric Container Node State and Attribute object type extension	Aris
17:20 - 17:30	10 min	draft-koutsiamanis-roll-traffic-aware-of-00 Traffic-aware Objective Function	Aris
17:30 - 17:50	20 min	draft-audeoudh-rpl-asymmetric-links-00 Experimental observation of RPL: routing protocol overhead and asymmetric links	Henry
17:50 - 18:05	15 min	RPL-BIER Status - How we proceed?	Toerless/Ines/Peter
18:05 - 18:10	05 min	Open Floor	Everyone

Milestones

Date	Milestone
Apr 2018	Initial Submission of a proposal with uses cases for RPI, RH3 and IPv6-in-IPv6 encapsulation to the IESG
Aug 2018	Initial submission of a root initiated routing state in RPL to the IESG
Dec 2018	Initial submission of a proposal to augment DIS flags and options to the IESG
Jan 2019	Initial submission of a proposal for Source-Route Multicast for RPL to the IESG
Jul 2018	Initial submission of a solution to the problems due to the use of No-Path DAO Messages to the IESG
Jul 2018	Initial submission of a reactive P2P route discovery mechanism based on AODV-RPL protocol to the IESG
Jul 2019	Initial submission of a Forwarder Selection Protocol for MPL to the IESG
Mar 2019	Initial submission of a YANG model for MPL to the IESG
Sep 2019	Recharter WG or close

State of Active Internet-Drafts

Draft	Status
draft-ietf-roll-aodv-rpl-06	WGLC - Discussion today
draft-ietf-roll-dao-projection-05	WGLC - Discussion today
Draft-ietf-roll-forw-select-00 (Expired)	On hold
draft-ietf-roll-useofrplinfo-25	New Version - WGLC ready
Draft-ietf-roll-dis-modifications-00 (Expired)	To be continued
Draft-ietf-roll-mpl-yang-02 (Expired)	On hold
Draft-ietf-roll-bier-ccast-01 (Expired)	Bier-roll design team takes over
draft-ietf-roll-efficient-npdao-09	Submitted to IESG :D
draft-ietf-roll-rpl-observations-00	Used as model to develop further drafts
draft-ietf-roll-nsa-extension-01	Discussion today

Related Internet-Drafts

Draft	Status
draft-thubert-roll-unaware-leaves-06	Discussion today :-)
draft-rahul-roll-mop-ext-00	
draft-koutsiamanis-roll-traffic-aware-of-00	
draft-audeoudh-rpl-asymmetric-links-00	

Open tickets

Ticket	Summary	Component
#179	Security considerations for dao projection	dao-projection
#180	13 issues to address in dao projection draft (lifetime, MOP, retransmissions, route cleanup)	dao-projection
#187	New version of RFC6550 - Topics to include	rpl
#188	Should 6LBR be included into the DODAG root?	rpl
#189	RPL and new MOP?	dao-projection

https://trac.ietf.org/trac/roll/report/2

Using RPL Option Type, Routing Header for Source Routes and IPv6-in-IPv6 encapsulation in the RPL Data Plane

draft-ietf-roll-useofrplinfo-25

- Operational Considerations added for unaware-leaves
- Operational Considerations of the new option 0x23
- Security Considerations modified - Tickets 190,191,192 and 193 fixed.
- Normative Terminology eliminated of the examples.

- Please read and comment into the ML => submitted to IESG, in LC

Asymmetric AODV-P2P-RPL in Low-Power and Lossy Networks (LLNs)

draft-ietf-roll-aodv-rpl-06

IETF 104, Prague

Satish Anamalamudi <satishnaidu80@gmail.com>

Mingui Zhang <zhangmingui@huawei.com>

Charlie Perkins <charles.perkins@earthlink.net>

S.V.R Anand <anand@ece.iisc.ernet.in>

Liu Bing <remy.liubing@huawei.com>

Changes from version 05 to version 06

- Added Security Considerations based on the security mechanisms defined in RFC 6550.

- Clarified the nature of improvements due to P2P route discovery versus bidirectional asymmetric route discovery.

- Editorial improvements and corrections

Security Considerations from RFC 6550

Previous Security Considerations referred the reader to RFC 6550. The new document presents the RFC 6550 framework in detail.

- Unsecured (rely on external mechanisms)
- Preinstalled (pre-configured shared key)
- Authenticated (node obtains a second key from a key authority)

P2P versus bidirectional asymmetric route discovery

- P2P routes reduce congestion at the root
- P2P routes often provide shorter paths
  - Higher performance routing, reduced interference

- Bidirectional asymmetric routes may exist that would not be found when only looking for symmetric routes
  - better network connectivity

Other editorial improvements

- Use “Target Node” in the Introduction (instead of *TargNode*, which wasn’t defined yet)
- Move some bibliographic references from Normative to Informational
- Clarifications to improve meaning, for instance about sequence number incrementation
- Various spelling and grammatical syntax corrections

Next steps

- WG Last Call comments have been resolved
- Shepherding
- Request publication
- More reviewer comments?

MOP Extension & Capabilities

draft-rahul-roll-mop-ext-00

- Rahul, Pascal @ IETF104, Prague

Need?

- Mode of Operation (MOP)
  - Defines mandatory primitives to be supported by the routers in the networks
  - 3-bit size
  - Already exhausted

MOP	Used for
0	No downward routes
1	Non-storing
2	Storing with no mcast
3	Storing with mcast
4	P2P-RPL
5,6,7 (Unused)	(AODV-RPL, P-DAO-NS, P-DAO-Storing)

MOP Extension

- MOPex Option
  - New RPL Control message option
  - Applicable only if base DIO-MOP = 0x7
  - Final MOP = base MOP + MOPex

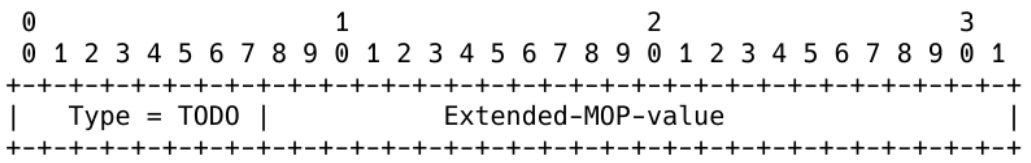


Figure 1: Extended MOP Option

Base MOP	MOPex	Final MOP
0	NA	0
1	NA	1
:	:	:
6	NA	6
7	0	7
7	1	8
7	2	9
:	:	:

Table 1: Final MOP calculation

Introducing Capabilities

- Capabilities indicate the set of features supported
  - DIO(cap): Capabilities supp by BR
  - DAO(cap): Capabilities supp by 6LR/6LN
  - Could be mandatory or optional
    - Specs defining new capability indicate whether it is mandatory/optional
- Why MOP is not sufficient?
  - MOP defines mandatory primitives needed by the routers to participate
  - With capabilities, you can have mandatory or optional with handshake (using DIO/DAO/ACK)
- Example use-case
  - Indicate support for 6LoRH capability ?
- Who else uses such capabilities field?
  - 802.11 Beacon

Capabilities Option

- Defined as new RPL Control message option
  - Can be part of DIO/DAO/ACK

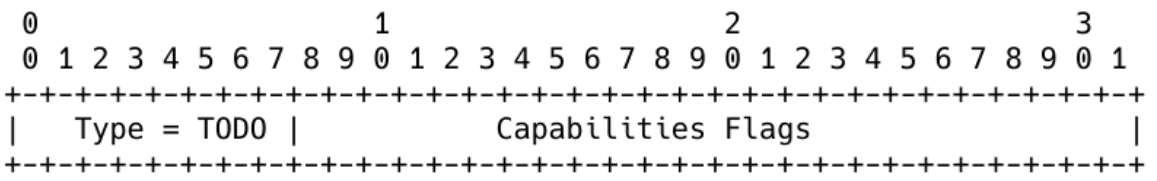


Figure 2: Capabilities Option

Points to ponder

- CAPs can work with existing MOPs
  - CAPs and MOPs is not dependent on each other
- CAPs to be sent in every DAO?
  - Can we elide/omit after initial DIO/DAO handshake?
- Impact on memory requirement?
  - 6LRs can independently generate DAO on behalf of any sub-child
  - With CAPs, will it increase memory requirement?
- Example capabilities handshake use-case?

Root initiated routing state in RPL

draft-ietf-roll-dao-projection

Pascal Thubert

IETF 104

Prague

Changes Highlights

- Invited Matthew to contribute from implementation experience
- New text on instances indicating risk of loops if the PCE adds routes to an instance with distributed RPL
- Added text to notify the root in case of an error detected in the datapath. The ICMP is similar to that in non storing mode.

Discussions needed to progress

How is the topology known to the root?

- Could use external management or Non-Storing DAO
- Suggestion to add a peer info option similar to transit info option

How are the node capabilities known to the root?

- Suggestion to add a node capability option similar

Complexity of mixed modes and route concatenation

MOP saturation: addressed by draft-rahul-roll-mop-ext

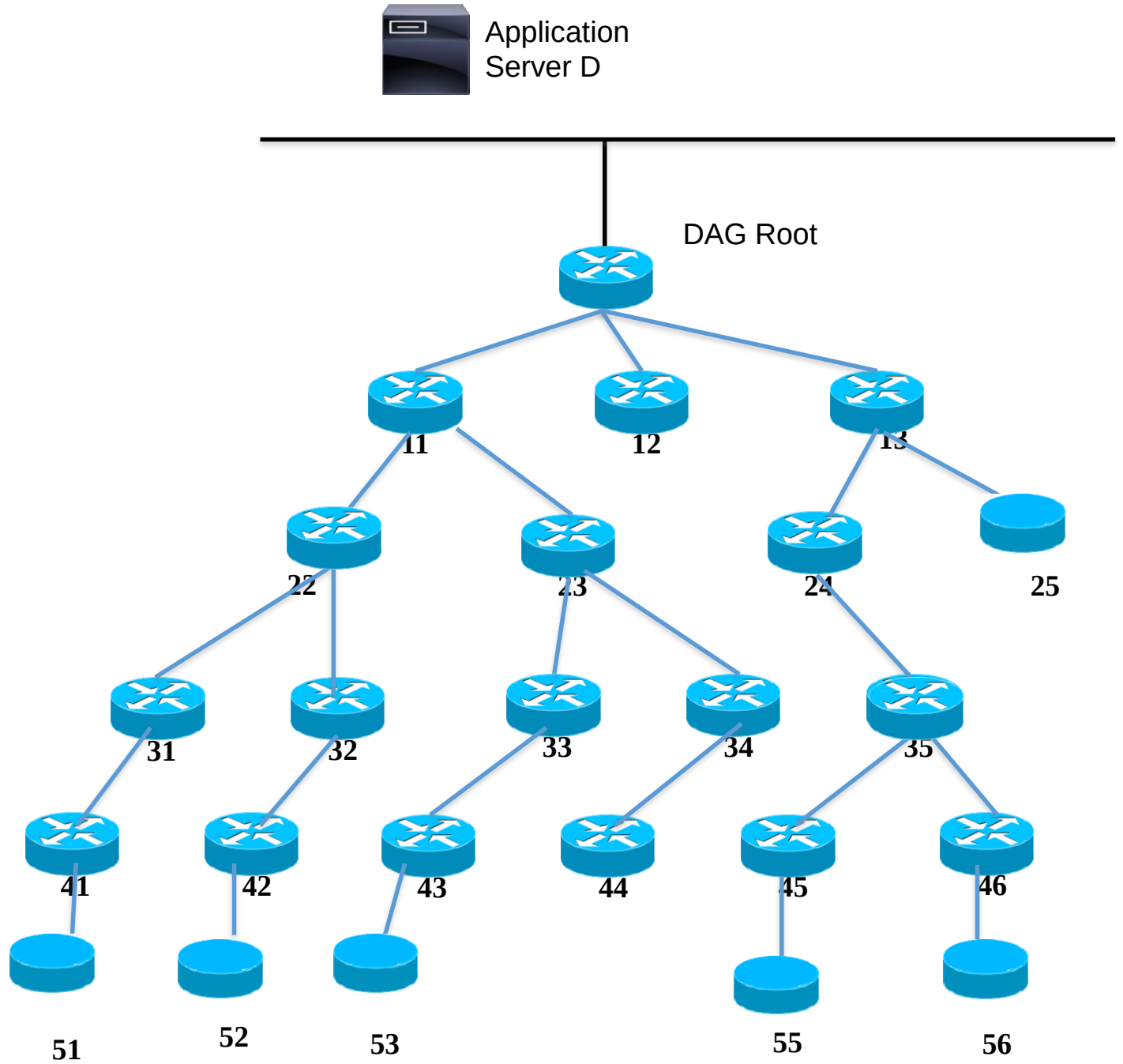
Compression of the Via Info option (so far full addresses)

Loop avoidance

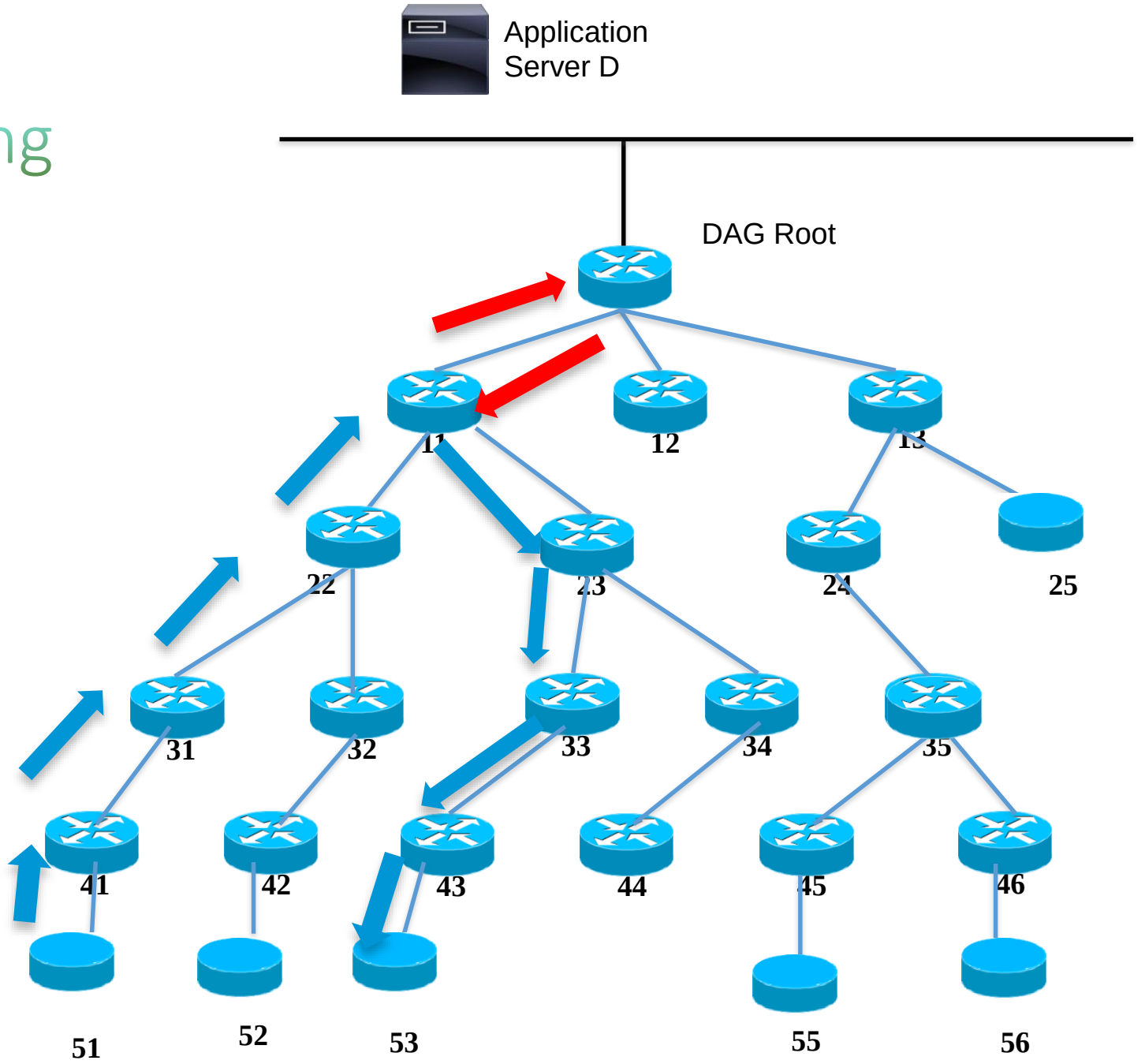
- Text on constraints on SRVIO to avoid loops with other routing

Backup

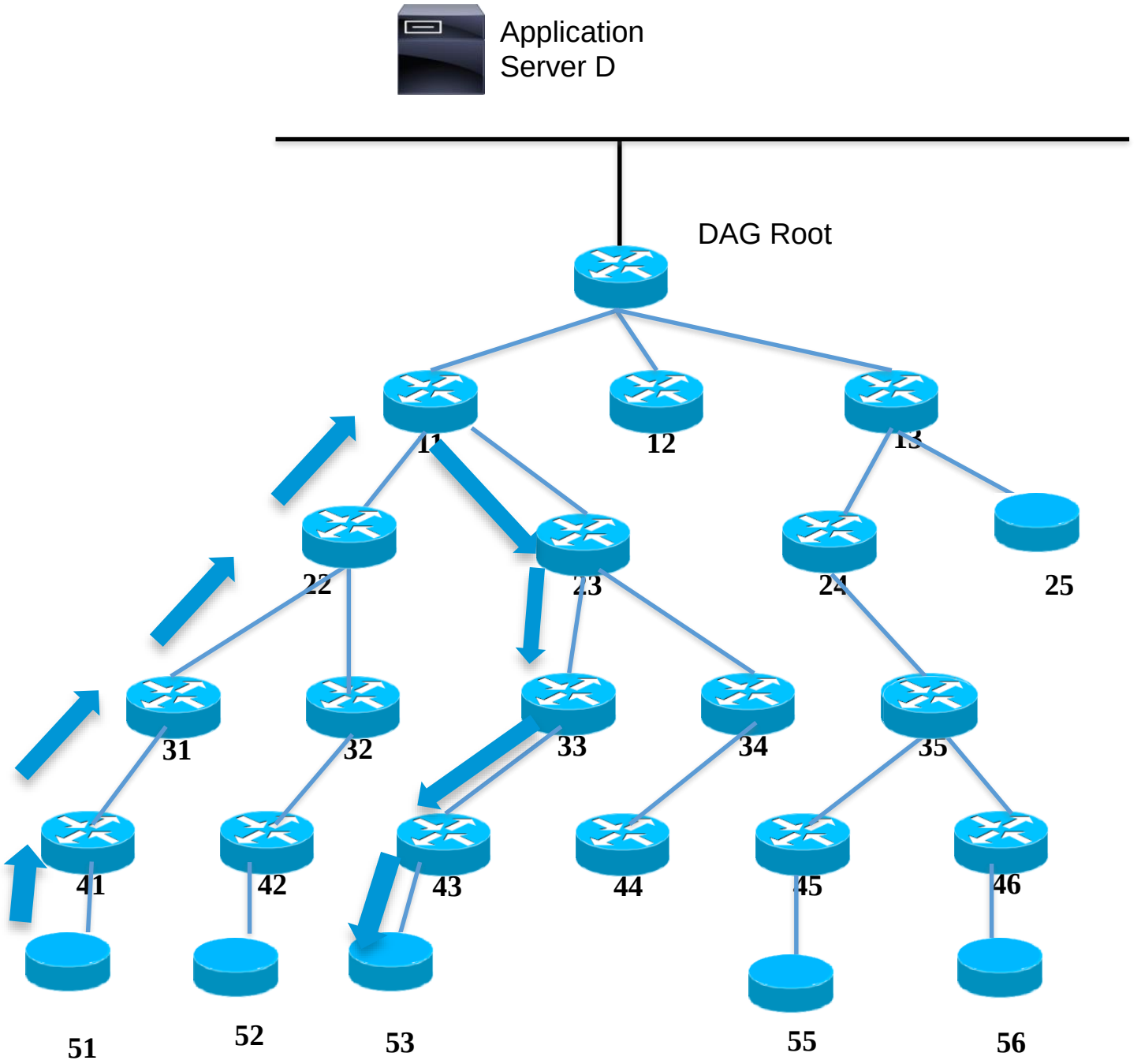
New: non-storing mode transversal route


Application
Server D
DAG Root
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56

Stretch in non-storing mode

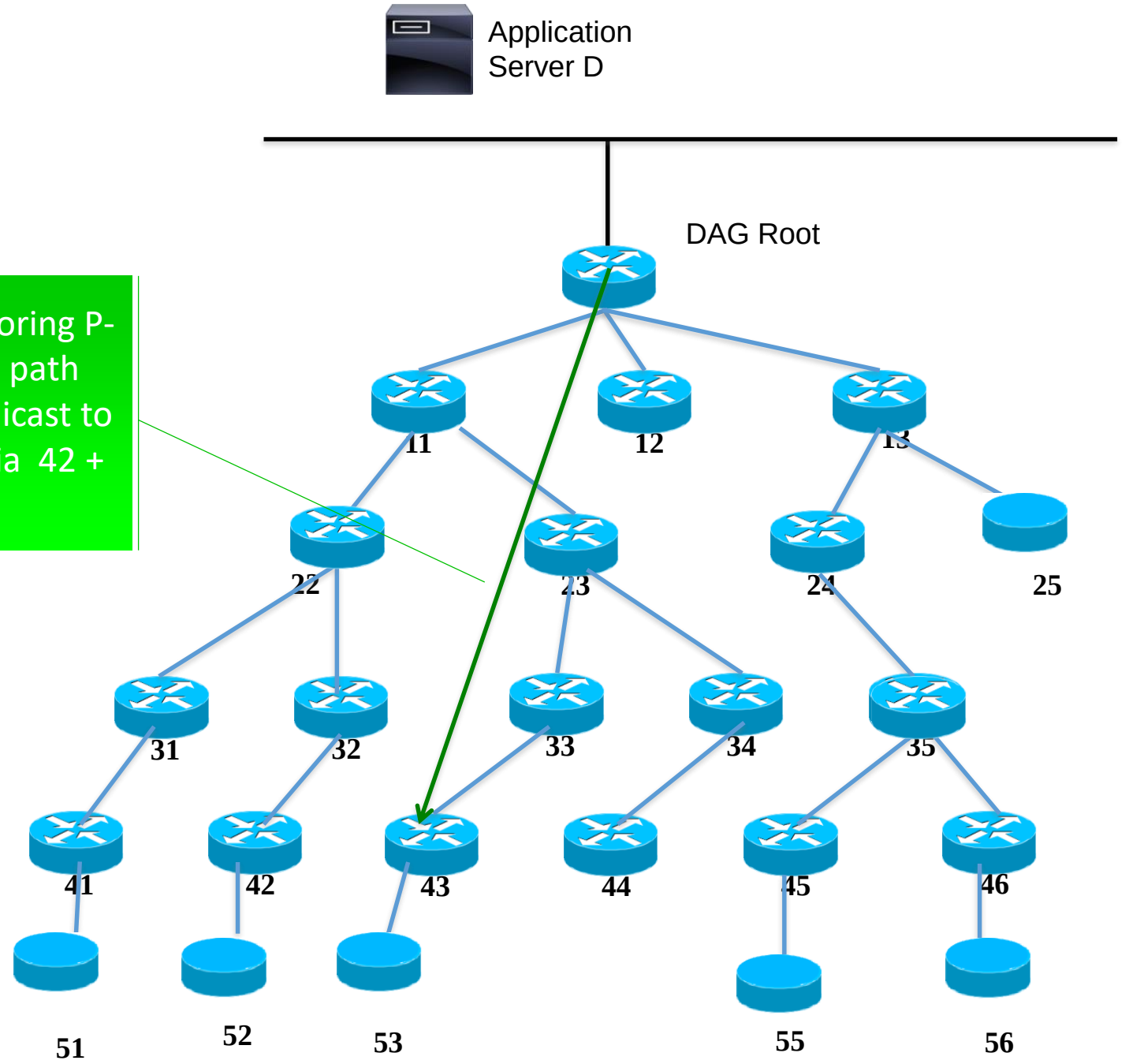


Stretch in storing mode

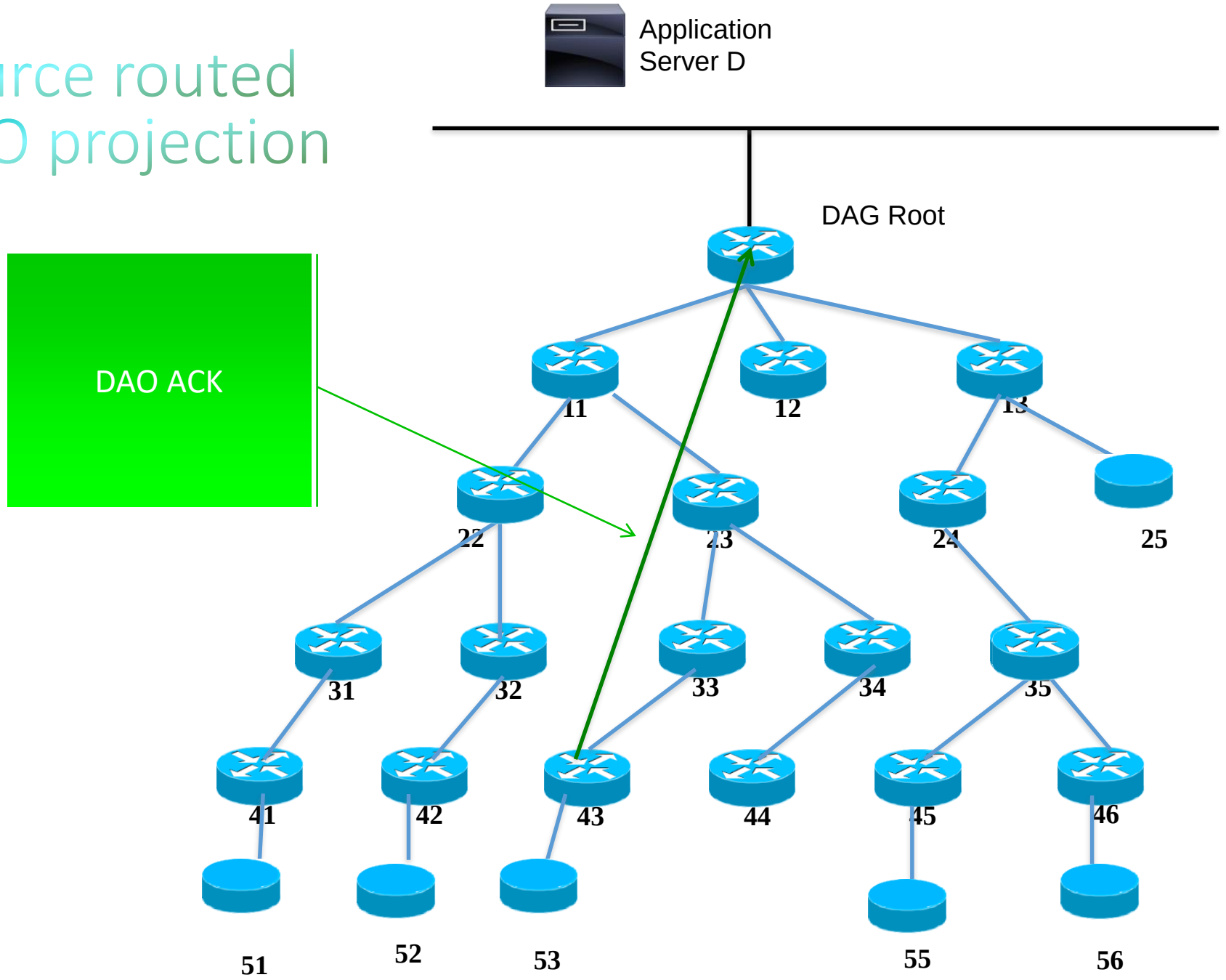


DAO projection

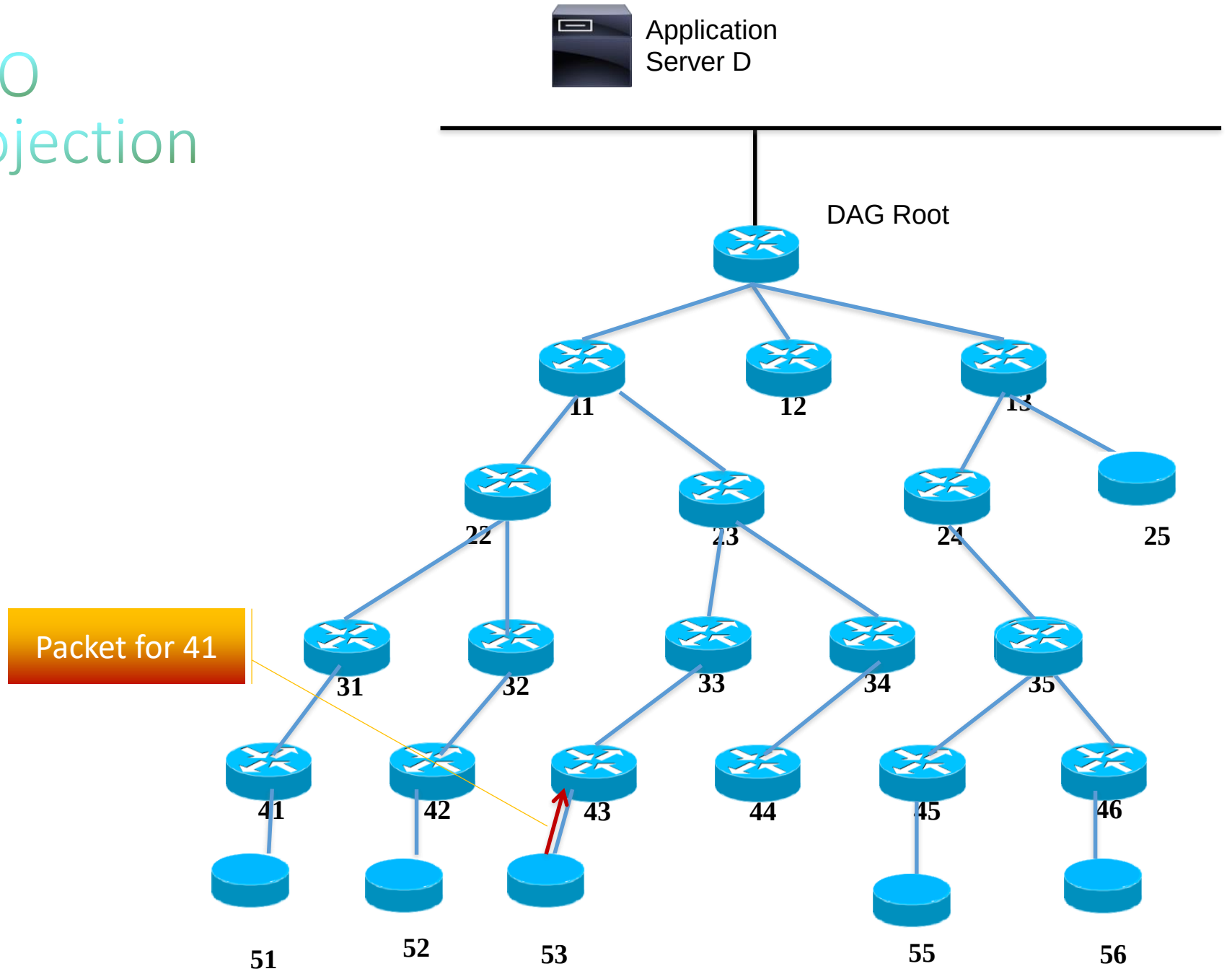
New non-storing P-DAO with path segment unicast to target 41 via 42 + 43



