IETF 99 ROLL

Routing over Low-Power And Lossy Networks

Chairs:

Peter van der Stok

Ines Robles

Secretary:

Michael Richardson

Note Well

Any submission to the IETF intended by the Contributor for publication as all or part of an IETF Internet-Draft or RFC and any statement made within the context of an IETF activity is considered an "IETF Contribution". Such statements include oral statements in IETF sessions, as well as written and electronic communications made at any time or place, which are addressed to:

- The IETF plenary session
- The IESG, or any member thereof on behalf of the IESG
- Any IETF mailing list, including the IETF list itself, any working group or design team list, or any other list functioning under IETF auspices
- Any IETF working group or portion thereof
- Any Birds of a Feather (BOF) session
- The IAB or any member thereof on behalf of the IAB
- The RFC Editor or the Internet-Drafts function

All IETF Contributions are subject to the rules of RFC 5378 and RFC 8179.

Statements made outside of an IETF session, mailing list or other function, that are clearly not intended to be input to an IETF activity, group or function, are not IETF Contributions in the context of this notice. Please consult RFC 5378 and RFC 8179 for details.

A participant in any IETF activity is deemed to accept all IETF rules of process, as documented in Best Current Practices RFCs and IESG Statements.

A participant in any IETF activity acknowledges that written, audio and video records of meetings may be made and may be available to the public.

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

Meeting Materials

- 13:30-15:30 Thursday Afternoon session I
- Remote Participation
  - Jabber Room: roll@jabber.ietf.org
  - Meetecho: http://www.meetecho.com/ietf99/roll
- Etherpad:
  - http://tools.ietf.org/wg/roll/minutes
- Audio Streaming:
- Minutes taker:
- Jabber Scribe:
- **Please sign blue sheets :-)**

Agenda

13:30-15:30 Thursday Afternoon session I

Item	Time	Presenter
State of the WG	13:30 - 13:40 (10mins)	Ines & Peter
RPL-Info	13:40 - 14:00 (20min.)	Michael
Multicast-bier	14:00 - 14:20 (20 min.)	Carsten +
No-path-dao	14:20 - 14:35 (15 min.)	Rahul
AODV-RPL	14:35 - 14:50 (15 min.)	Charlie
Parent-selection	14:50 - 15:05 (15 min.)	Jianqiang
DAO-projection	15:05 - 15:25 (20 min)	Pascal
Q&A	15:25 - 15:30 (5 min.)	Ines & Peter

Milestones

Sep 2018	Recharter WG or close
Jul 2018	Initial submission of a solution to the problems due to the use of No-Path DAO Messages to the IESG
Nov 2017	Initial submission of a proposal to augment DIS flags and options to the IESG
Nov 2017	Initial submission of a reactive P2P route discovery mechanism based on AODV-RPL protocol to the IESG
Jul 2017	Initial submission of a Forwarder Selection Protocol for MPL to the IESG
Jul 2017	Initial submission of a proposal for Source-Route Multicast for RPL to the IESG
Mar 2017	Initial submission of a root initiated routing state in RPL to the IESG
Mar 2017	Initial submission of a YANG model for MPL to the IESG
Jan 2017	Initial Submission of a proposal with uses cases for RPI, RH3 and IPv6-in-IPv6 encapsulation to the IESG

State of Active Internet-Drafts

Draft	Status
draft-ietf-roll-aodv-rpl-01	Presentation today
draft-ietf-roll-dao-projection-01	Discussion today -- **IPR**
draft-ietf-roll-forw-select-00	Sleeping
draft-ietf-roll-useofrplinfo-16	New version, presentation today
draft-ietf-roll-dis-modifications-00	No presentation today
draft-ietf-roll-mpl-yang-00	Waits for co-author
draft-ietf-roll-bier-ccast-00	Presentation today
draft-ietf-roll-efficient-npdao-00	New WG doc, presentation today -- **IPR**

Open Tickets

Ticket	Status
178: Editorial comments for dao projection draft	New Defect, Created
179: Security considerations for dao projection	New Defect, Created
180: 13 issues to address in dao projection draft (lifetime, MOP, transmissions, route cleanup)	New Defect, Created
182: useofrplinfo review	New Defect, Created

Related Internet-Drafts

Load Balancing Objective Function in RPL draft-qasem-roll-rpl-load-balancing-00	Presented ietf98
An energy optimization routing scheme for LLSs draft-wang-roll-energy-optimization-scheme-00	----------
Optimization of Parent-node Selection in RPL-based Networks draft-hou-roll-rpl-parent-selection-00	Presented today

When to use RFC 6553, 6554 and IPv6-in-IPv6

draft-ietf-roll-useofrplinfo-16

Michael Richardson
Pascal Thubert
Ines Robles

IETF 99

New version (16):

- Updates 6553 (**Million thanks to Mike Heard for his comments**)
- Updates 6550 (**Million thanks to Mike Heard for his comments**)
- Text clarification

Why we update the RFC 6553?

Background:

IPv6 Extension Headers - Options

[draft-ietf-6man-rfc2460bis-13#section-4.2]

```
00 - skip over this option and continue processing the header.

01 - discard the packet.

10 - discard the packet and, regardless of whether or not the
     packet's Destination Address was a multicast address, send an
     ICMP Parameter Problem, Code 2, message to the packet's
     Source Address, pointing to the unrecognized Option Type.

11 - discard the packet and, only if the packet's Destination
     Address was not a multicast address, send an ICMP Parameter
     Problem, Code 2, message to the packet's Source Address,
     pointing to the unrecognized Option Type.
```

Why we update the RFC 6553?

Processing of the Hop-by-Hop Options header is now optional,

If the nodes are configured to process the header, and if such nodes encounter an option with the first two bits set to **01 (0x63)** they will **drop** the packet (RPL Option type in RFC 6553).

Hex Value	Binary Value				
	act	chg	rest	Description	Reference
0x63	01	1	00011	RPL Option	[RFC6553]

Figure 1: Option Type in RPL Option.

But, we need that,

If an IPv6 (intermediate) node (RPL-not-capable) receives a packet with an RPL Option, it should **ignore** the HBH RPL option

Ignore = skip over this option and continue processing the header.

Thus, we propose,

Hex Value	Binary Value				
	act	chg	rest	Description	Reference
0x23	00	1	00011	RPL Option	[RFCXXXX]

Figure 2: Proposed change to the Option Type in RPL Option.

The first two bits **(0x23)** indicate that the IPv6 node **MUST skip** over this option and continue processing the header

This ensures that a packet that leaves the RPL domain of an LLN (or that leaves the LLN entirely) **will not be discarded** when it contains the [RFC6553] RPL Hop-by-Hop option known as RPI.

But,

This change creates a **flag day** for existing networks which are currently using 0x63 as the RPI value. A move to 0x23 will not be understood by those networks.

Flag day: A "flag day" is a procedure in which the network, or a part of it, is changed during a planned outage, or suddenly, causing an outage while the network recovers [RFC4192]

So,

In order to avoid a flag day caused by lack of interoperation between new RPI **(0x23)** and old RPI **(0x63)** nodes, the new nodes need to be told that there are old RPI nodes present

This can be done via a **new DIO Option (DIO Flag)** which will propagate through the network

New DIO Option vs DIO Flag

DIO Option: **0x05** RPI 0x23 enable MCRXXX

DIO Flag:

Figure 3: A DIO Flag to indicate the RPI-flag-day.

FR(RPI-flag-day): the flag with values of 1 indicates that RPL Option field is set to "**00**", values of 0 indicates that RPL Option field is set to "**01**"

New DIO Option vs DIO Flag

In this way we **update RFC 6550**

So,

New DIO Option vs DIO Flag

Which approach do you think is better?

https://i.gyazo.com/0898fbcde619c18d070708693cbff4a4.png

IP-in-IP encapsulation in **Storing mode**
(based on the updates)

Interaction between	Use Case	IP-in-IP	IP-in-IP dst
Leaf - Root	Raf to root	No	--
	root to Raf	No	--
	root to ~Raf	No	--
	~Raf to root	Yes	root
Leaf - Internet	Raf to Int	No	--
	Int to Raf	Yes	Raf
	~Raf to Int	Yes	root
	Int to ~Raf	Yes	hop
Leaf - Leaf	Raf to Raf	No	--
	Raf to ~Raf	No	--
	~Raf to Raf	Yes	dst
	~Raf to ~Raf	Yes	hop

Headers needed in **Non-Storing mode**: RPI, RH3,
IP-in-IP encapsulation. (based on the updates)

Interaction between	Use Case	RPI	RH3	IP-in-IP	IP-in-IP dst
Leaf - Root	Raf to root	Yes	No	No	--
	root to Raf	Opt	Yes	No	--
	root to ~Raf	No	Yes	Yes	6LR
	~Raf to root	Yes	No	Yes	root
Leaf - Internet	Raf to Int	Yes	No	Yes	root
	Int to Raf	Opt	Yes	Yes	dst
	~Raf to Int	Yes	No	Yes	root
	Int to ~Raf	Opt	Yes	Yes	6LR
Leaf - Leaf	Raf to Raf	Yes	Yes	Yes	root/dst
	Raf to ~Raf	Yes	Yes	Yes	root/6LR
	~Raf to Raf	Yes	Yes	Yes	root/6LN
	~Raf to ~Raf	Yes	Yes	Yes	root/6LR

Thanks!