Source routed DAO projection

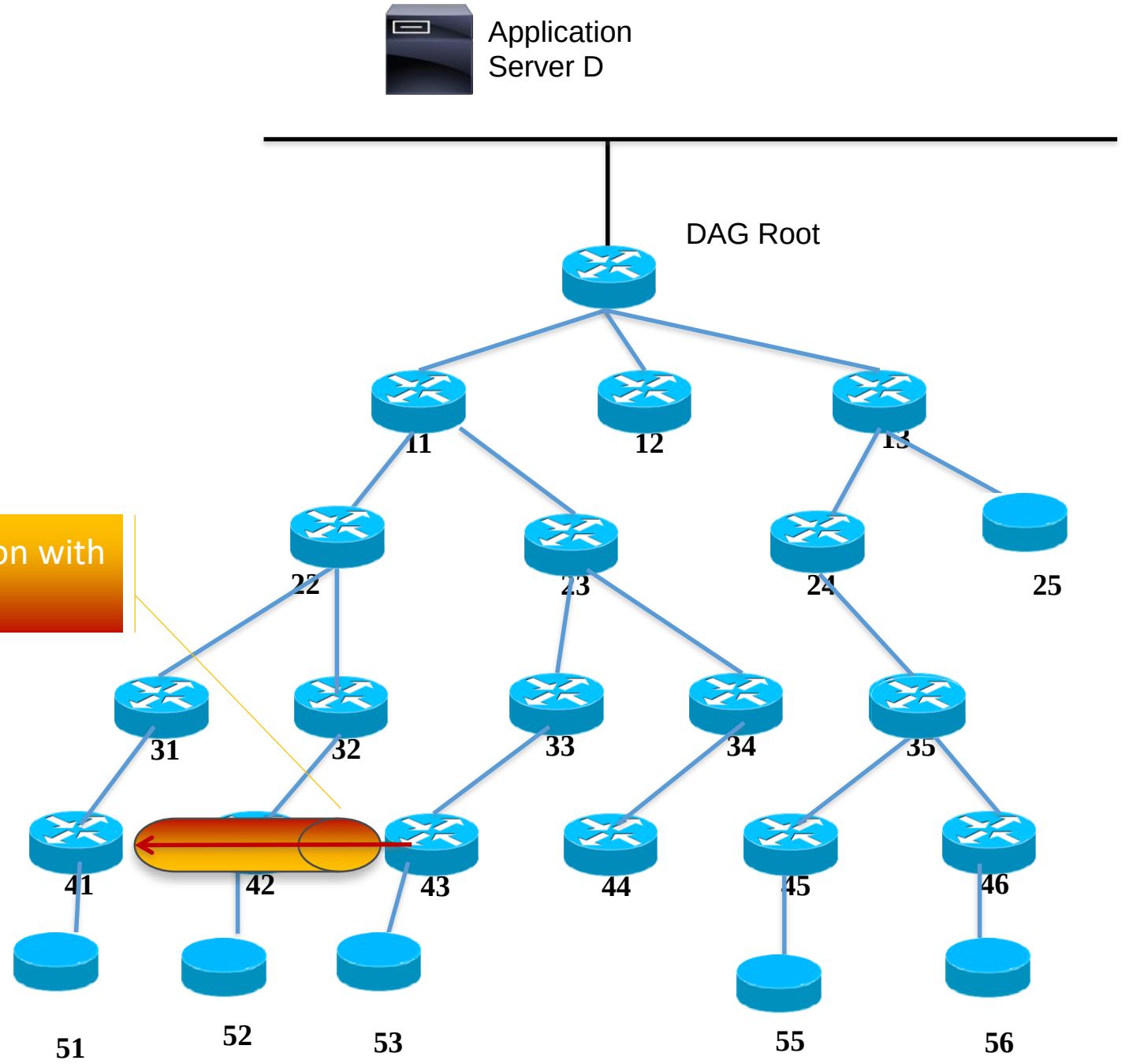


DAO projection

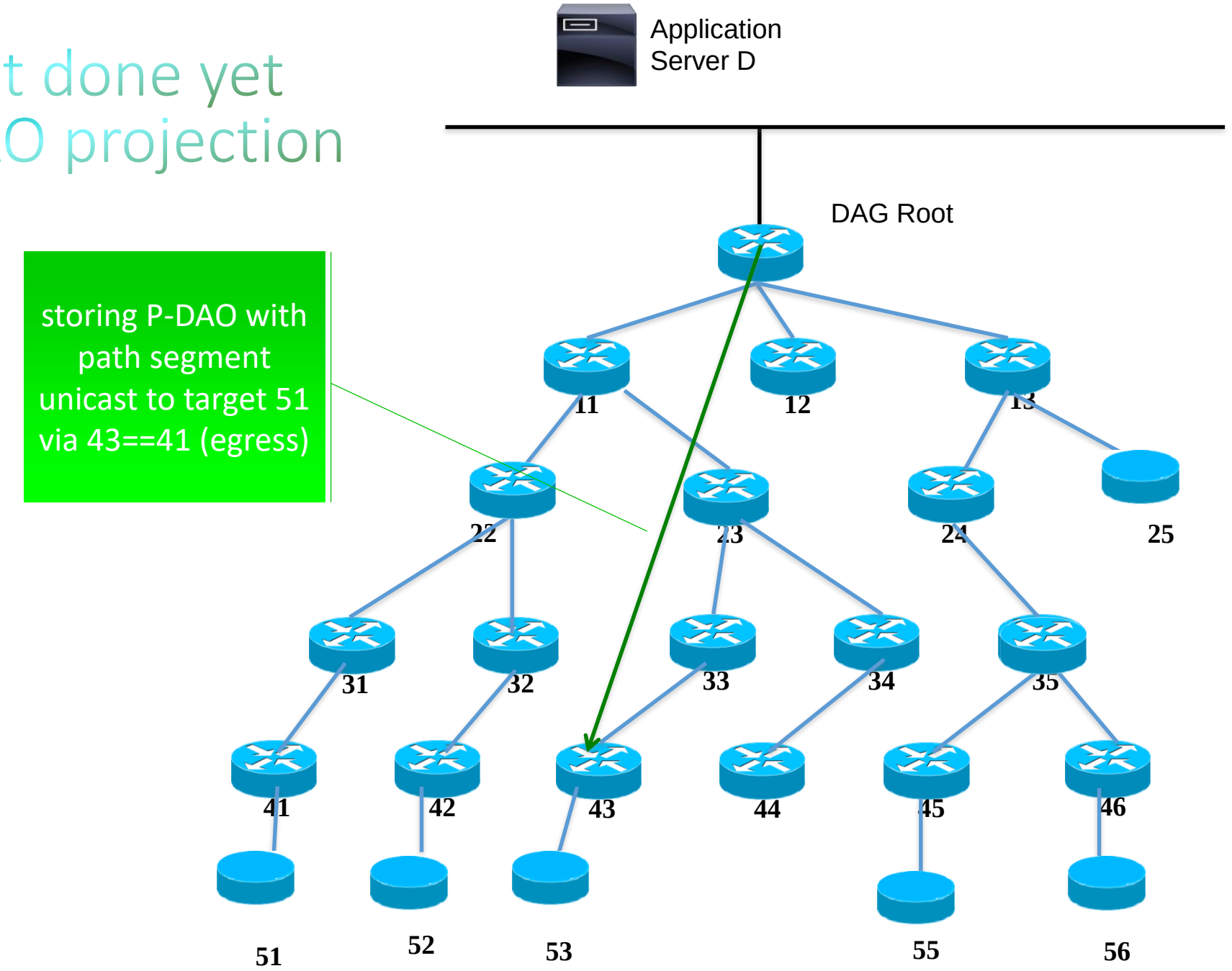


DAO projection

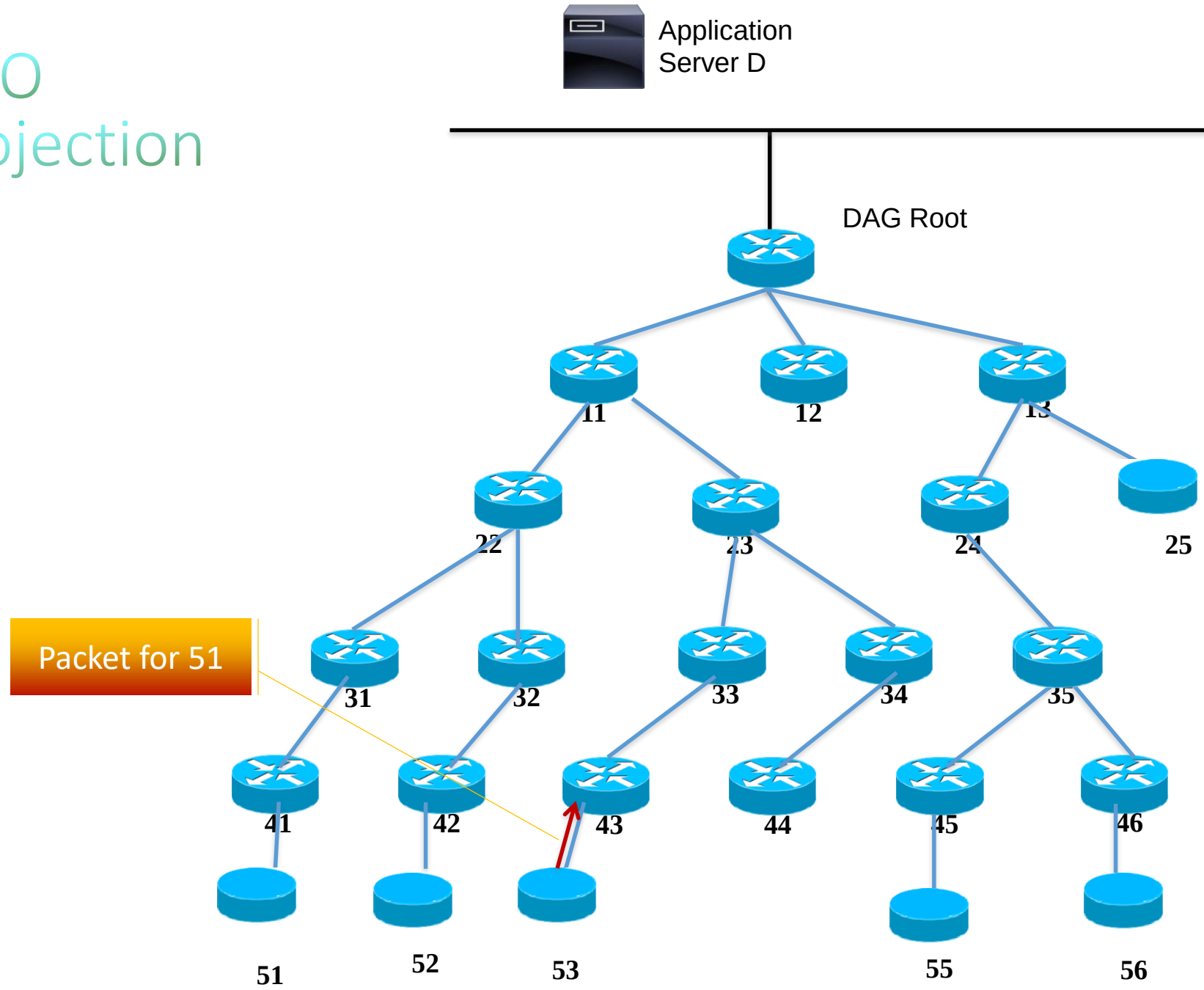
IP in IP encapsulation with SRH 42, 41



Not done yet
DAO projection

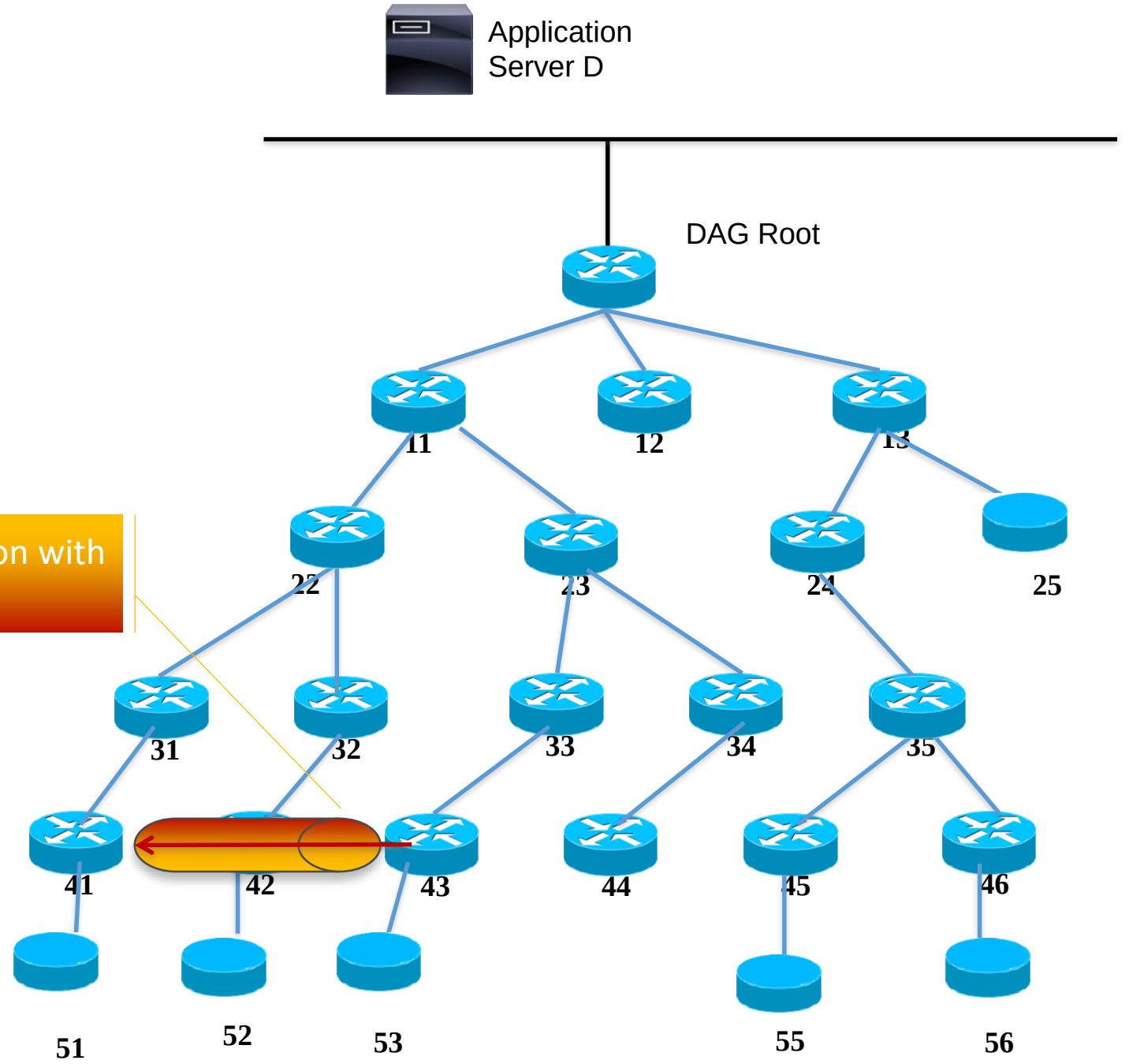


DAO projection

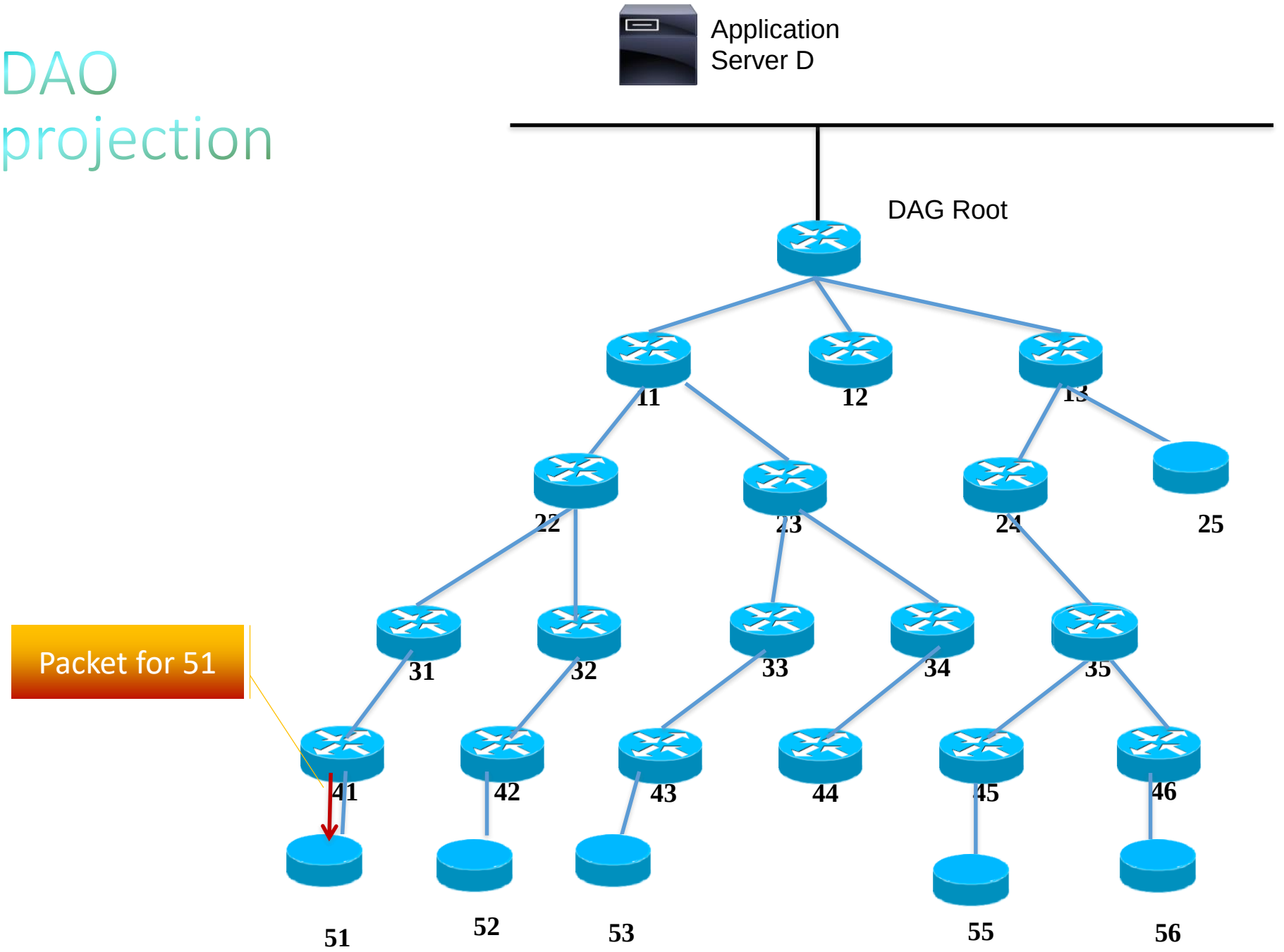


DAO projection

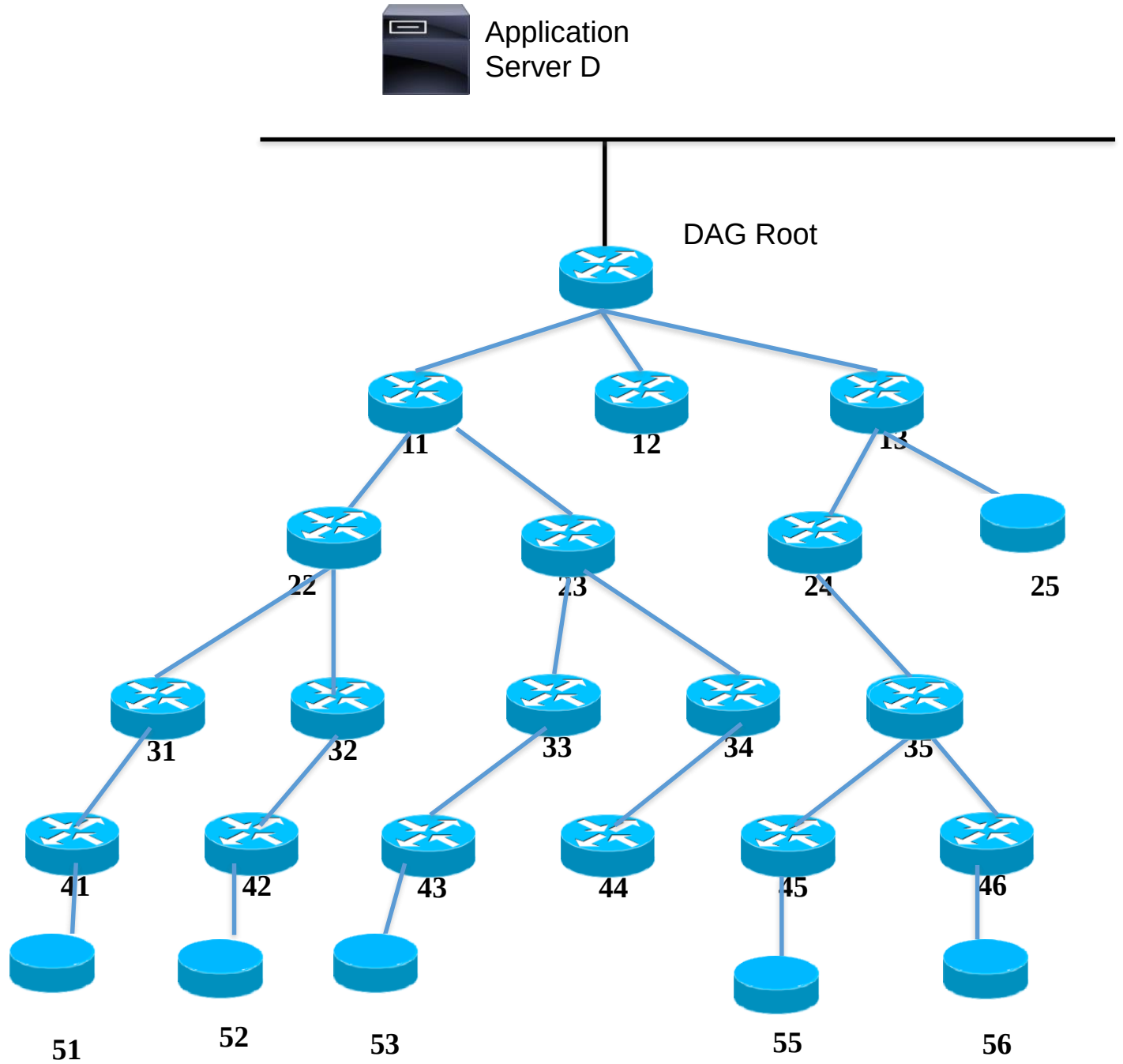
IP in IP encapsulation with SRH 42, 41



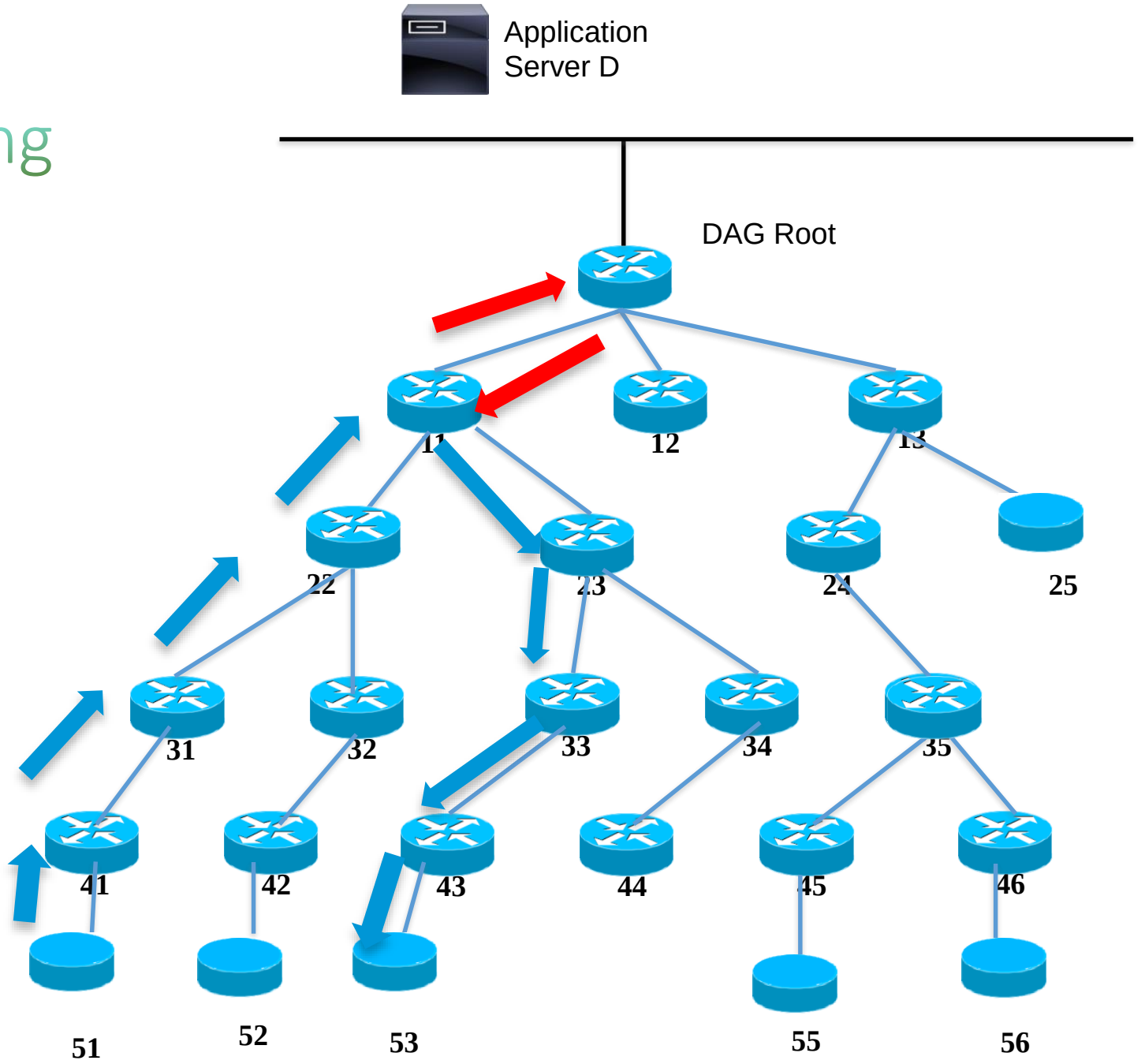
DAO projection



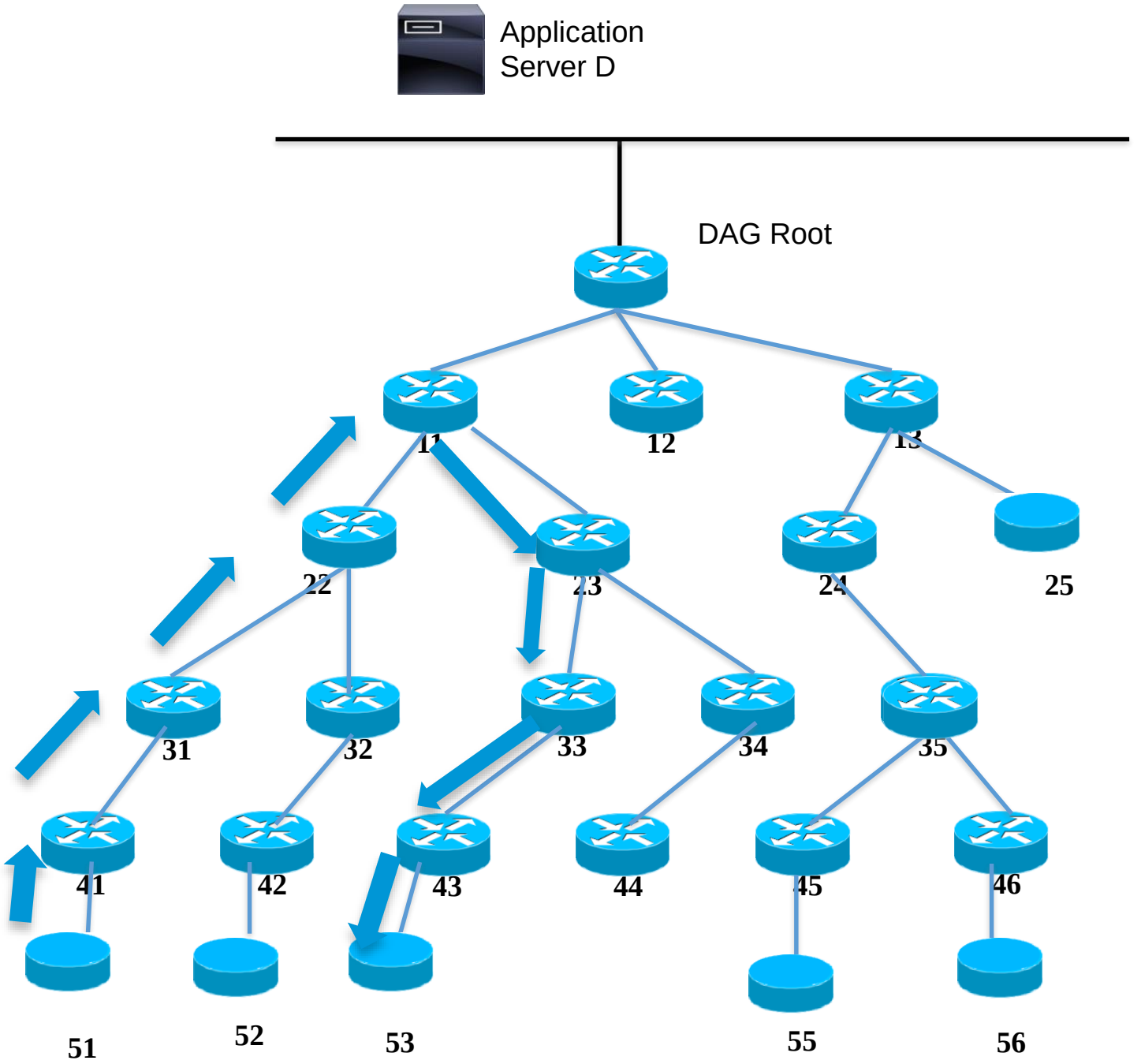
Storing mode transversal route


Application
Server D
DAG Root
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56

Stretch in non-storing mode

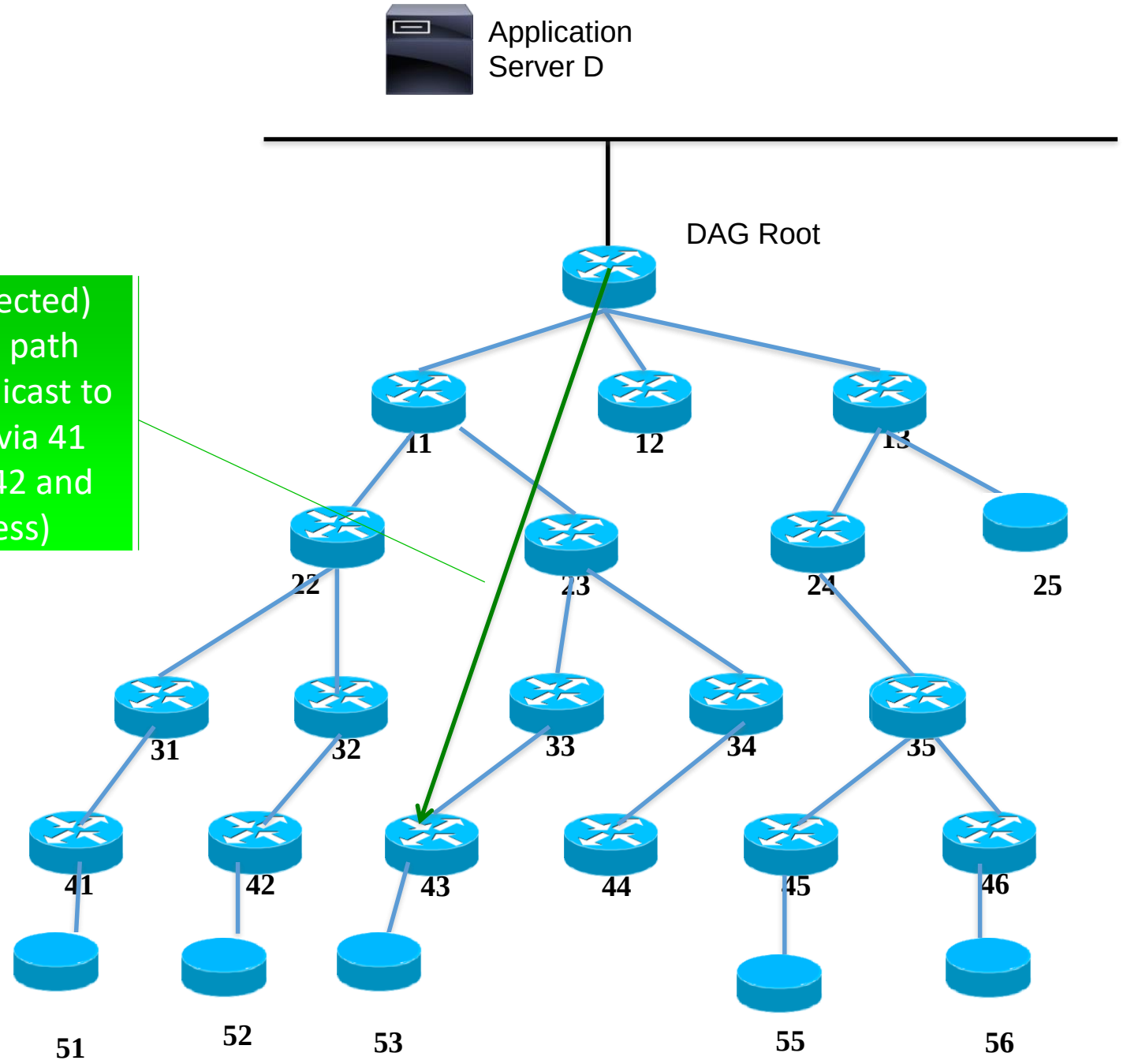


Stretch in storing mode

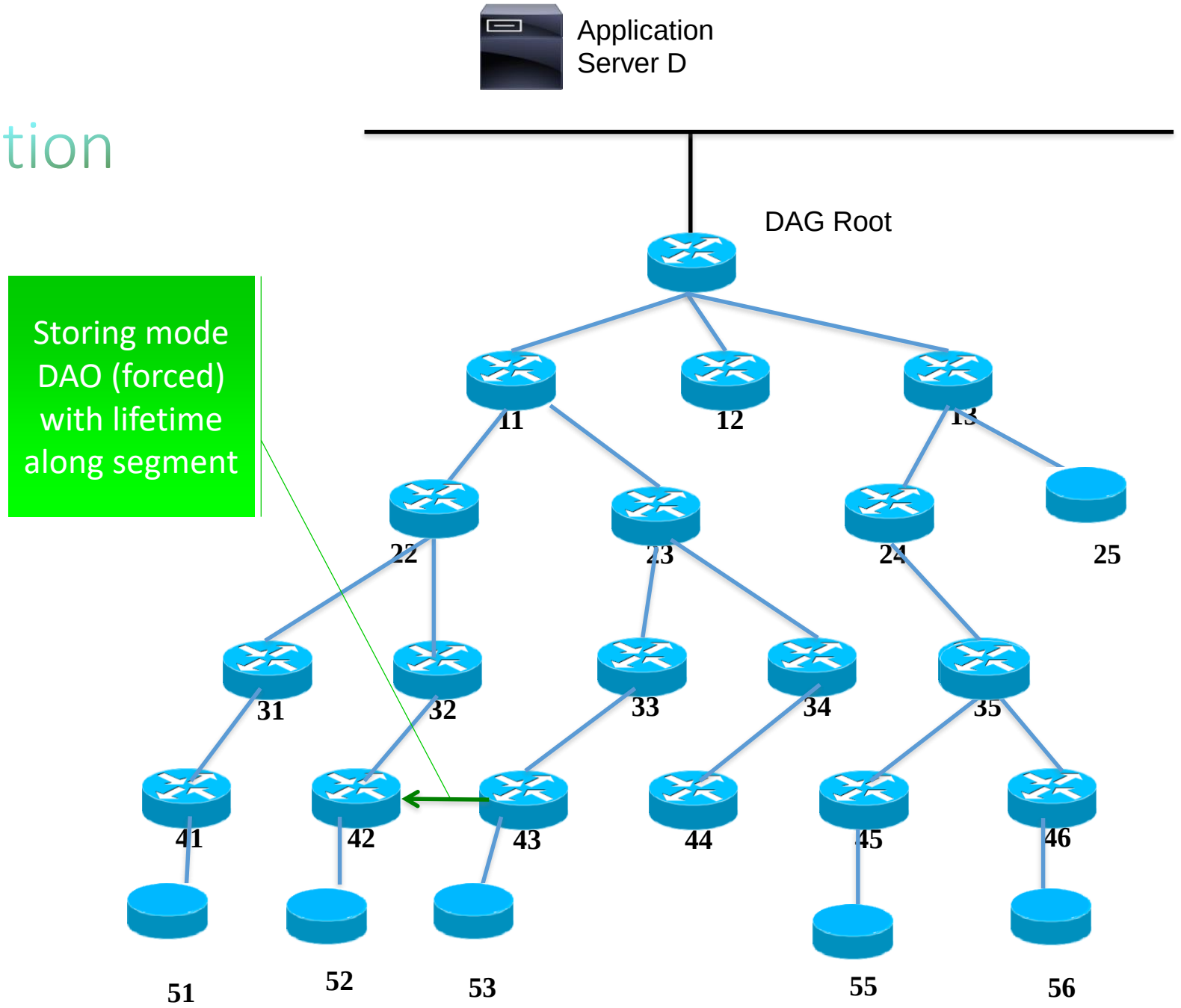


DAO projection

New (projected) DAO with path segment unicast to target 53 via 41 (ingress), 42 and 43 (egress)

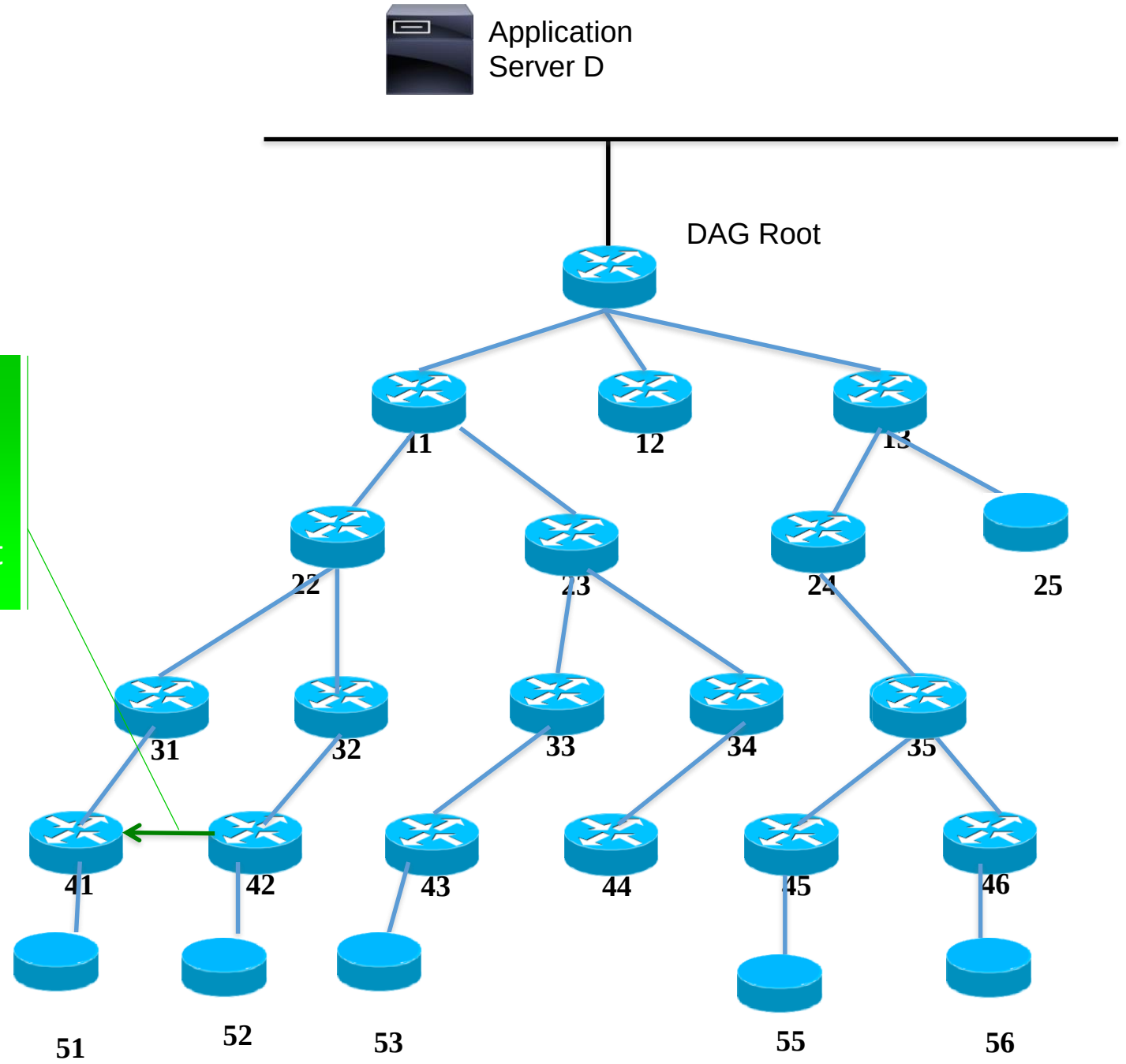


DAO projection

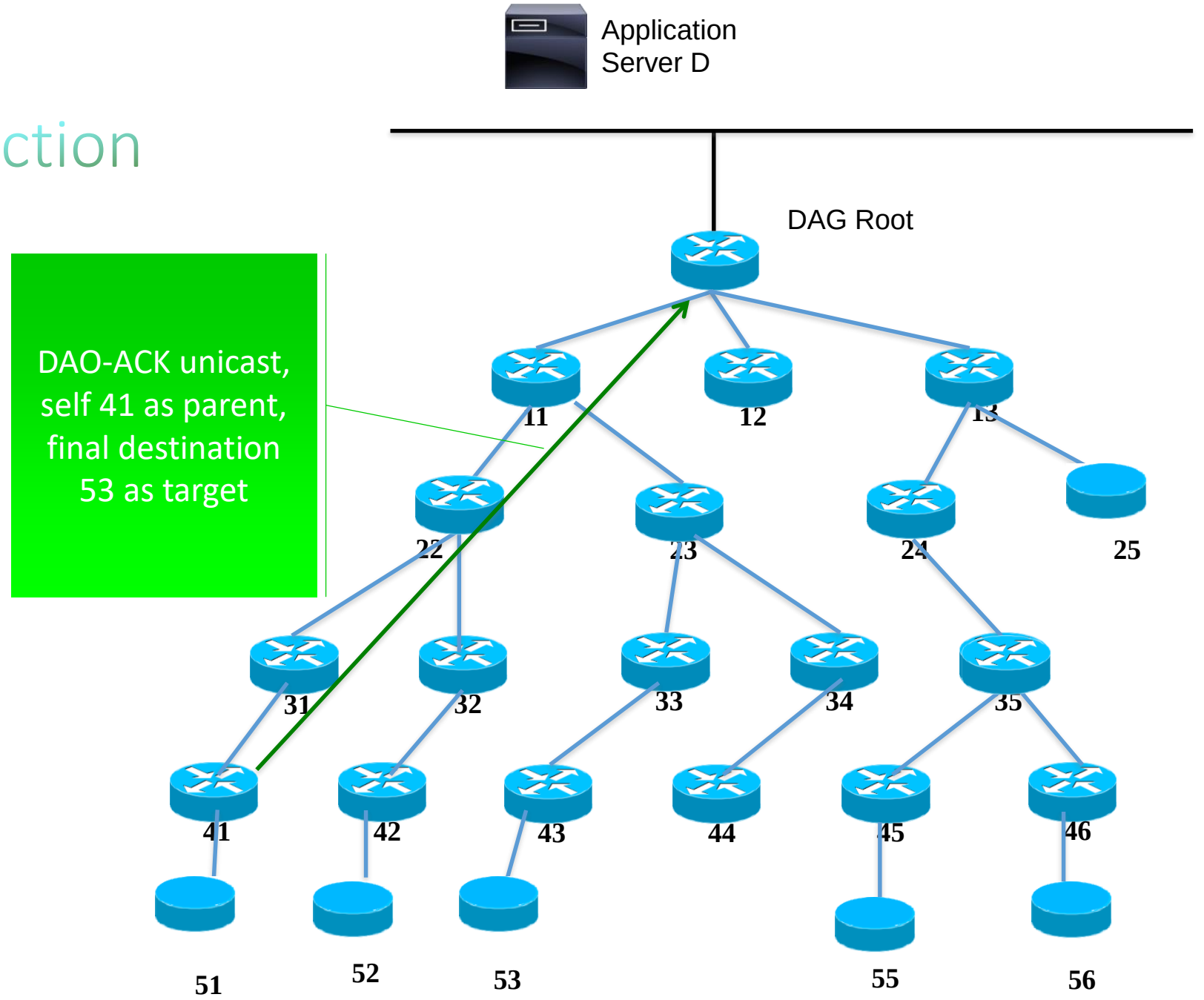


DAO projection

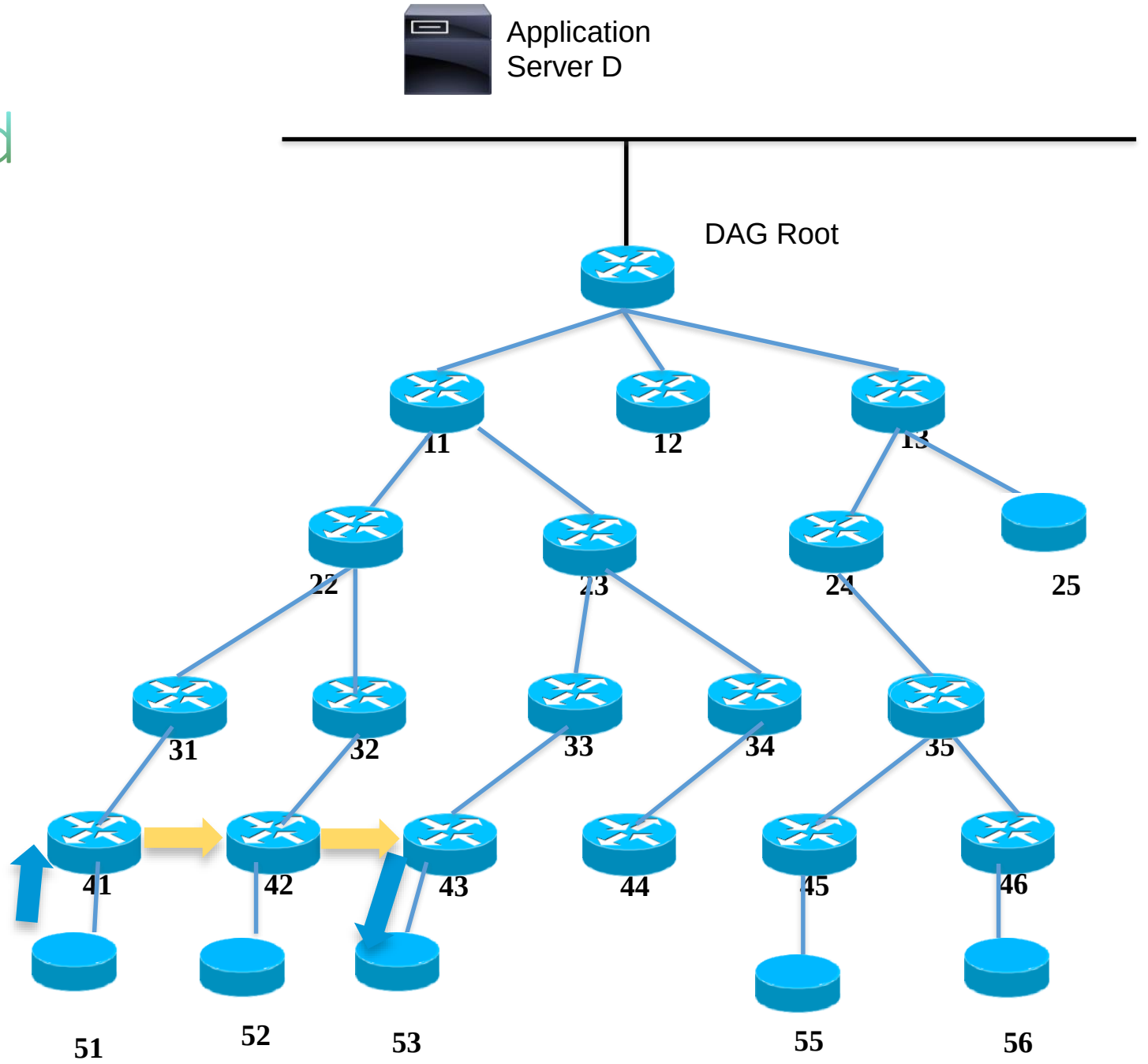
Storing mode DAO (forced) with lifetime along segment



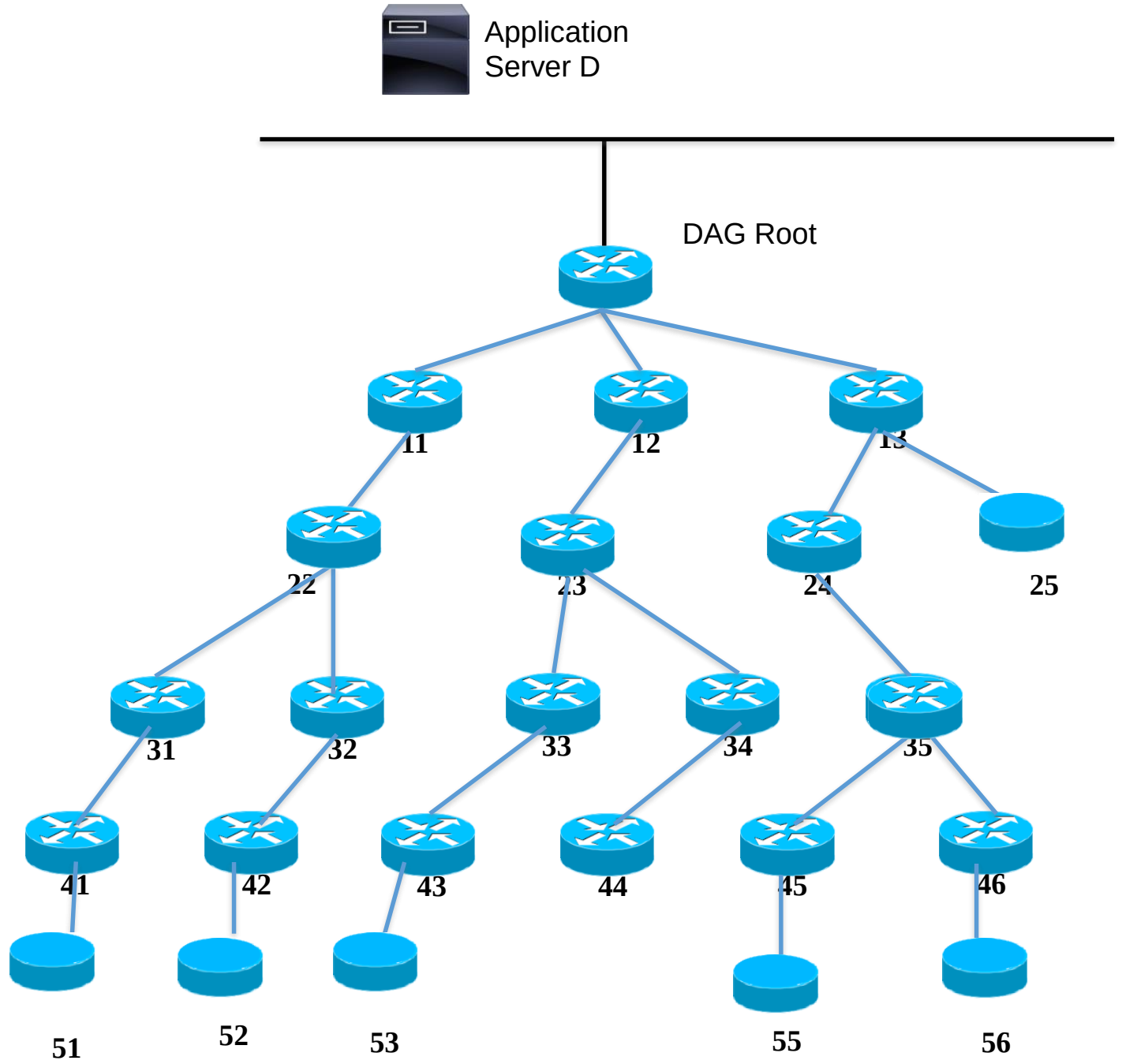
DAO projection

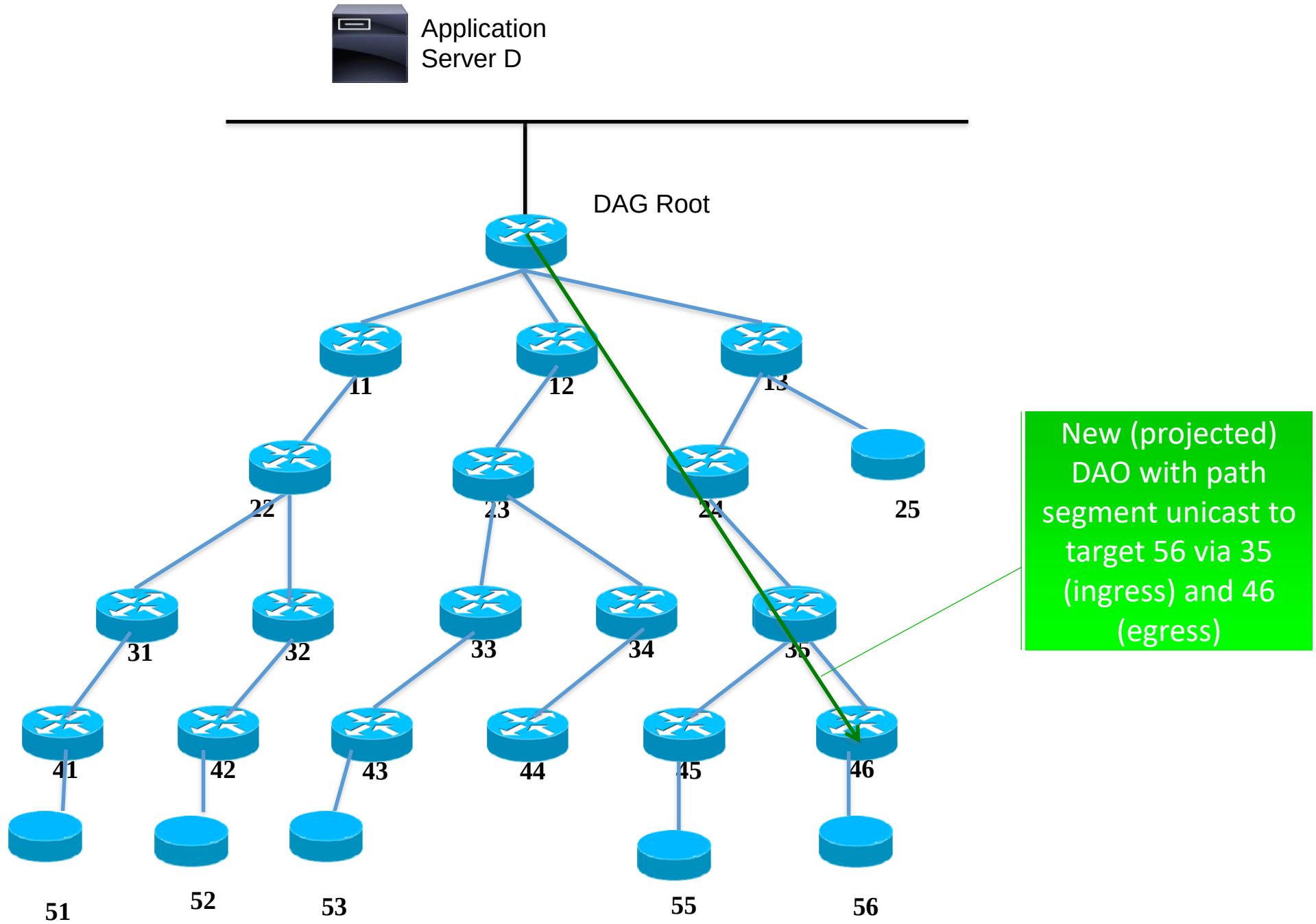


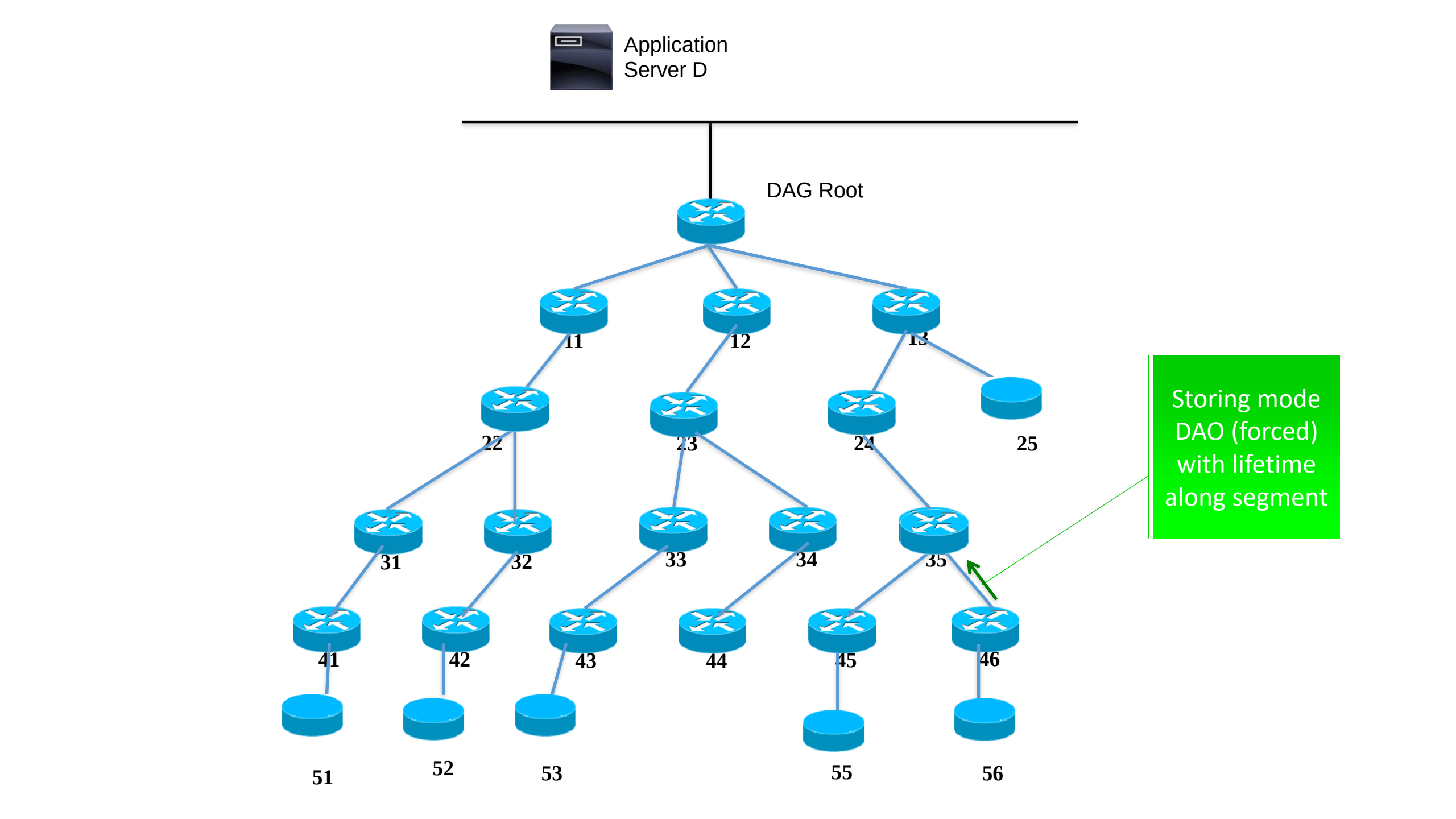
Optimized Path

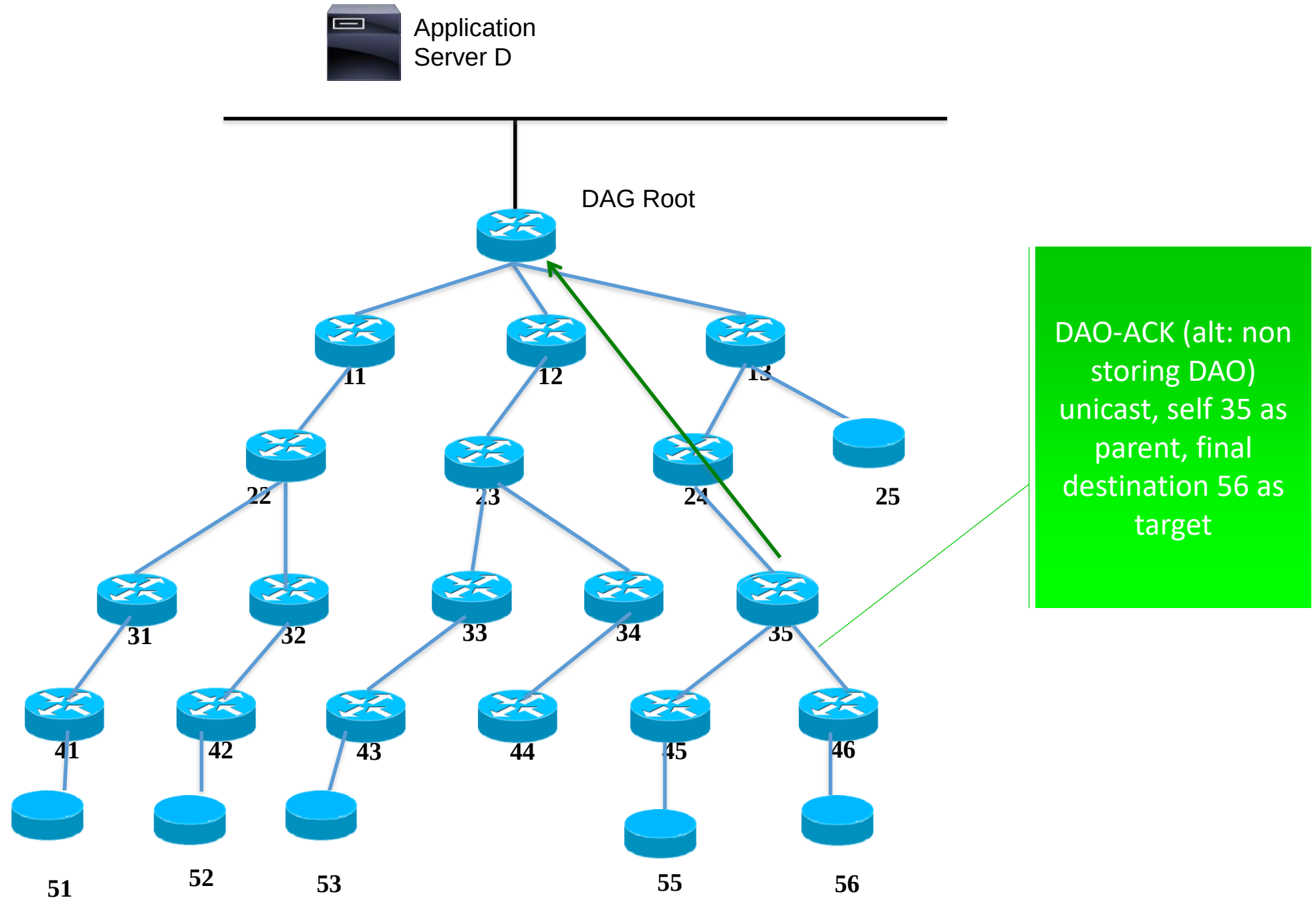


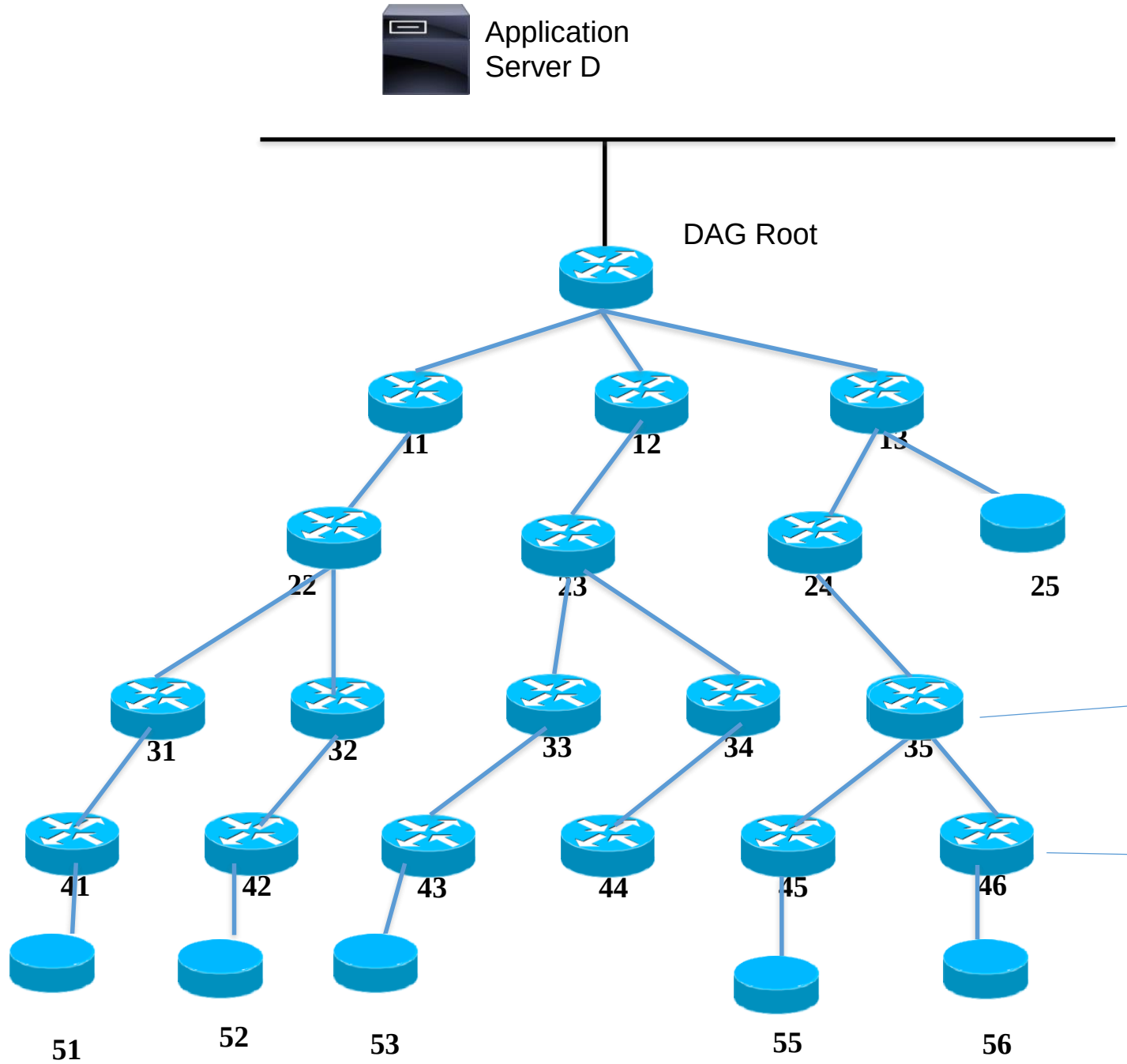
Existing non storing optimization


Application
Server D
DAG Root
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56


Application
Server D
DAG Root
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56
New (projected) DAO with path segment unicast to target 56 via 35 (ingress) and 46 (egress)


Application
Server D
DAG Root
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56
Storing mode
DAO (forced)
with lifetime
along segment