Q&A

BIER / RPL

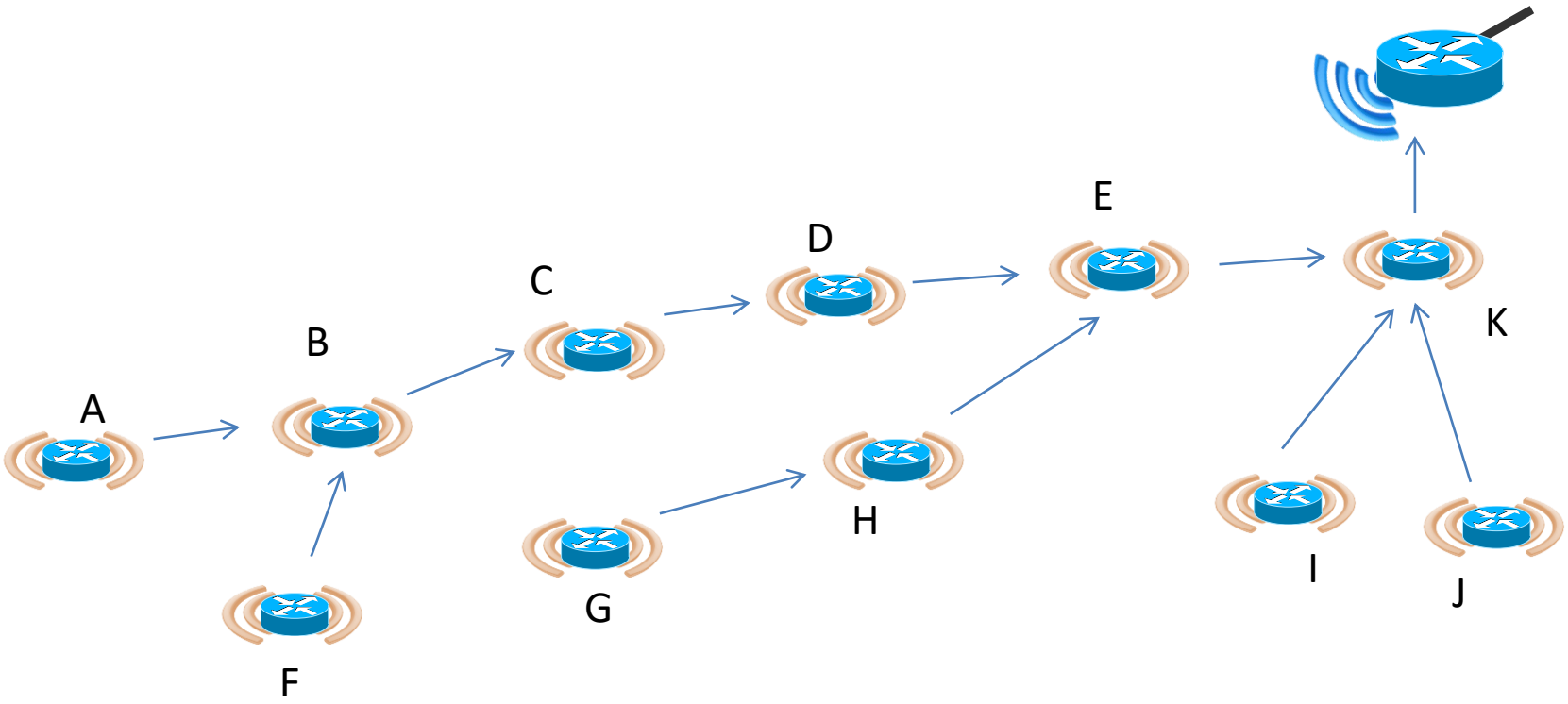
Pascal Thubert

IETF 99

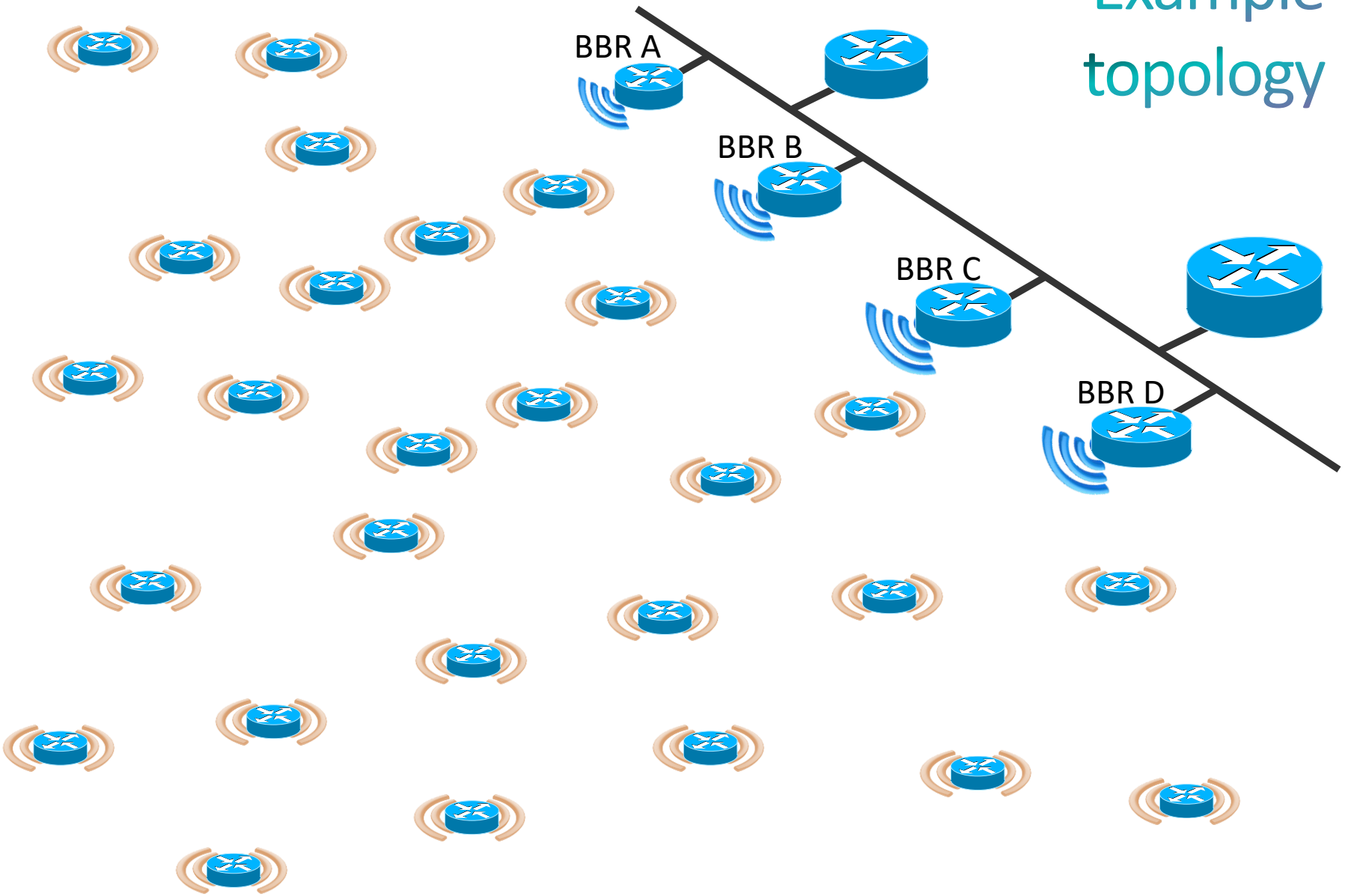
Prague, July 2017

Unreliable BIERPL

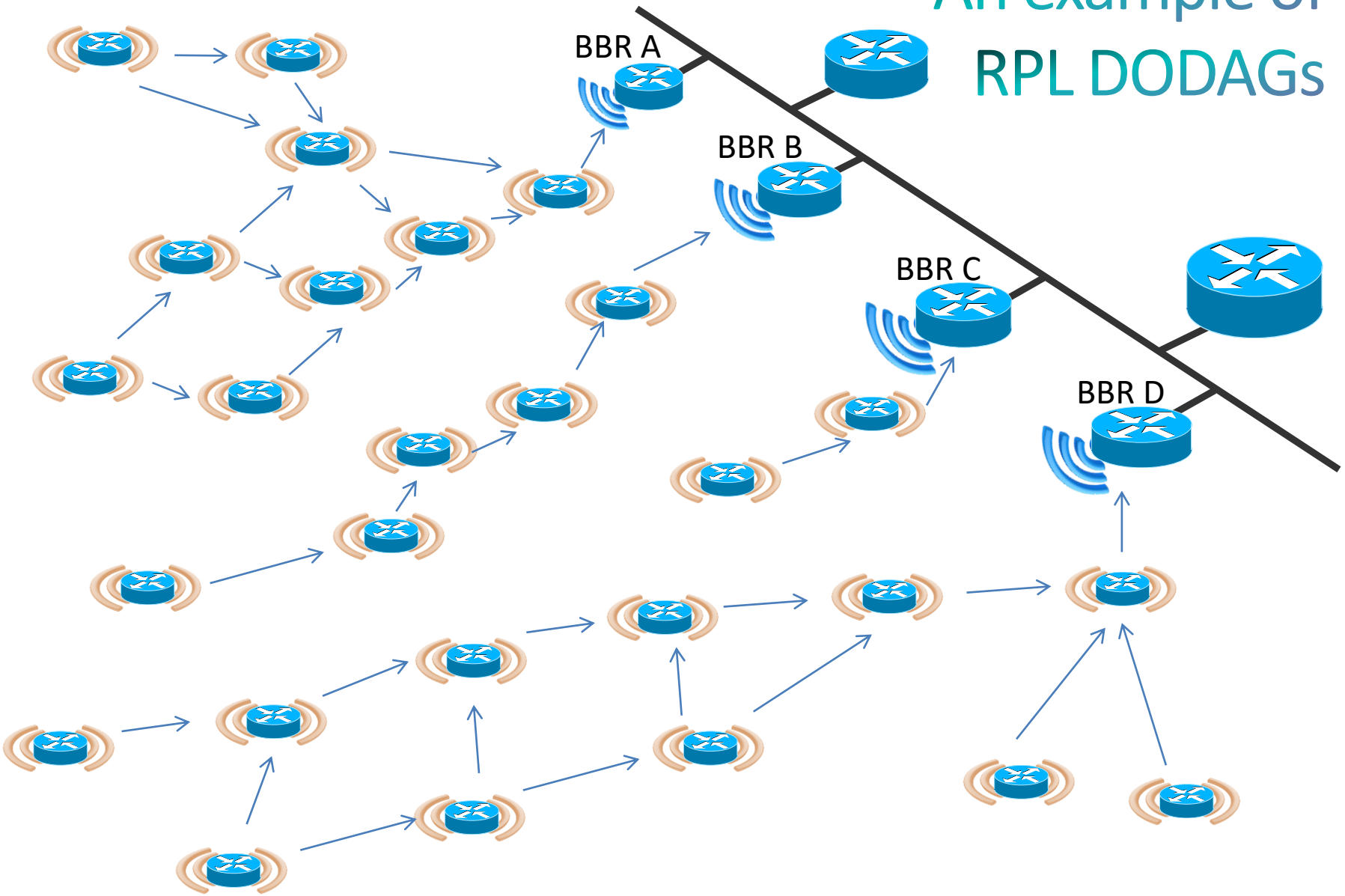
Classical addressing



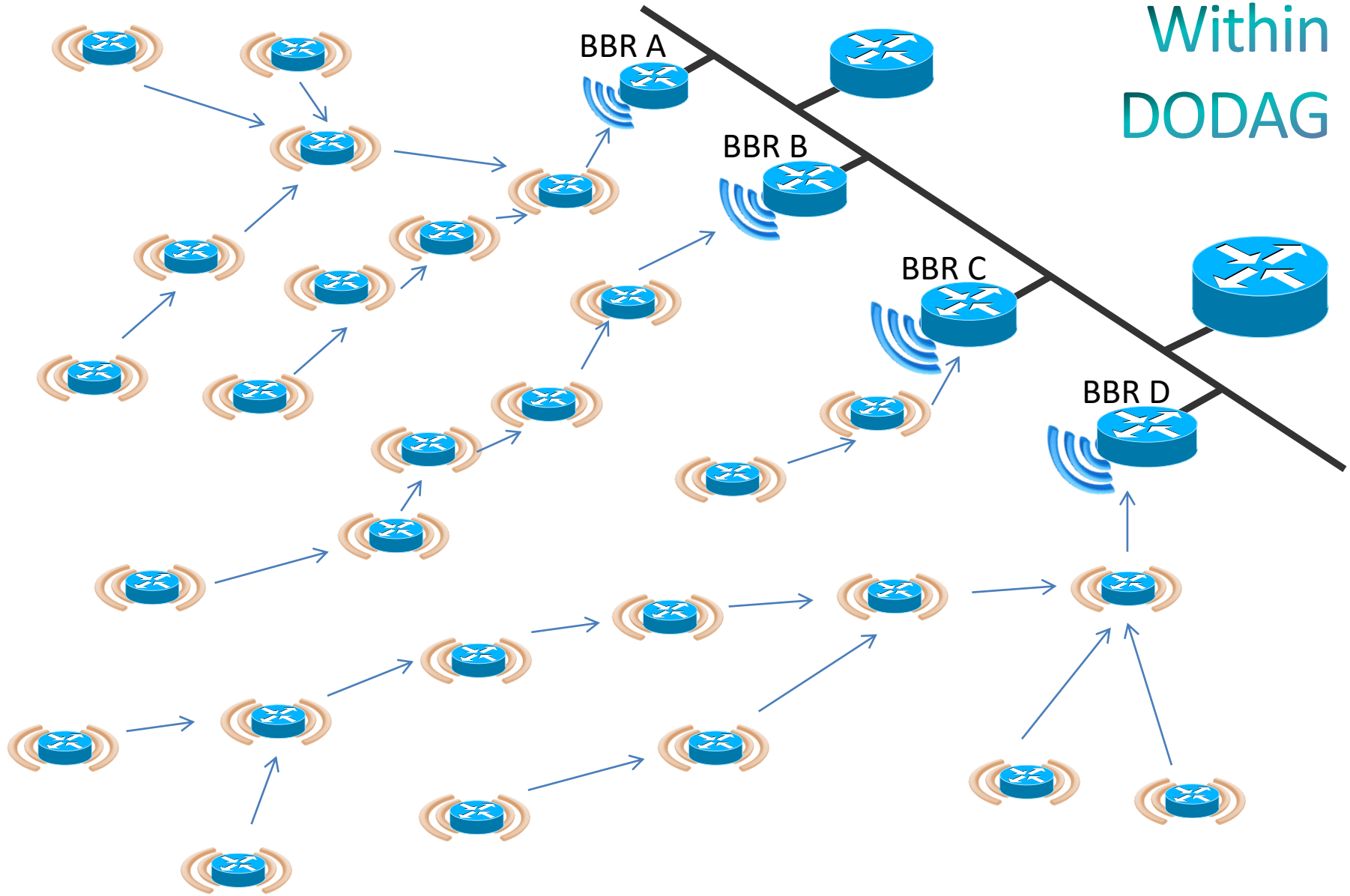
Example topology



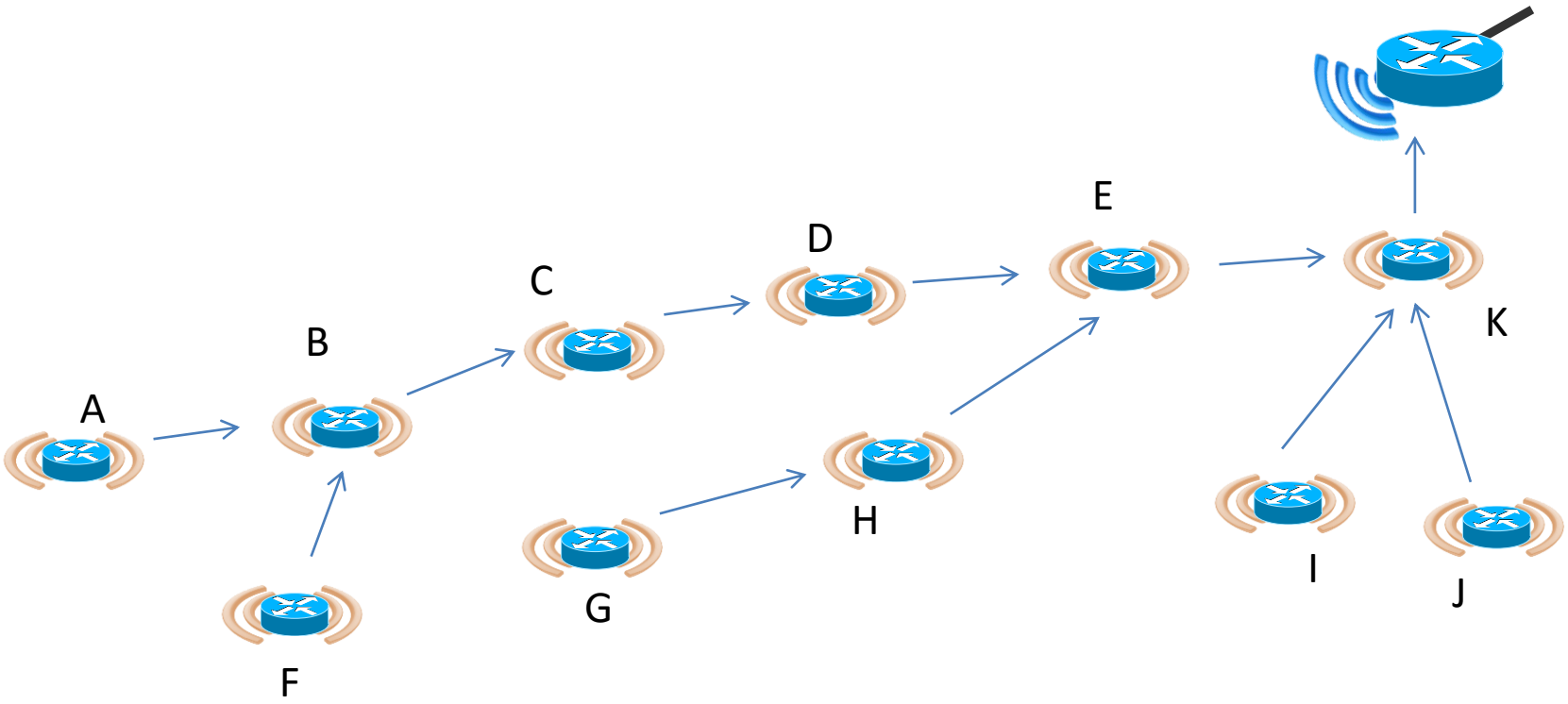
An example of RPL DODAGs



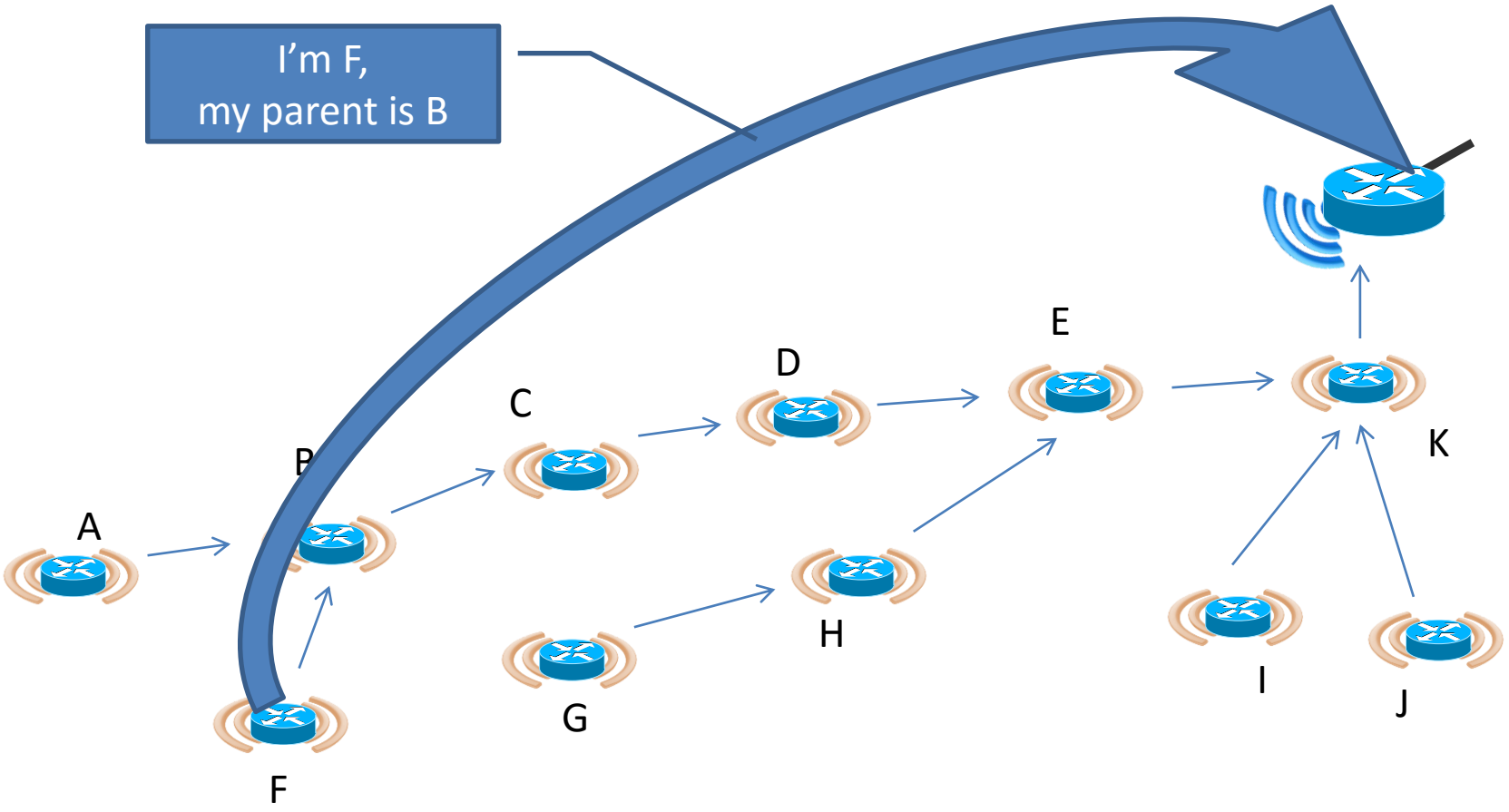
Preferred tree Within DODAG



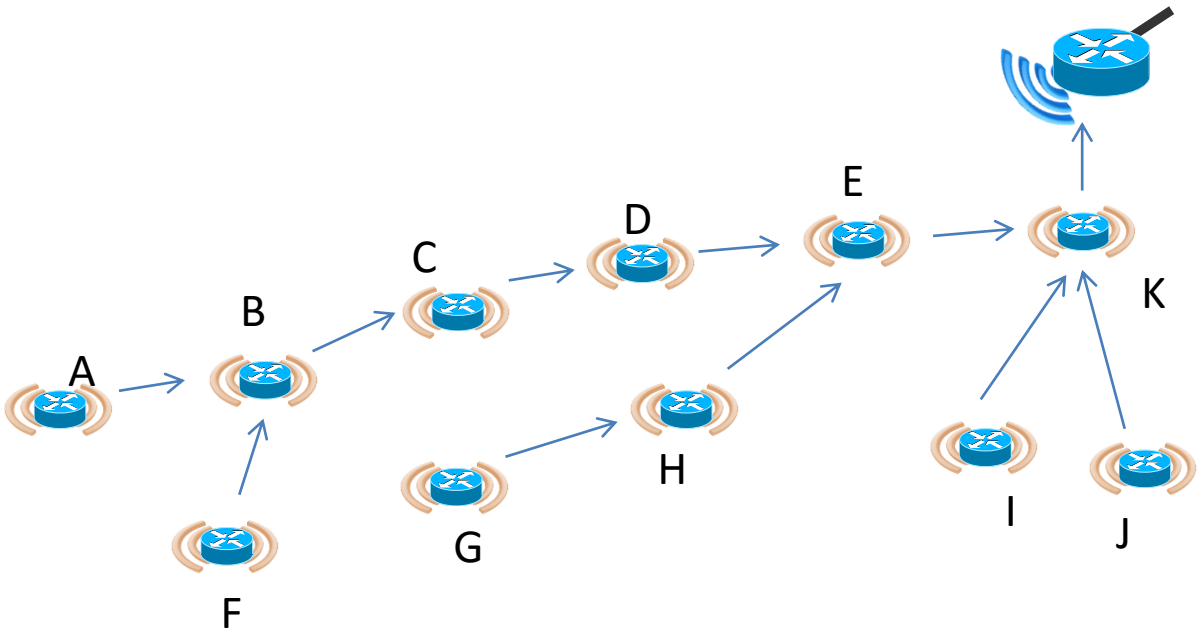
Classical addressing



Classical non storing routing



Classical non storing routing



Target	Transit
J	K
I	K
E	K
D	E
C	D
B	C
F	B
A	B
…	

Allocating bit positions

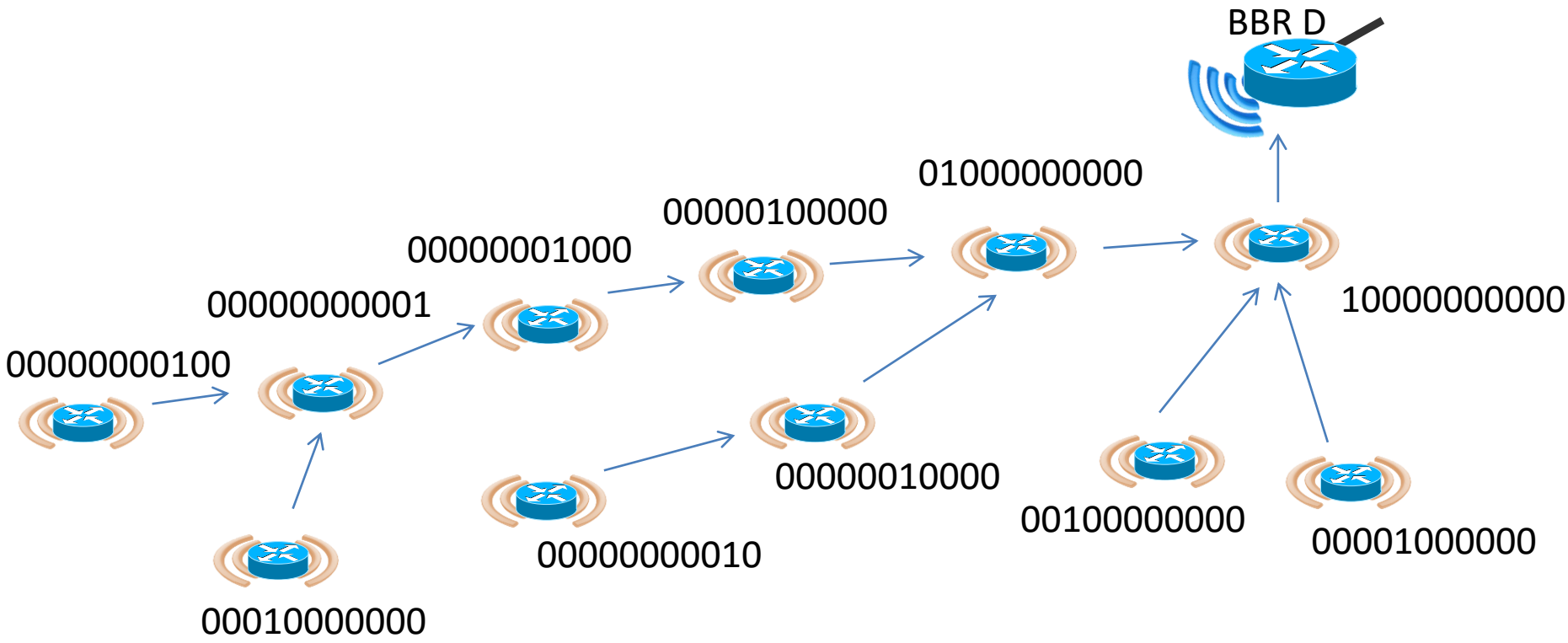
Option 0 - > static

Option 1 -> autoconf. RAs carry the current bitmap of allocated bits like they carry 6lowpan context info, and nodes pick a free bit. Collisions are handled as part of 6lowpan ND, like DAD.

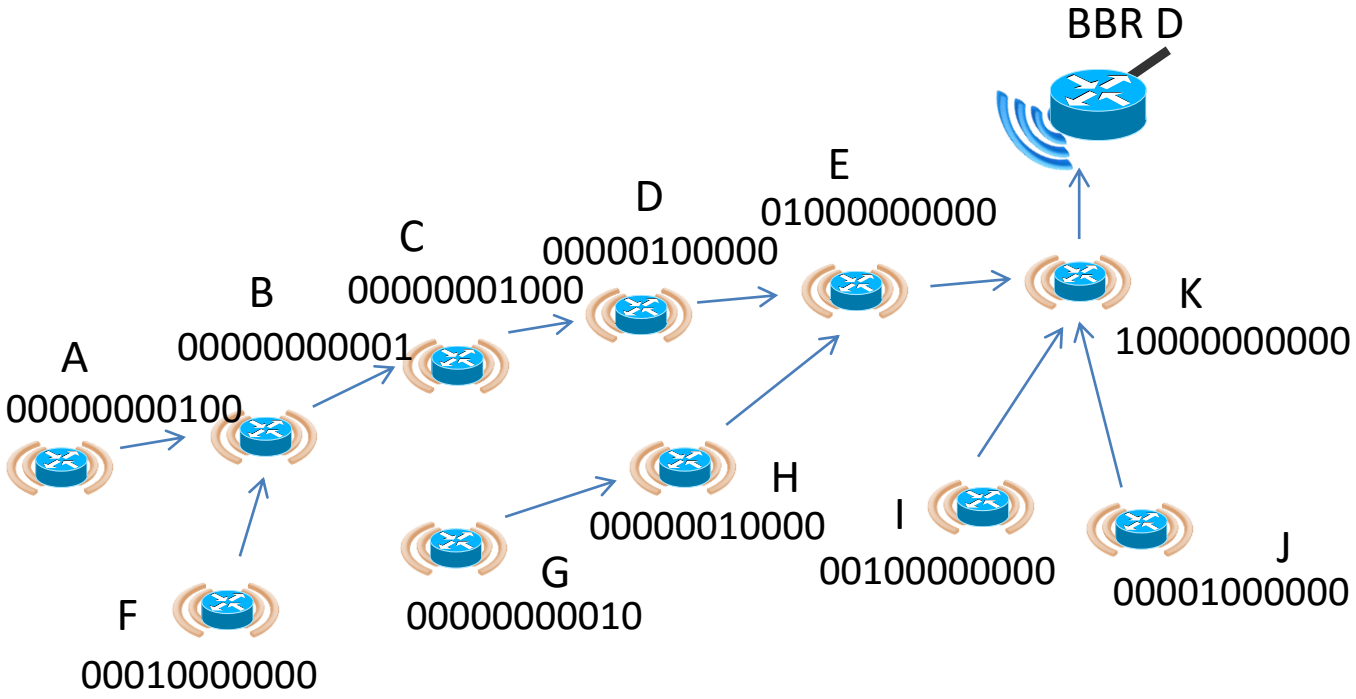
Option 2 -> the 6LBR assigns a bit and returns it on the DAR/DAC exchange

Note: Upon mobility to new 6LBR, a new bit has to be assigned.