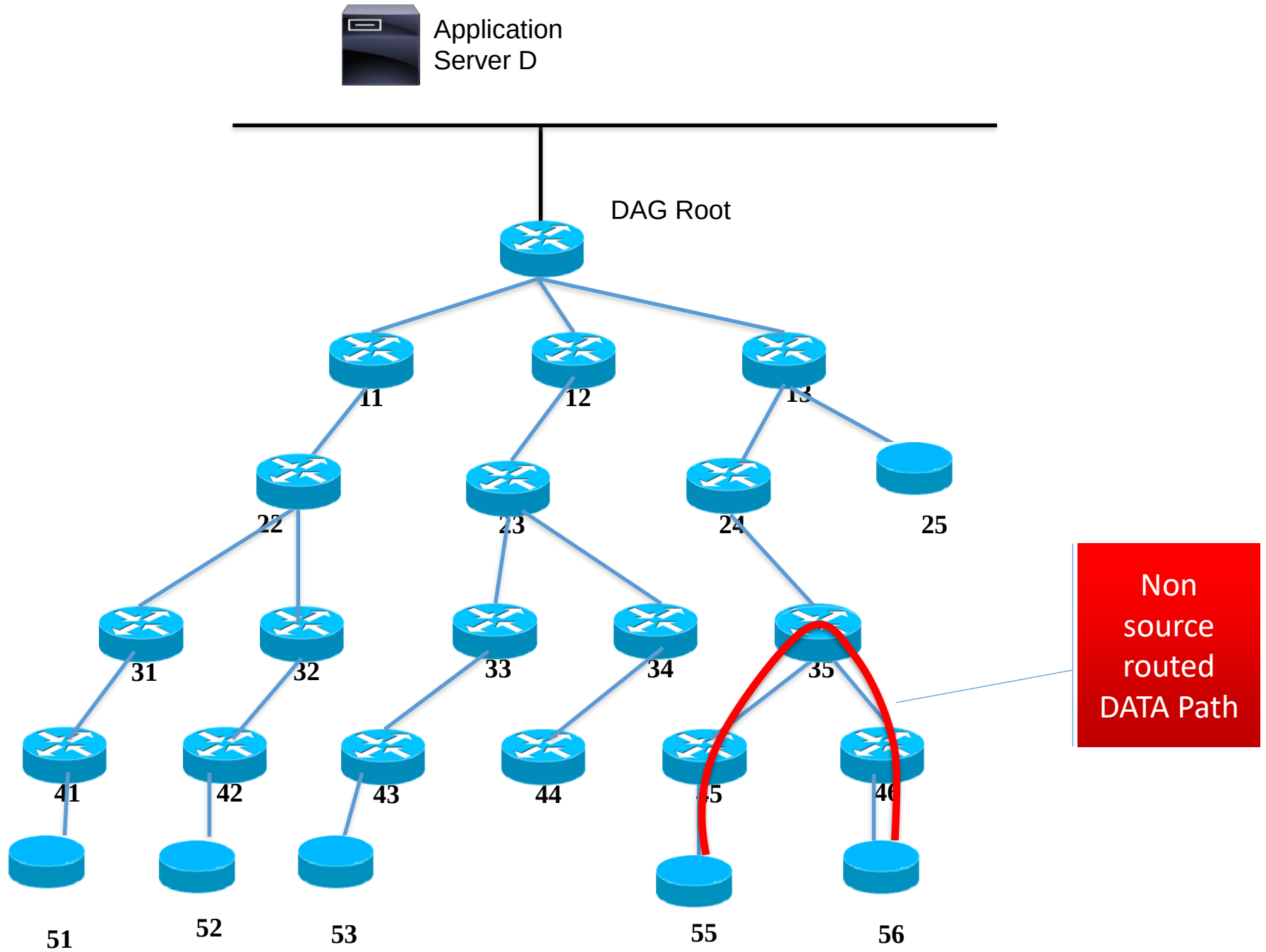

Application
Server D
DAG Root
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56
DAO-ACK (alt: non storing DAO) unicast, self 35 as parent, final destination 56 as target

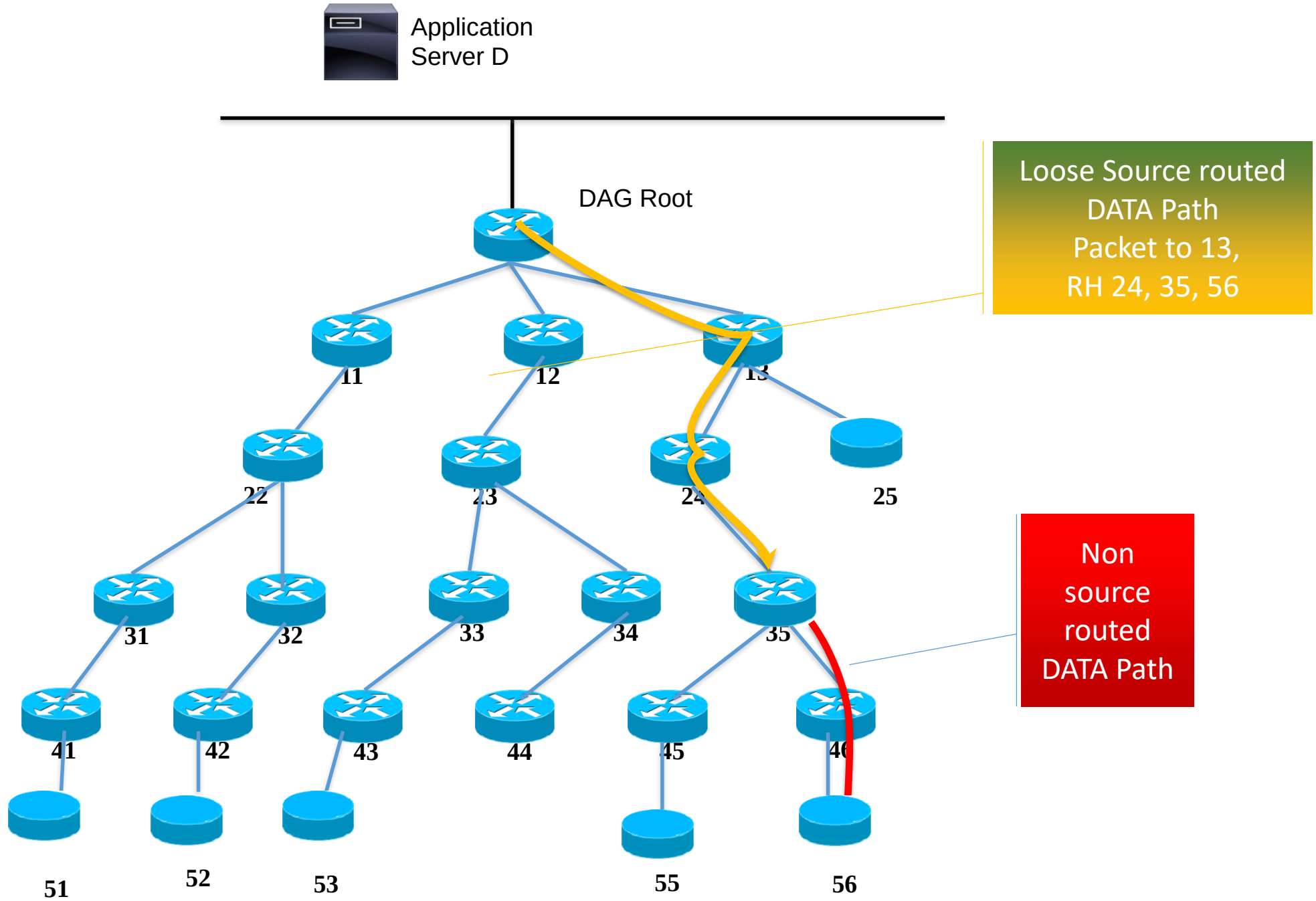


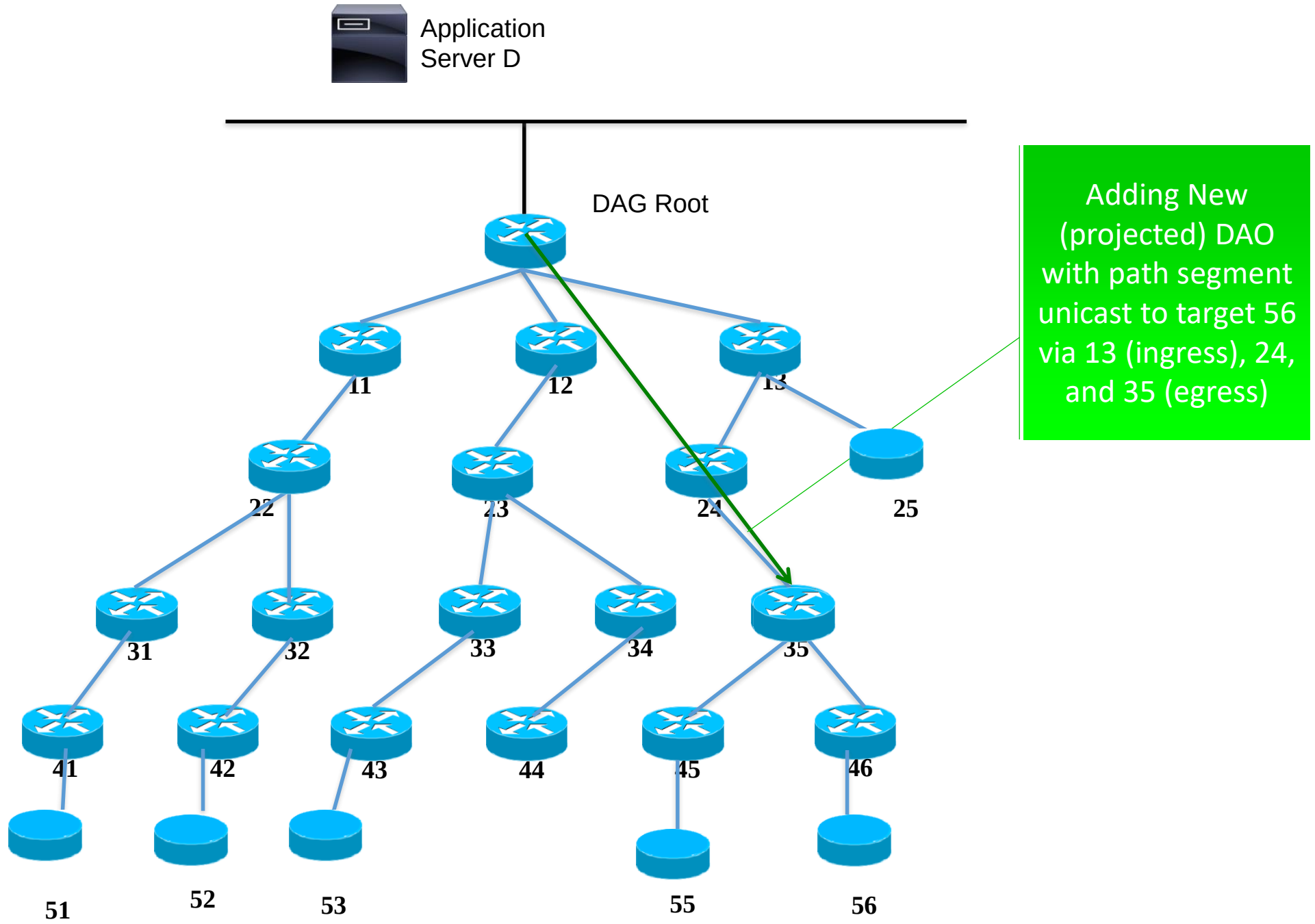
DAO from 46 installs a route to 56 in 35
(all nodes in projected route from ingress included to egress excluded)
=> egress should already have a route to target

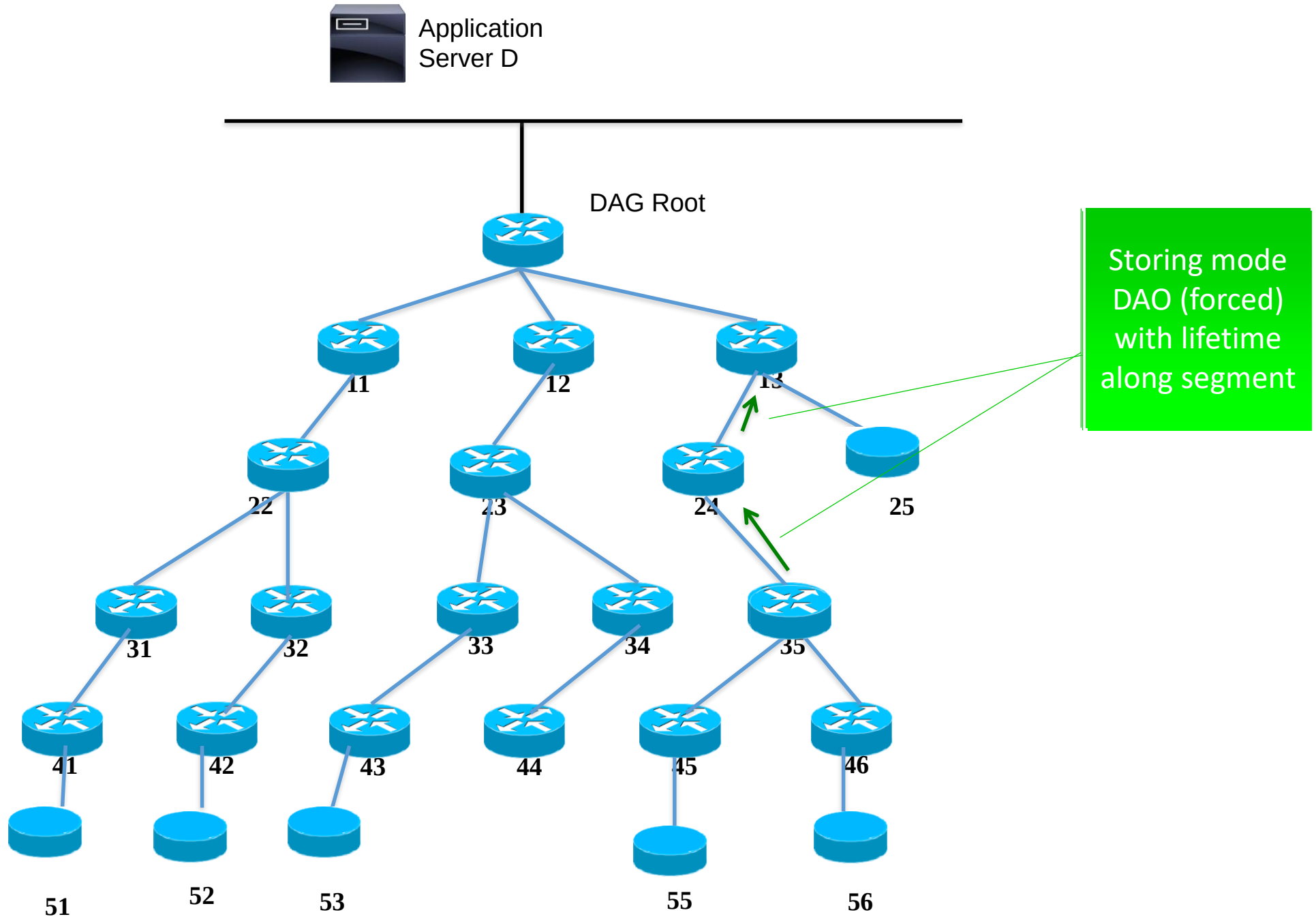
56 via 46

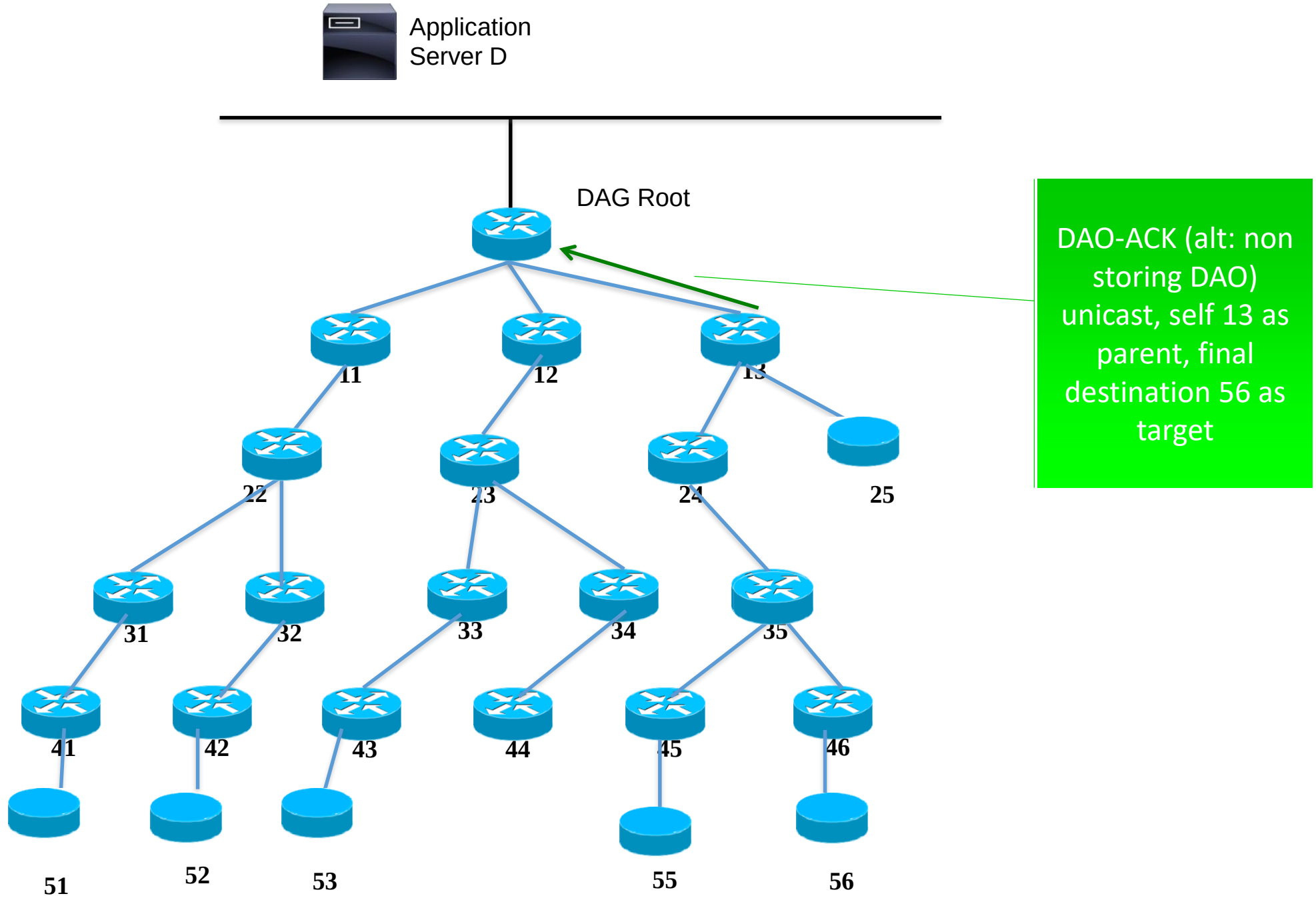
Preexisting connected route to 56

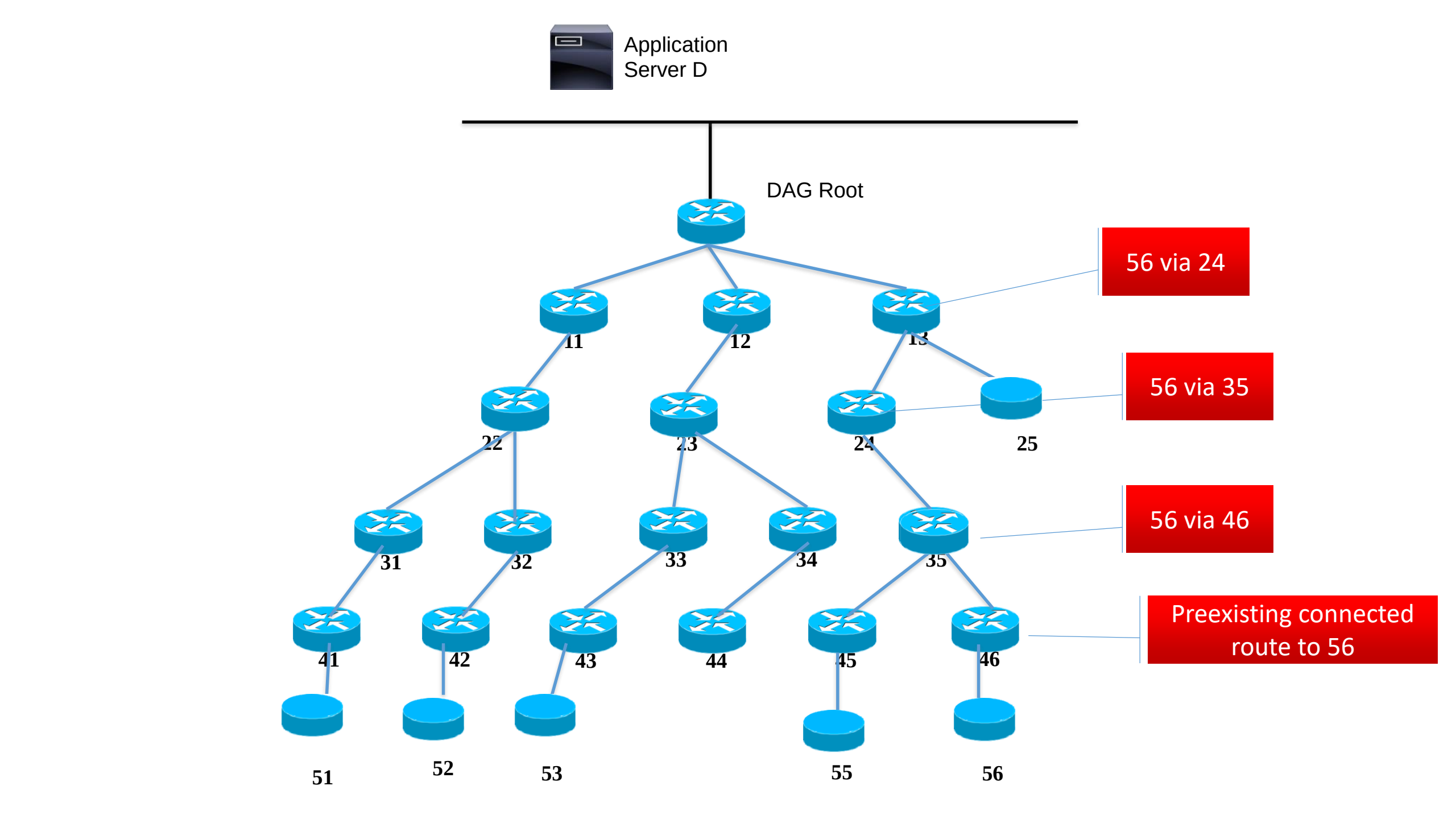

Application
Server D
DAG Root
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56
Non source routed DATA Path

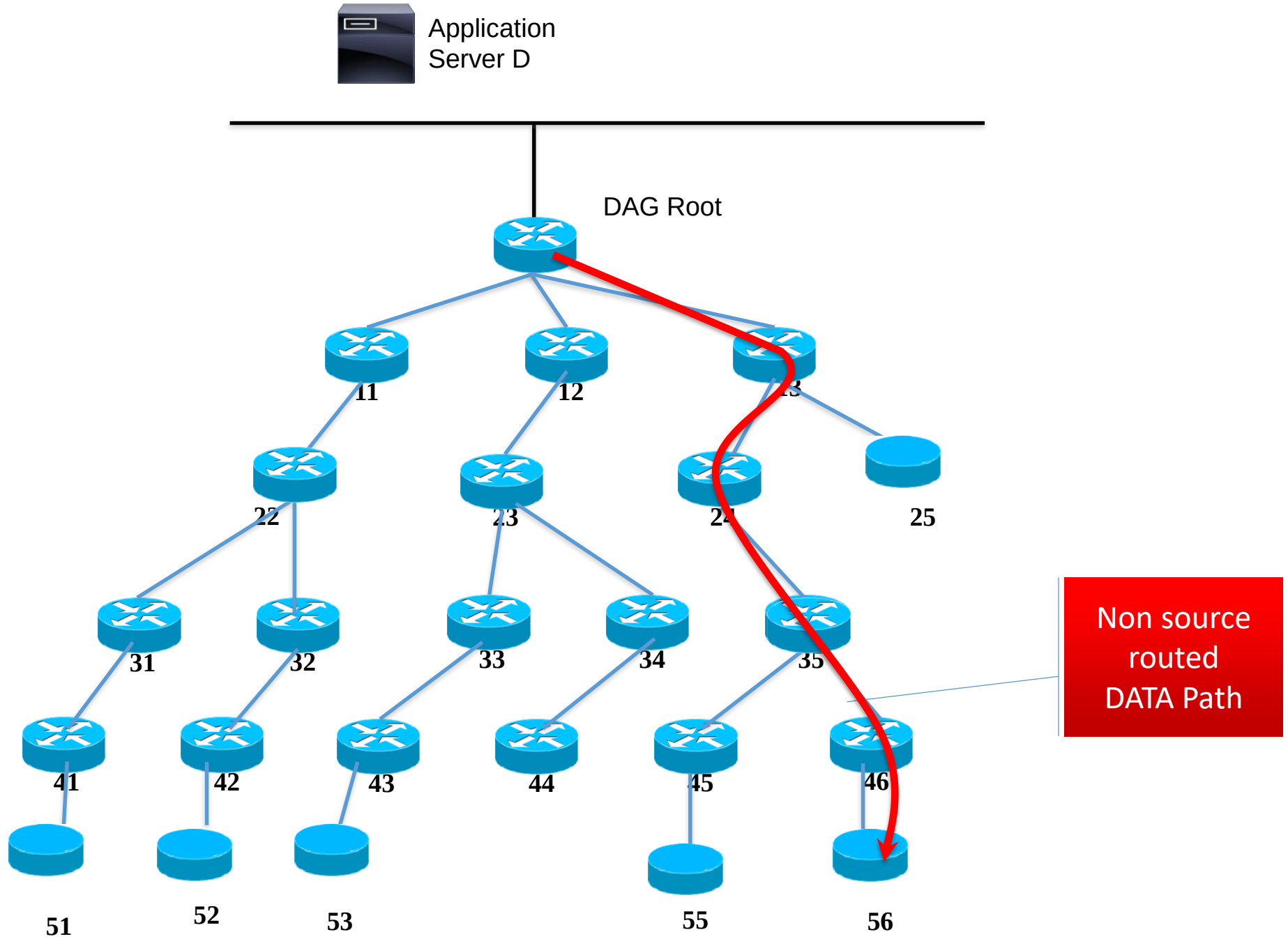

Application
Server D
DAG Root
Loose Source routed
DATA Path
Packet to 13,
RH 24, 35, 56
Non
source
routed
DATA Path
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56


Application
Server D
DAG Root
Adding New (projected) DAO with path segment unicast to target 56 via 13 (ingress), 24, and 35 (egress)
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56

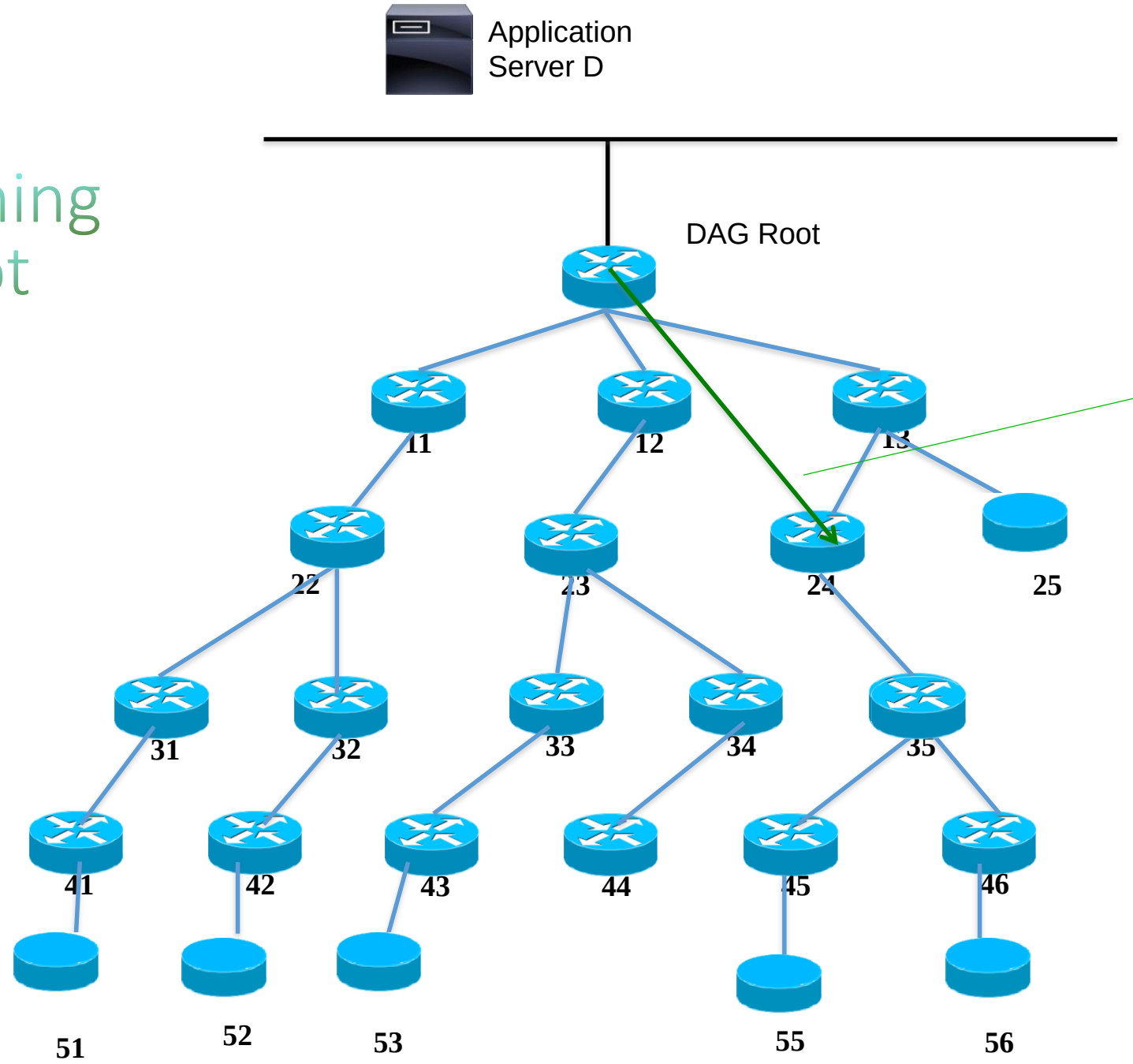

Application
Server D
DAG Root
Storing mode
DAO (forced)
with lifetime
along segment
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56


Application
Server D
DAG Root
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56
DAO-ACK (alt: non storing DAO) unicast, self 13 as parent, final destination 56 as target

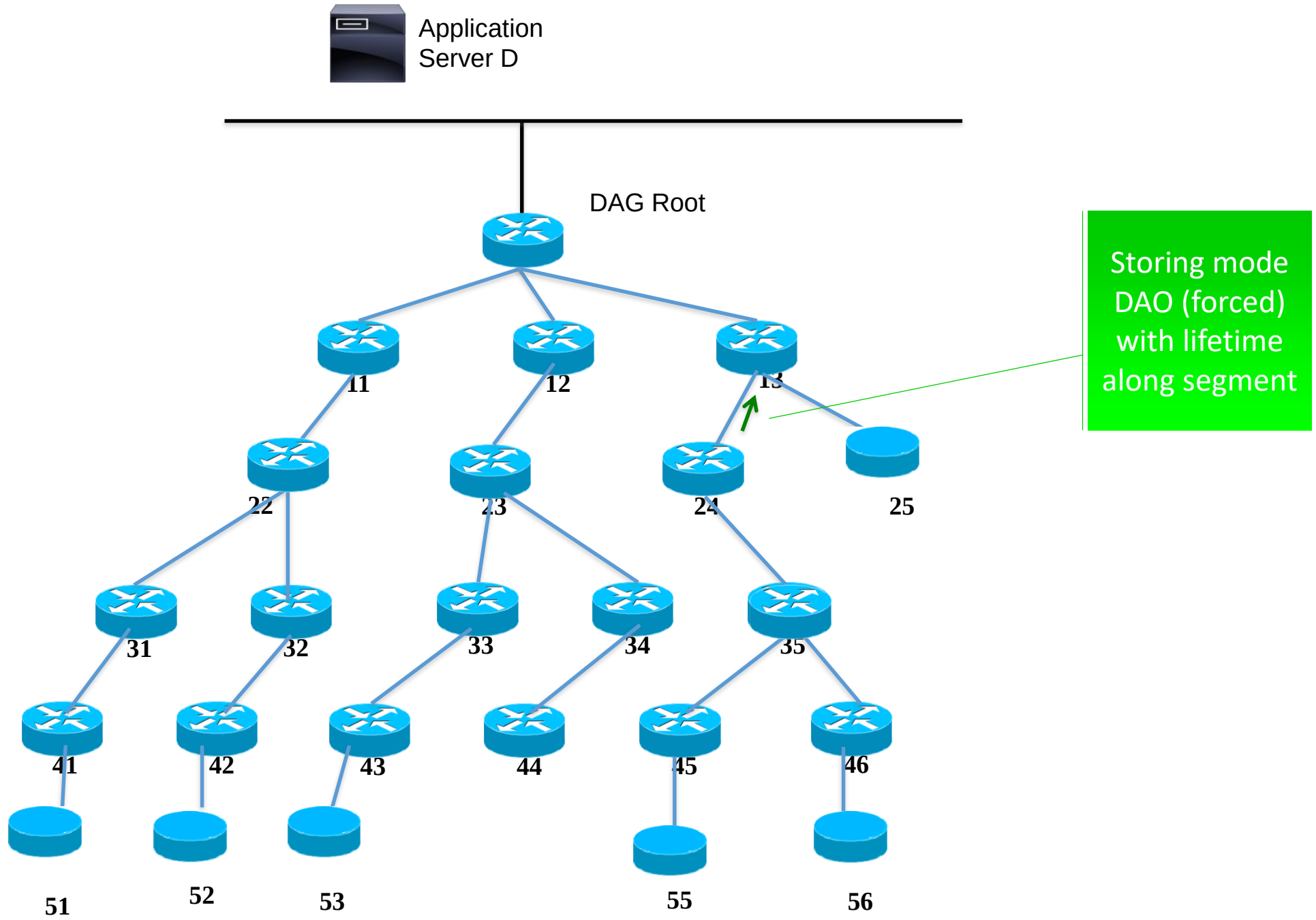

Application
Server D
DAG Root
56 via 24
56 via 35
56 via 46
Preexisting connected
route to 56
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56

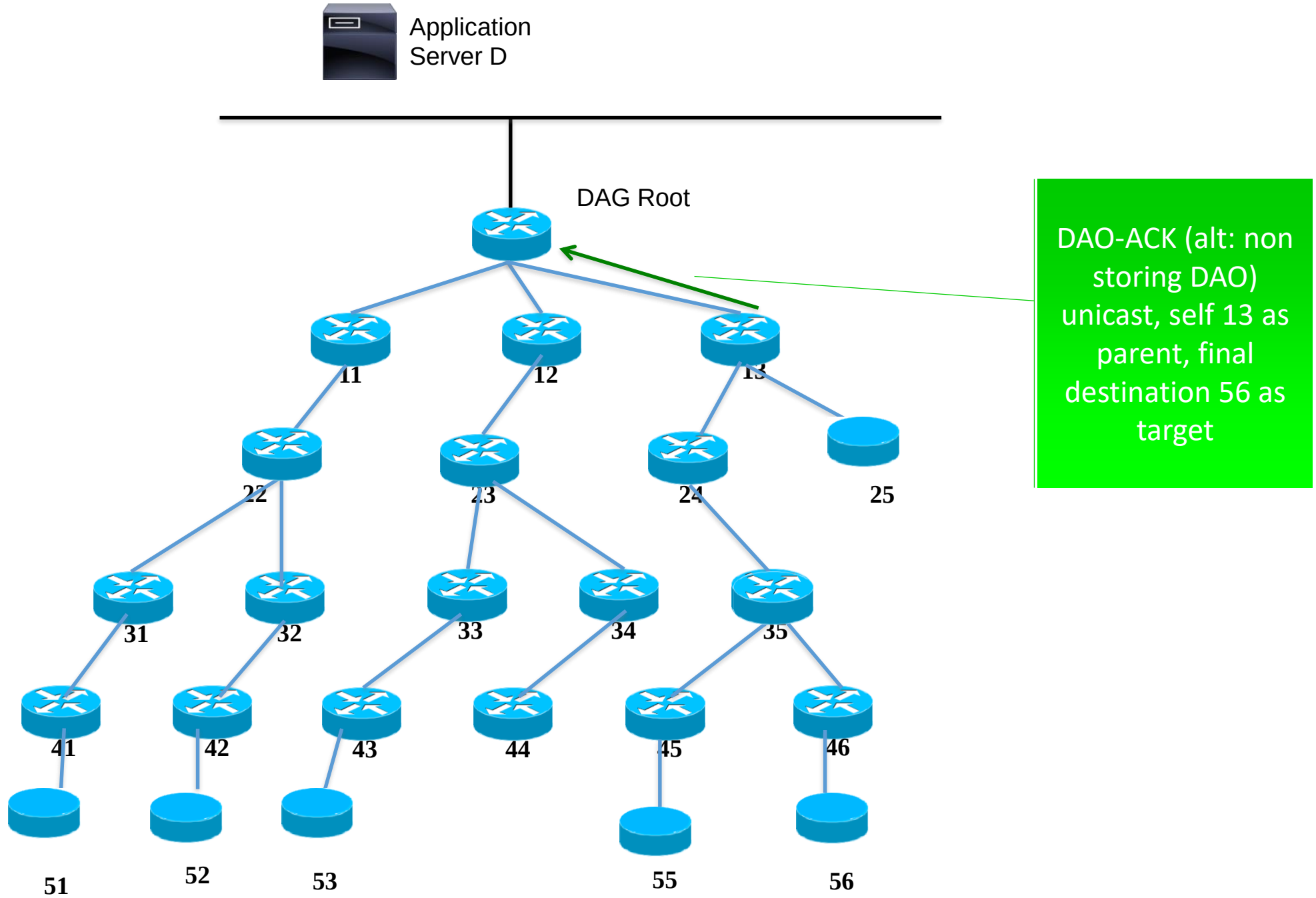

Application
Server D
DAG Root
11
12
22
23
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56
Non source
routed
DATA Path

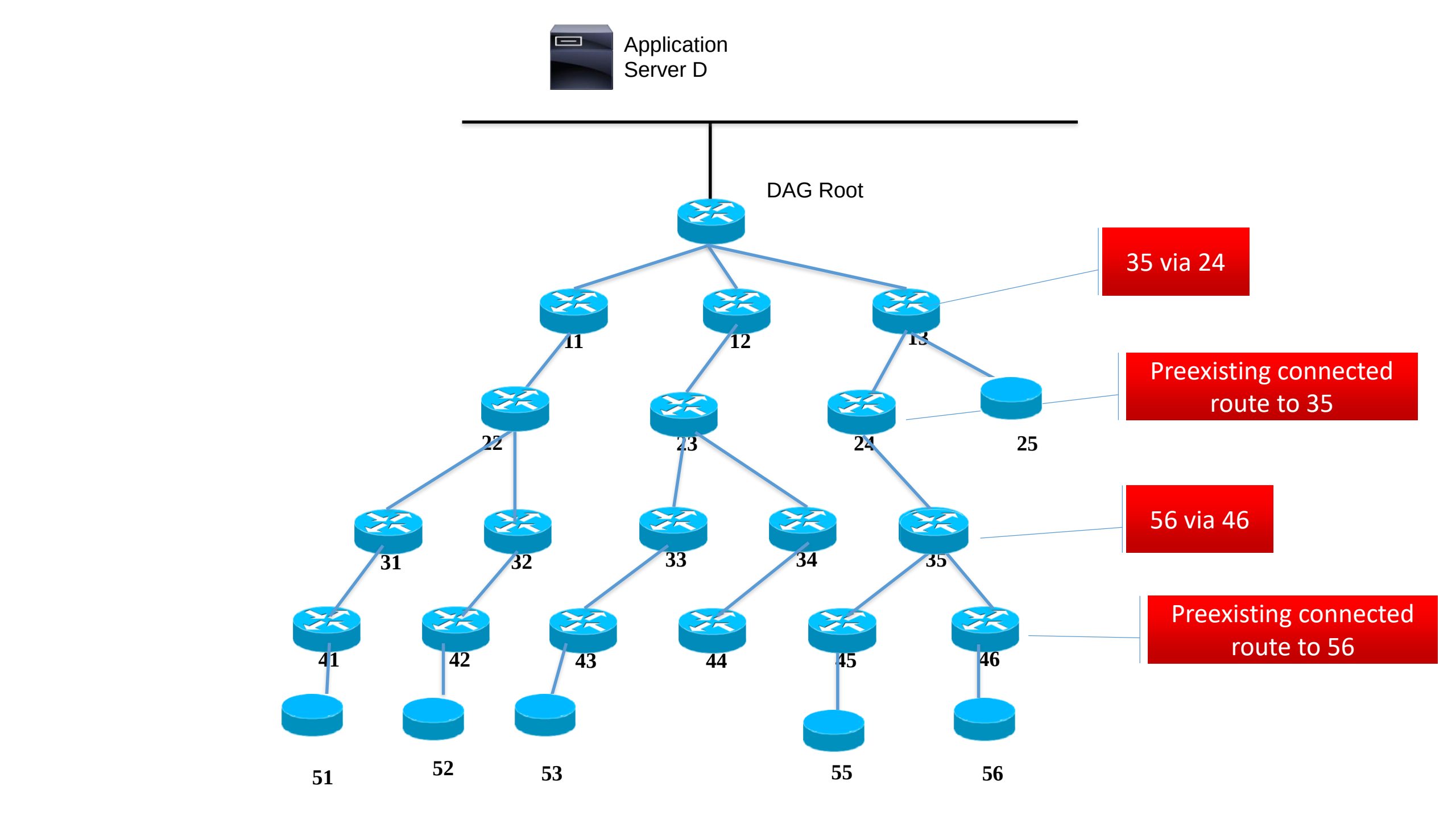
Alternate Programming By the root (Michael)

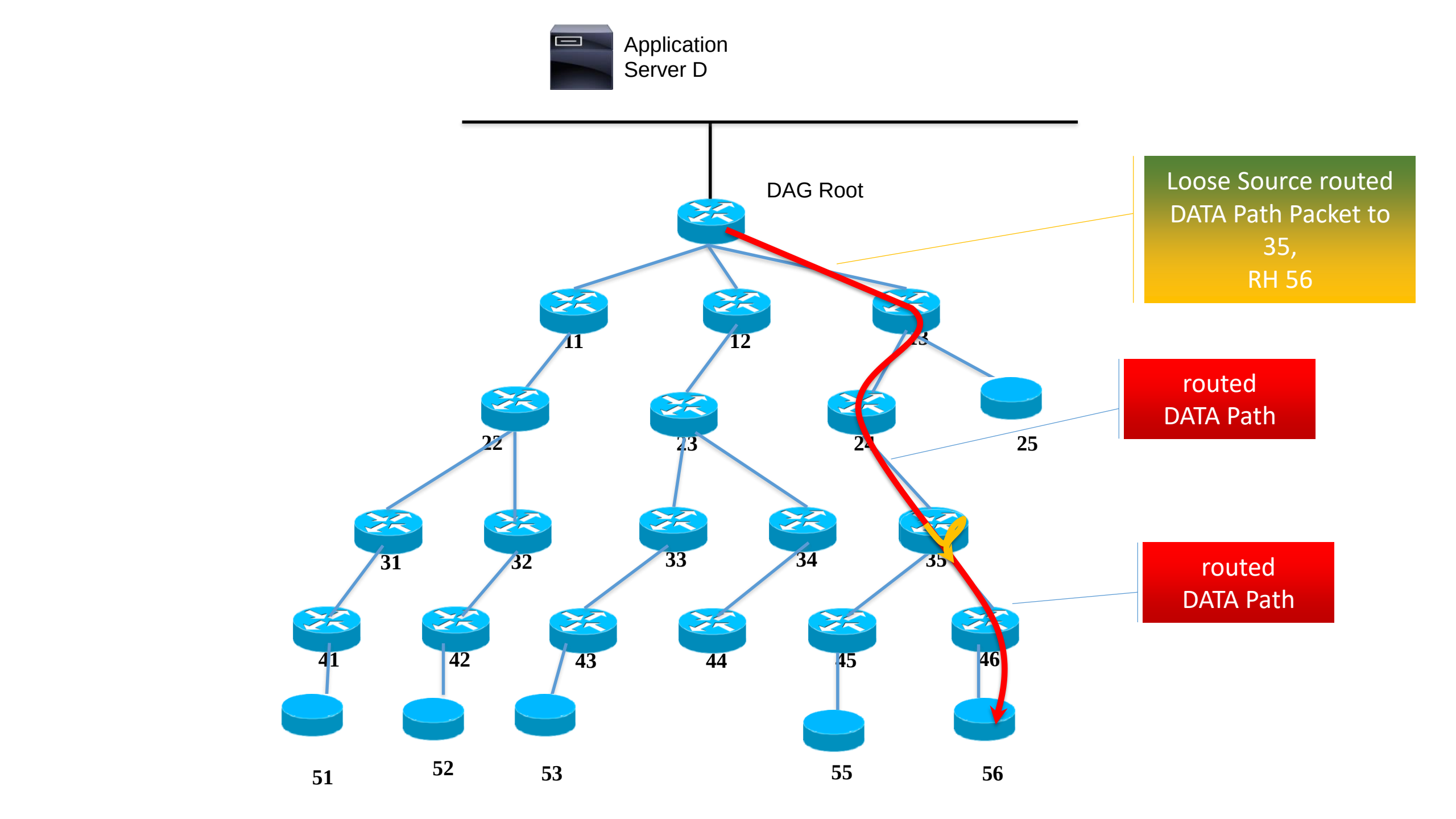


ALT: Adding New (projected) DAO with path segment unicast to target 35 via 13 (ingress) and 24 (egress)


Application
Server D
DAG Root
Storing mode
DAO (forced)
with lifetime
along segment
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56


Application
Server D
DAG Root
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56
DAO-ACK (alt: non storing DAO) unicast, self 13 as parent, final destination 56 as target


Application
Server D
DAG Root
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56
35 via 24
Preexisting connected
route to 35
56 via 46
Preexisting connected
route to 56


Application
Server D
DAG Root
Loose Source routed
DATA Path Packet to
35,
RH 56
routed
DATA Path
routed
DATA Path
11
12
22
23
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56

Routing for RPL Leaves

draft-thubert-roll-unaware-leaves-06

Pascal Thubert

IETF 104

Prague

6lo standard work

A proactive setting of proxy/routing state to avoid multicast due to reactive Duplicate address detection and lookup in IPv6 ND

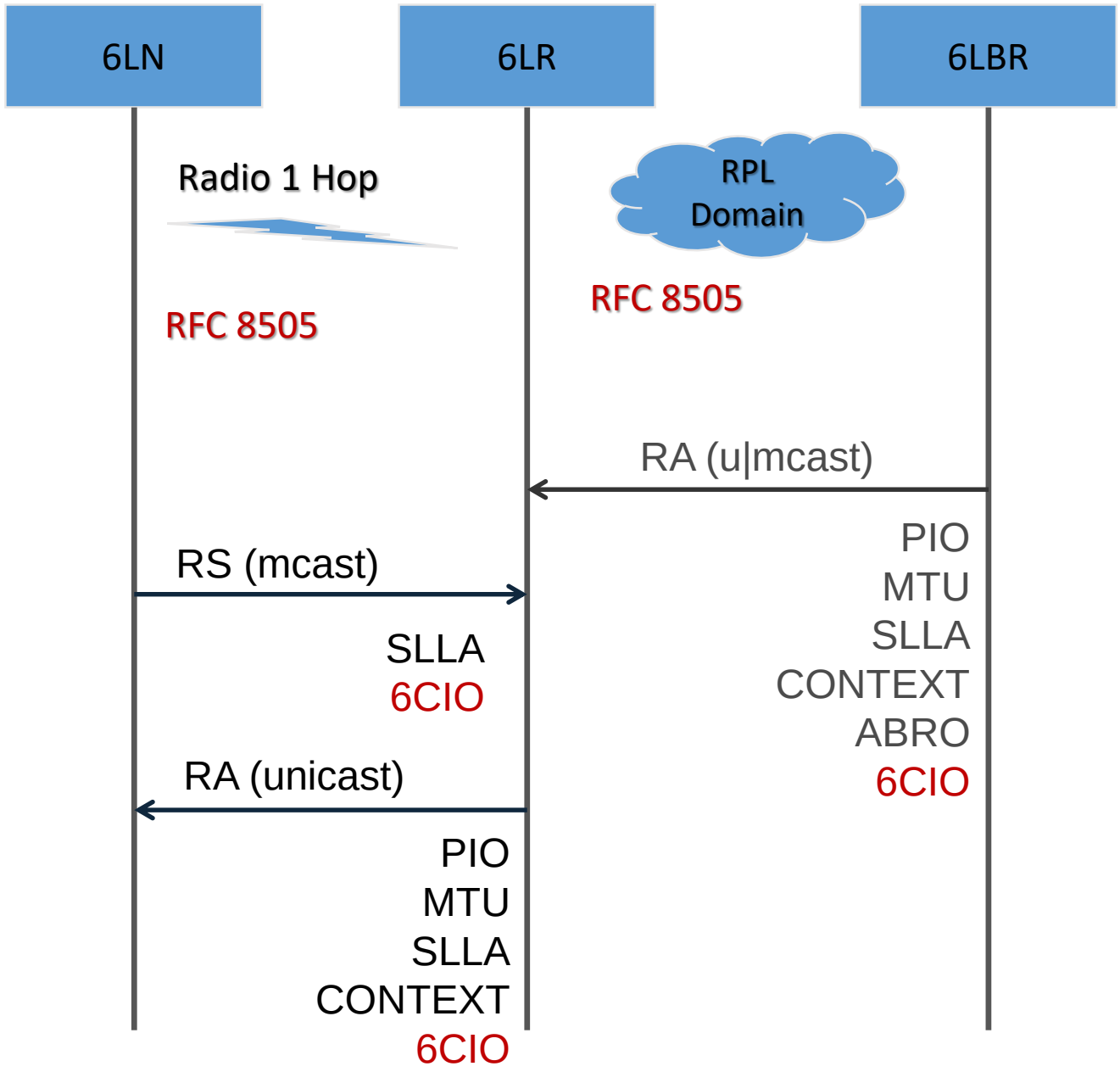
- RFC 8505 (Issued 11/2018)
  - The registration mechanism for proxy and routing services
  - Analogous to a Wi-Fi association but at Layer 3
- draft-ietf-6lo-backbone-router (WGLC complete 1/25)
  - Federates 6lo meshes over a high speed backbone
  - ND proxy analogous to Wi-Fi bridging but at Layer 3
- draft-ietf-6lo-ap-nd (WGLC complete 3/26)
  - Protects addresses against theft (Crypto ID in registration)
- draft-thubert-6lo-unicast-lookup (new draft)
  - Provides a 6LBR on the backbone to speed up DAD and lookup

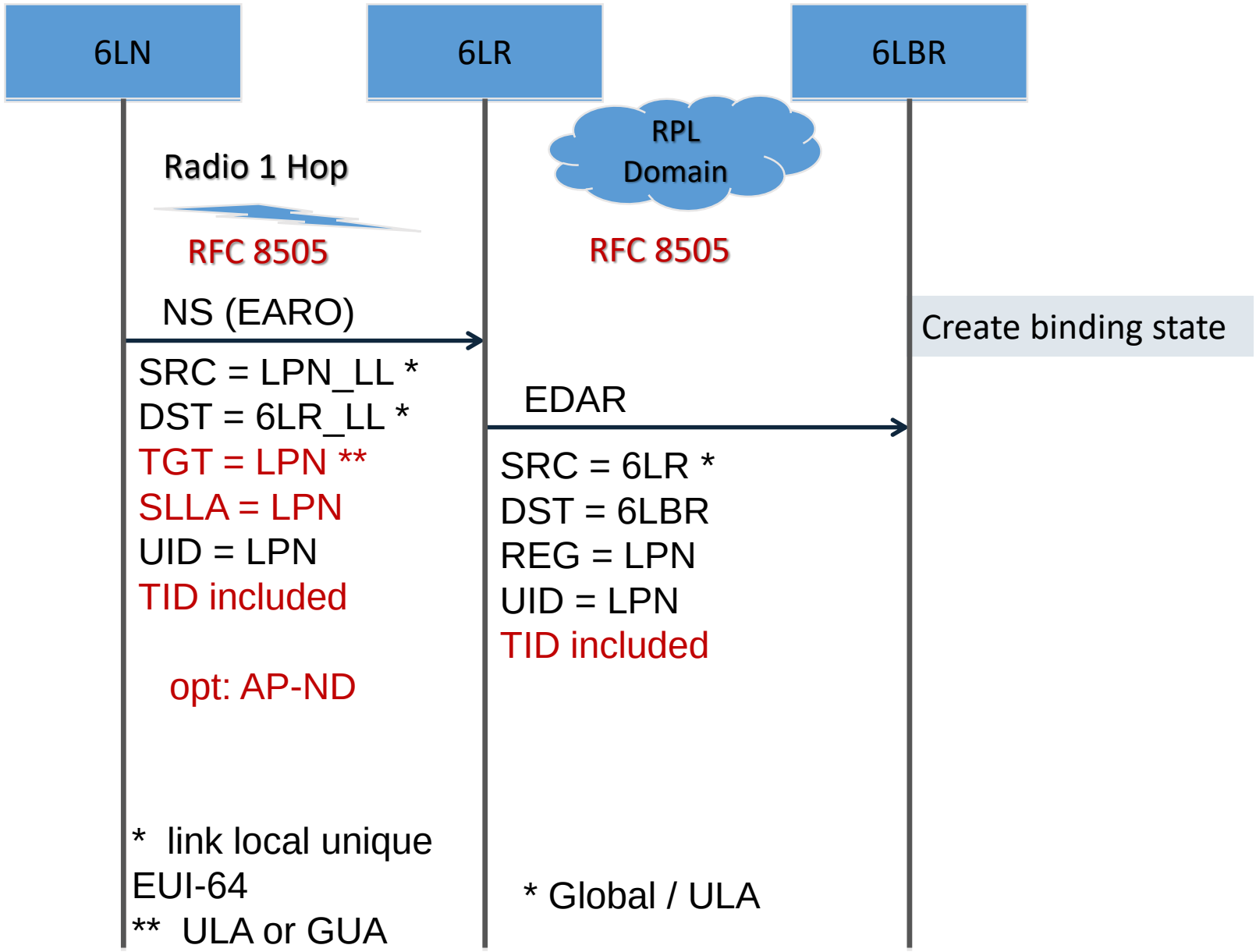
Unmet expectations

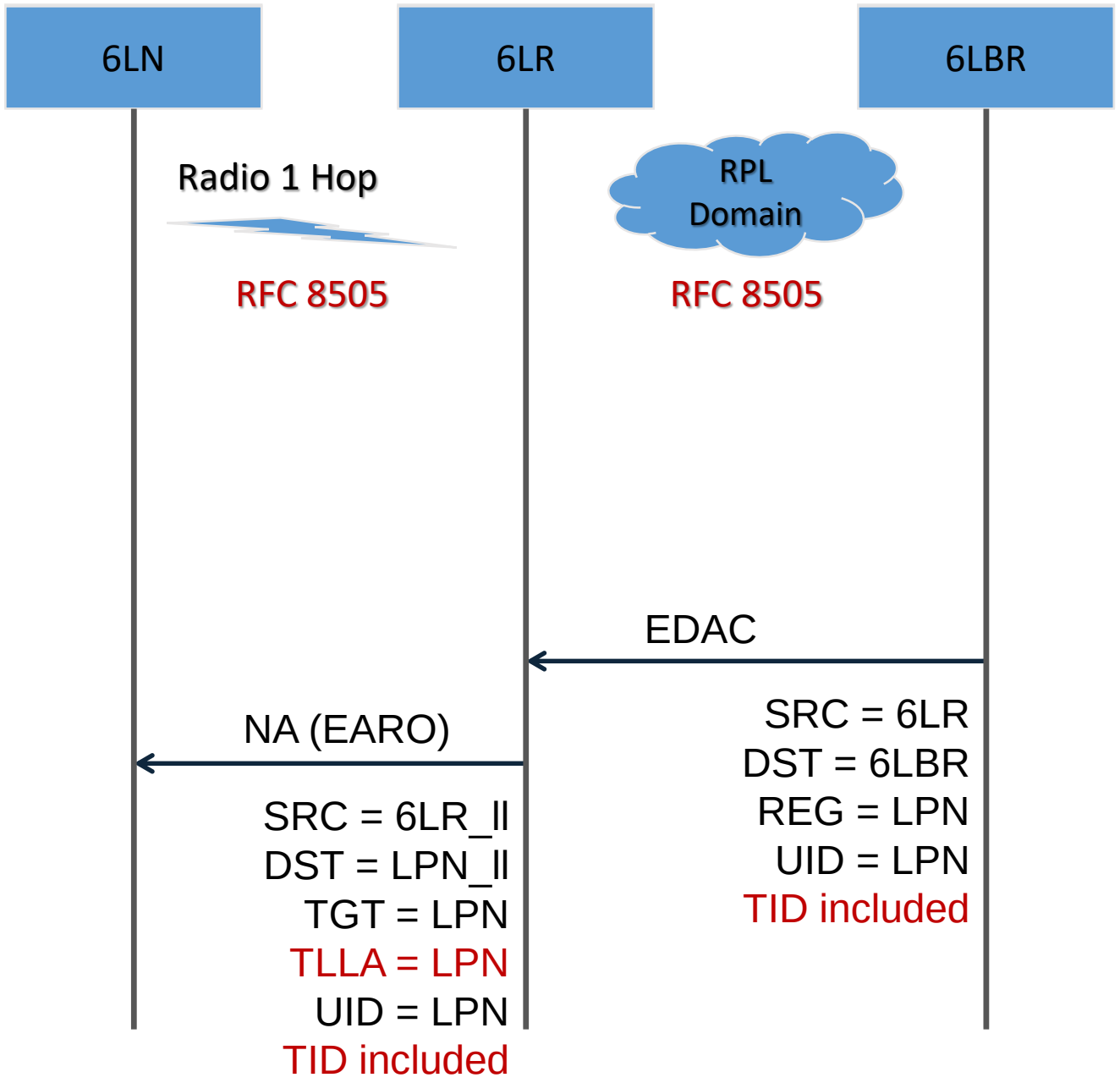
- Connectivity for a Non-RPL aware node in a RPL domain
  - Forwarding is described but not the control plane
- Integration of the EDA Exchange (EDAR/EDAC) used as keep-alive with the RPL signaling to avoid duplication
  - At the moment both are needed periodicallThis spec uses a common lifetime and the EDA exchange is proxied
- Separation of the RPL Root and the 6LBR and proxy registration to the 6BBR
  - The RPL root proxies the EDA with the 6LBR and the NS(EARO) with the 6BBR
  - Do we really want that actually?

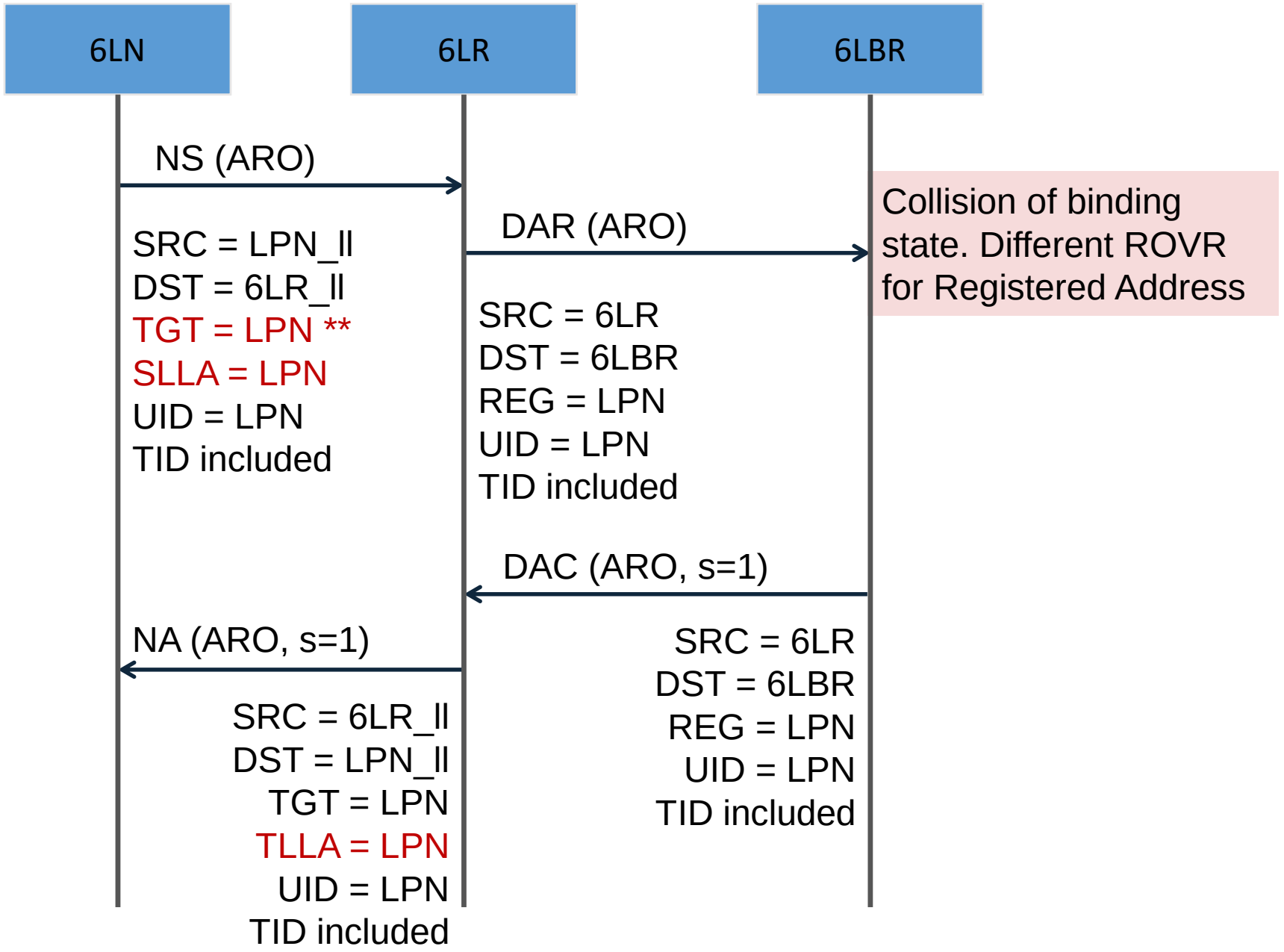
RFC 8505

P. Thubert, E. Nordmark, S. Chakrabarti, C. Perkins


6LN
6LR
6LBR
Radio 1 Hop
RPL
Domain
RFC 8505
RFC 8505
RA (u|mcast)
PIO
MTU
SLLA
CONTEXT
ABRO
6CIO
RS (mcast)
SLLA
6CIO
RA (unicast)
PIO
MTU
SLLA
CONTEXT
6CIO


6LN
6LR
6LBR
RPL
Domain
Radio 1 Hop
RFC 8505
RFC 8505
NS (EARO)
SRC = LPN_LL *
DST = 6LR_LL *
TGT = LPN **
SLLA = LPN
UID = LPN
TID included
opt: AP-ND
EDAR
SRC = 6LR *
DST = 6LBR
REG = LPN
UID = LPN
TID included
Create binding state
* link local unique EUI-64
** ULA or GUA
* Global / ULA


6LN
6LR
6LBR
Radio 1 Hop
RFC 8505
RPL
Domain
RFC 8505
EDAC
SRC = 6LR
DST = 6LBR
REG = LPN
UID = LPN
TID included
NA (EARO)
SRC = 6LR_ll
DST = LPN_ll
TGT = LPN
TLLA = LPN
UID = LPN
TID included


6LN
6LR
6LBR
NS (ARO)
SRC = LPN_ll
DST = 6LR_ll
TGT = LPN **
SLLA = LPN
UID = LPN
TID included
DAR (ARO)
SRC = 6LR
DST = 6LBR
REG = LPN
UID = LPN
TID included
Collision of binding state. Different ROVR for Registered Address
DAC (ARO, s=1)
SRC = 6LR
DST = 6LBR
REG = LPN
UID = LPN
TID included
NA (ARO, s=1)
SRC = 6LR_ll
DST = LPN_ll
TGT = LPN
TLLA = LPN
UID = LPN
TID included

draft-thubert-roll-unaware-leaves

P.Thubert

IETF 104

Prague

Terminology

- RFC 6550:
  - A RPL leaf may understands RPL
  - But does not act as a router
- This draft:
  - A RPL-unaware leaf does not implement anything specific to RPL,
  - but it MUST support RFC 8505,
  - and it MUST ask the 6LR for abstract reachability services

Status

- Added support for Multicast addresses
  - Requires RPL storing mode + multicast support
  - Maps MLD Reports onto DAO
- Added text on applicability to smartgrid
- Dependency on draft-rfc6775-update gone (now RFC 8505)
- Otherwise stable…
- Time for adoption…

Notes on the ‘R’ flag (defined in RFC 8505)

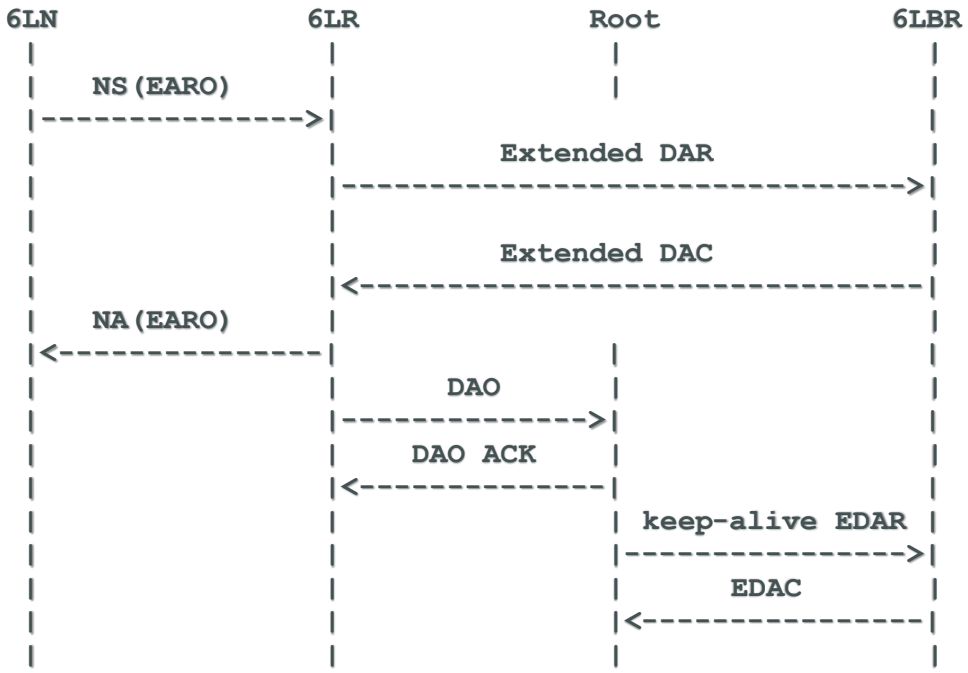
- A RPL Unaware Leaf does not know that there is routing in place and that the routing is RPL; draft-thubert-roll-unaware-leaves does not require anything from the Leaf.
- RFC 8505 specifies a new flag in the EARO, the 'R' flag.
- If the 'R' flag is set, the Registering Node expects that the 6LR ensures reachability for the Registered Address, e.g., by means of routing or proxying ND.
- Conversely, when it is not set, the 'R' flag indicates that the Registering Node is a router, which for instance participates to RPL and that it will take care of injecting its Address over the routing protocol by itself.
- A 6LN that acts only as a host, when registering, MUST set the 'R' to indicate that it is not a router and that it will not handle its own reachability.
- A 6LR that manages its reachability SHOULD NOT set the 'R' flag; if it does, routes towards this router may be installed on its behalf and may interfere with those it injects.