Allocating bit positions

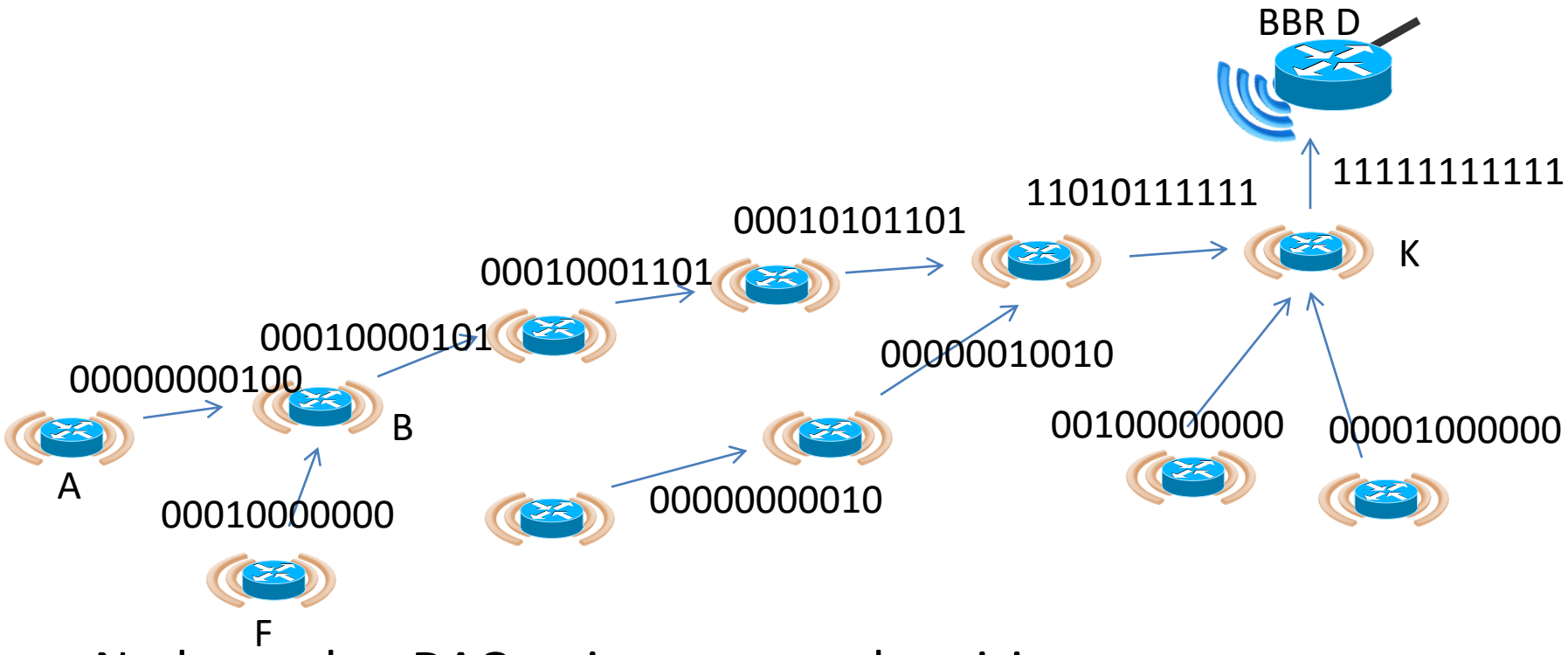


Address to bit position table stored in root



Address	Bit
A	9
B	11
C	8
D	6
E	2
F	4
G	10
H	7
I	3
J	5
K	1

Aggregating bits in DAO operation



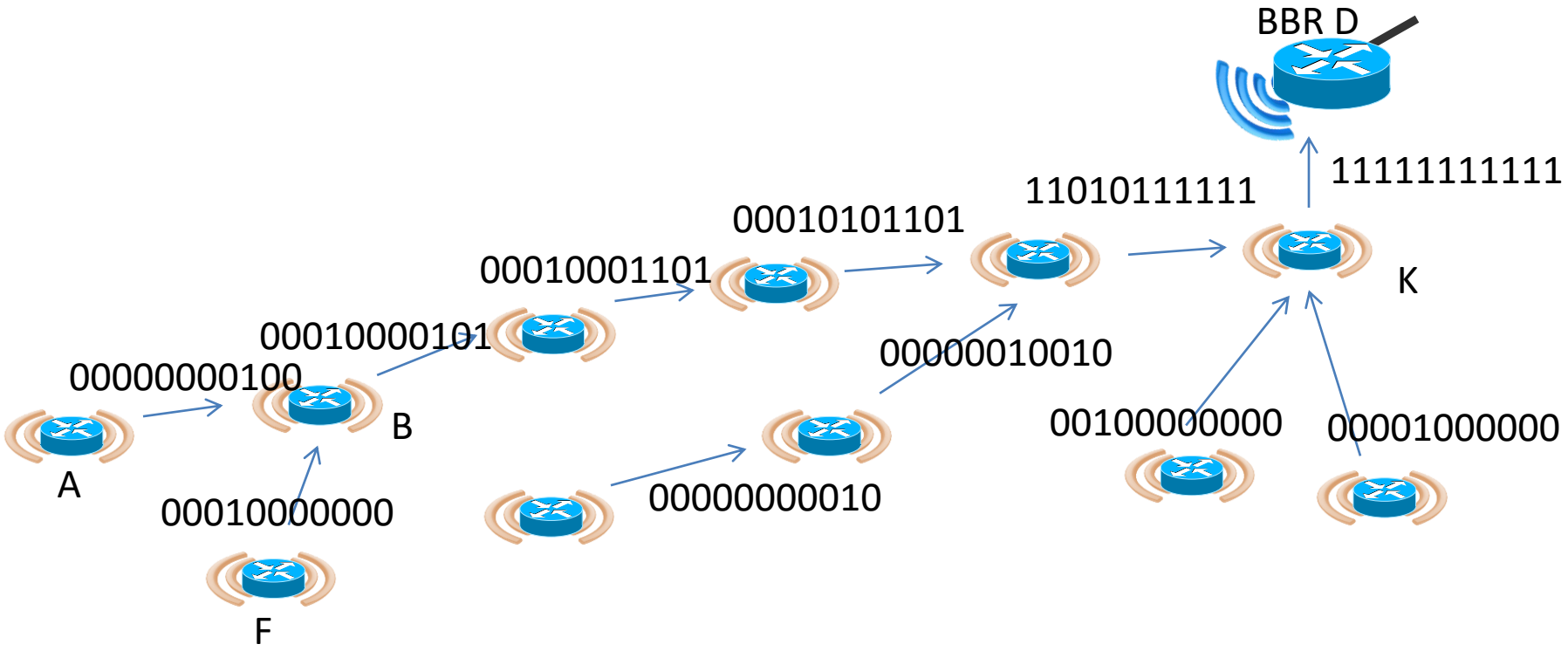
Node sends a DAO to its parent, advertising

$$n$$

Node's bitmap = (OR child I's bitmap) OR Node's bit

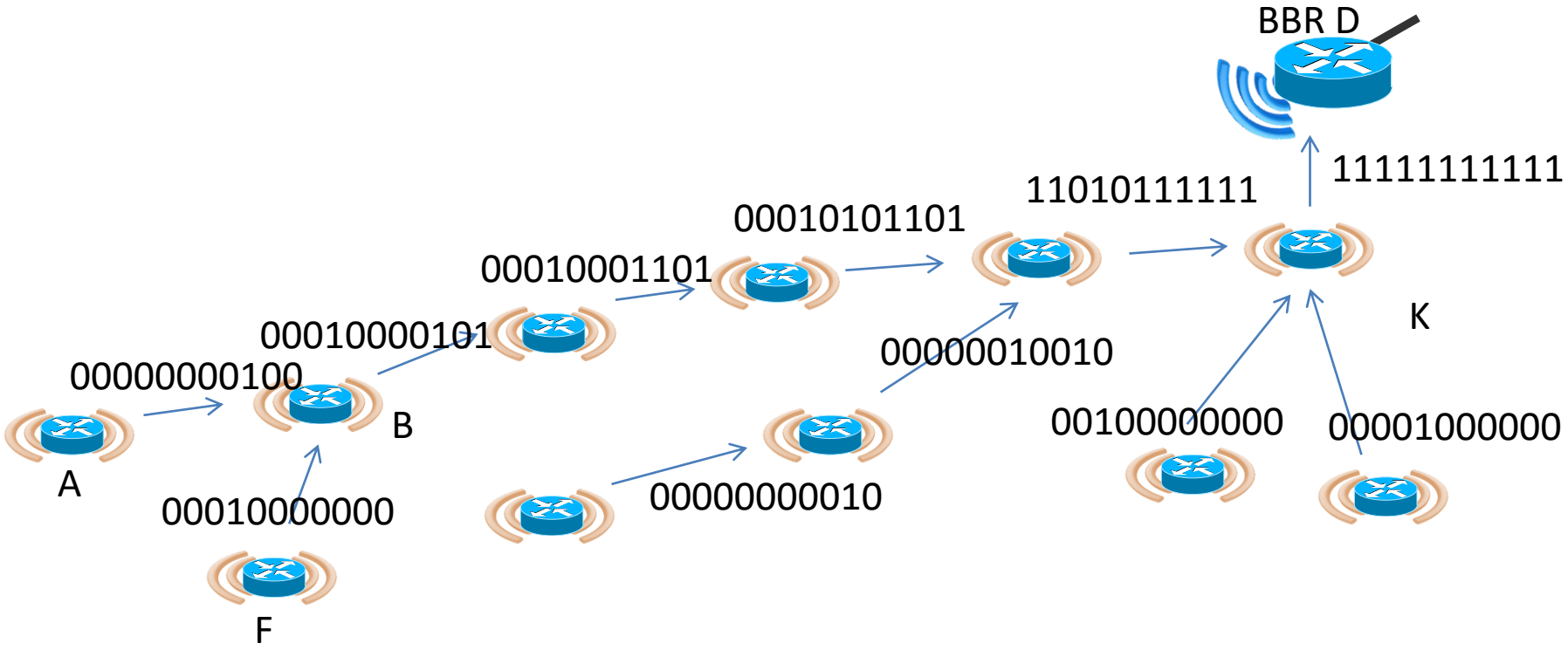
$$i=1$$

e.g. B's DAO operation



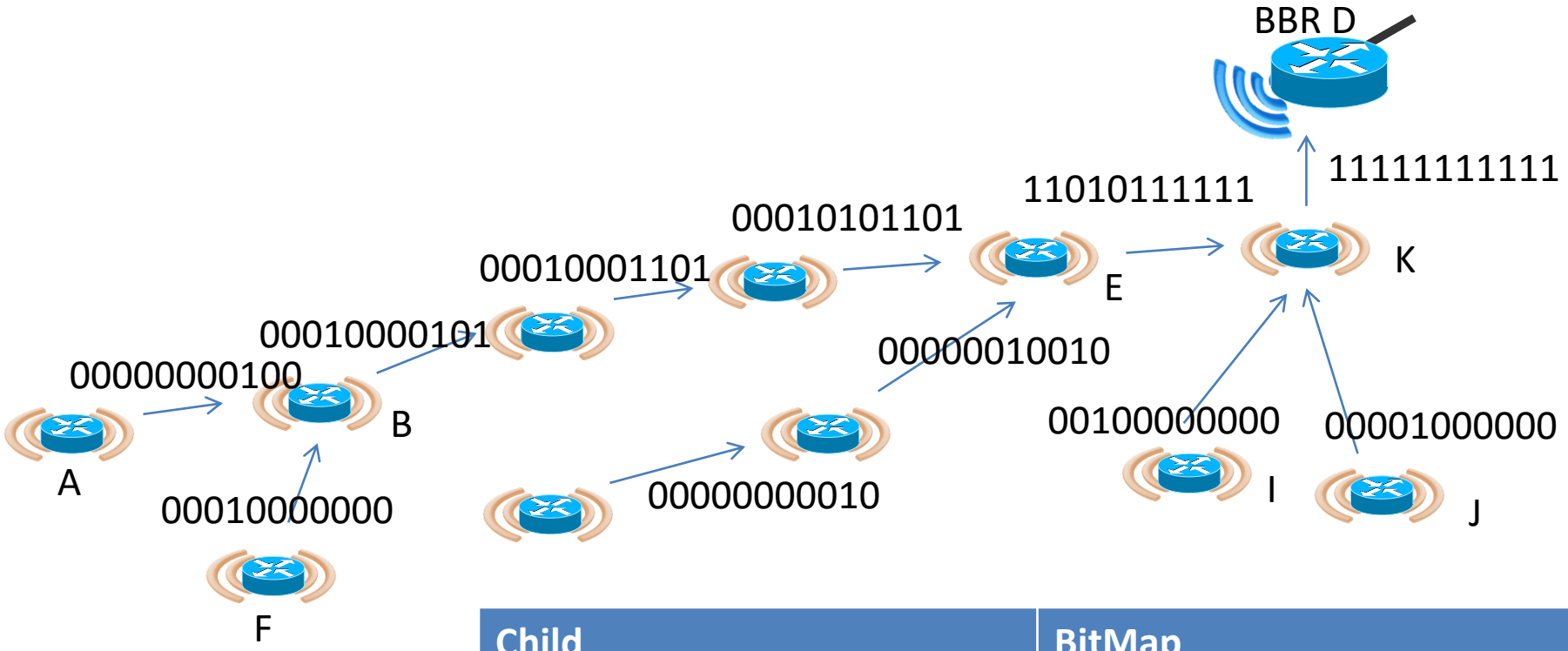
B sends a DAO to its parent, advertising
B's bitmap = (A's bitmap OR F's bitmap OR B's bit)

State in B



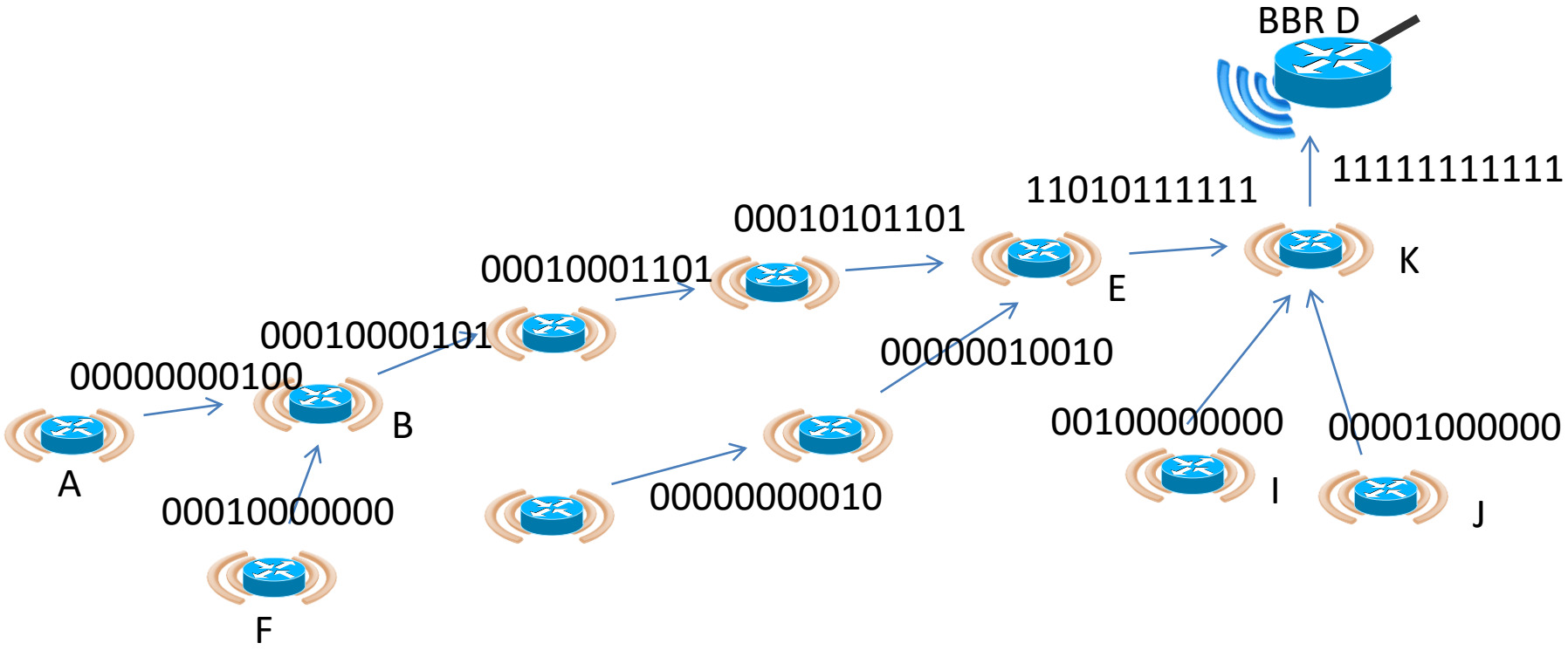
Child	BitMap
A	00000000100
F	00010000000

State in K



Child	BitMap
E	11010111111
I	00100000000
J	00001000000

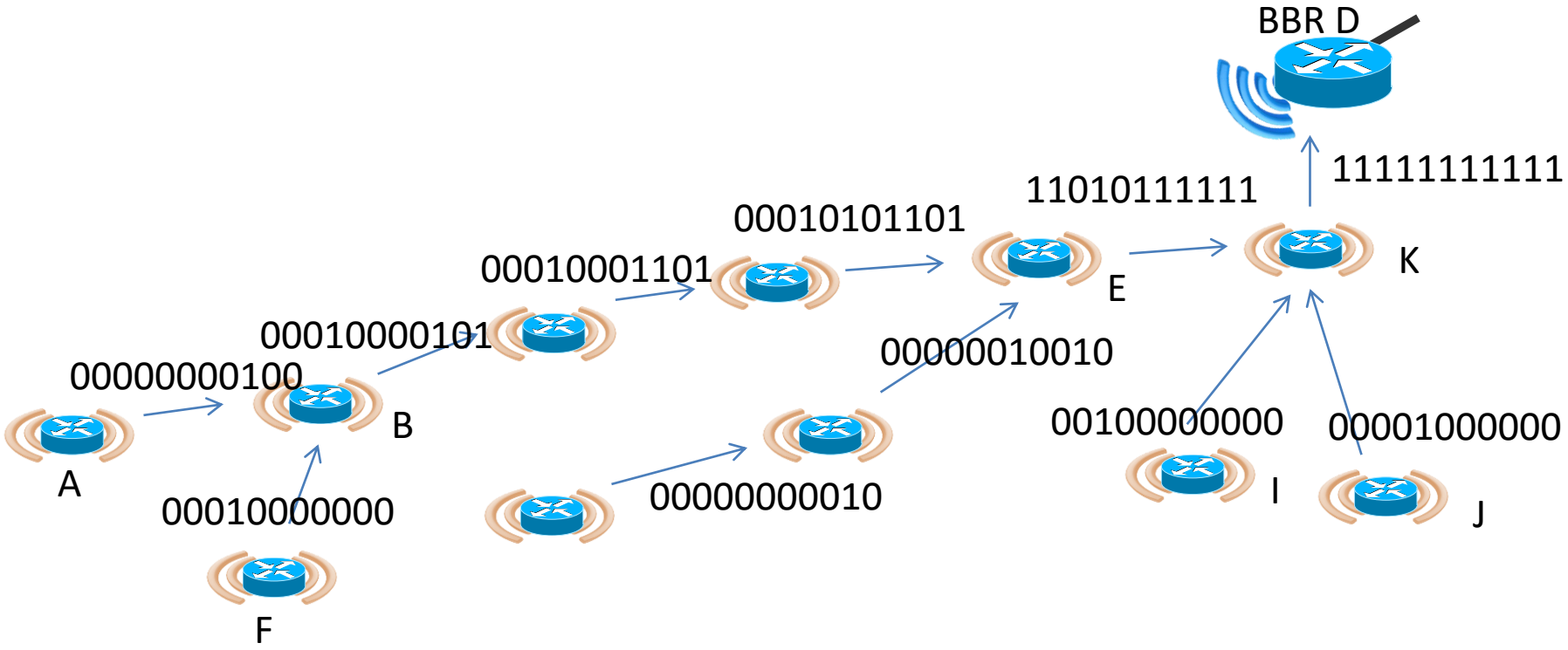
Message to n nodes



Root computes destination bitmap as

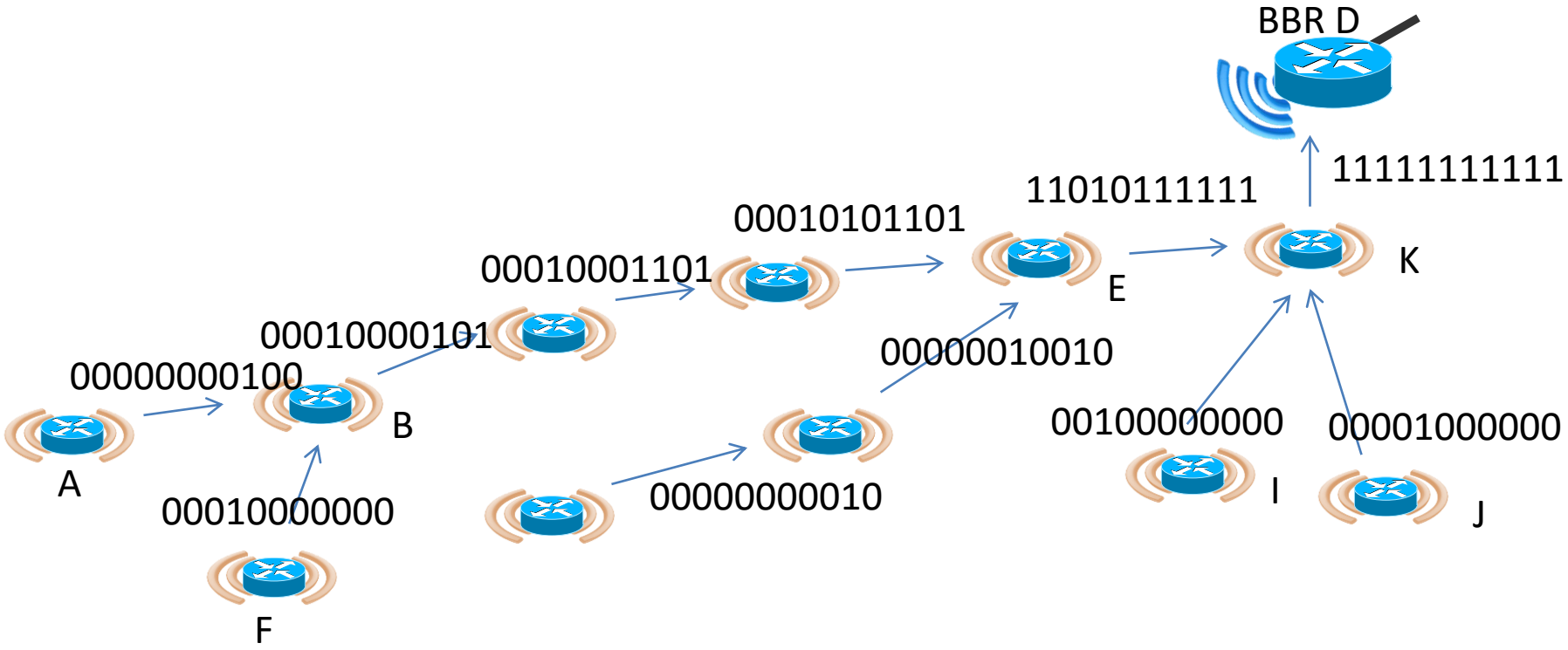
$$\text{Dest bitmap} = \bigvee_{i=1}^{n} (\text{OR node } i\text{'s bit})$$