Mapping Fields from RPL DAO to NS(EARO) and EDA

- The Registered Address in a RPL Target Option is a direct match to the Registered Address field of the EDAR message and in the Target field of the NS, respectively
- EARO's TID is a direct match to Path Sequence in Transit Information option (TIO)
- EARO's opaque field carries the RPLInstanceID, 0 means 6LR's default
- EARO's Lifetime unit is 60s. RPL uses Lifetime Units that is passed in the DODAG Configuration Option. Converting EARO to DAO and back requires mapping of units.
- The Registration Ownership Verifier (ROVR) field in keep-alive EDAR messages by the Root is set to 64-bits of all ones to indicate that it is not provided. It is obtained in the EDAC from the 6LBR and used in proxy registration.
  - Q: Should we carry it in a RPL option in DAO messages?

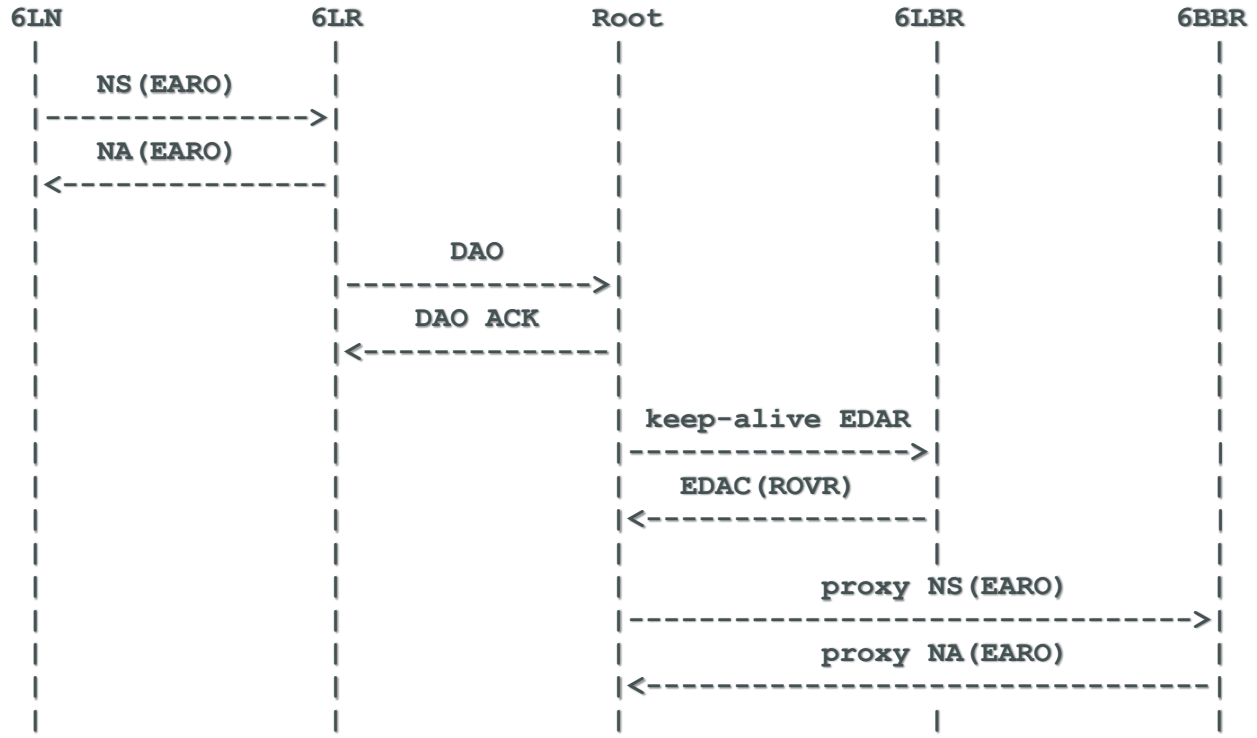
First registration

- Upon the first registration, the EDAR / EDAC populates a state in the 6LBR including the ROVR field and the 6LR sends a first DAO message.
- The RPL Root acts as a proxy on behalf of the 6LR upon the reception of the DAO propagation initiated at the 6LR. Should we allow splitting from the 6LBR, e.g.:



EDA (DAR, DAC) message Proxying

- Upon the renewal of a 6lowPAN ND registration: if the 'R' flag is set, the 6LR injects a DAO targeting the Registered Address, and refrains from sending a DAR message.
- With a Root/6LBR split that could give:



RPL DAG Metric Container Node State and Attribute object type extension

draft-ietf-roll-nsa-extension-01

Remous-Aris Koutsiamanis
Georgios Z. Papadopoulos
Nicolas Montavont
Pascal Thubert

ROLL@IETF104

Updates on adopted -01 version

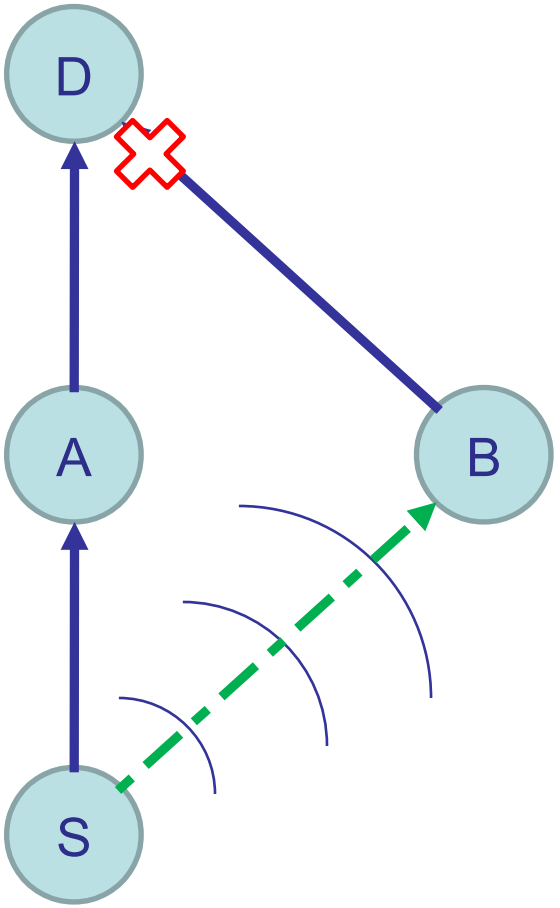
- Changes addressing Rahul's review (many thanks!)
  - Removed strict dependency on 6TiSCH
  - Explained Alternative Parent (AP) selection via Common Ancestor
  - Allowed flexible PS usage (any AP method)
  - Allowed flexible AP usage (nodes may use different AP methods)
  - Allowed flexible PRE (paths with different reliability requirements)
  - Added Appendix with Implementation status + results
- Updated Contiki implementation
  - to do 6LoRH-based IPv6 address compression of Parent Set
  - Selected reference address to be the DODAG ID
  - Added example in code

Brief Overview

- Goal: “Determinism”
  - High reliability
  - Low jitter
  - Bounded delay
- How:
  - Send multiple copies of packets (PRE) over different paths
- Challenges:
  - Avoid flooding with copies
  - Keep jitter low
- This draft:
  - Information in DIO to help with challenges

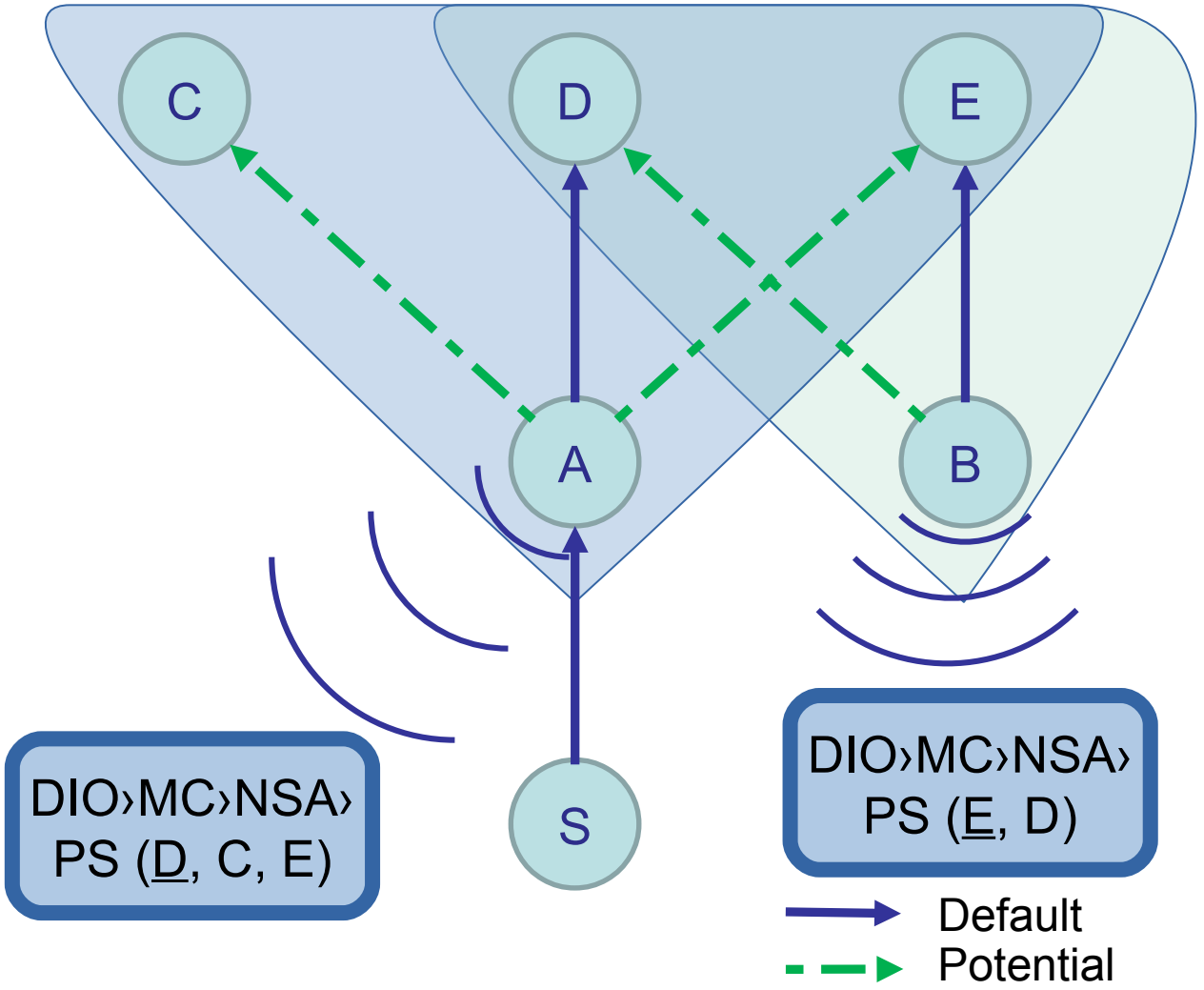
PRE over 6TiSCH

- Low jitter → bounded delay
- Reliable communication
- Packet Replication Elimination
  - Replication
  - Elimination
  - Promiscuous Overhearing (optionally)

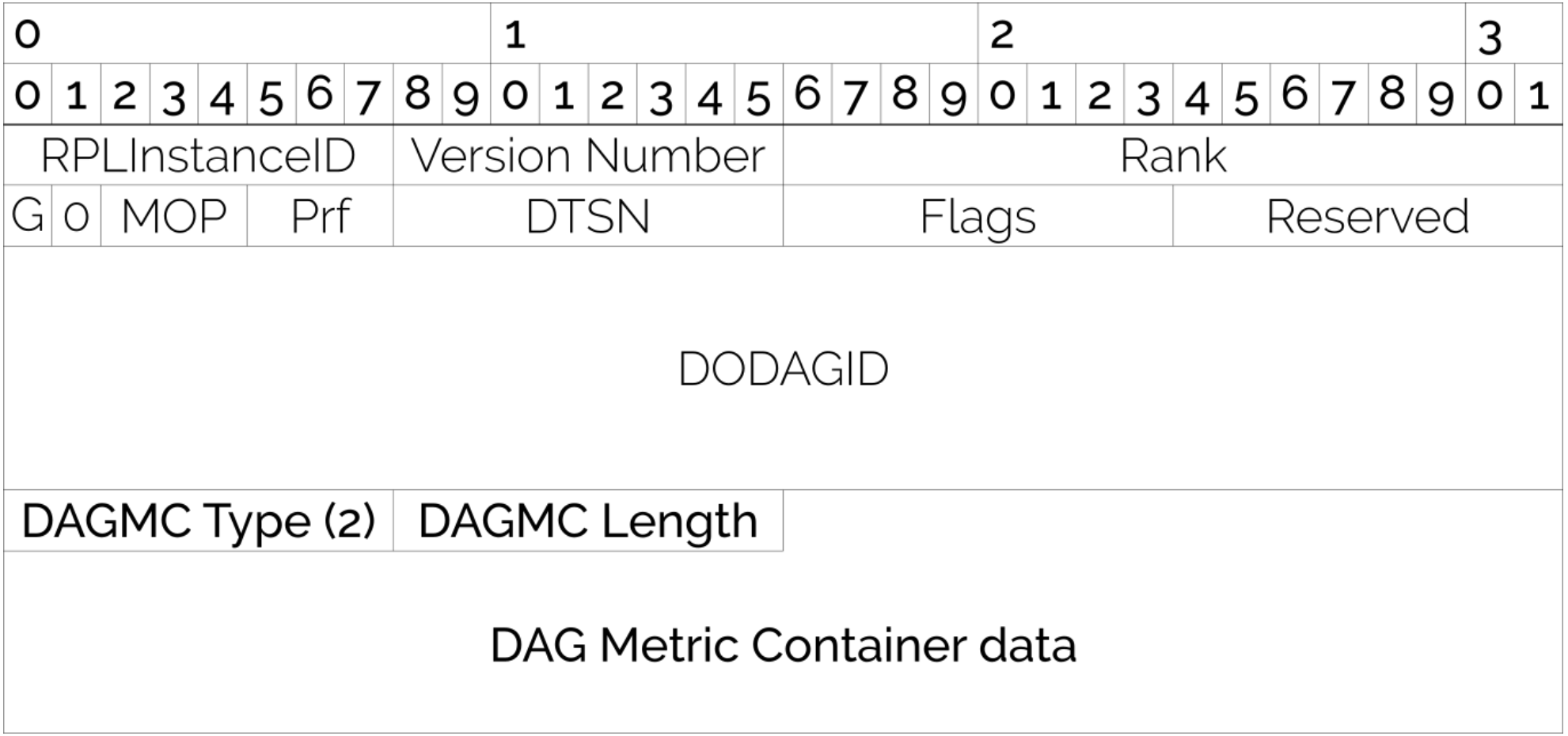


Parent Selection - DIO Messages

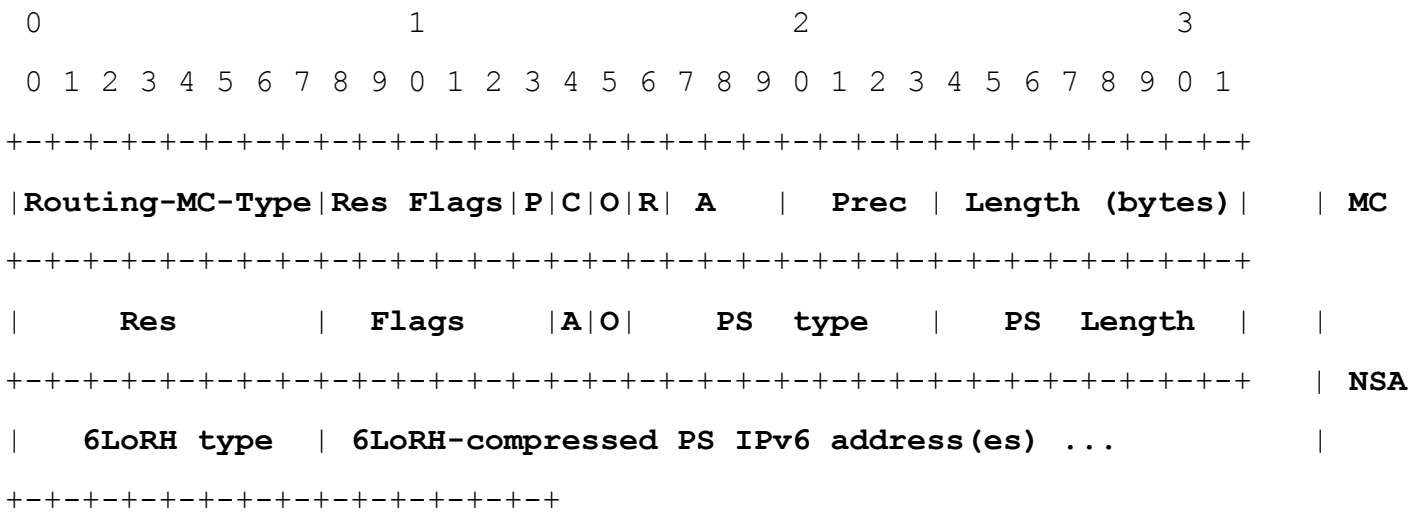
- Parent Set A:
  - {D, C, E}
- Parent set B:
  - {E, D}



DIO Format Example



MC Format Example (1)



- Parent Set (PS)
  - Node State and Attributes Option
  - PS type = 1 (8 bits)
  - PS Length = # of PS addresses x IPv6 address size (8 bits)
  - 6LoRH type = 1 (8 bits) As in Source Routing Header 6LoRH - RFC8138
  - PS IPv6 addresses = 1 or more 6LoRH compressed IPv6 addresses

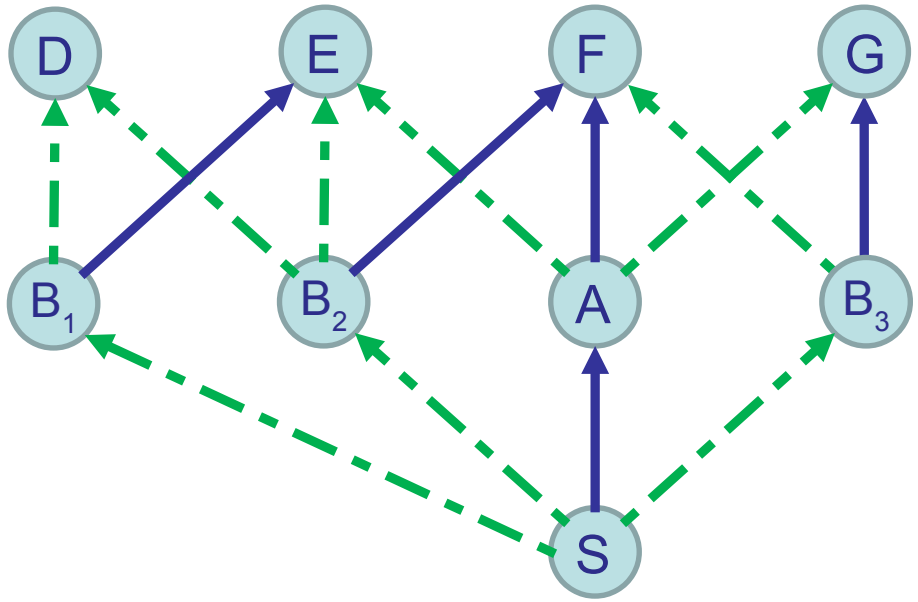
6LoRH-based PS address compression

6LoRH Type	Length of compressed IPv6 address (bytes)
0	1
1	2
2	4
3	8
4	16

- 6LoRH type
  - Indicates size of omitted IPv6 prefix (RFC8138)
  - Used for all the compressed IPv6 addresses → Lowest common denominator compression level
  - Compression uses common DODAG ID for reference address (But, what if multiple DODAGs?)

Implementation of CA: Medium

- PP(PP) ∈ PS(AP)
  - PP(A) = F
  - PS(B_1) = (E, D) ✖
  - PS(B_2) = (F, D, E) ✔
  - PS(B_3) = (G, F) ✔



Preferred Parent (PP)

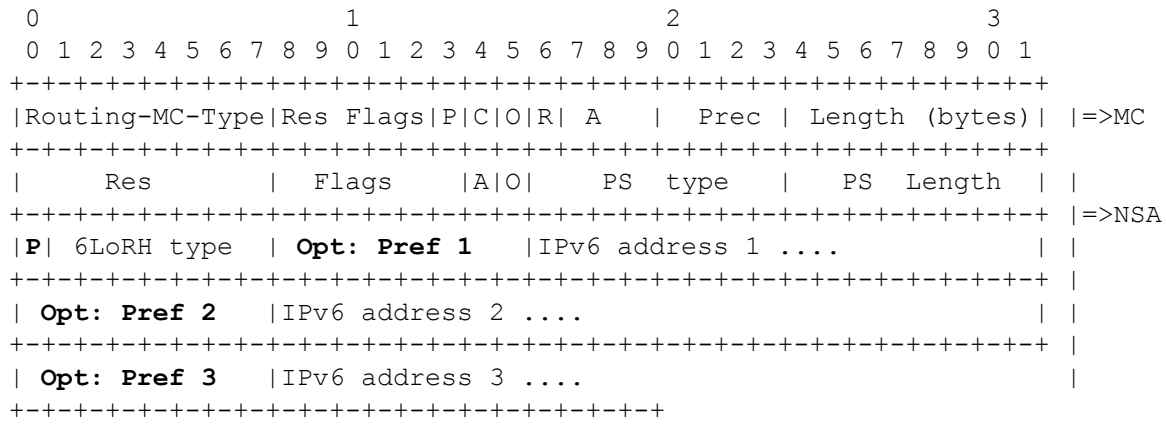
Alternative Parent (AP)

Road Forward (1)

- Adopted at the last IETF meeting
- Code is available here:
  - Contiki NSA extension https://github.com/ariskou/contiki/tree/draft-ietf-roll-nsa-extension
  - Wireshark dissectors (for the optional TLV, i.e., PS): https://code.wireshark.org/review/gitweb?p=wireshark.git;a=commit;h=e2f6ba229f45d8ccae2a6405e0ef41f1e61da138
    Pending: decode 6LoRH compressed addresses

Road Forward (2)

- We received and addressed reviews from Rahul. Pending issues:
  - Draft scope: Prescribe specific AP selection method?
  - Mandatory 6LoRH compression: 6LoRH type → 1 byte overhead
  - Extend each parent with optional *priority value* to quantify preference



  - We are looking for more reviews
- Next steps?

Thanks!

Questions?

Traffic-Aware Objective Function

draft-koutsiamanis-roll-traffic-aware-of-00

Remous-Aris Koutsiamanis aris@ariskou.com
Georgios Z. Papadopoulos
Eduardo Ingles Sanchez
Chenyang Ji
Diego Dujovne
Nicolas Montavont

ROLL@IETF104

New since -03

- Changes addressing Rahul's review (many thanks!)
  - THROUGHPUT_PERIOD → THROUGHPUT_WINDOW (it is a slinding window)
  - Made configuration more flexible
    - External configuration
    - Hard-coded values
    - Optional TLV to carry THROUGHPUT_WINDOW
    - Optional TLV to carry THROUGHPUT_WINDOW_UNIT
  - Refer to Oscillations issue (re: Pascal) (But, no solution proposed)
  - Defined behaviour of P,C,O,R,A bits for the metric
  - Address additive semantics

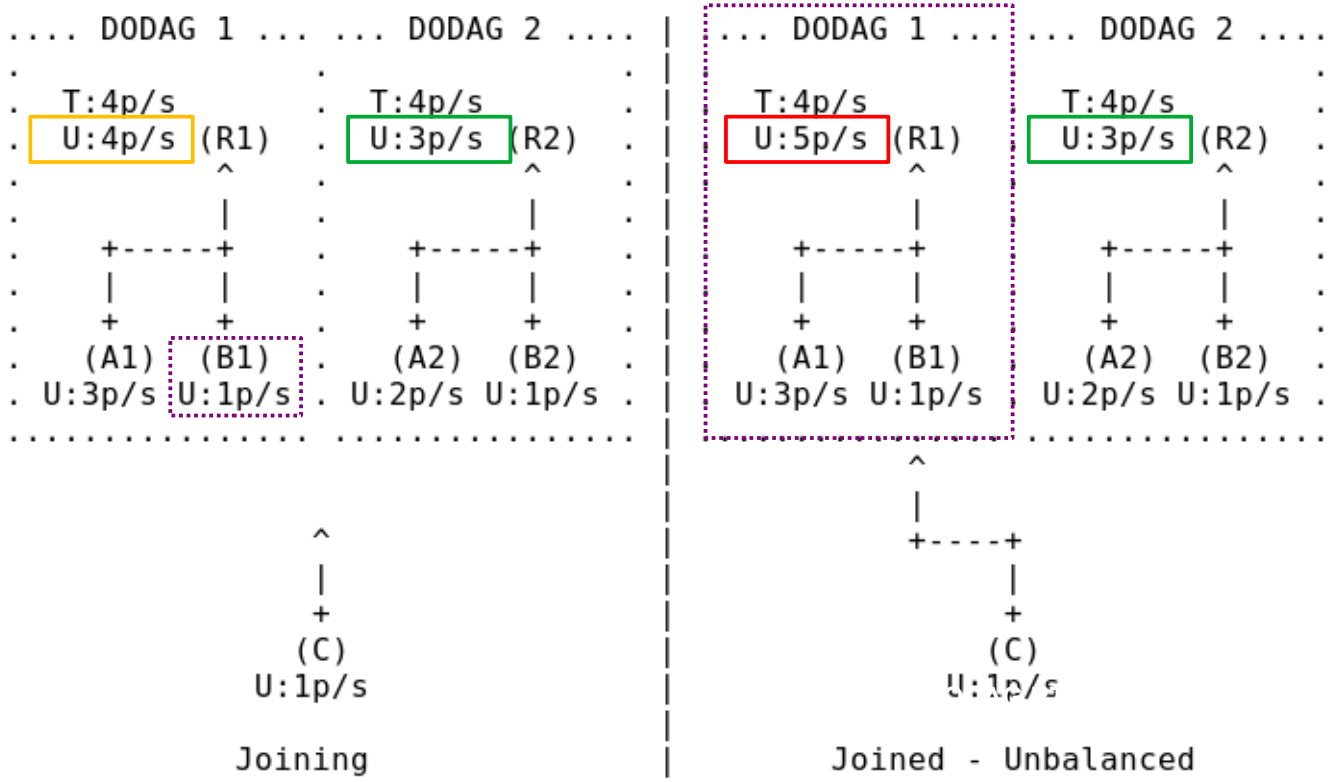
Standardisation Efforts

- Objective Function → Preferred Parent
  - OF0
    - ETX
    - No hysteresis
  - MRHOF
    - ETX, Energy, other additive metrics
    - Hysteresis
  - *Load balanced OF (LB-OF)*
    - Uses parent's child-count
    - Hysteresis

Problem statement

- Using standard OFs (OF0, MRHOF) leads to unbalanced network:
  - Some nodes overloaded (forwarding)
  - Some DODAGs overloaded (forwarding)
  - Lower network and node lifetime
  - Higher packet losses (queueing)
  - Higher packet delay (queueing)

DODAG Selection Example (1)

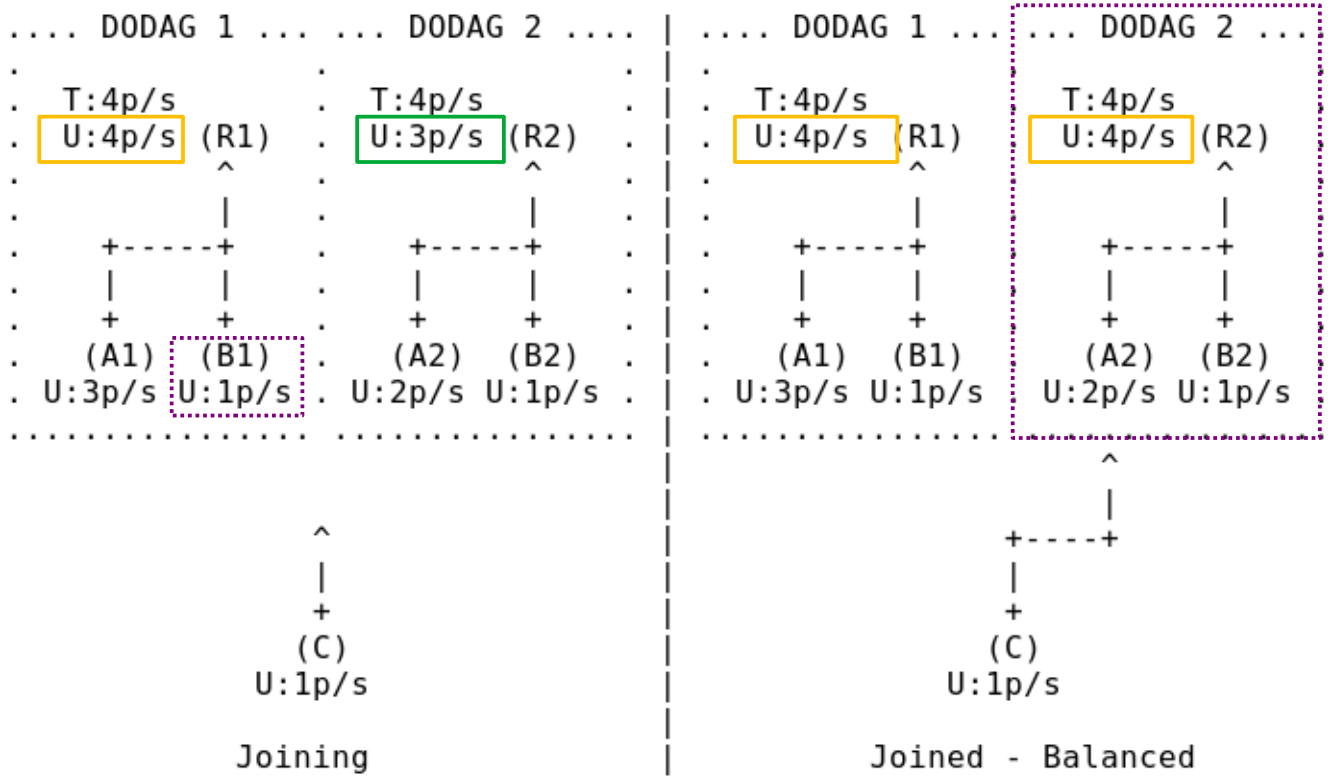


T → Total Throughput Capacity

U → Used Throughput Capacity

DODAG selection without RT leading to unbalanced traffic

DODAG Selection Example (2)



T → Total Throughput Capacity

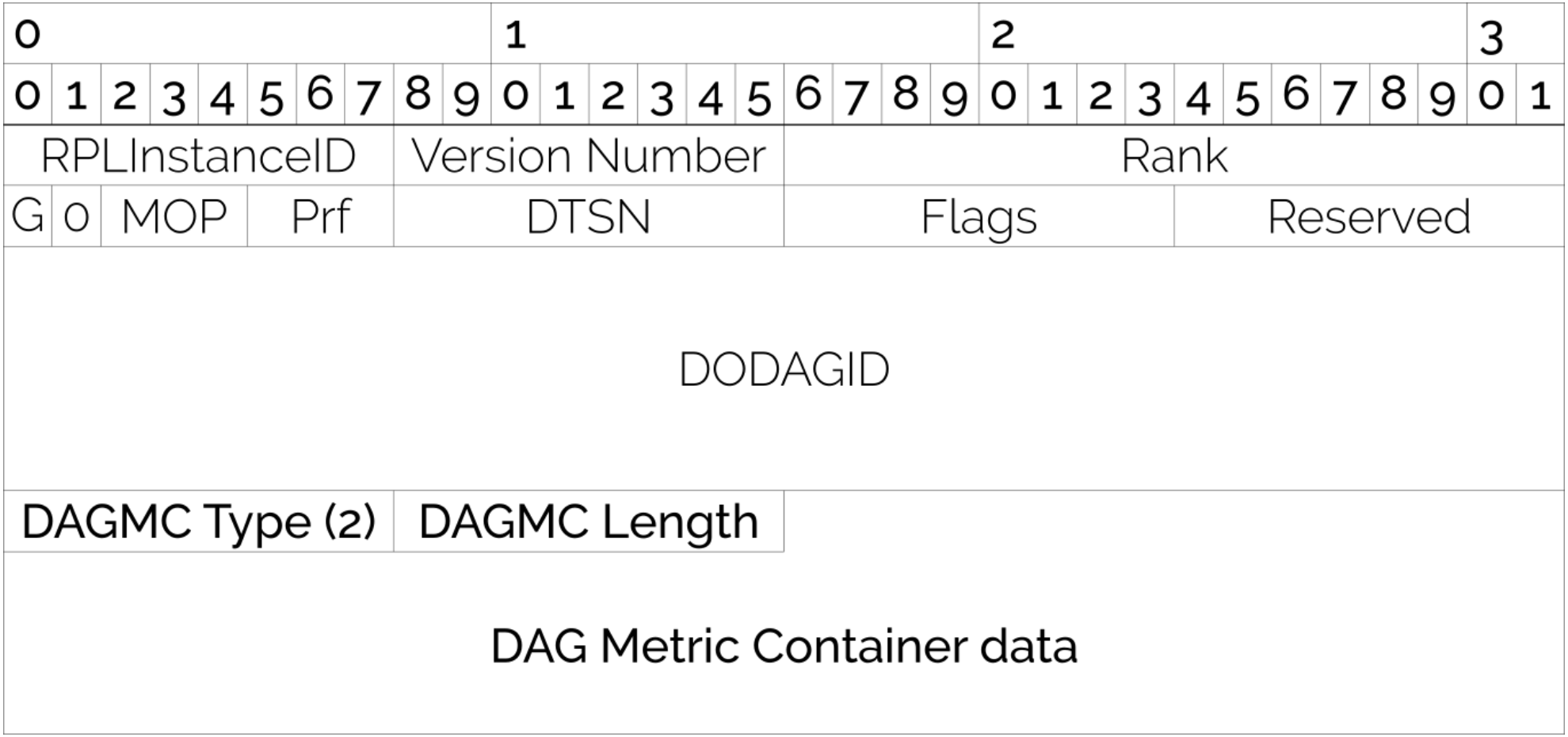
U → Used Throughput Capacity

- DODAG selection with RT leading to balanced traffic

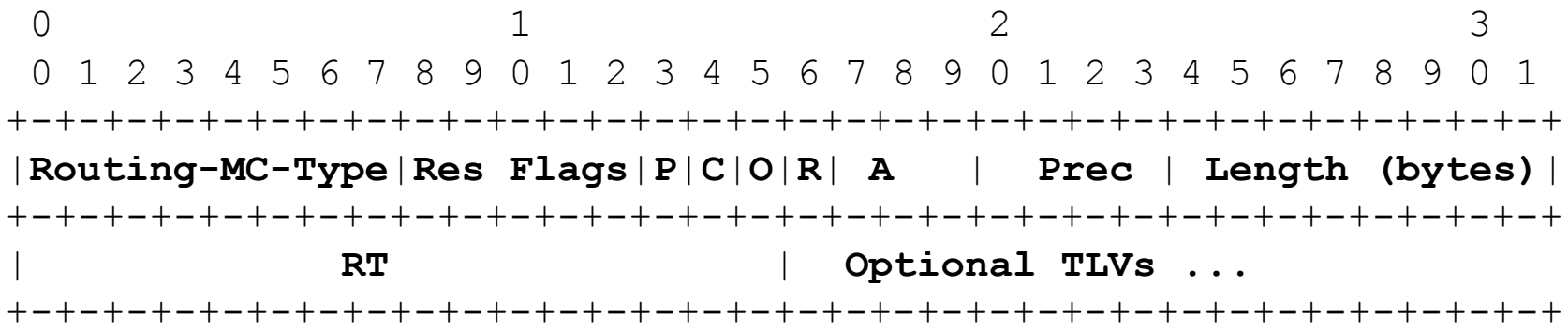
Traffic-Aware OF

- New Remaining Throughput (RT) metric
  - Aggregated → Use DAGMC.A=MINIMUM
  - Sliding window tracking of used throughput capacity
    - Window size: THROUGHPUT_WINDOW (time)
  - Calculate RT (in THROUGHPUT_WINDOW)
    - Total Throughput Capacity – Used Throughput
- Traffic-Aware OF
  - Highest RT → Preferred parent
  - Use with MRHOF-style hysteresis

DIO Format Example

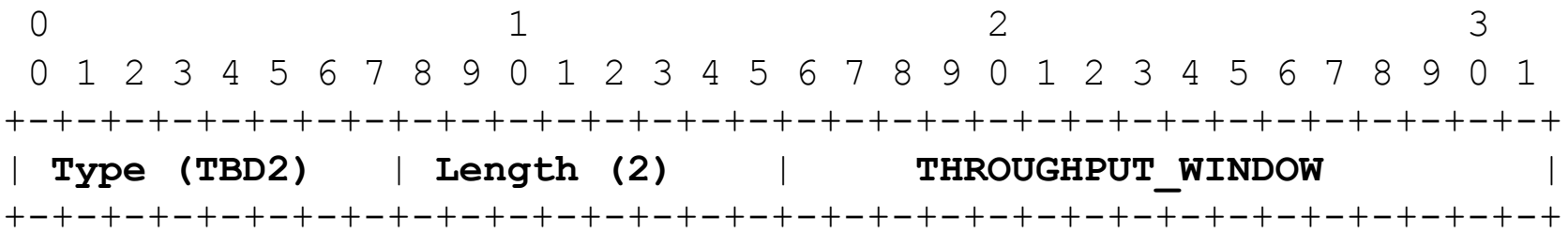


Remaining Throughput



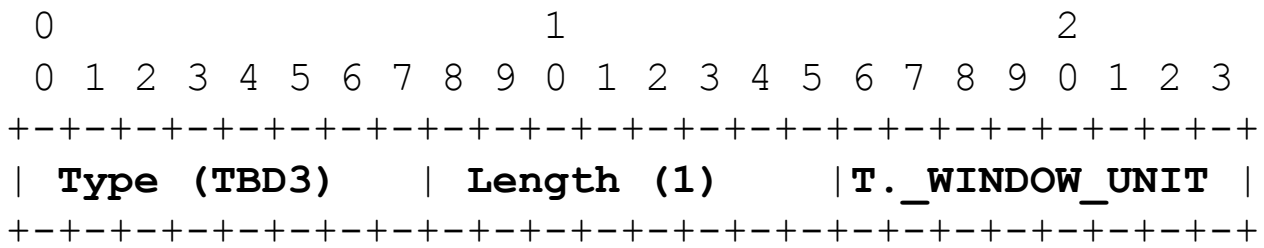
- Node Metric Object (RT)
  - 2 octets – unsigned integer
  - A = 2 = Minimum
  - C = Constraint, set minimum acceptable RT value
  - Two optional TLVs ...

Remaining Throughput Window TLV



- RT Window TLV
  - 2 octets – unsigned integer
  - Expresses window size in time
  - Time units = THROUGHPUT_WINDOW_UNIT

Remaining Throughput Window Unit TLV



- RT Window Unit TLV
  - 1 octets – unsigned integer
  - Expresses window time unit
  - Time unit = $2^{\text{THROUGHPUT\_WINDOW\_UNIT}}$ ms
  - Time unit range = 1 ms … ≈5.7*10^73 sec

Enrollment – Layer 2 + RPL

- L2 – Join network (EB)
  - ietf-6tisch-enrollment-enhanced-beacon-01
  - Report PAN priority in EB
    (PAN priority = 16 - FLOOR(LOG2(RT + 1))
- RPL – Initial DODAG selection
  - Directly pick DODAG with max RT
  - Use in conjunction with other metrics
    e.g. ETX

Road forward

- Reviewed by Rahul and addressed comments
- Contiki implementation almost ready
- Remaining issue: Oscillations

- Next steps
  - Other reviewers?
  - Other changes?
  - Adoption?

Thanks!
Questions?

Experimental observation of RPL:

routing protocol overhead and asymmetric links

Henry-Joseph Audéoud, Martin Heusse

March 25th, 2019 — IETF 104

Introduction

Introduction

- Context of WSNs (IEEE 802.15.4)
- We used RPL as reference for our own routing protocol development. . . and noticed interesting details

- **We report on three situations**:
  - **Background control traffic** in a standard network
  - RPL in the presence of a **deaf node**
  - RPL in the presence of a **muted node**

Introduction — Asymmetric links?

- Influence of the environment
  - Interferences have not the same impact on all nodes (near-far effect)

- Influence of the hardware
  - Variability of the amplifier gain and sensitivity

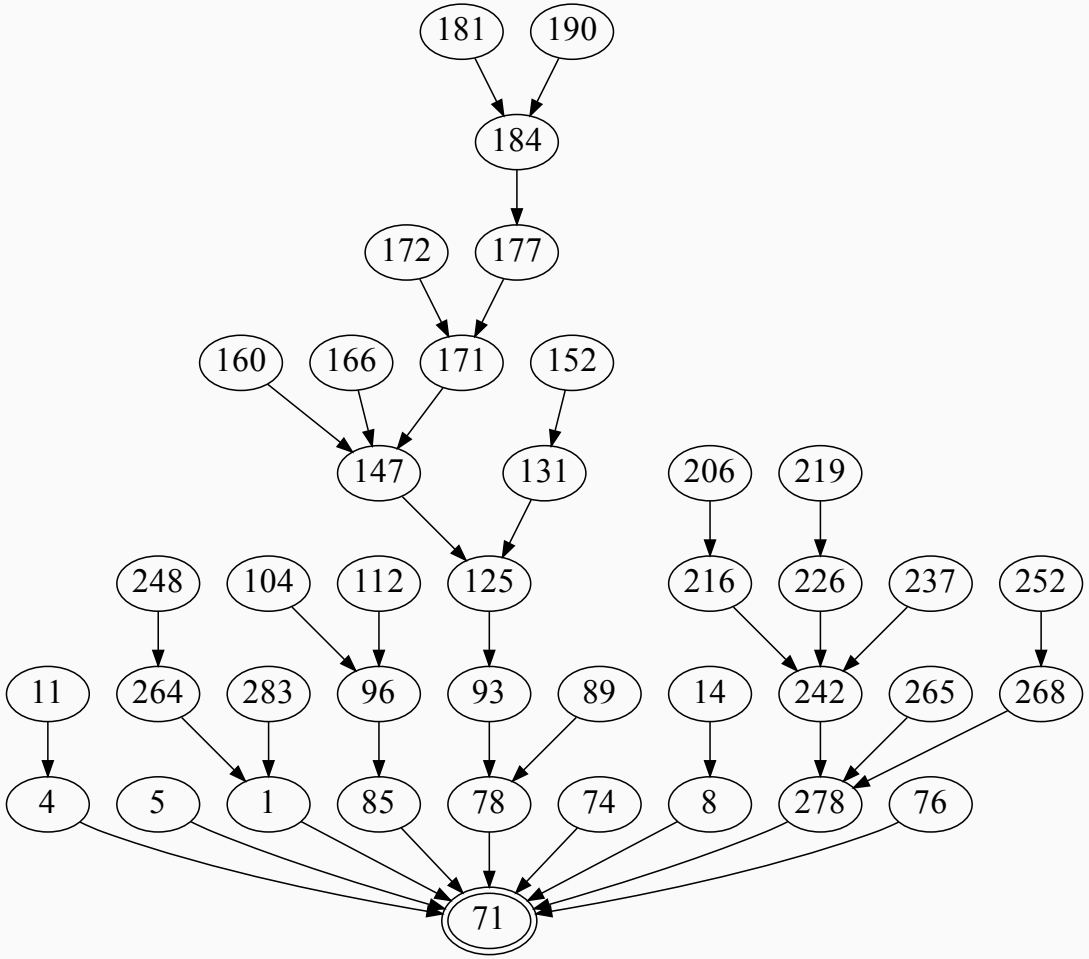
$\rightarrow$ Asymmetric links are rare but possible!

Background control traffic

Experimental setup

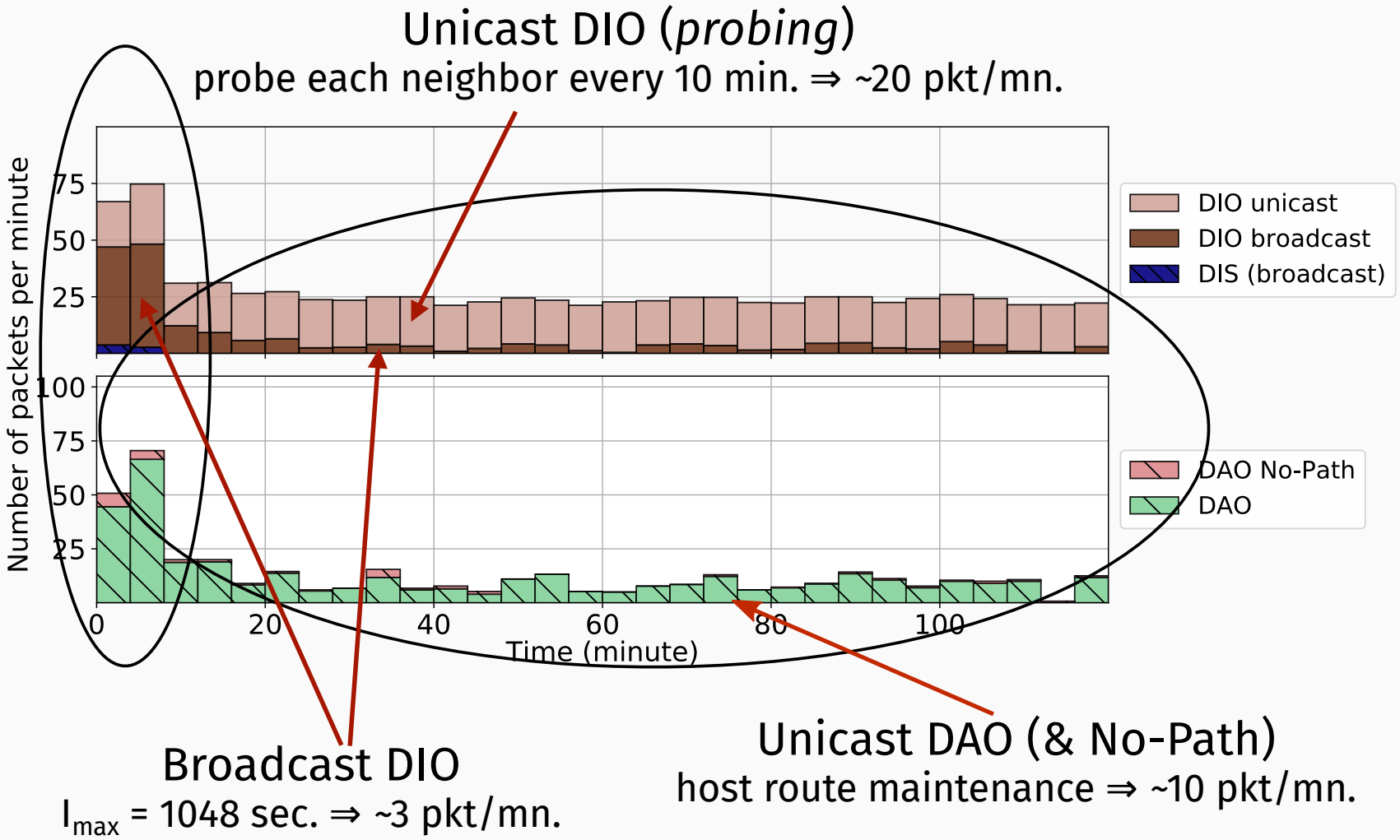
- Experimental scenario
  - **40 client nodes** + 1 sink, 2-hour duration
  - **1 UDP query/response** from each node to the sink every 5 minutes
- Testbed
  - **FIT/IoT-lab**, https://www.iot-lab.info/
  - $>$ 1500 wireless sensor nodes, monitoring infrastructure
  - We use ARM Cortex M3 nodes and Atmel 802.15.4 @2.4GHz radio chips
- RPL implementation from **Contiki 3.0**
  - MRHOF with ETX
  - ContikiMAC RDC

Experimental setup — DODAG



- average number of hops to the sink: 3
- average number of neighbors: 5

Background control traffic



What are those *unicast* DIOs?

> *RPL expects an **external mechanism** to be triggered during the parent selection phase in order to **verify link properties** and neighbor reachability.* — *[RFC6550, RPL]*

> *A node should **compute the path cost** for the path through each candidate neighbor reachable through an interface.* — *[RFC6719, MRHOF]*

Contiki $\Rightarrow$ uses unicast DIO for this task[1].

[1] Duquennoy, S. *et al.*, "Five-Nines Reliable Downward Routing in RPL", arXiv

Background control traffic — Discussion

- Link estimation is necessary & closely related to routing. . .

  even RPL-related docs recognize this!

- . . . But it is left aside. . .

  not an interoperability issue!

- . . . Leaves a few questions without answer
  - What is a node supposed to do when it receives a **unicast DIO**?
  - Do they count as **redundancy** for Trickle?
  - What about **traffic overhead**?

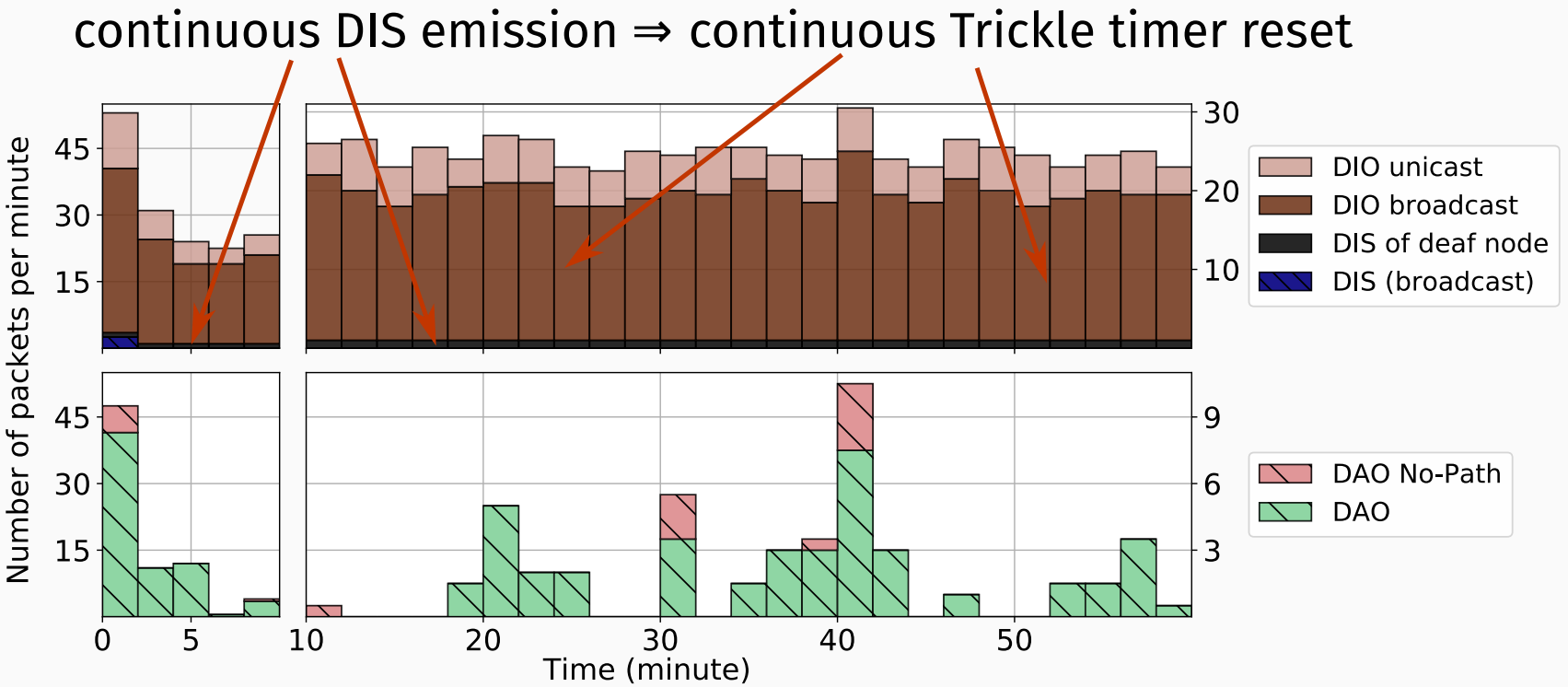
RPL in presence of a deaf node

Deaf node — Experimental setup

- Deaf node?
  - emits correctly, but do not receive packets

- Experimental scenario
  - **10** client nodes + 1 sink + **1 deaf node**, 1-hour duration
  - **1 UDP query/response** from each node to the sink every 5 minutes

Deaf node — Control traffic



Deaf node — Discussion

- It's more a question of neighbor managment...

  MAC layer should blacklist!

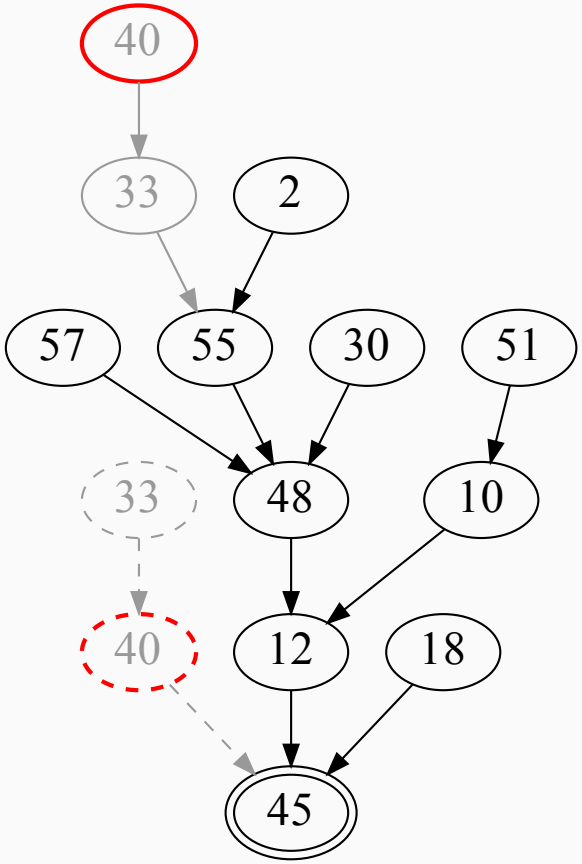
- ... But DIS and DIO packets are broadcast ($\rightarrow$ not acknowledged)

  MAC layer cannot estimate this link!

RPL in presence of a muted node

Muted node?

- Receives correctly, but is (almost) not heard
- Not completely muted (of course!)
- . . . Only after 15min.

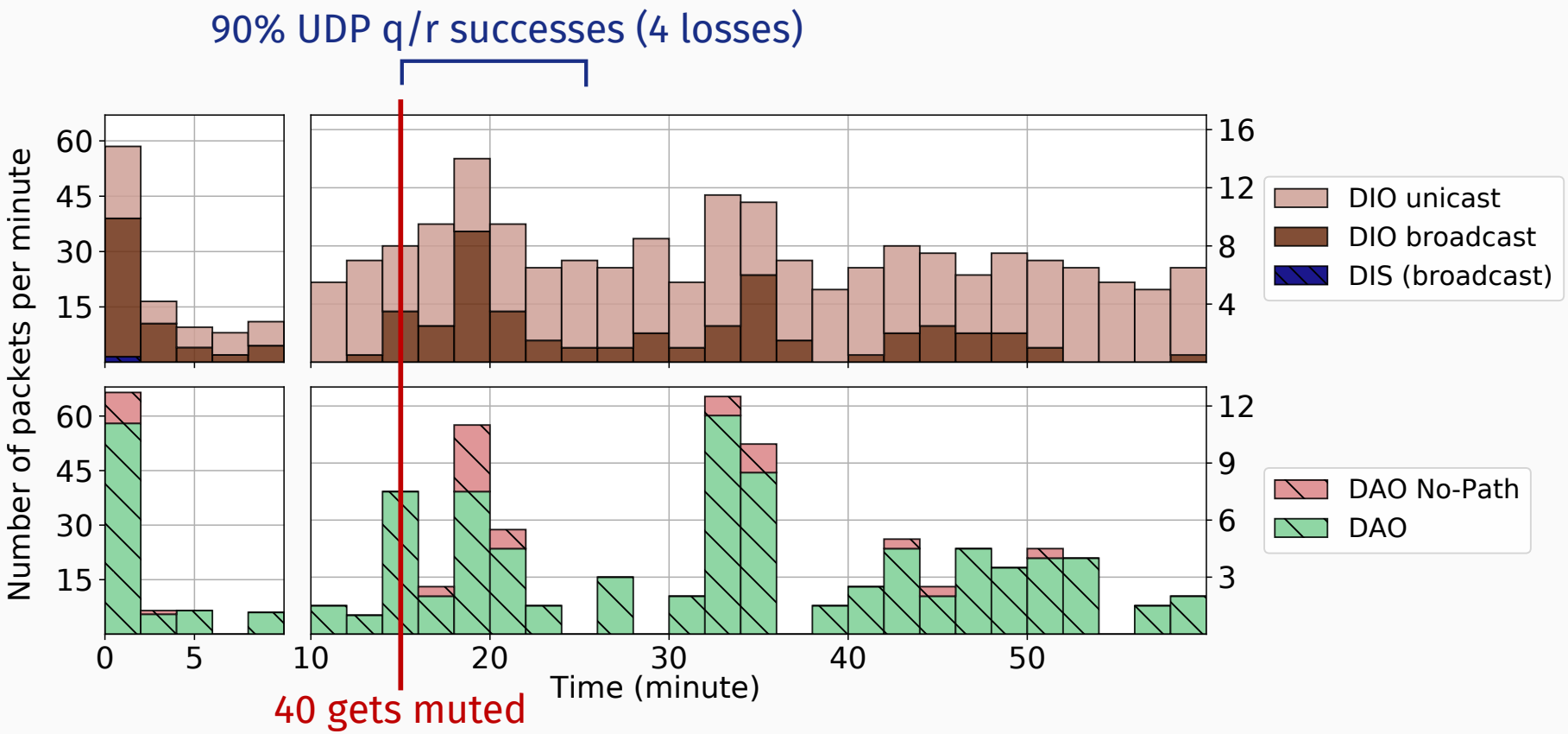


$\Rightarrow$ there is a **local repair**!

Muted node — Experimental setup

- Muted node scenario
  - **10** client nodes + 1 sink + **1 muted node**, 1-hour duration
  - **1 UDP query/response** from each node to the sink every 5 minutes

Muted node — Control traffic



Muted node — Discussion

- Application layer, OK
- Limited topology change, not solved by a few additional packets but acceleration of RPL traffic due to Trickle.

Conclusion

Conclusion

- It is tempting (and practical) to use routing packets to the purpose of link metric estimation, as in Contiki!
  - Reserve a packet format for metric estimation?
  - Explicitly allow a possible side-use of existing packet?
- Integrate (or combine) the broadcast DIS/DIO exchange to the link metric estimation?[2]
- Watch out for RPL overhead

[2] Relation with `draft-ietf-roll-dis-modification`?

Thank you for listening!

BACKUP SLIDES

Detailed parameters

Table 1: Main parameters used during the experiments

Platform	IoT-lab
Sensors	IoT-lab's (ARM Cortex) M3 motes
Sensor radio	802.15.4 @2.4GHz (AT86RF231)
OS & RPL implementation	Contiki 3.0
Radio Duty Cycling (RDC)	ContikiMAC
RDC Check interval	125 ms
RPL Mode of Operation	Storing
Imin	4 seconds
Imax	1048 seconds
DIORedundancyConstant	10 (standard's default)
DAO re-generation period	15 to 22 minutes
Objective function	MRHOF with ETX
# UDP traffic intensity per client	1 request-response every 5 minutes

Background traffic expe. detailed results

Table 2: Number of messages sent during the experiment of background traffic

Message	# of occurrence
DIS multicast	26
DIO multicast	760
DIO unicast	2497
DAO (unicast, counted on each hop)	1407
DAO No-Path (unicast, counted on each hop)	179
Data packet successfully routed end-to-end	1795 (97%)
Data packet emitted (counted on each hop)	5516

Deaf node expe. detailed results

Table 3: Number of messages sent during the experiment with the deaf node

Message	# of occurrence
DIS multicast (deaf node excluded)	5
DIO multicast	1177
DIO unicast	315
DAO (unicast, counted on each hop)	202
DAO No-Path (unicast, counted on each hop)	40
Data packet successfully routed end-to-end (deaf node excluded)	233 (99%)
Data packet emitted (on each hop, deaf node excluded)	583

Muted note expe. detailed results

Table 4: Number of messages sent during the experiment with the muted node

Message	repair (10 min)	elsewhere	total
DIS multicast	0	3	3
DIO multicast	37	172	209
DIO unicast	58	313	371
DAO (unicast)	41	258	299
DAO No-Path (unicast)	11	40	51
Data pkt successfully routed end-to-end	37 (90%)	220 (99%)	251 (98%)
Data pkt emitted (counted on each hop)	103	598	701

BIER - ROLL Design Team

RPL-BIER SIDE MEETING - PROPOSAL https://datatracker.ietf.org/meeting/104/materials/			
Time	Duration	General Information	Room
Monday 25 March 18:00 - 19:00	60 min	Minute: https://etherpad.tools.ietf.org/p/notes-roll-bier-side-meeting-ietf104 Room reservation: https://datatracker.ietf.org/meeting/104/floor-plan?room=paris#lobby	Paris
Tuesday 26 March 18:00 - 19:00	60 min	https://datatracker.ietf.org/meeting/104/floor-plan?room=berlin-brussels Slides IETF 103 for BIER: https://datatracker.ietf.org/meeting/104/materials/slides-104-roll-bier-rpl-ietf-103-00	Paris
Wednesday 27 March 15:00 - 16:30	90 min	Presenter: Everyone - General Discussion -	Berlin/Brussels

Q&A - AOB

Thank you very much!!!!!

APPENDIX

ROLL-Bier slides for IETF 103

IETF 103 Bangkok
ROLL-BIER Design Team

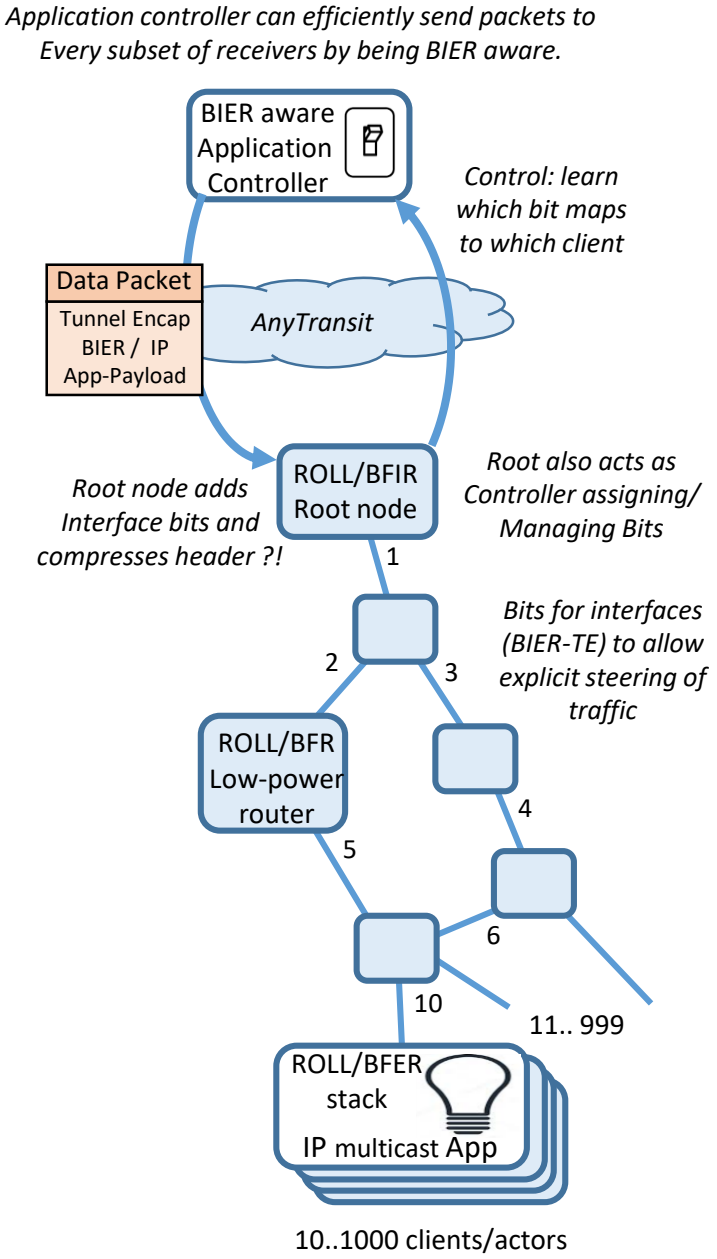
Toerless Eckert (Huawei), <tte@cs.fau.de>

BIER in ROLL Vision

Design Team

- Consider joining/collaborating in BIER design team in ROLL-WG:
- Email: roll-bier-dt@ietf.org (normal subscribe)
- https://trac.ietf.org/trac/roll/wiki/roll-bier-dt (to be filled)
- Issues: tte@cs.fau.de

- What could be cool about this (if design team decides to do it) ?
- End-to-end BIER (with TE) in low-power networks (e.g.: building control)
  - Example: Application Controller sends BIER packet to subset of clients (lightbulbs)
  - Each client is BFER (has a bit)
  - Every packet can address a separate subset of actors through bitstring
  - Only controller app needs to be BIER aware. Receivers can think its just IP multicast.
- BIER TE bits to save power/memory
  - Routers are low-power (memory/CPU). Do not want to keep large routing table (1000 lightbulbs). Links are low power too.
  - Every interface has a bit. Routers only need to route on bits to directly connected downstream neighbors.
- No ASIC constraints. Everything is software
  - Headers/Bitstring can be compressed (loss free, lossy (bloom) to support long bitstrings.
  - Should result in header more compact than existing ROLL/RPL headers even for unicast: Only hop-by-hop bits sets to one receiver: Would also be used for unicast forwarding.