Forwarding operation



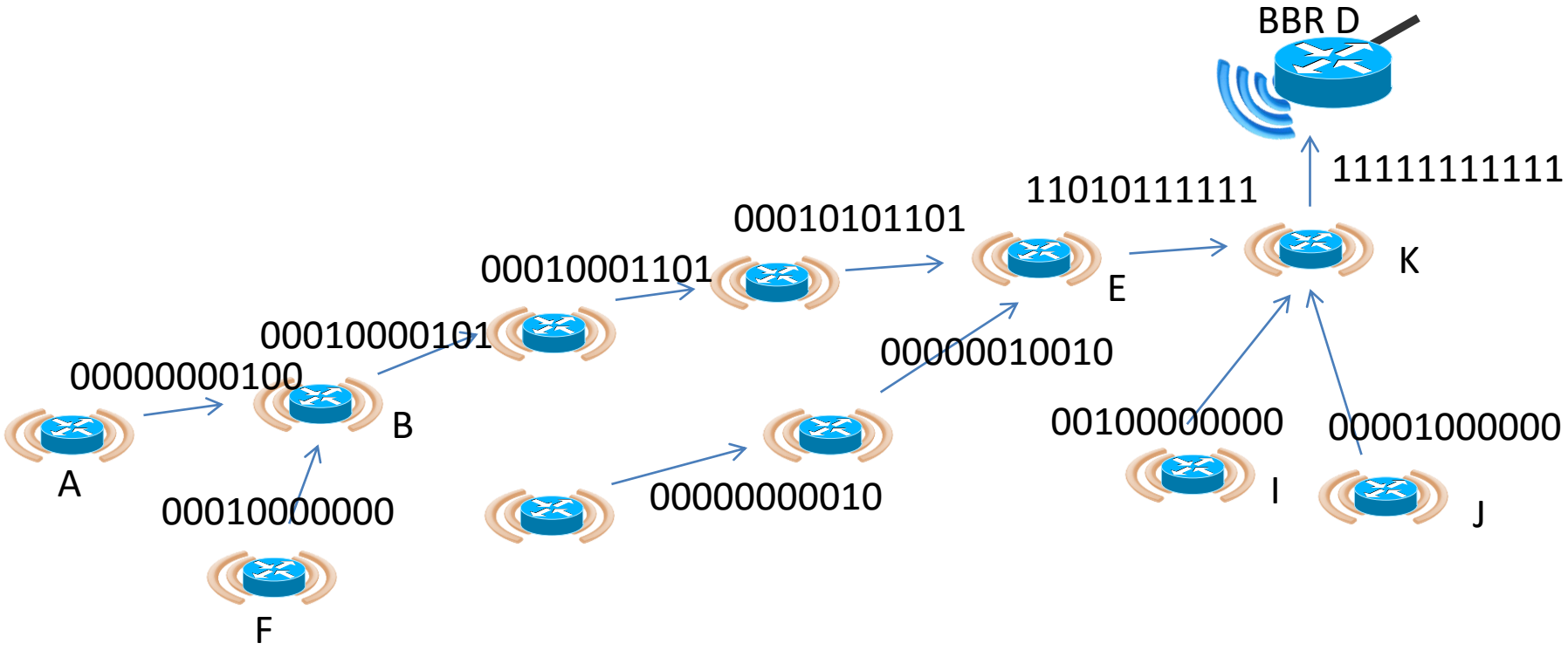
Node computes (Dest bitmap AND child's bitmap) for all children
When result is TRUE (non-zero), node copies the packet as a MC level unicast to child.

Alt Forwarding operation



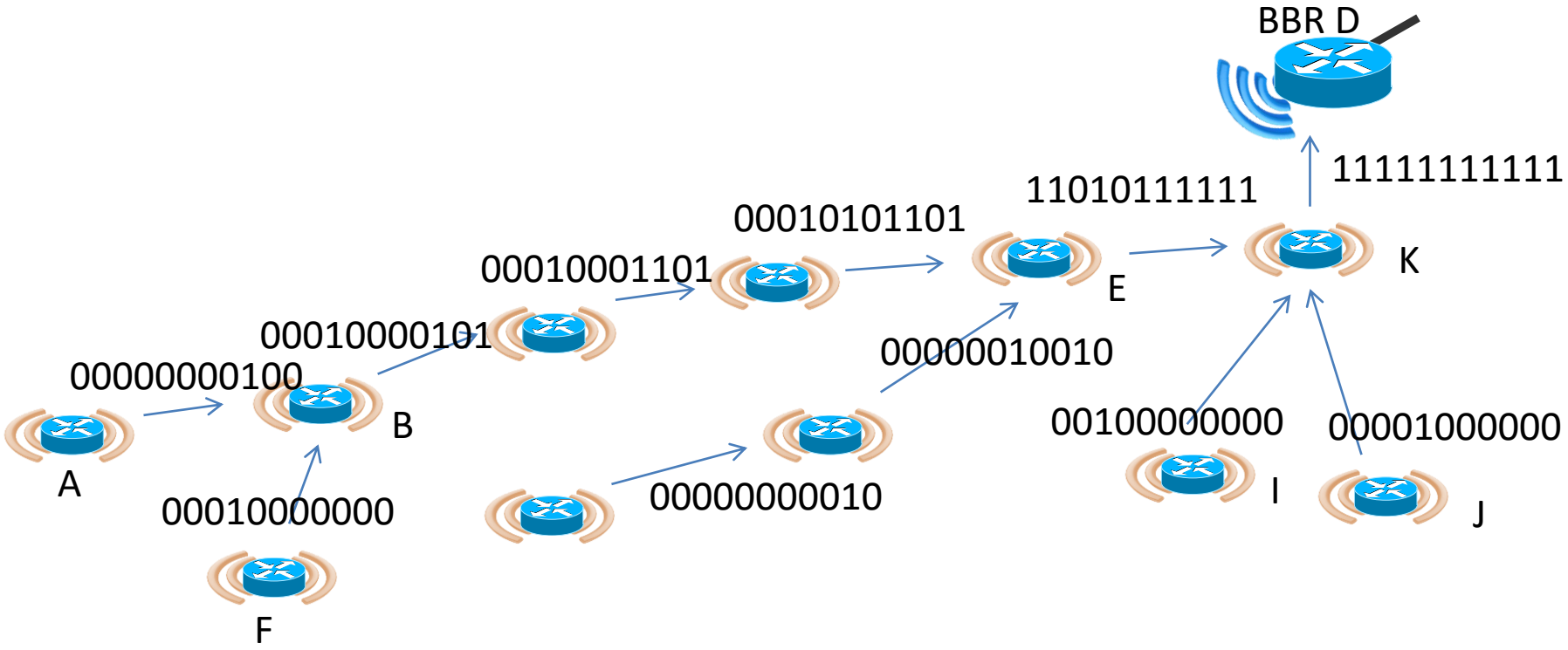
If most of the children are targetted, it makes sense to broadcast the message to all children. In that case, receiving children perform the OR operation with the bitmap they advertise in DAO and drop on receive is the result is not TRUE

e.g. Message to A, F and J



Root computes destination bitmap as
Dest bitmap = (A's bit OR F's bit OR J's bit)
= 00011000100

K's Forwarding operation



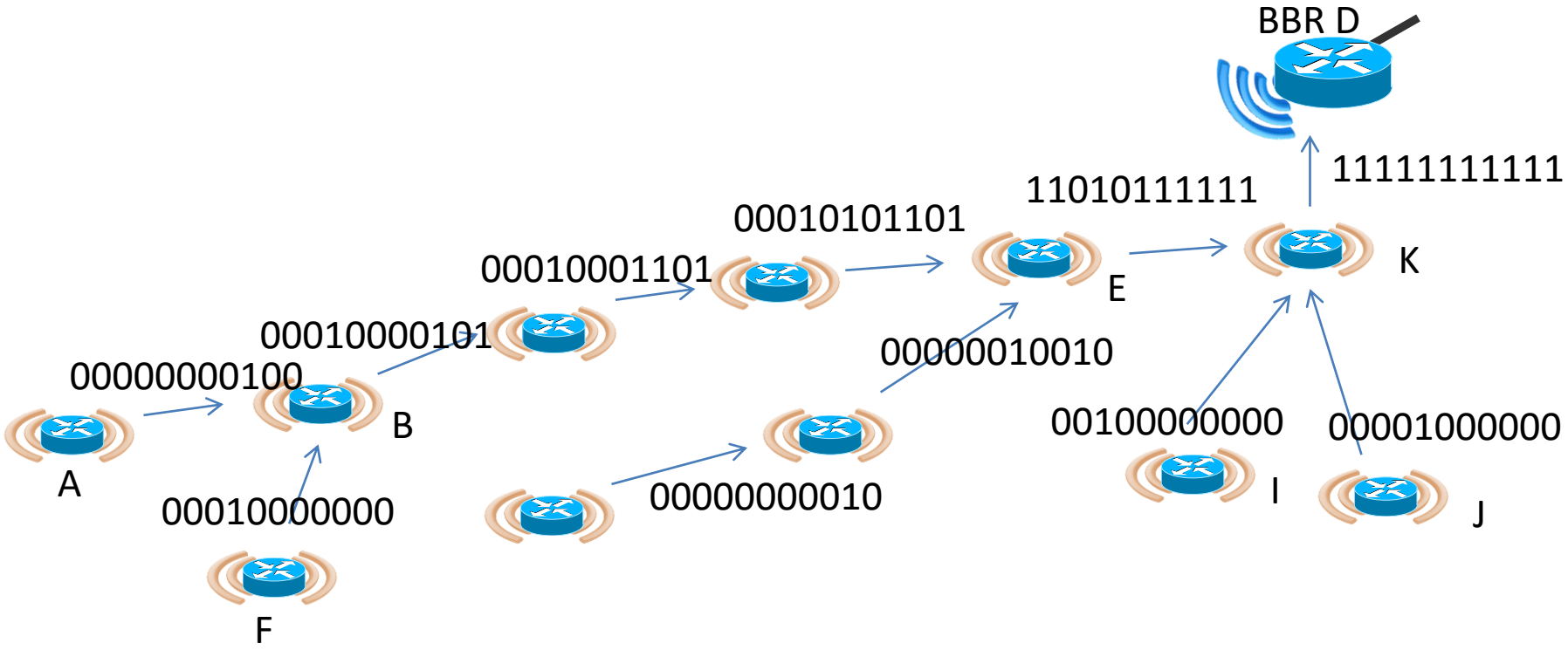
Dest bitmap = 00011000100

(Dest bitmap AND J's bitmap) = 00001000000 -> MAC unicast to J

(Dest bitmap AND I's bitmap) = 00000000000 -> NO copy to I

(Dest bitmap AND E's bitmap) = 00010000100 -> MAC unicast to E

B's Forwarding operation



Dest bitmap = 00011000100

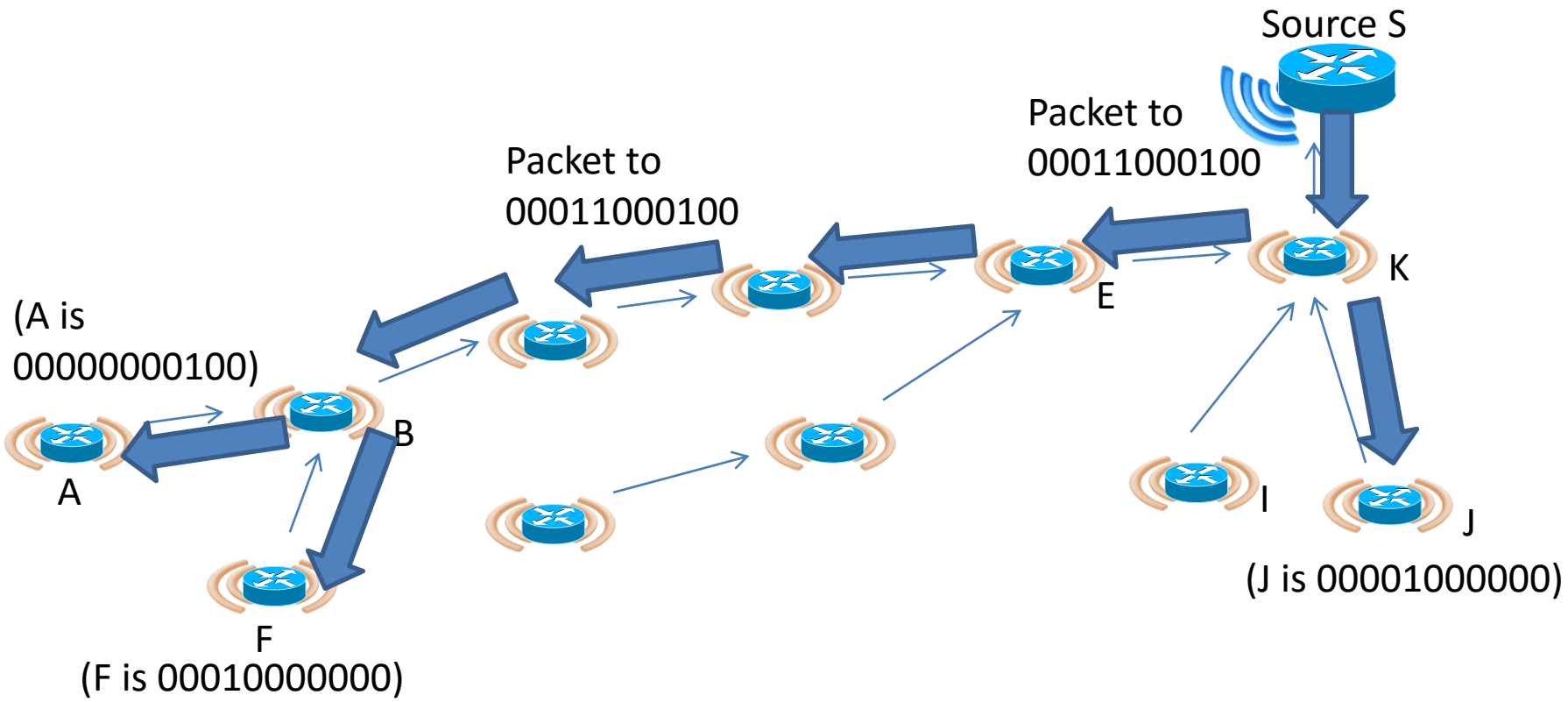
In B: (Dest bitmap AND B's bitmap) = most bits set -> B broadcasts