Status of dt

- Slow start but getting more organized
  - Try weekly meeting via IETF webex, Wed. 7 PM GMT
  - until we can get work items assigned to individual stakeholders
    - Getting participants up to speed
  - Will redo role call for meeting time/cycle after IETF103
- Started to put slide deck (this) and notes etc. into github
  - www.guthub.com/toerless/roll-bier (no ROLL WG github repo group ?)
- Worked through bunch of arch level details
  - But in middle of writing them down

Relevant docs

- draft-thubert-roll-bier
  - Proposed arch of BIER via ROLL
    - Core – not IP Multicast overlay, not L2.5 encap
    - Intends to support BIER/BIER-TE x explicit/bloom-filter mode of operations
- draft-ietf-roll-ccast
  - BIER for ROLL with bloom filters. Expect separate (IP multicast) overlay to handle false positives
- draft-thubert-6lo-bier-dispatch
  - L2.5 proposed encap for BIER packets
  - Comparable layer/function to MPLS/Ether BIER encap (RFC8296) ?!
    - Aka: would not use RFC8296 because in 6LO networks it all about compression
- draft-thubert-roll-unaware-leaves
  - Not covering BIER, but introduces type of nodes we want to support via roll-bier too.

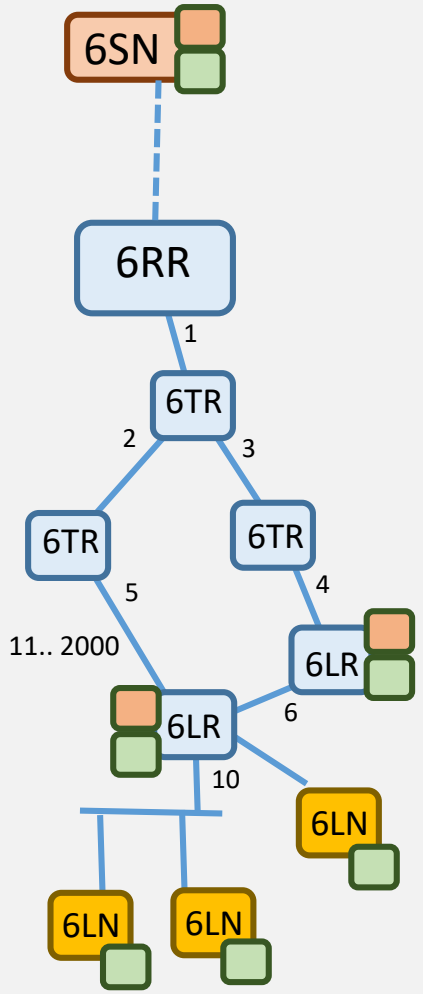
- draft-other-apologies-forgetting-you
  - Please help complete this list

Framework/ Terminology

ROLL+BIER+6LoRH

- Tunnel Mode
  - BIER, BIER-TE: Bits assigned to 6LR
  - BIER-TE: Bits also assigned to adjacencies/6TR
  - No bits assigned to 6LN
- Services
  - BIER 6SN-> 6LR + BA
  - IP unicast 6SN -> 6LR/6LN + IA
- 6LoRH + BIER
  - Compress/uncompress unicast/multicast packets
  - 6RR … 6LR
  - 6LR ... 6LN ???

Framework



Terminology

6SN = Server Node / Application
Not running ROLL/BIER
able to send Multicst
packets with
BIER bitstring to target
set of destinations explicit

RPL + BIER routers
Roles:
6RR = root.
needs to support 6LoRH for BIER
6TR = transit – no bit in BIER mode
6LR = leaf (LoWPAN) Router
bit in BIER mode,
connects to 6LN
needs to support 6LoRH BIER

6LN = Lightweight Node
no knowledge of ROLL/BIER
MAY want to support
6LoRH with BIER
extensions for compression

BA = BIER aware app able to send/rcv packets with bitstring, no IP multicast needed

IA = IP unicast/multicast app. No BIER awareness

Continuing to limit work scope by eliminating “below the line / do not work” options

- Only consider IP multicast payload for ROLL-BIER multicast now
  - Required for 6LN (non ROLL-BIER capable) nodes
  - For 6LR receivers we do primarily care that we can address them directly from the sender via bitstring, but not so important to get rid of potentially unnecessary IP Multicast header
    - IP Multicast header should be compressed also by 6LoRH ?!
- No new “faked” IP packets (see later slide)
- Only “transport mode” to 6LR, only “tunnel mode” to 6LN
  - See explanations later
- Only consider BIER-TE semantic, not BIER ?
  - TBD. Maybe we can start defining common forwarding and add BIER control plane later (when seen necessary)

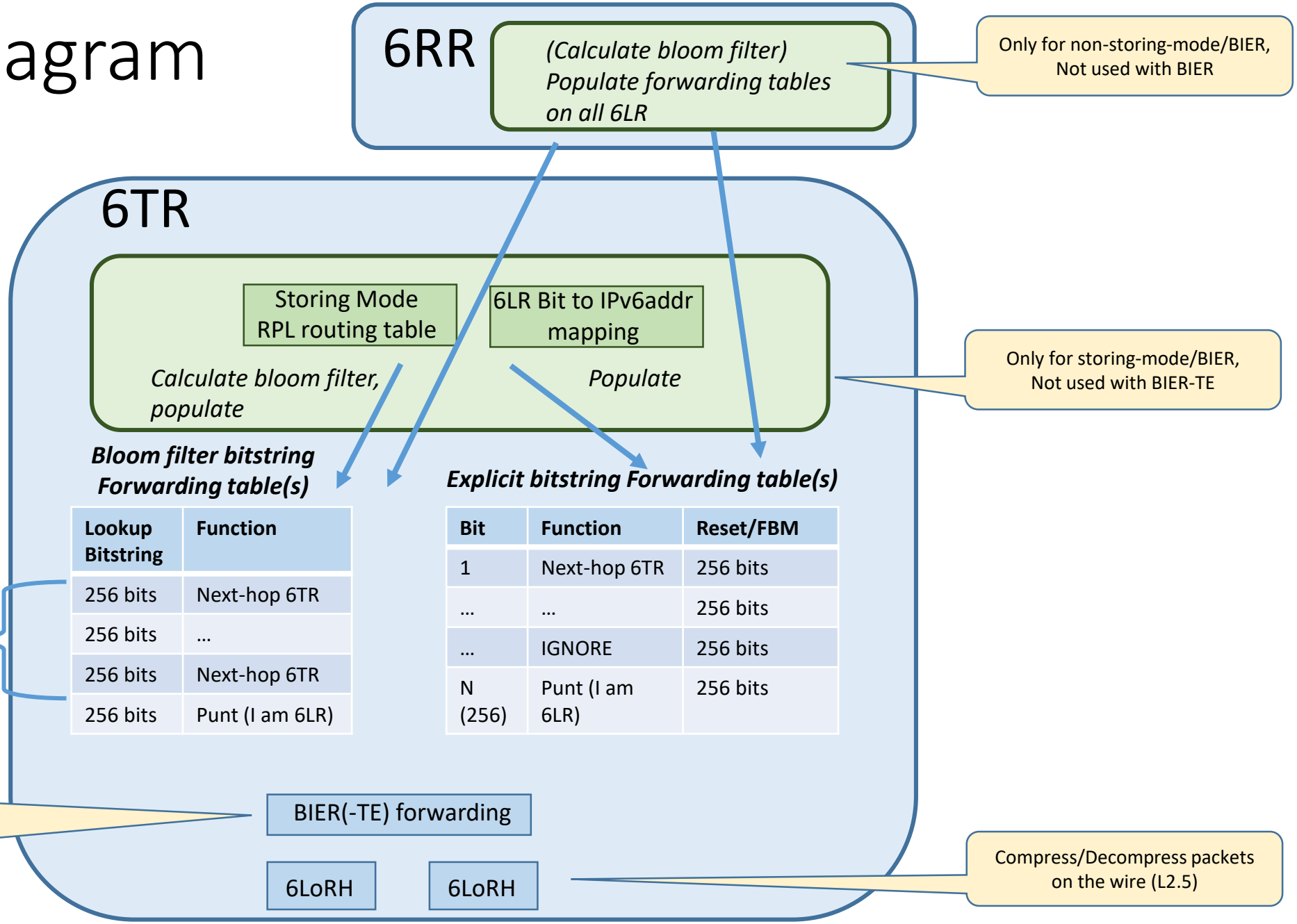
Details, Stack options

- First option: “Tunnel Mode” to 6LN, “Transport mode to 6LR”. Common header
  - Lowest layer is 6LoRH with BIER bitstring. Also all RPL artefact.
  - Next: 6LoPAN compressed IP packet (RFC6282)
- Transport Mode to 6LR:
  - 6LoRH indicates this mode, compres/uncompress destination address
  - Save 6LoPAN state for Dst address and compression field in 6LoPAN header (compressed state)
- Tunnel Mode to 6LN:
    - *Simple IP packet without any RPL/BIER artefacts.*
      - *High compression of IP address through state on every RPL hop for src/dst addresses (stateful)*
      - *Stateless – assumes MAC address derived IPv6 address.*
        - *But only works single L2 hop because it relies on the L2 destination “MAC”.*
      - *Aka: Stateful compression + BIER is big benefit because with BIER only the 6RR and 6LR would need to have state for the addresses because 6TR would just forward based on bits.*
      - *Need to show in slides how BIER benefit applies here:*
          - *No changes required for 6LoPAN to get desired benefit*
      - *FUTURE: define details of “transport mode” options.*

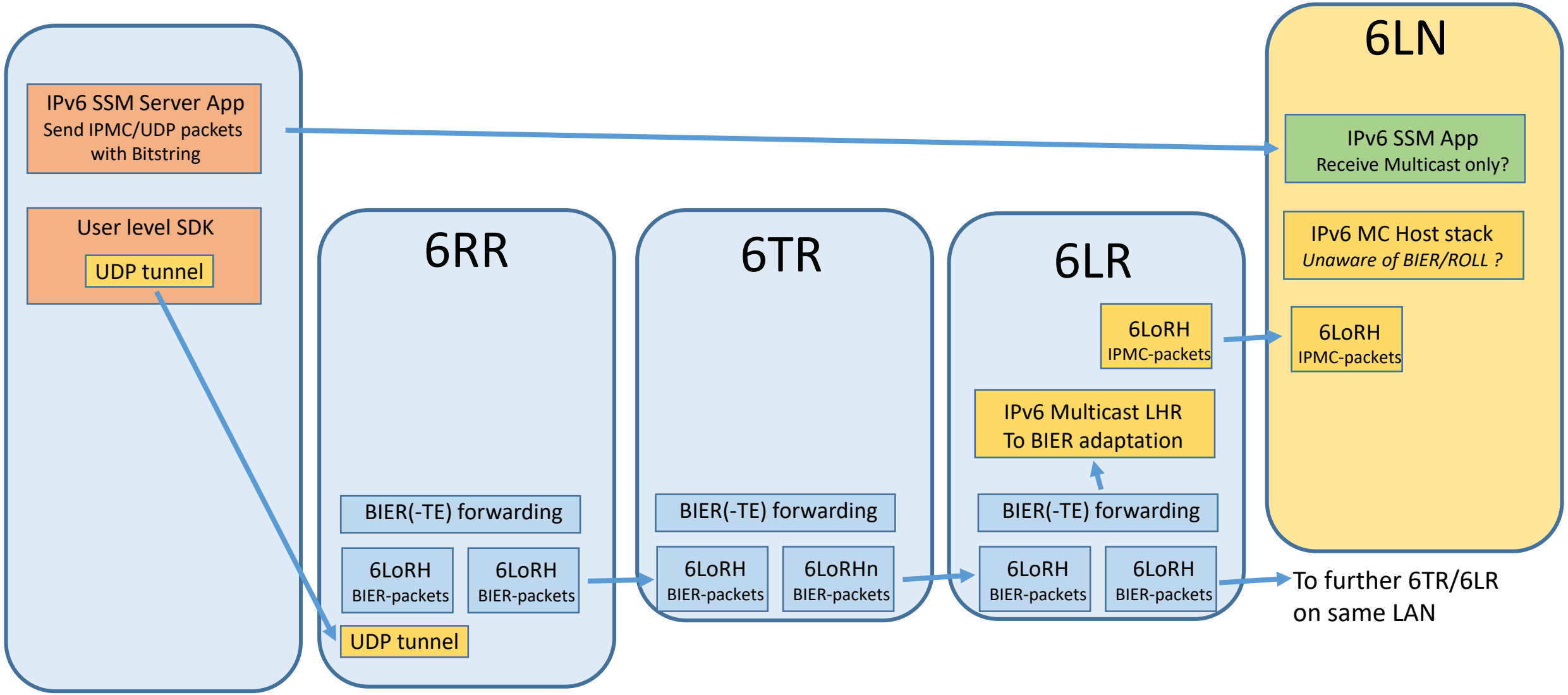
6RR/6LR diagram

Forwarding table does Not need to distinguish Between BIER/BIER-TE

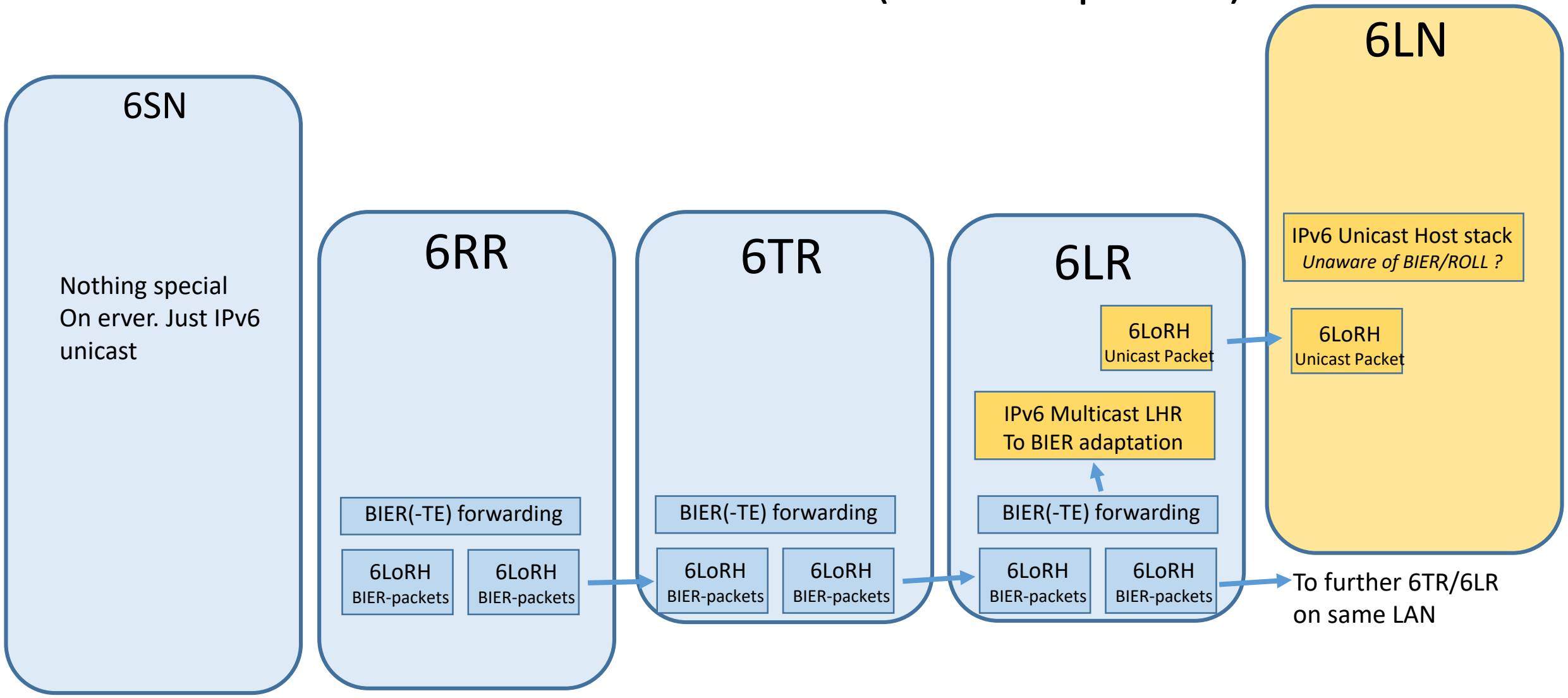
Difference just in how Next-hop entries/FBM for Bits are created



IP multicast over ROLL-BIER



IP unicast over ROLL-BIER (incomplete)



Bit-Reset/ Bloom Filter considerations (not reviewed)

- With bloom filter bitstring, bits can not be reset
- BIER logic
  - Leads to duplicates. Other radio-network issues can also lead to duplicates, so maybe not a big issue – radio networks MUST be prepared to deal with duplicates
  - Routing protocol microloops are avoided by reset. If RPL can have microloops they would only be protected against by TTL expiry (eg.: > 100 packet replications possible).
  - False positives in bloom filter can create more duplicates and more replicated packet when there is a routing microloop
- With BIER-TE
  - No routing protocol involved = no IGP microloops.
  - False positives can create duplicates AND looping (like badly set bits by BIER-TE controller can create loops too)
- Solutions ?
  - Have ROLL-BIER domain TTL (6LoRH TTL ?) that is set to be as small as possible: longest path to any receiver. Calculated by root node ?!
    - With common network diameter < 8 ?! This should be good enough ?! (4 bits enough to encode ?)
  - Duplicate elimination by packet-ID ? Probably takes too much memory…
- What to do ?
  - Reset bits when using explicit bitstring, not-reset only when using bloom-filter bitstring ?!
  - Figure out we can live without resets for all modes ?

BIER consideration (not reviewed)

- With bitstring length N, we need N * N bits for FBM – reset bits
  - With bloom filters, we can not have FBM (reset bits). But we try to resolve that issue
  - Do we want to forego FBM for non-bloom filter ?
  - Less problems to solve than with bloom filter: No false positives! Just more duplicates, routing microloop duplicates.
- Simple option ?
  - Make FBM a “SHOULD” – low-end devices could optimiz them away.
- Quantify benefit of explicit BIER bitstring (not BIER-TE)
  - Unicast: Do we save anything over existing storing mode with 6LoRH ? (header already well compressed)
    - Maybe there is NO reason to use BIER bitstring for unicast (only BIER-TE)
  - Multicast: assume existing storing mode network (aka: overhead of unicast routes acceptable)
    - BIER avoids to introduce another multicast routing protocol (PIM, RPL-multicast state,…)
    - More efficient use of bits (no bits used for adjacencies, just 6LR

BIER consideration (2, not reviewed)

- Option position summary ? All need to be validated/quantified.
- Non-storing mode networks
  - Use BIER-TE == non-storing. Unicast+multicast.
  - Can further minimize state avoiding FBM memory
  - Smaller networks use explicit bitstring. Larger ones use bloom filter.
- Networks with storing mode routers
  - Use BIER, maybe just multicast. No benefit? to use for unicast.
  - Smaller networks use explicit bitstrings, larger ones use bloom filter.

Bitstring / BIER considerations

- Explicit BIER-TE bitstring allows to save memory on 6TR/6LR:
  - Do not need RPL routing entries for all 6TR/6LR.
- With BIER, we do need RPL routing table for all 6TR/6LR
  - What do we save (why BIER) ? Unclear / need to better caracterize.
  - Comparison complex ? Because we also have to take savings from 6LoRH into account.
- Aka: Do we need to consider Explicit Bitstring + BIER ?
  - Forwarding plane can be the same BIER/BIER-TE. Eliminating BIER just reduces control plane work to consider (RPL -> create BIER forwarding table entries).
  - BIER/BIER-TE forwarding more similar than outside of ROLL:

Open

- With BA (bier aware application) we could send non-IPv6 payload. Any benefit in that ?
  - No ? 6LoRH would compress IP header away so there is no benefit eliminating IP ??
- Unicast between 6LR/6LN ? Unicast 6LR/6LN towards 6RR ?
  - How do we deal with that ?

- IEF Multicast likes SSM.
  - SSM could help to address 6SN (which server needs to know about IGMP memberships). Part of

Refuse

Refuse - Does not work

- Does not work: Multicasting unicast packets
  - Aka: unicast packet replicated via bitstring to more than one destination
  - Could recreate packet with each destinations unicast dest-addr, but:
    - No architecturally clean solution to do so – e.g.: UDP checksum calculation
  - Make destination address a “unicast-group-address” – IPv6 unicast address same on all nodes
    - Avoid problems like UDP checksum
    - But would just reinvent IP multicast service with IP unicast addresses (which we will have).
    - No added value.

Refuse - Below the line (now)

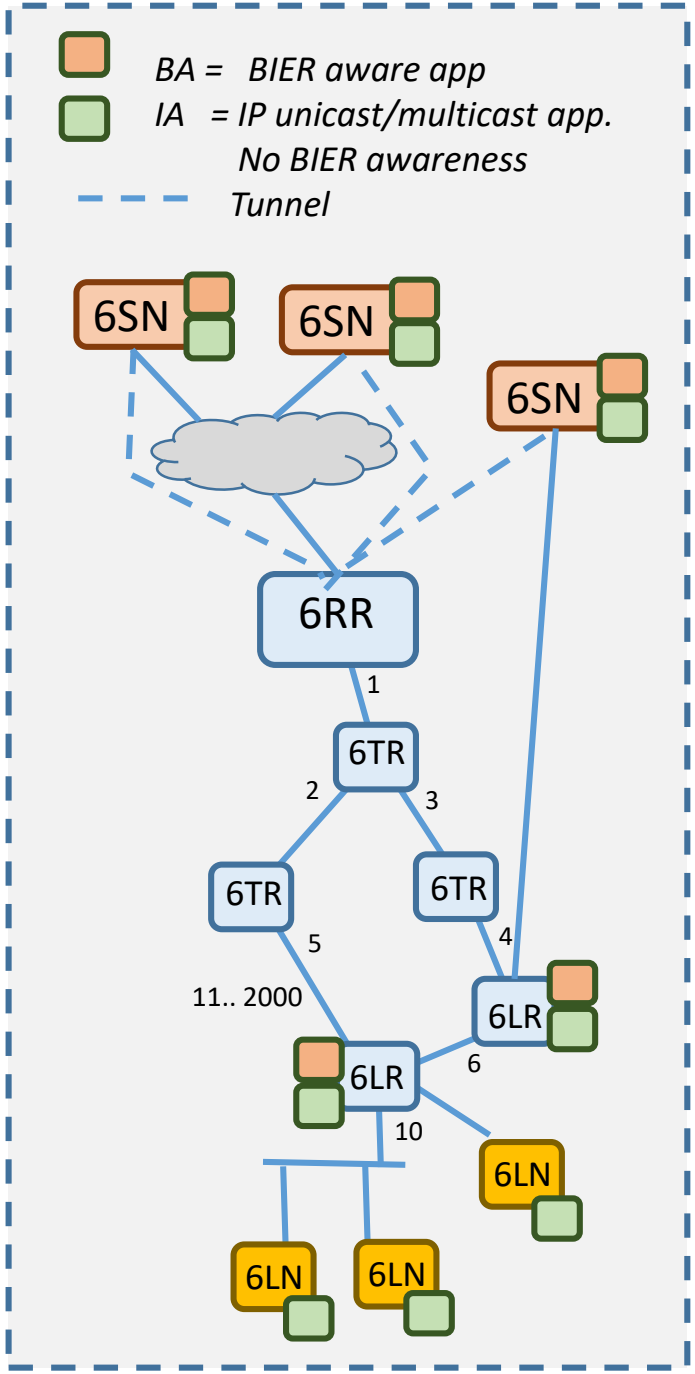
- Using BIER to carry non-IPv6 packets
  - Idea was: BIER packets can directly carry any payload, no IPv6-multicast heade required, eliminate overhead/complexity of IPv6
  - Claim: 6LoRH can compress the IPv6 headers so much that there is not enough initial value in designing header stacks to eliminate IPv6.
  - But: We need to make sure 6LoRH will also equally well compresses the IPv6 multicast header
  - Also eliminates need to figure out how to deal with false-positives for non-IPv6—multicast BIER packets
- Below the line: Transport mode
  - In transport mode, 6LN nodes will have bits assigned to them in the bitstring. This introduces the need for them to be more aware of BIER, e.g.: introduce more BIER awareness into 6LoRH all the way into 6LN and receiver applications.
  - For unicast
    - Will work fine but below line until we have evidence that it could provide better compression than tunnel mode with 6LoRH+BIER compression.
  - Transport mode for multicast
    - Just too much work trying to figure out in first round how to build a lightweight node that ALSO has to be aware of BIER bitstrings.

TO BE DISCUSSED

IP Multicast layer

IP Multicast layer overview

- Primary target for IP Multicast in ROLL-BIER
  - Efficient sending/replication of multicast packets from IP multicast sender (app) on 6SN to IP multicast receiver (app) on 6LN/6LR
  - App on 6SN should be able to explicitly indicate set of receivers. Key benefit not possible with BIER not possible with standard IP multicast.
    - Example from design-team slide: send “ON”/“OFF” message to subset ot light-bulbs that should be switched “ON”/”OFF”
- Not needed: Send IP multicast packets from 6LR/6LN
  - Application client/server mode:
  - Application has one or more server (6SN) sending IP multicast through tunnel into 6RR. Clients (6LR/6LN) can just receive IP multicast.
  - Client->client multicast “emulated” by client sending unicast to Server and then server sends IP multicast
    - Client may receive its own IP multicast packet back, so application need to be able to recognize/filter packets from “itself”
    - Client on 6LR can be excluded from bitstring by 6SN, so this additional filtering only required for 6LN



IP Multicast layer: SSM only

- Majority of IETF Multicast experts prefer SSM IP Multicast over classical ASM IP Multicast
  - Proposal follows that direction
  - SSM: receivers send (S,G) membership instead of (G) membership (MLDv2).
  - S = sender (Server) IP unicast address.
- Benefits
  - If we want to avoid implement directly client->client IP multicast in ROLL-BIER, we need Client/Server model, only servers allowed to send IP multicast, client unicasting to server if they need to be able to send IP multicast (previousslide). This is exactly the model how IP multicast would be done with SSM.
  - SSM avoid need to coordinate IP multicast group addresses across applications (potentially big operational issue).
  - When SSM IP multicast application on a sender wants to create a new SSM stream, it simply needs to allocate an SSM IP multicast group that is unused on this sender (like allocating an TCP port unused for a new unicast service).
  - Client discovery of SSM (S,G) ? Use same scheme a for Unicast server discovery in the absence of IP multicast (e.g.: DNS). As necessary, extend discovery to also discover IP Multicast group (e.g.: Put Server IP unicast address and group-address into DNS. E.g.: group-address in TXT record).
  - Ideal: New IP multicast application on 6SN can allocate new local IPv6 address (independent from IPv6 address used by other IP multicast app running on same 6SN)
    - Benefit: Applications that can have their own SSM IP Multicast sender IPv6 addresses can use static/well-known IPv6 Multicast group application – no need for group discovery by receivers.
    - Best method for local address allocation ? IPv6 privacy addresses ?... ?

IP Multicast layer: MLDv2

- Why do we even need MLDv2 ?
- Not for 6LR: Receiver application on 6LR will receive multicast packet because of bit in BIER bitstring. No IGMP/IP Multicast needed.
- Receiver on 6LN will receive link-layer multicast packet across access-subnet with potentially many 6LN.
  - (S,G) IP multicast header indicates whether packet is of interest to receiver
  - 6LR needs to know whether to send packet to a particular subnet. Some signaling needed to learn this from receivers. No need to reinvent wheel: MLDv2 does this.
  - Receivers need to open link-layer filter (eg: destination) to receive multicast and send packets up to right application. OS-level MLDv2/multicast-socket-API does this. No need to reinvent the wheel.
- BUT!! Above reasoning to reuse/not-modify 6LR->6LN IP multicast (SSM) is ONLY true if IP Multicast packet actually contains IPv6 Multicast header.
  - If we want compressed packet on 6LR->6LN link, then the above rules may not be true: If new OS-level code is needed on 6LR/6LN to support 6LoRH compressed IP multicast packets, then we should re-evaluate what the most pragmatic way is to get a working solution.
  - Unclear what exists today wrt. 6LoRH/IP-multicast/compressed-packets.

IP Multicast layer: proposal (1) - INCOMPLETE

- 6LN:
  - Signaling: SSM subset of RFC3810 (MLDv2), host side
    - Note: This should also be subset of “lightweight IGMPv3/MLDv2” (RFC5790)
  - Only need to be able to receive, not send IP multicast
  - 6LoRH (extension?) to receive compressed IP multicast packets
- 6LR:
  - Signaling: SSM subset of RFC3810 (MLDv2), router side
  - (Modified/Extended) SSM subset of RFC4605 (IGMP/MLD proxy routing)
    - 6LR aggregates SSM receiver state from all subnets: 6LN + internal virtual subnet for IP Multicast receiver application on 6LR itself.
    - 6R sends aggregated membership state to 6SN (join/leave): HOW? (later slide)
    - Any BIER packet with 6LR “bit” set or 6LR comb filter match will be passed to MLD proxy routing forwarding as coming ”from upstream”
    - NO receiving of IP multicast packets from downstream nodes (6LR, 6LN) – just discard
- 6TR:
  - Do not participate in P multicast layer. Just forward BIER. Become 6LR when BIER bitmask/comb-filter entry exists for them. Every 6LR is also always 6TR (forwards BIER independent of processing IP Multicast)
- 6RR:
  - Just like 6TR except that it also must be able to receive tunneled IP Multicast + (pseudo) BIER header from 6SN.

IP Multicast layer: issue

- Simple: let 6SN “just” send SSM IP multicast. No BIER improvements
  - Receipt of IP multicast ONLY determined by MLDv2 membership from receivers (6LN / 6LR).
  - Foregoes application benefits of using BIER (sender determines receivers)
- Problem: Can not get BIER benefit for 6LN without non-heuristic bitstring transport mode”
  - Give BIER bits to 6LN, carry bitstring all the way to 6LN
  - Can not use heuristic likely not useful here: need to be able to eliminate false positives at application level.
- Design for ONLY SSM IP multicast different / simpler as if we want to introduce application layer BIER benefits to application

IP Multicast ONLY model

- 6LR send aggregate MLD membership (join/leave) to RR.
- 6LR keeps (S,G) -> { 6LRi } state.
- Keep table of Bits(6LRi) -> bitstring:
  - bitstring with one bit (non-bloom, BIER)
  - bitstring with > 1 bits (non-bloom, BIER-TE):
    - BIER bit for 6LR plus bits for each hop towards it.
  - bitstring with > 1 bits (bloom BIER/BIER-TE). Bloom compression bitstring result for 6LR (BIER) plus Bloom compression bitstring result for each hop towards it (BIER-TE).
- 6SN just send IP multicast to RR via some tunnel
  - RR may perform RPF check (source address must be 6SN unicast address).
  - Then encaps/forwards BIER/BIER-TE:
    - Loop up (S,G) of packet, Ors Bits(6LRi) for all { 6LRi }. Sends packets
- At minimum same type of hop-bits determination for BIER-TE / non-storing mode as used for unicast non-storing mode today. Just encoded as direct/bloom-filter bits.

IP Multicast + 6SN BIER model

- 6SN should indicate set of destinations instead of “just” MLDv2 join state
- As long as we assume we always use IPv6 multicast packet / MLD, the set of destinations indicated by 6SN must be subset of the “joiners”
  - Simple app example: Each application just has ONE IPv6 multicast group, all application members on 6LR / 6LN join to that group. But 6SN only indicates subset of those application clients for each packet.
  - Does not work for 6LN without transport mode (prior slide).
- Most simple? Hybrid solution (introduce BIER benefit to 6SN with minimum additional work ?)
  - RR forwards (S,G) -> { 6LR } state changes to 6SN. Use of SSN makes this easy: Each 6SN only gets updates for those (Si,G) with its own Si.
  - 6SN send IP Multicast packet with additional BIER pseudo-header
    - Indicating subset of 6LR to which packet should go
  - 6RR then only OR’s bitstrings for this subset as opposed to full { 6LR } it maintains
- INCOMPLETE

Next steps ?!

- Push discussion about BIER interface for 6SN to BIER-WG ?!
  - ROLL should not come up with somethin ROLL specific that does not have to be ROLL specific – better try to find general purpose solution in BIER (ROLL defining requirements)
- SSM IP multicast (ONLY) solution
  - ROLL: (optional) ? compression for 6LoRH extension on access LAN
  - “Tunnel” from 6SN node to 6RR - to send IP Multicast packets from 6SN to 6RR
    - Two cases ?
    - A) 6SN is inside 6LoRH network: simple 6LoRH header extension: one bit to indicate “packet goes to root”, but also encode IP Multicast Group address.
    - B) 6SN is “behind backbone”: Need an actual IP tunnel (path between 6SN/6RR not natively supporting 6LoRH). Just encapsulate same 6LoRH packet inside ?
  - MLDv2 proxy operations on 6LR
    - 6LR -> 6RR covered by ‘draft-ietf-bier-mld-01.txt’, but discovery of 6RR and constraints (only receive, not sending) and 6LR->6LN not covered.