In A: (Dest bitmap AND A's bitmap) = 00000000100 -> A accepts

In F: (Dest bitmap AND F's bitmap) = 00010000000 -> F accepts

Reliable BIERPL

e.g. reliable mcast message to A, F and J

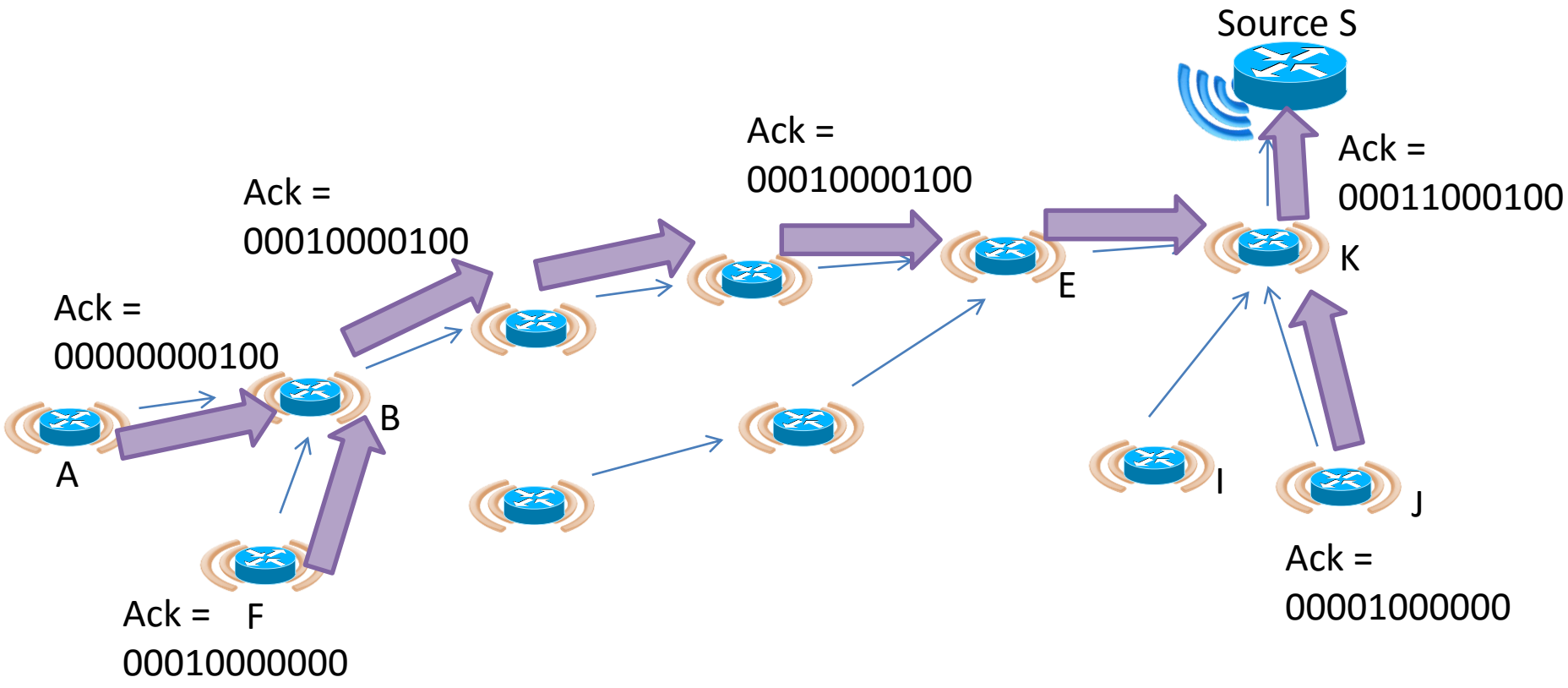


Root computes destination bitmap as

Dest bitmap = (A's bit OR F's bit OR J's bit) = 00011000100

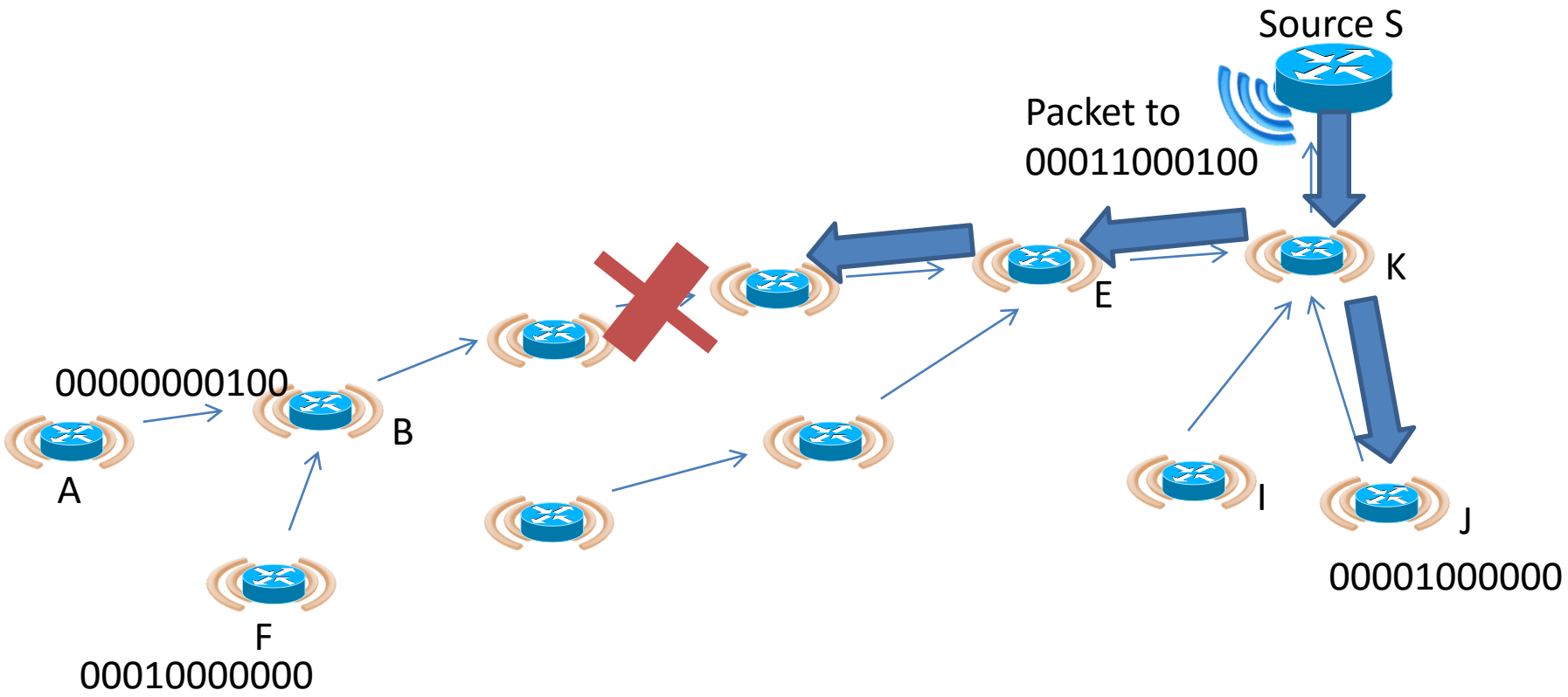
Forwarding expected to follow

e.g. reliable mcast message to A, F and J



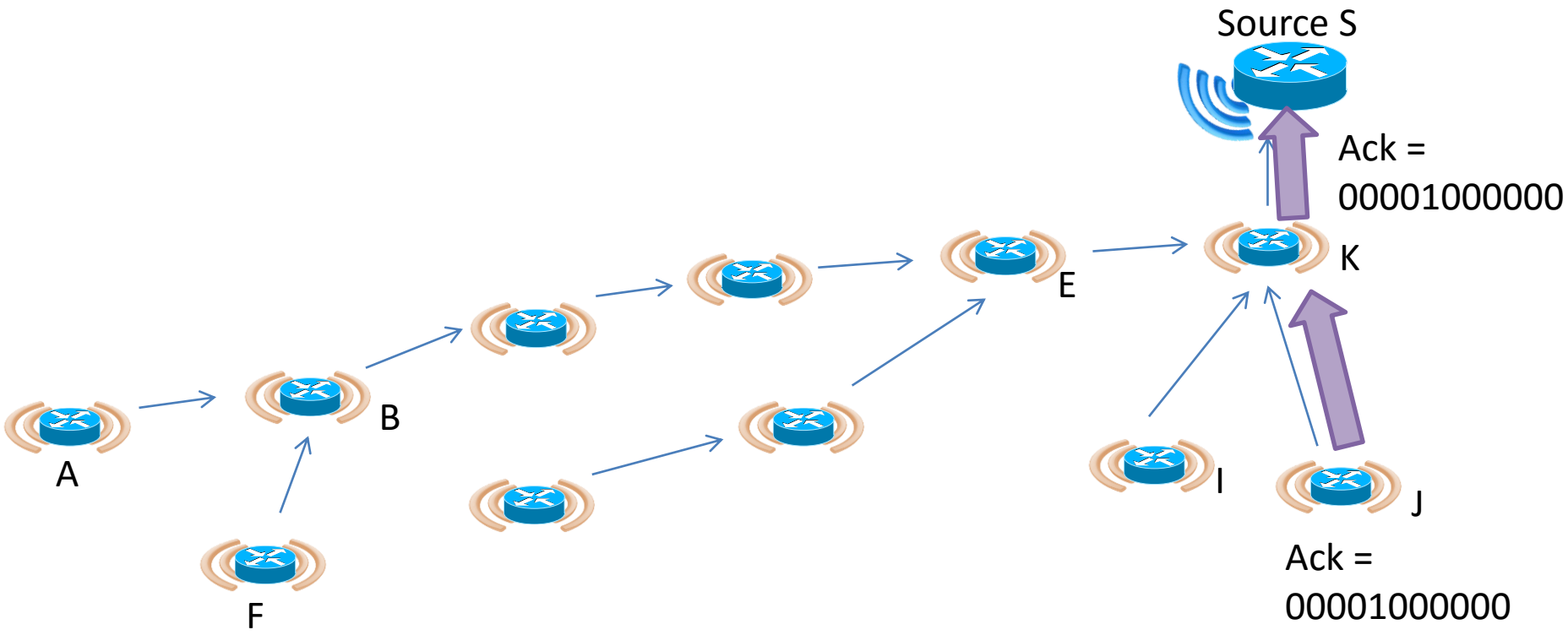
Acks are aggregated on the return path

Transmission failure down a branch



loss on the way in the branch that leads to A and F

Ack bitmap reports only J reception



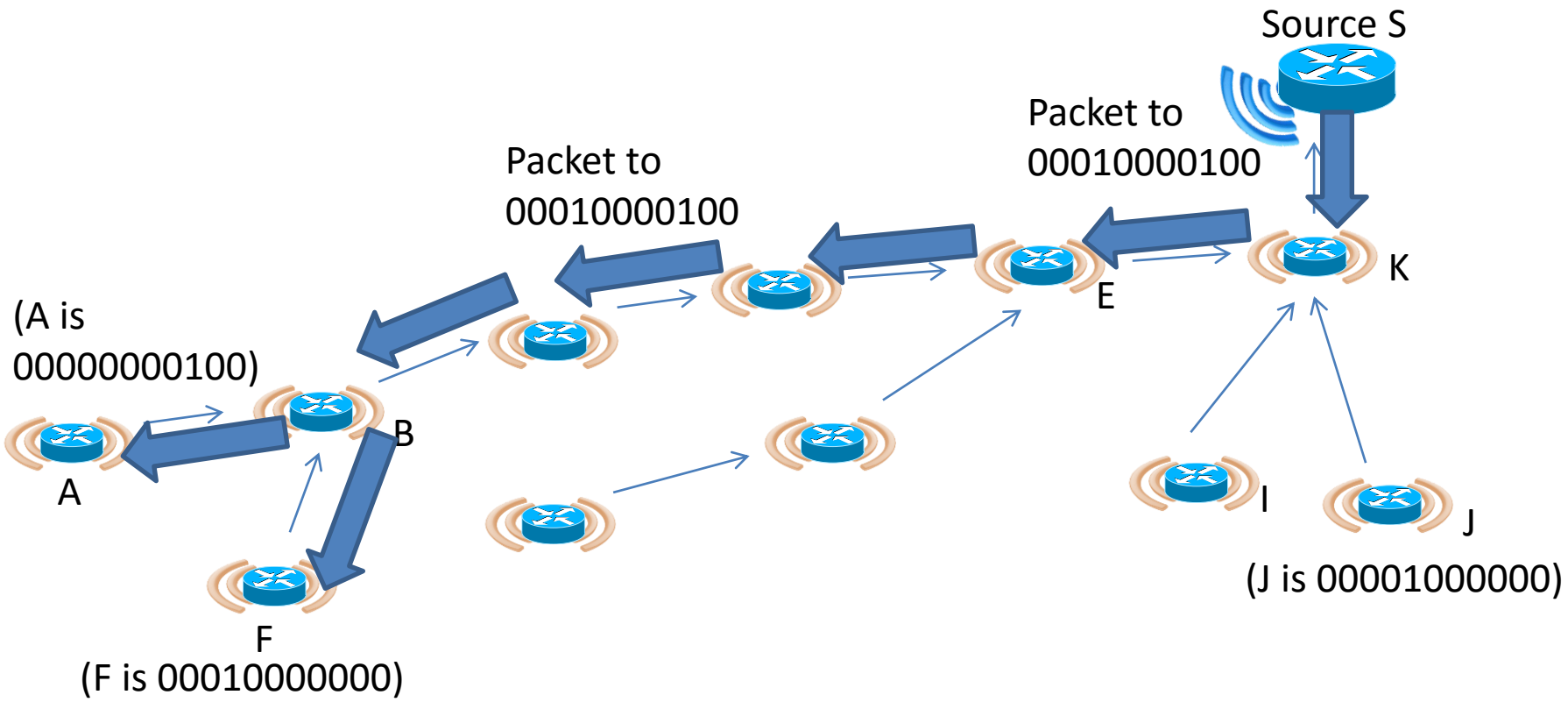
Root computes destination bitmap – ack bitpmap

Retrans bitmap = Dest bitmap - Ack bitmap

= 00011000100 - 00001000000

= 00010000100

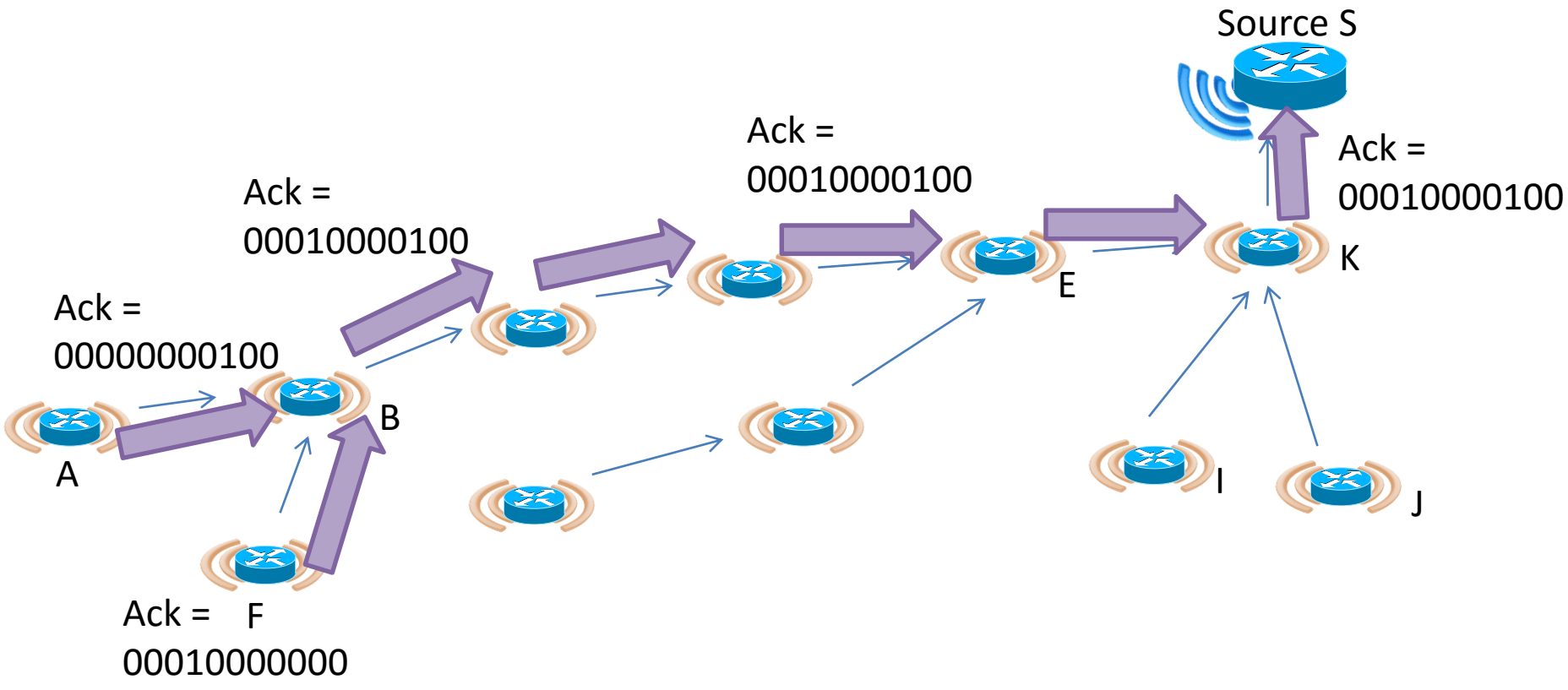
e.g. reliable mcast message to A, F and J



Retransmission bitmap indicates only along failed branch(es)

Forwarding expected to follow

Ack from A and F



Acks are aggregated on the return path

Efficient No-Path DAO for RPL Storing MOP

(clarification according to discussion during adoption call)

https://tools.ietf.org/html/draft-ietf-roll-efficient-npdao-00

Rahul, Rabi, Zhen@ Huawei

IETF99, Prague

History:

IETF95 - Presented the problem statement

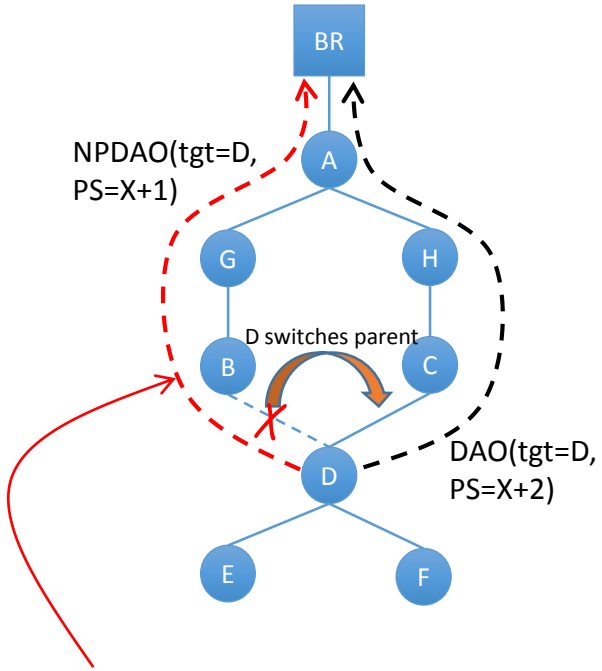
IETF96 - Presented existing solutions based on comments rcvd and why those fall short

IETF98 – Presented new solution for improving route invalidation

Pre IETF99 – adopted as WG document , THANK YOU SO MUCH FOR THE REVIEW

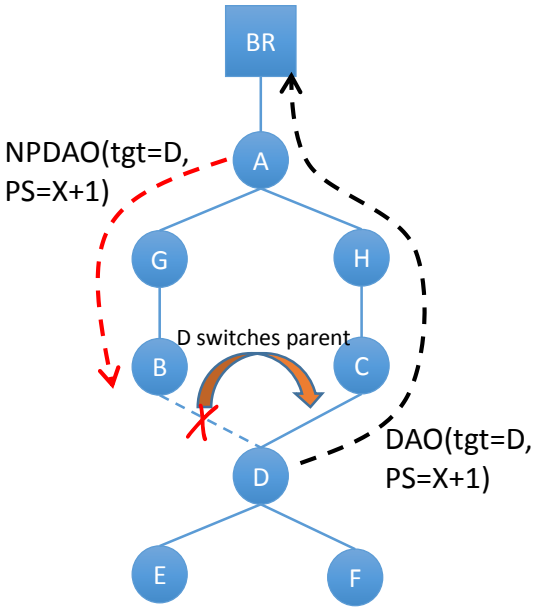
Recap: the problem and the solution

Current RPL NPDAO



NP-DAO via broken links will cause many problems such as reachability and efficiency

Proposed Change



- Send the DAO via the new parent;
- Trigger the common parent to send the NP-DAO downstream to invalidate the broken path

PS = PathSequence
Tgt = Target

Clarification#1: Compatibility with current NPDAO…

Is it needed to be compatible?

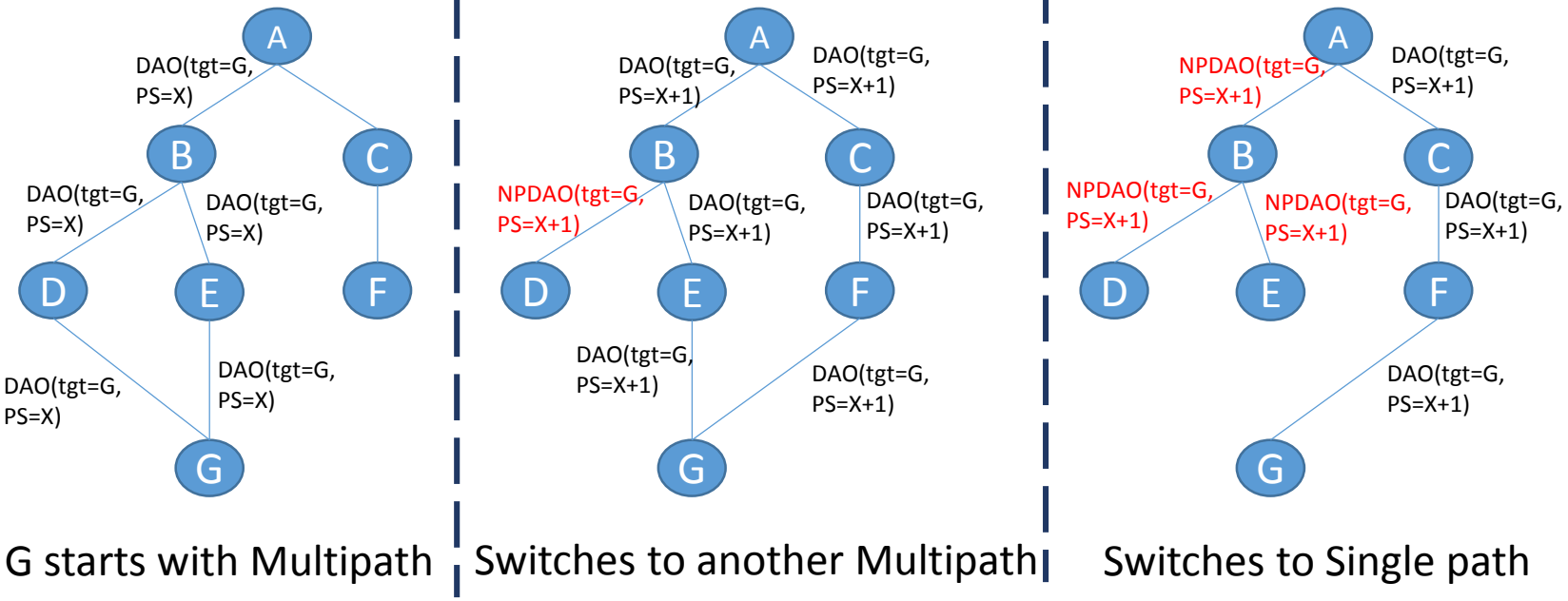
- Yes. Scenario: Gracious node shutdown should result in an upstream NPDAO.

- The newly proposed downstream NPDAO is compatible with upstream NPDAO.

Clarification#2: Impact on Multipath routing

Node may send the same DAO with the same PathSequence to multiple preferred parents to establish multipath routing.

Proposed downstream NPDAO is compatible with current multipath routing semantics.



Clarification #3: a new DAO is needed?

- We extended the existing DAO messaging since most of the containers and the header flags that are used would be same.
  - For e.g. Target container, Transit Information container
- Current NPDAO clears the route only when it is received from the same next hop based on which the route entry was previously established.
  - Downstream NPDAO does not follow this rule.

Next Step

- Contiki based implementation in-progress…
- Welcome any feedback from the Open-source community (while we believe the technique description is stable enough)

Thank you

AODV-RPL

To be added

Optimization of Parent-node Selection in RPL-based Networks

draft-hou-roll-rpl-parent-selection-00

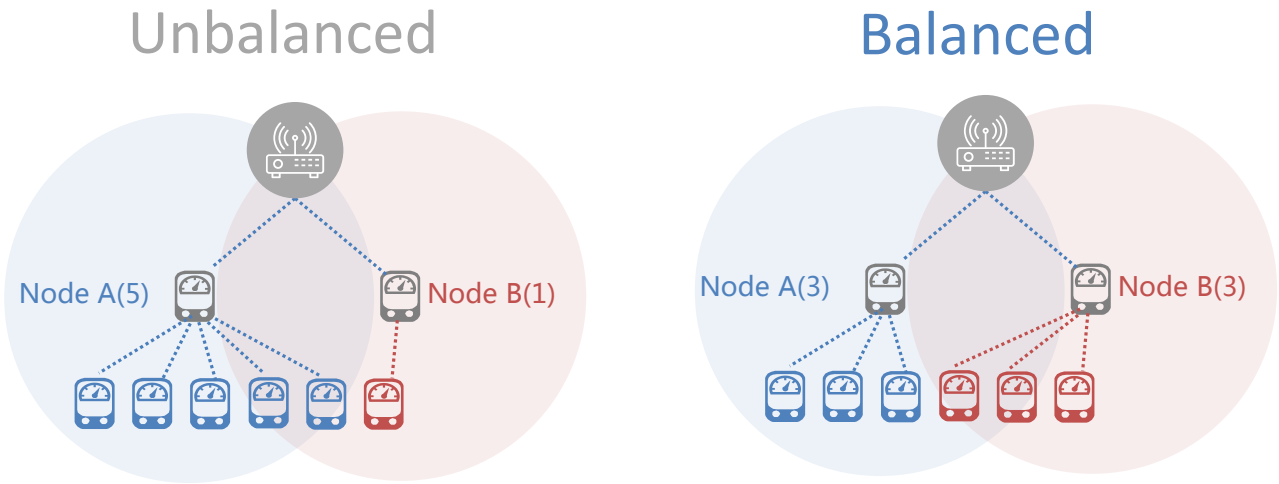
Jianqiang Hou (houjianqiang@huawei.com)
Rahul Arvind Jadhav (rahul.ietf@gmail.com)
Zhenhui Luo (luozhenhui@huawei.com)

Overview

- Draft version 00 (March 12, 2017)
  - Two problems are addressed
    - 1. "Thundering Herd" problem
    - 2. Randomly Unbalanced Networking
  - Problem 2 has been illustrated by Qasem Mamoun in IETF98
  - Today I will introduce our solution for Problem 2

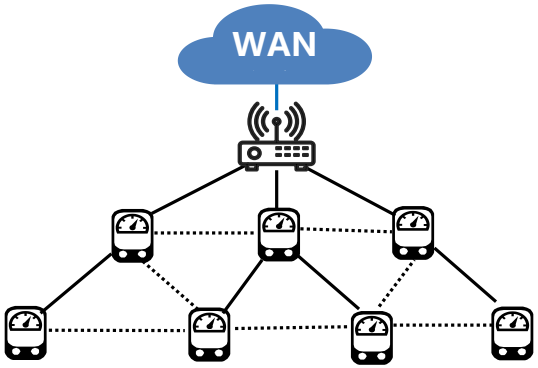
Problem Statement

- Randomly Unbalanced Networking
  - This problem has been stated in IETF98



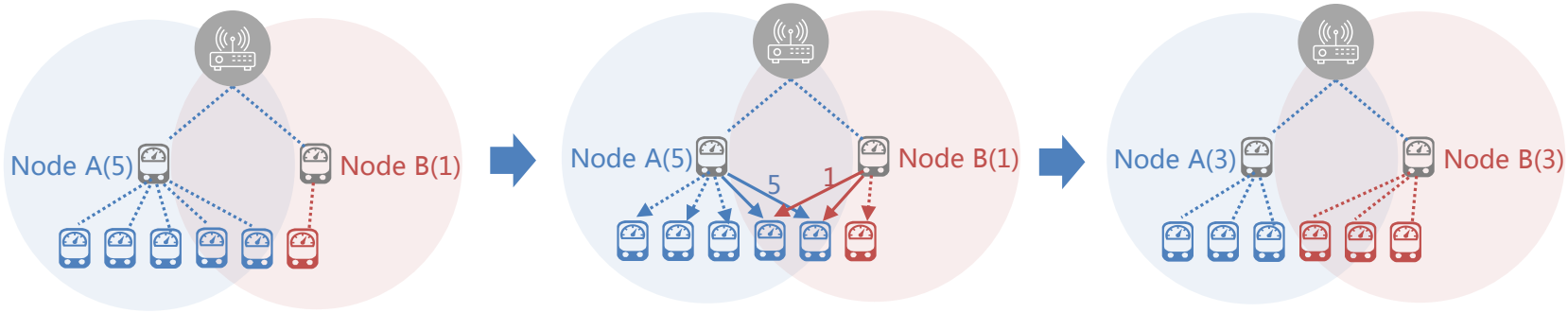
Possible Using Scenarios

- **Goal: balancing the number of child nodes**
  - Indirect way of reflecting the traffic load
  - Can be used when all nodes send packets in the same size and frequency
    - number of children = traffic load
    - Scenarios: **Advanced Metering Infrastructure**



Our Solution (1/5)

- **Compute the number of children from Neighbor Table**
- Create a new Metric
- Combine the number of children with other metrics/constraints, e.g. ETX, HopCount, Latency

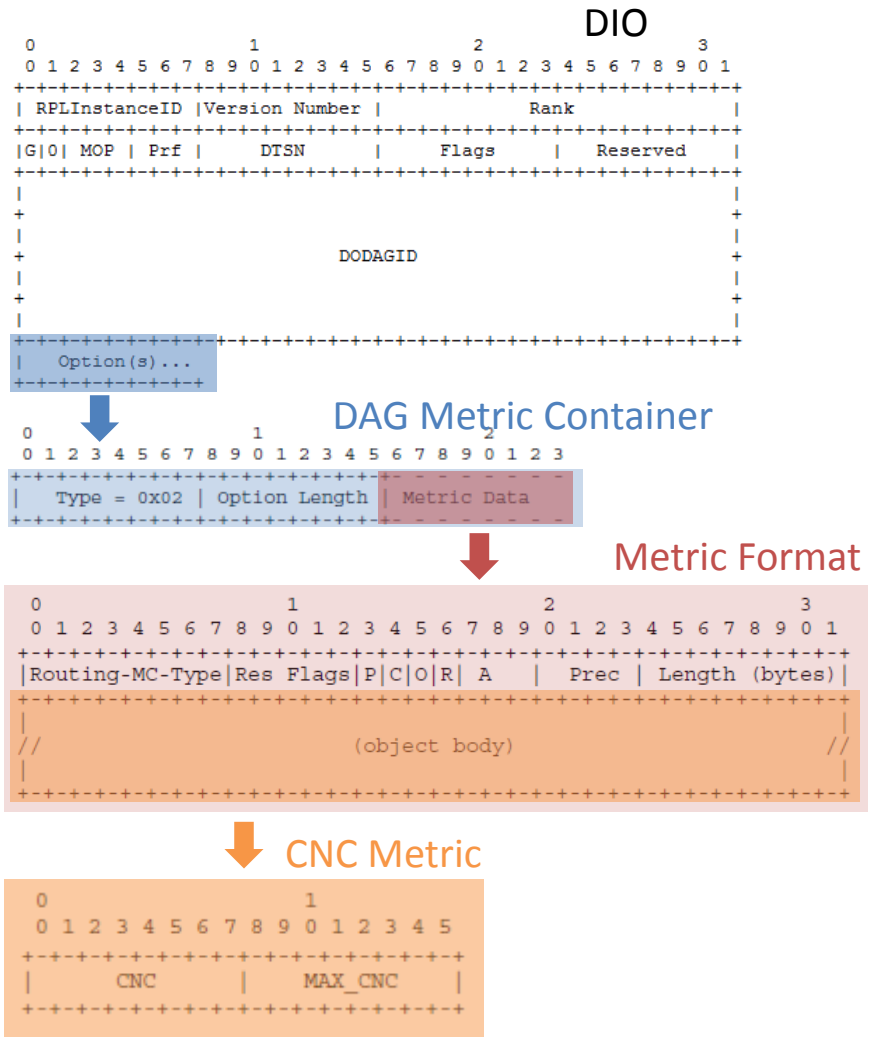


Parent nodes send DIO + Number_of_Children

Nodes select their parent node based on Number_of_Children

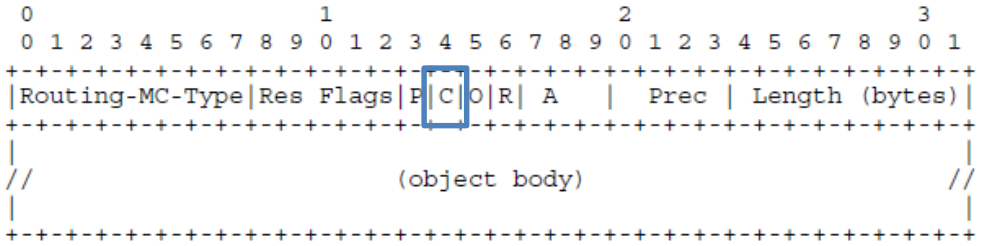
Our Solution (2/5)

- Create a new metric
  - Child Node Count (CNC)
  - DIO->DAG Metric Container->CNC Metric
  - MAX_CNC
    - Minimize the use of DAO-NACK

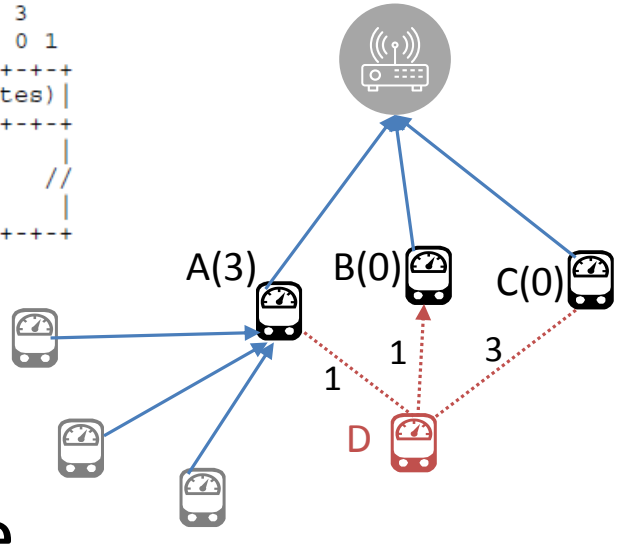


Our Solution (3/5)

- Constraint + Metric demo: ETX + CNC
  - ETX (**C=1**, O=0, A=00, R=0, Prec=0), MaxValue=2
  - CNC (**C=0**, O=0, A=00, R=0, Prec=0)

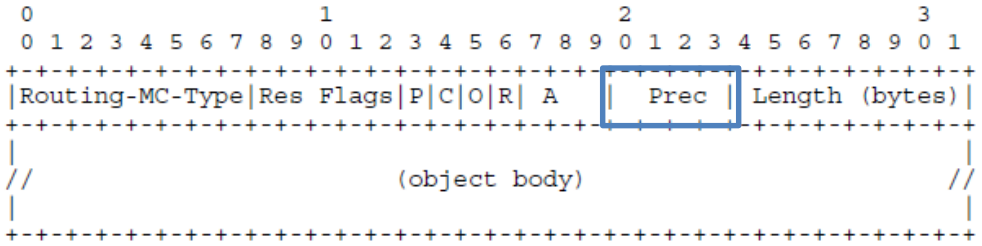


  - ETX: A-D 1, B-D 1, ~~C-D 3~~
  - CNC: ~~A 3~~, B 0
- D choose B as parent node

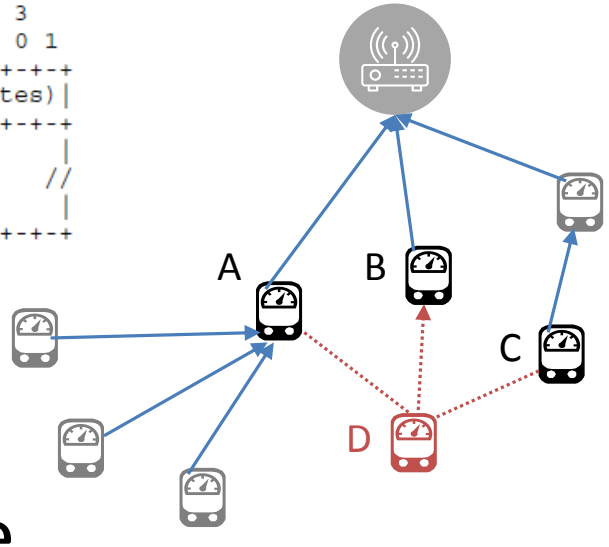


Our Solution (4/5)

- Hybrid RANK demo: HopCount + CNC
  - HopCount (C=0, O=0, A=00, R=0, **Prec=0**)
  - CNC (C=0, O=0, A=00, R=0, **Prec=1**)

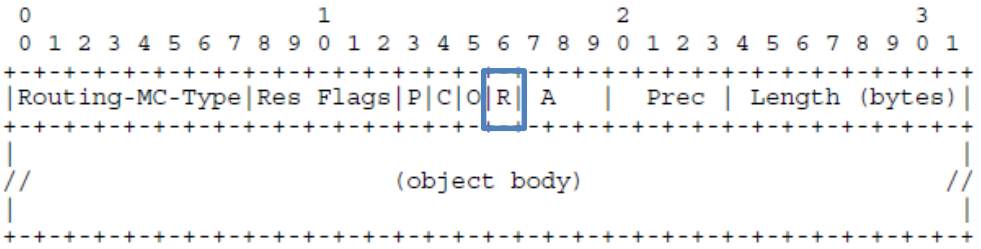


  - HopCount: A 2, B 2, ~~C 3~~
  - CNC: ~~A 3~~, B 0
- D choose B as parent node

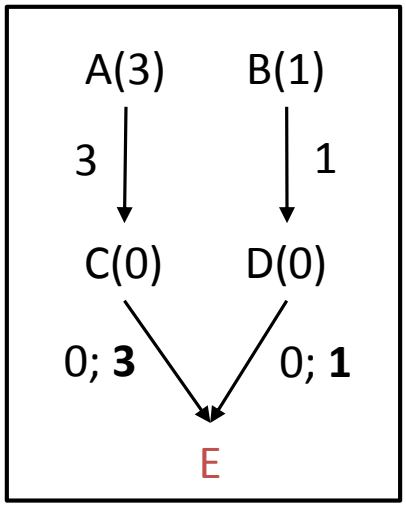


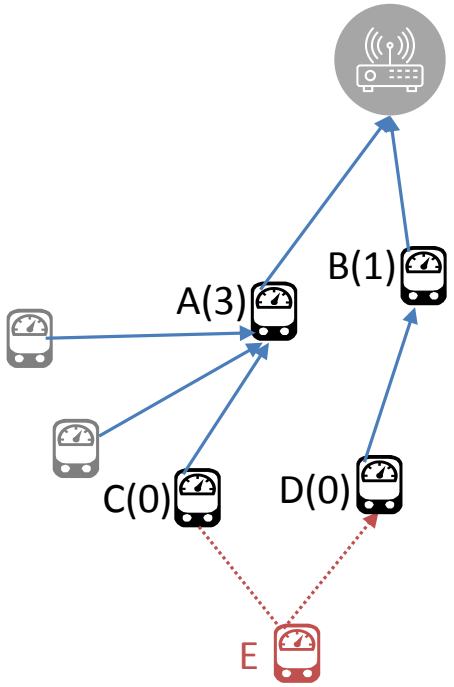
Our Solution (5/5)

- Aggregation CNC demo
  - (C=0, O=0, A=00, **R=1**, Prec=0)



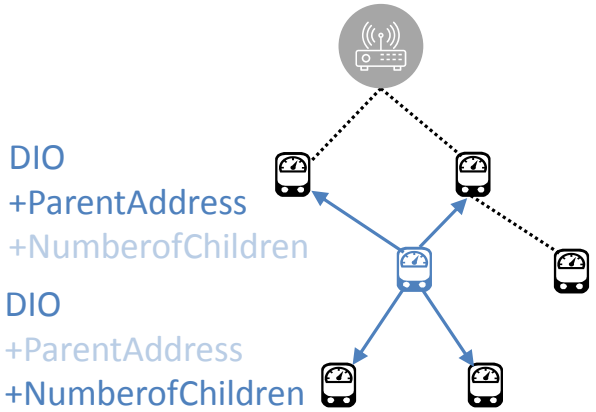
- CNC:
  - A 3, B 1
  - C 0, D 0
    - E->C or E->D?
- E choose D

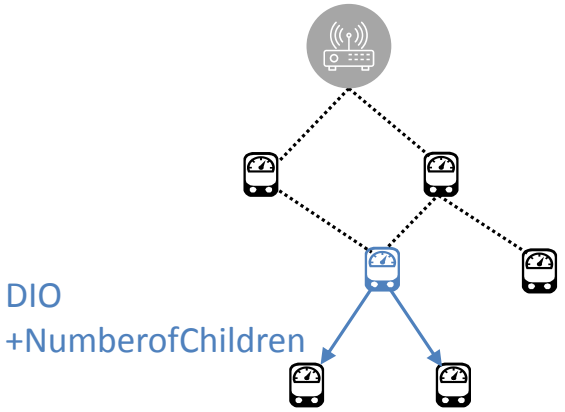




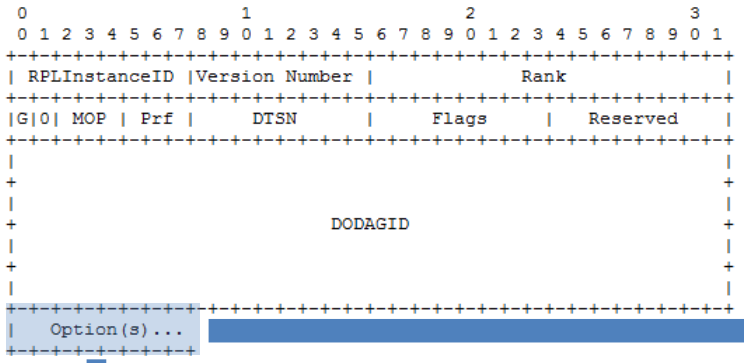
Modification Comparison (1/2)

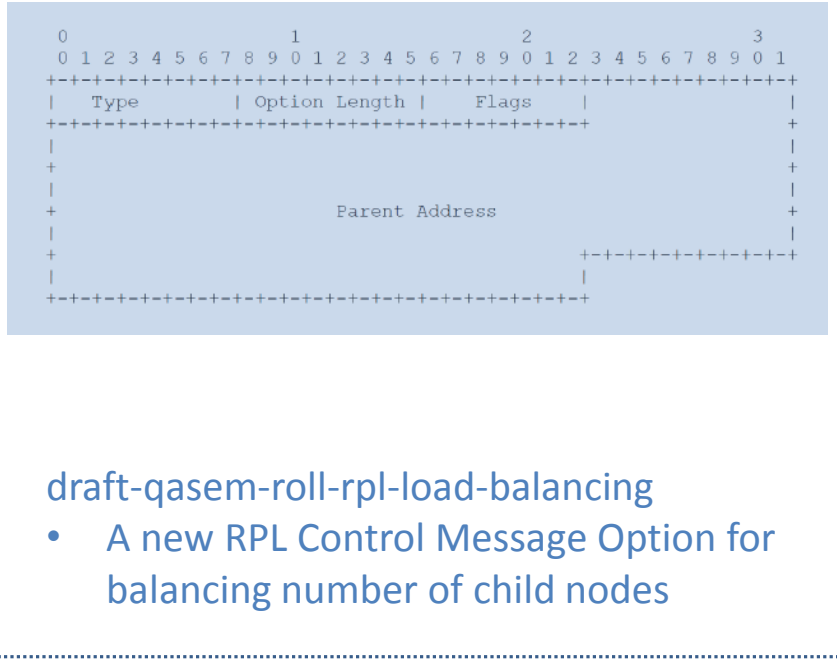
	draft-qasem-roll-rpl-load-balancing	draft-hou-roll-rpl-parent-selection
Main Idea	Put the parent address in the DIO Option field	Compute the number of children from Neighbor Table
	Send DIO to compute number of children	Create a new Metric
	Create a new Objective Function	Combine the Number_of_Children with other metrics
Advantage	For both storing & non-storing modes	Cost efficient, metric/constraint combination
Disadvantage	Extra ParentTable; Extra traffic load from ParentAddress	Not accurate in non-storing mode
Similarity	Both drafts are trying to balance number of children, but not real traffic load	





Modification Comparison (2/2)

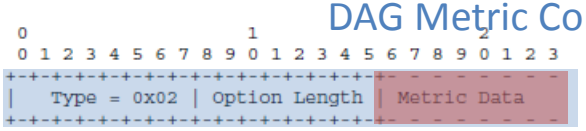




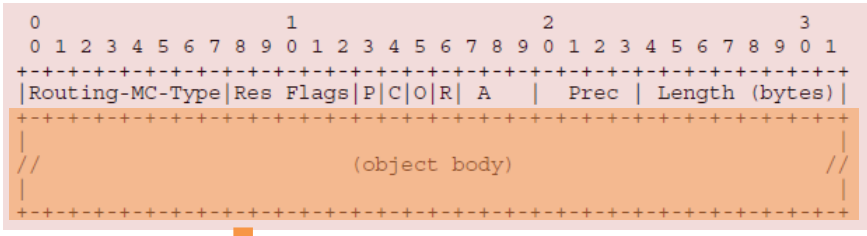
draft-qasem-roll-rpl-load-balancing

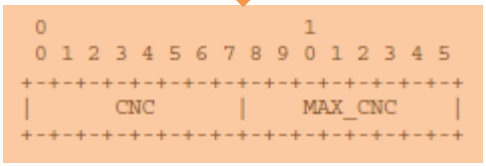
- A new RPL Control Message Option for balancing number of child nodes

DAG Metric Container



Metric Format





draft-hou-roll-rpl-parent-selection

- A new metric in the Metric Container
- CNC (Child Node Count) = Number_of_Children

Next Step

- Comments and Questions

Thank you!

Root initiated routing state in RPL

draft-ietf-dao-projection

Pascal Thubert

IETF 99

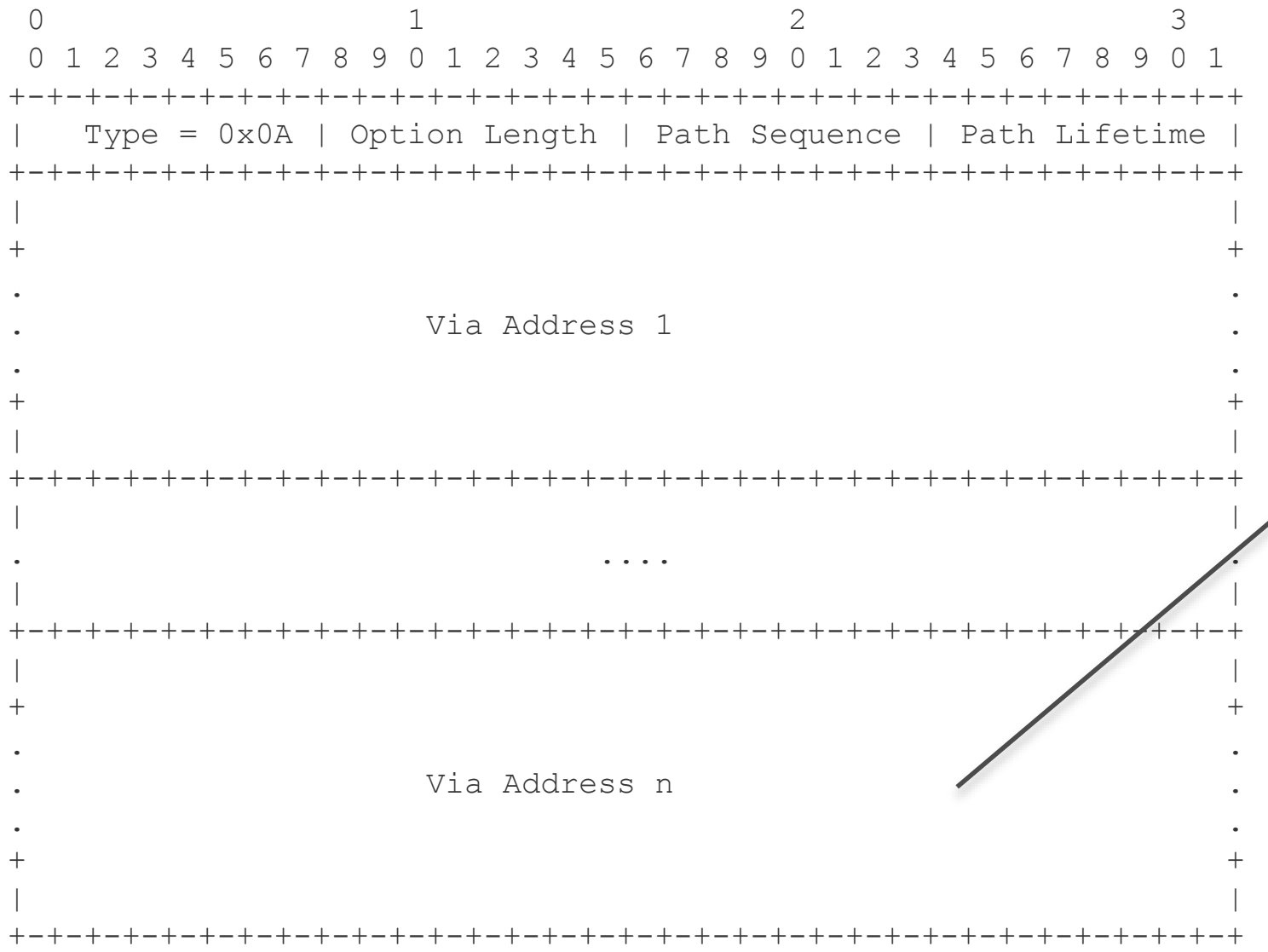
Prague, July 2017

Changes Highlights

- Now a work group document, draft ietf .. 01
- Added Source Routed projected route (Rahul)
  - Multiple addresses in one Via Info option
  - Allows any mix of projected route and RPL MOP
- Clarified NP-DAO operation
- Restructured
  - A lot of example text moved to appendix
- Added Encapsulation discussion
  - Operation by root, egress, ingress and intermediate nodes

3.1. Via Information Option

The Via Information option MAY be present in DAO messages, and its format is as follows:



Now multiple hops

Differentiating Storing vs. non-storing

- The non-storing mode P-DAO discussed in section Section 4.1 has a single VIO with one or more Via Addresses in it, the list of Via Addresses indicating the source-routed path to the target to be installed in the router that receives the message, which replies to the root directly with a DAO-ACK message.
- The storing mode P-DAO discussed in section Section 4.2 has at least two Via Information options with one Via Address each, for the ingress and the egress of the path, and more if there are intermediate routers. In normal operations, the P-DAO is propagated along the chain of Via Routers from the egress router of the path till the ingress one, which confirms the installation to the root with a DAO-ACK message.

Discussions

How is the topology known to the root?

How are the node capabilities known to the root?

Complexity of mixed modes

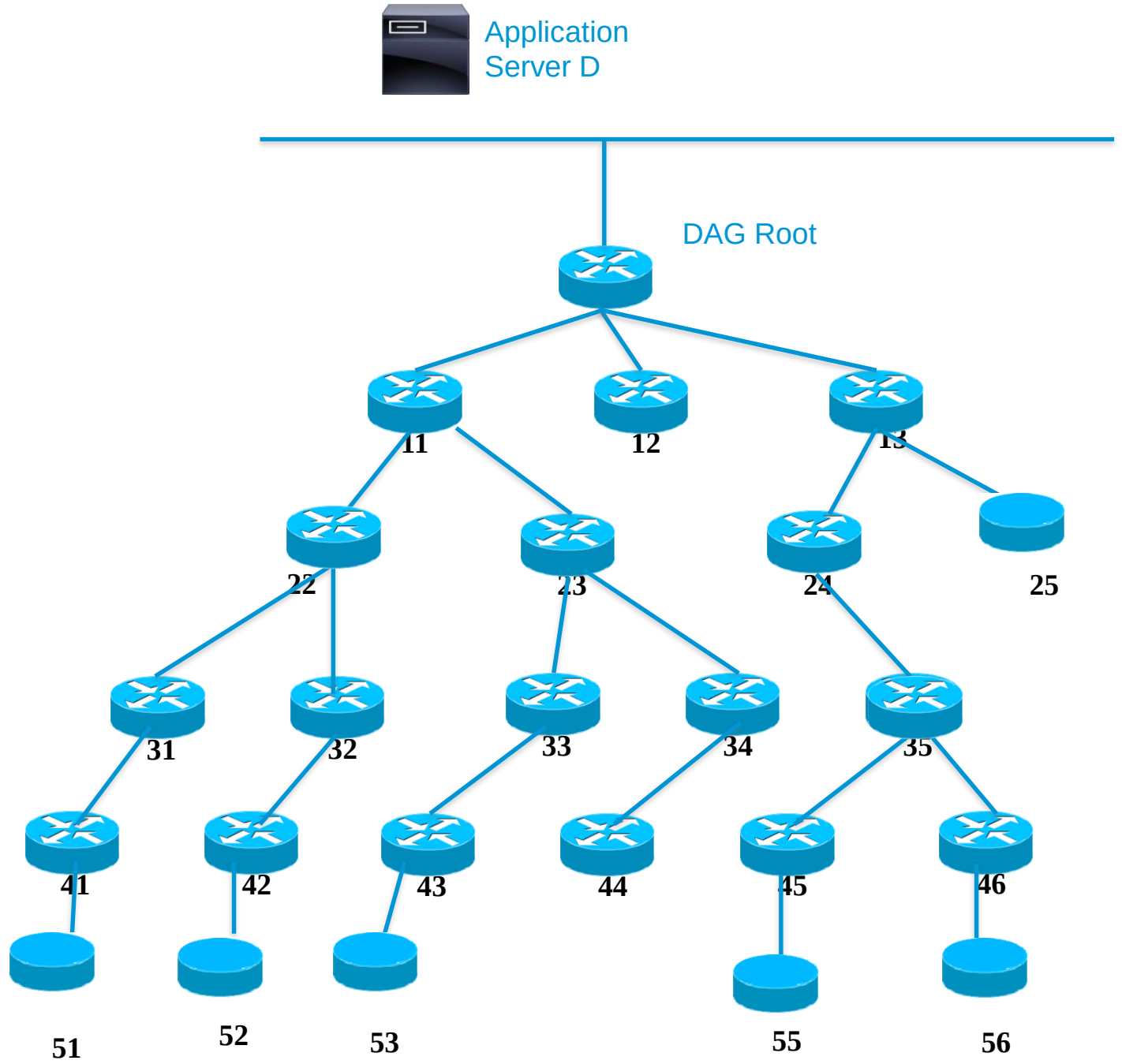
Compression of the Via Info option (so far full addresses)

Loop avoidance

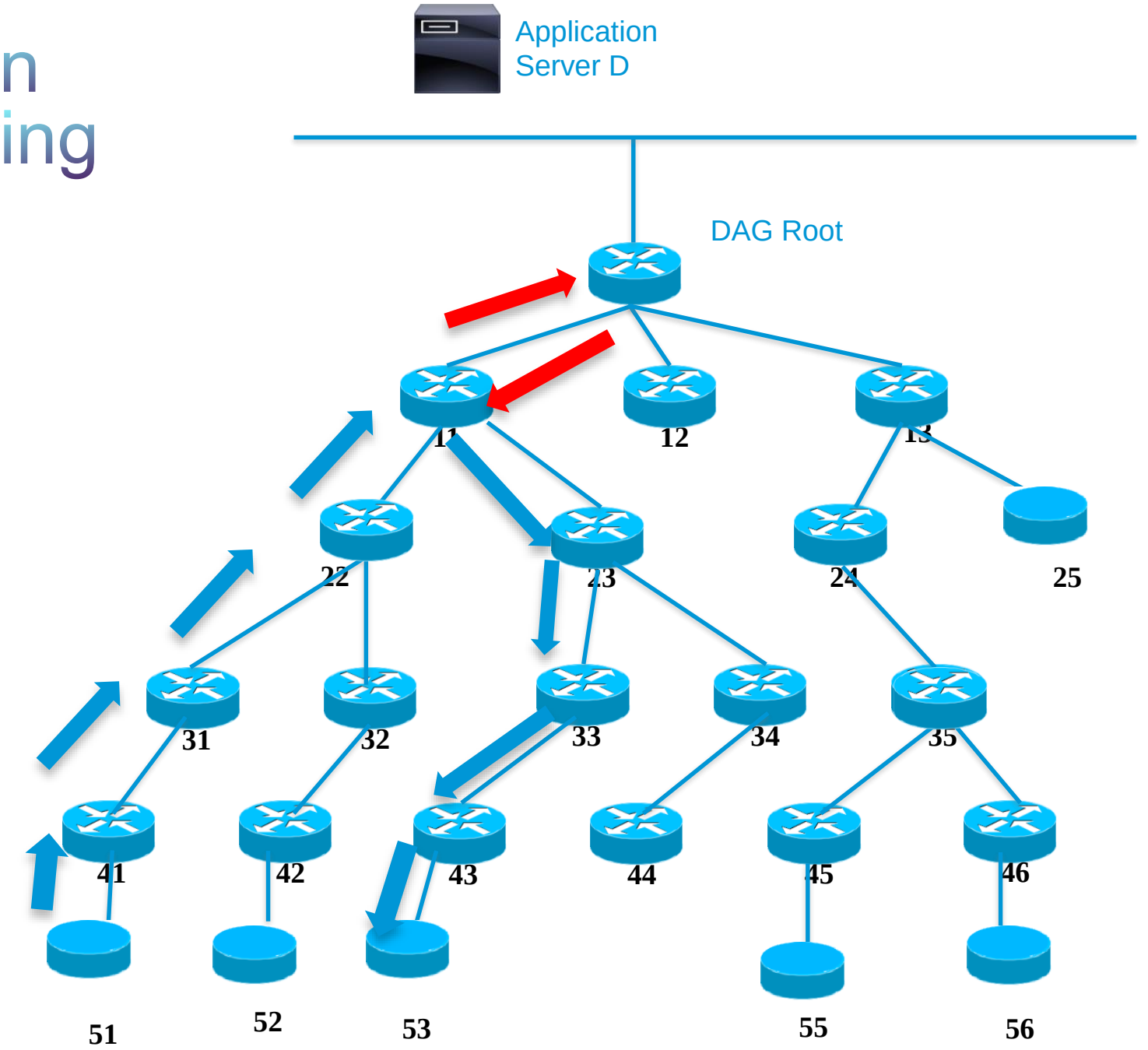
- in particular for loose and not end to end route
- Recommend Setting the ‘O’ bit

<RFC6550>: “Down 'O': 1-bit flag indicating whether the packet is expected to progress Up or Down. A router sets the 'O' flag when the packet is expected to progress Down (using DAO routes), and clears it when forwarding toward the DODAG root to a node with a lower Rank). A host or RPL leaf node MUST set the 'O' flag to 0.”

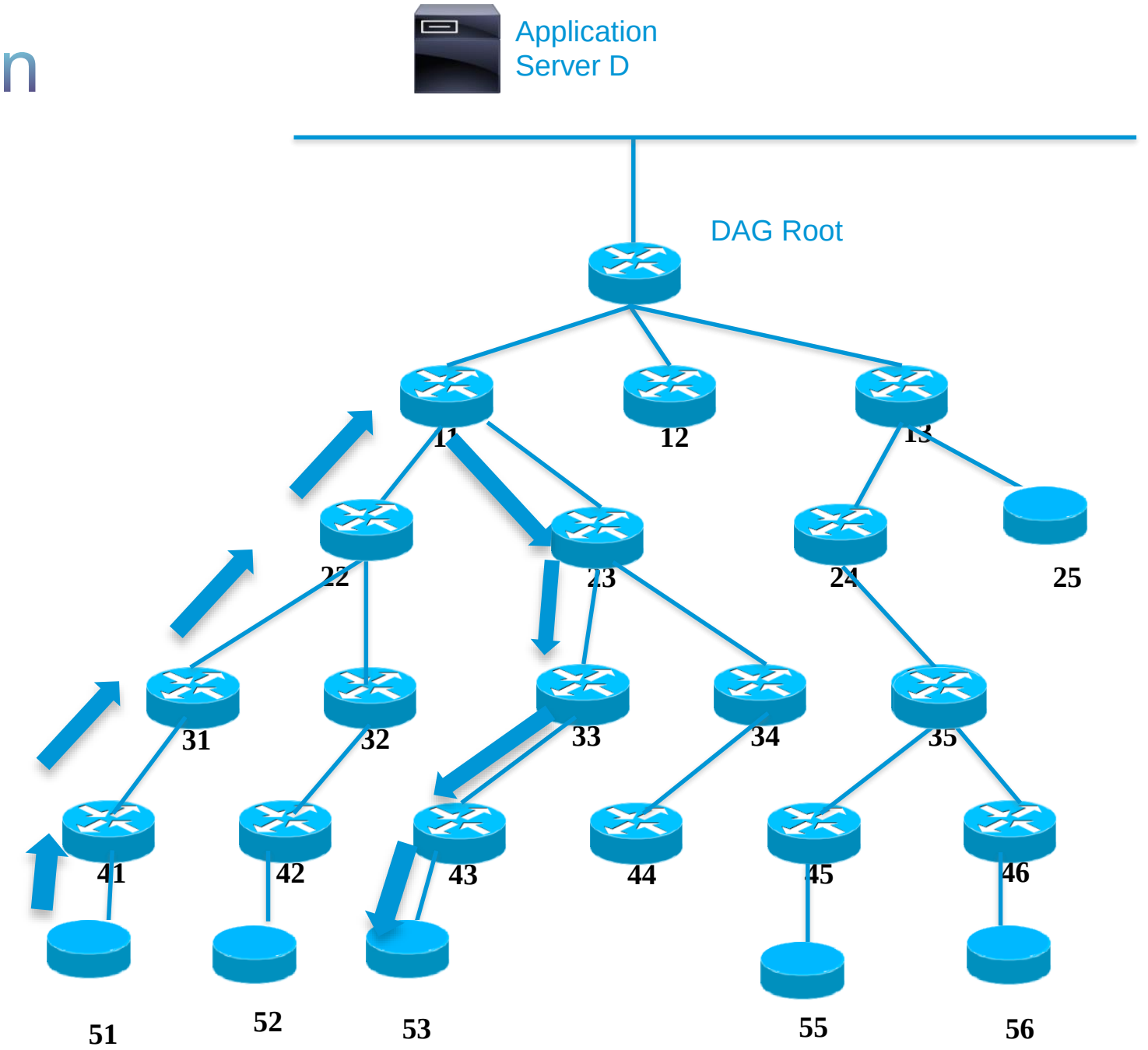
New: non-storing mode transversal route


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

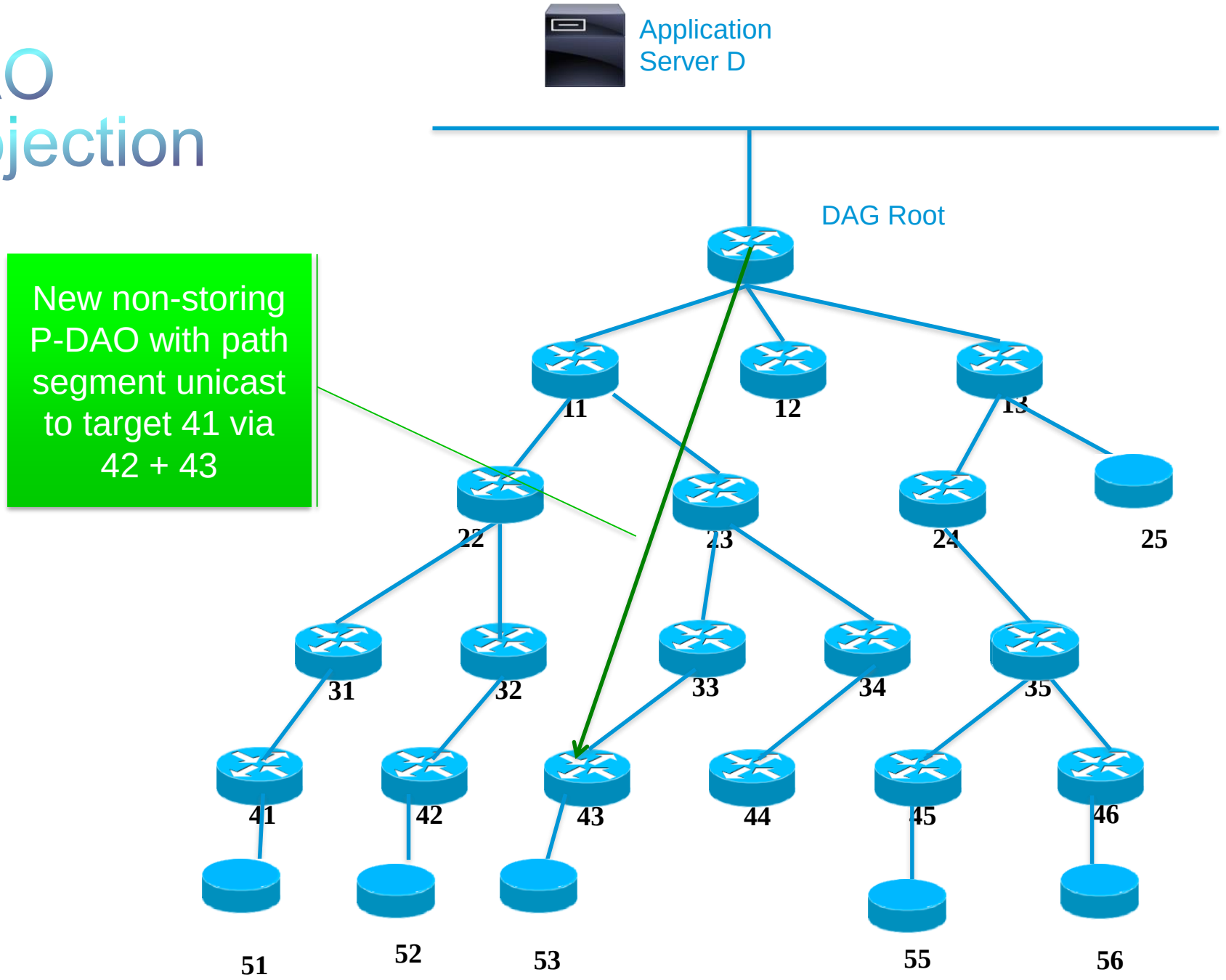
Stretch in non-storing mode



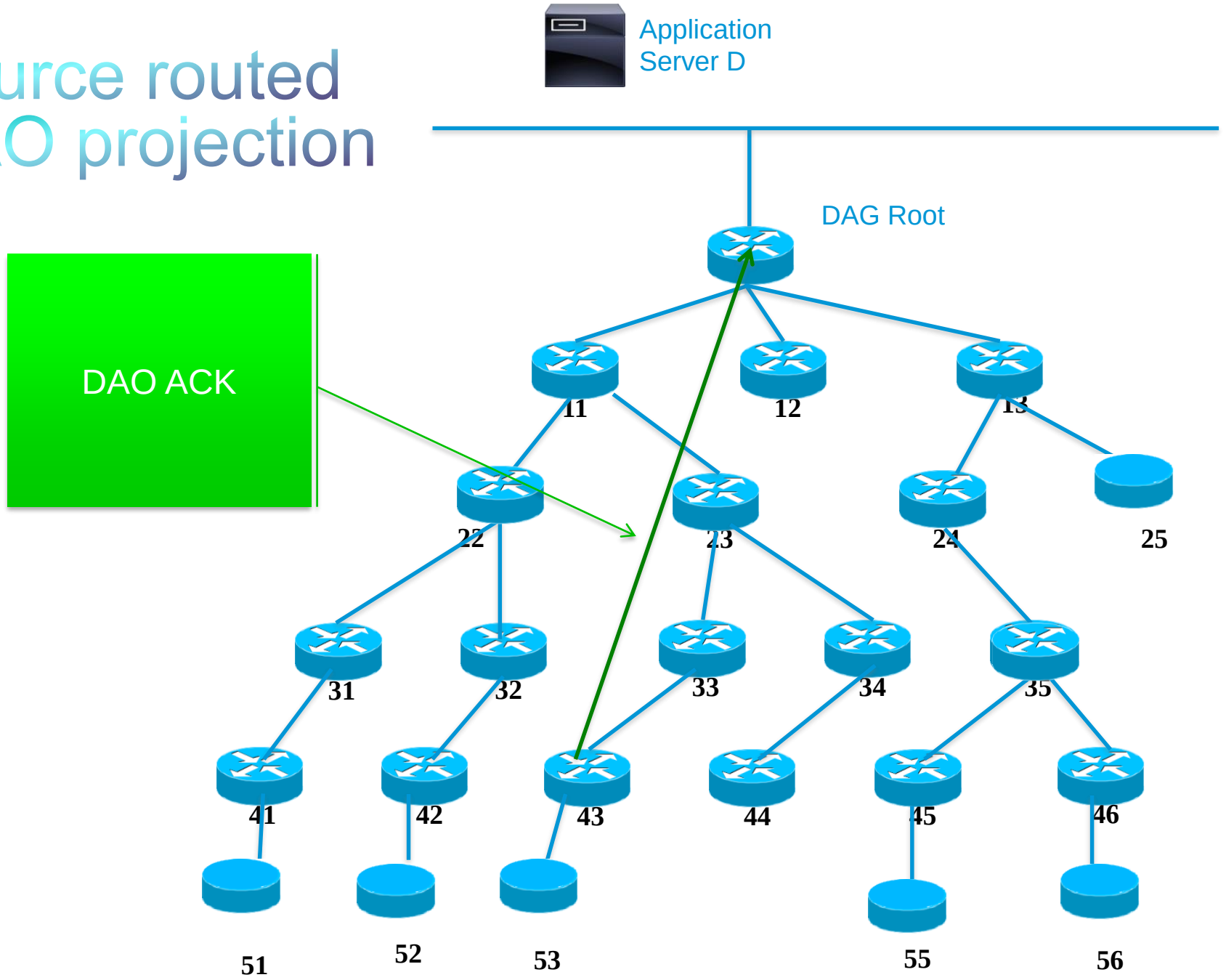
Stretch in storing mode



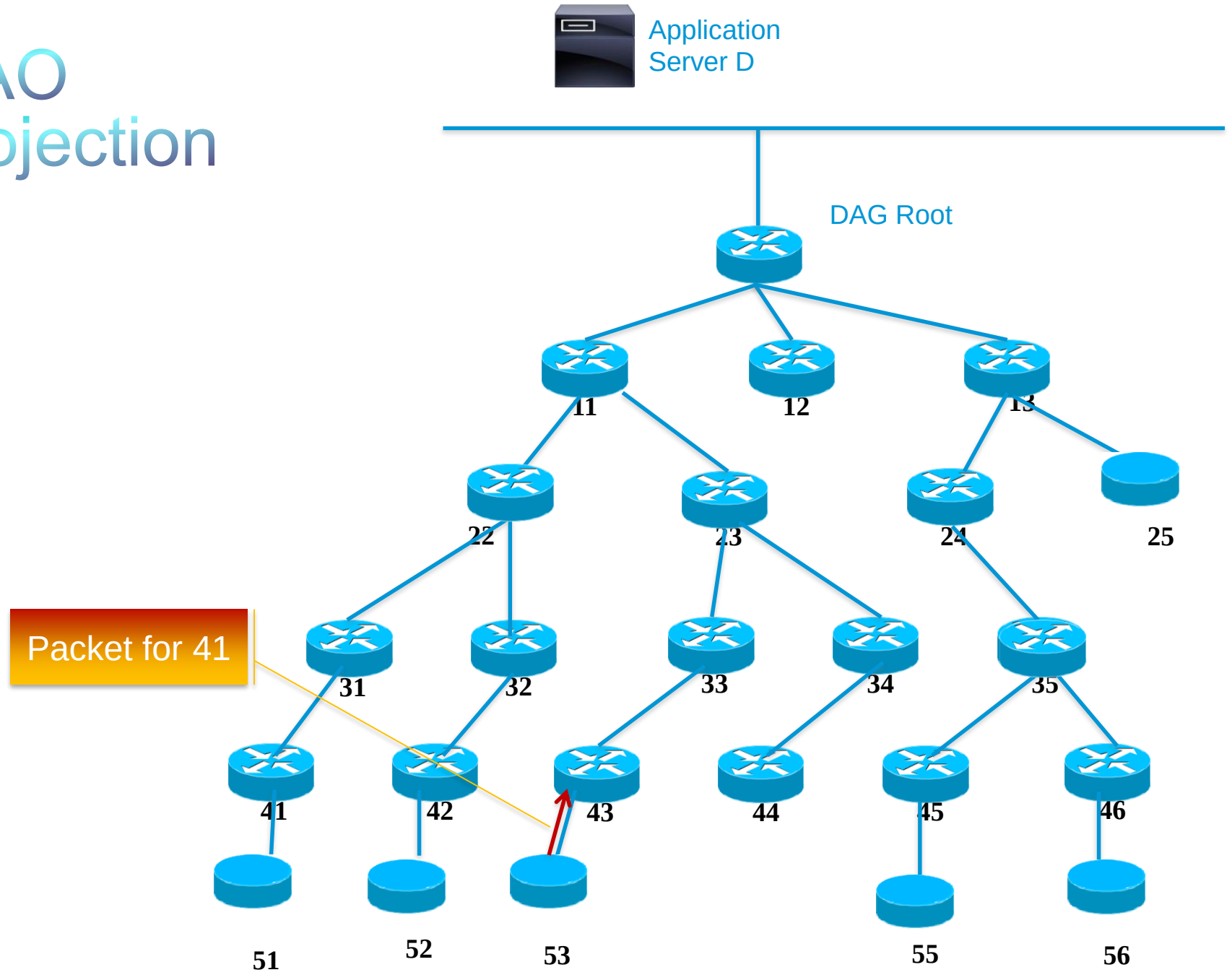
DAO projection



Source routed DAO projection



DAO projection

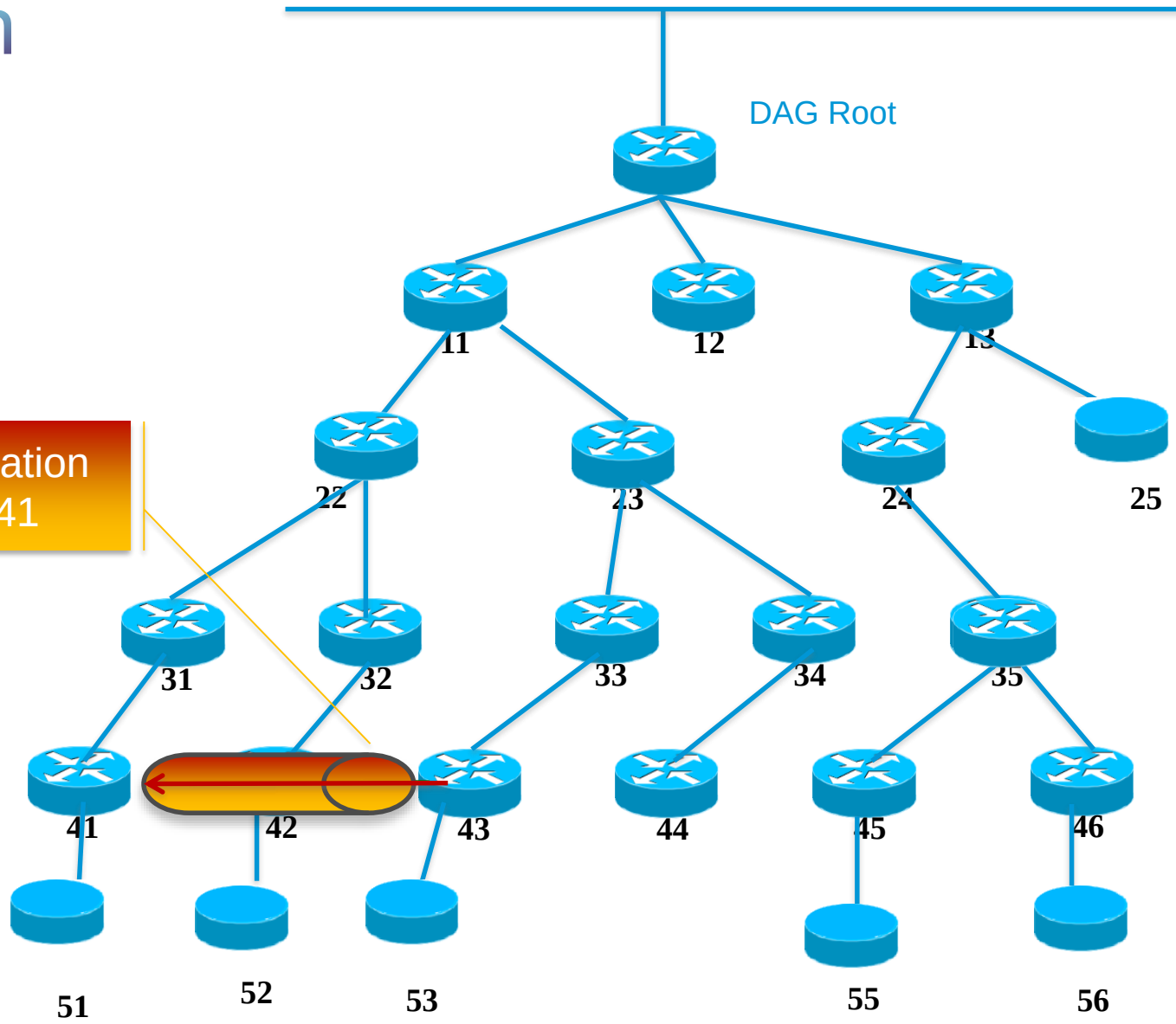


DAO projection

Application Server D

DAG Root

IP in IP encapsulation with SRH 42, 41

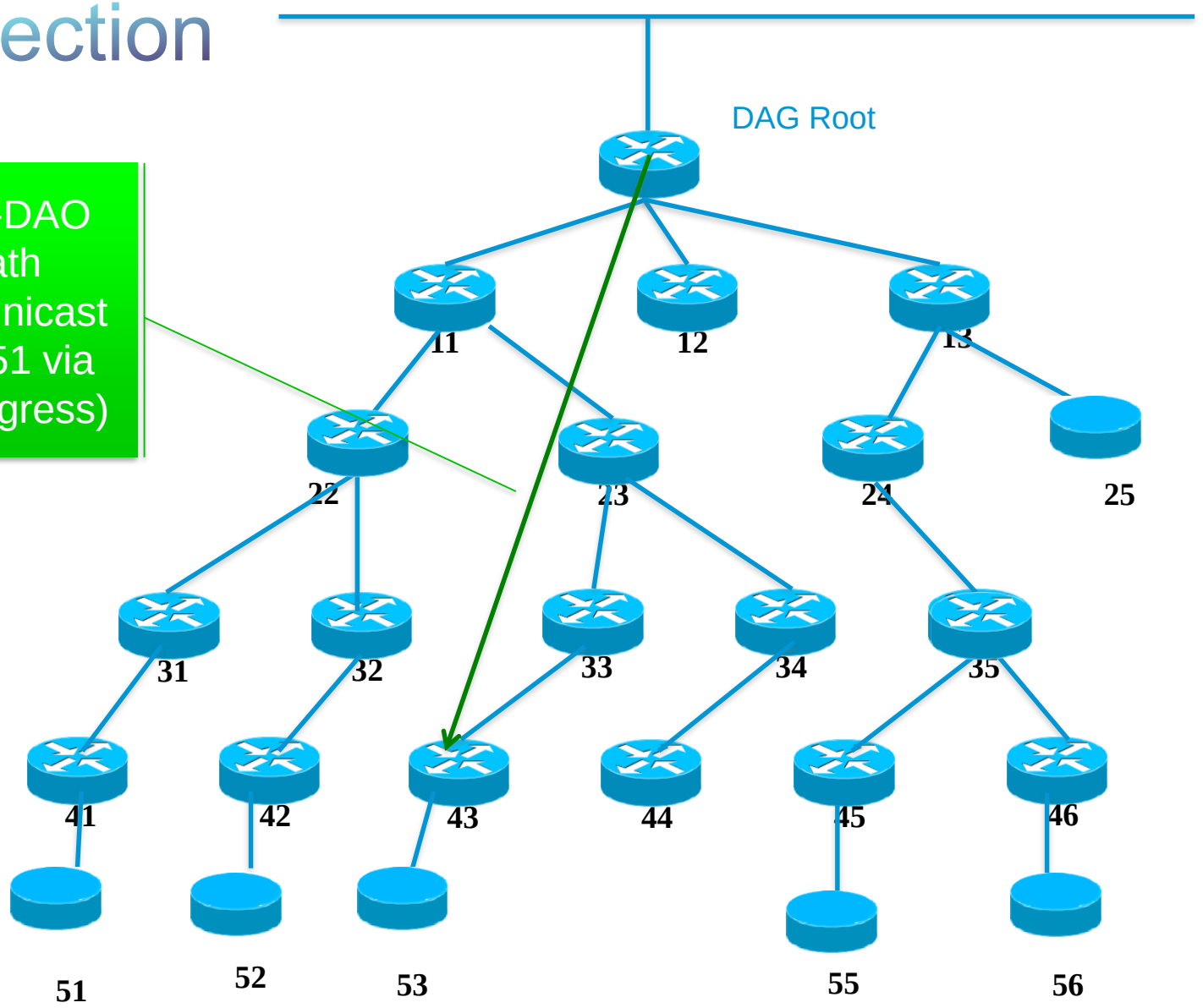


Not done yet
DAO projection

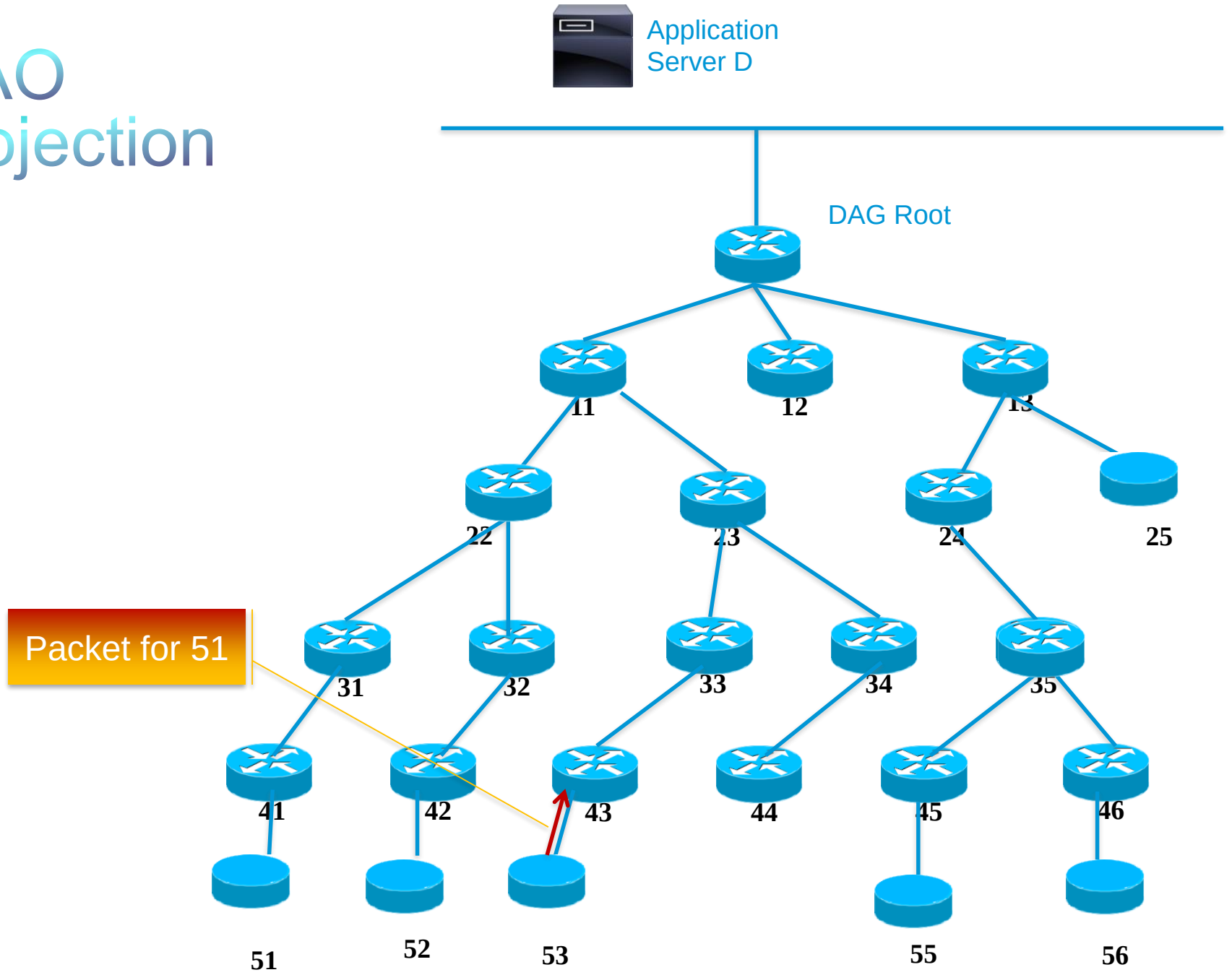
Application
Server D

DAG Root

storing P-DAO with path segment unicast to target 51 via 43==41 (egress)



DAO projection

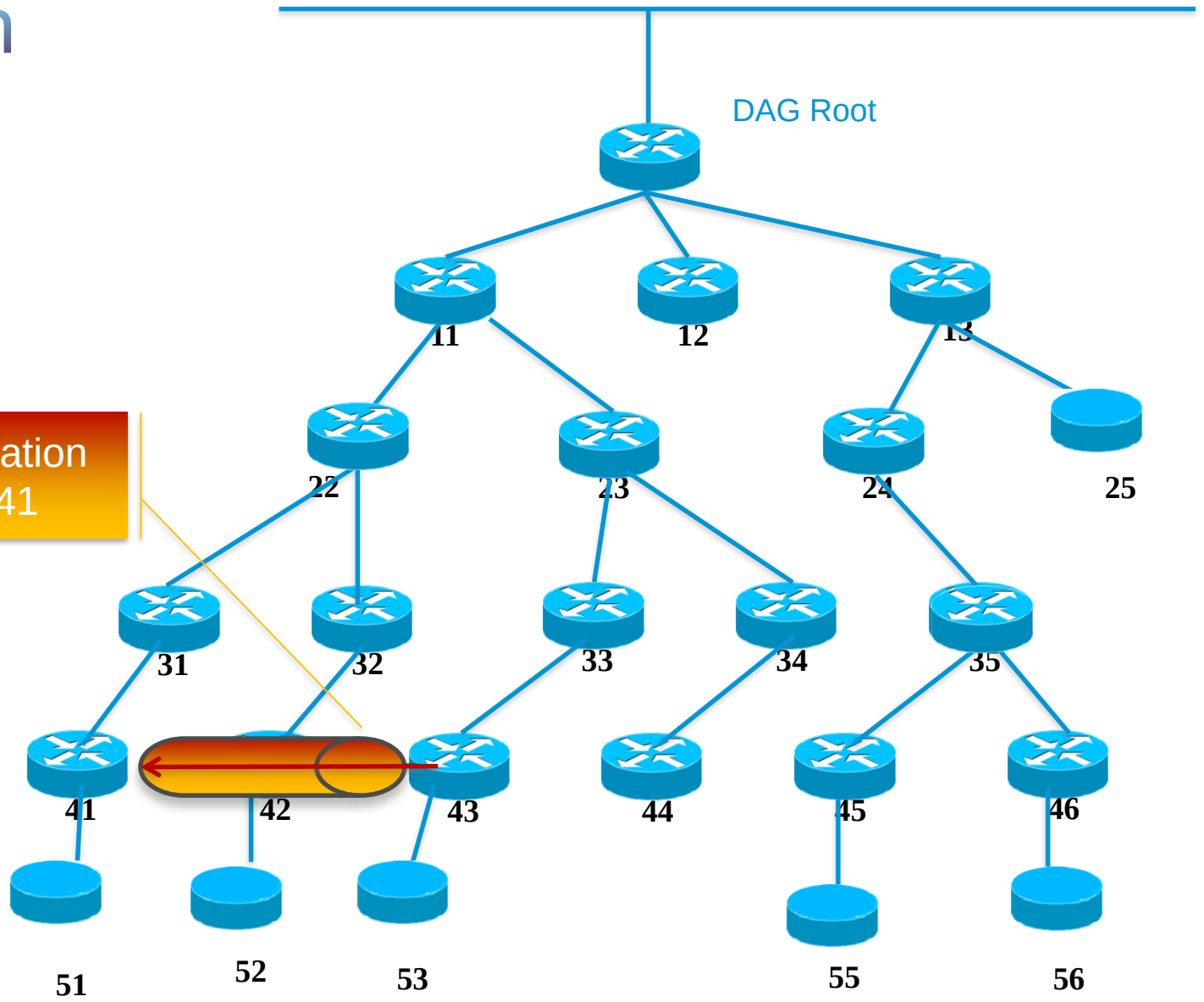


DAO projection

Application Server D

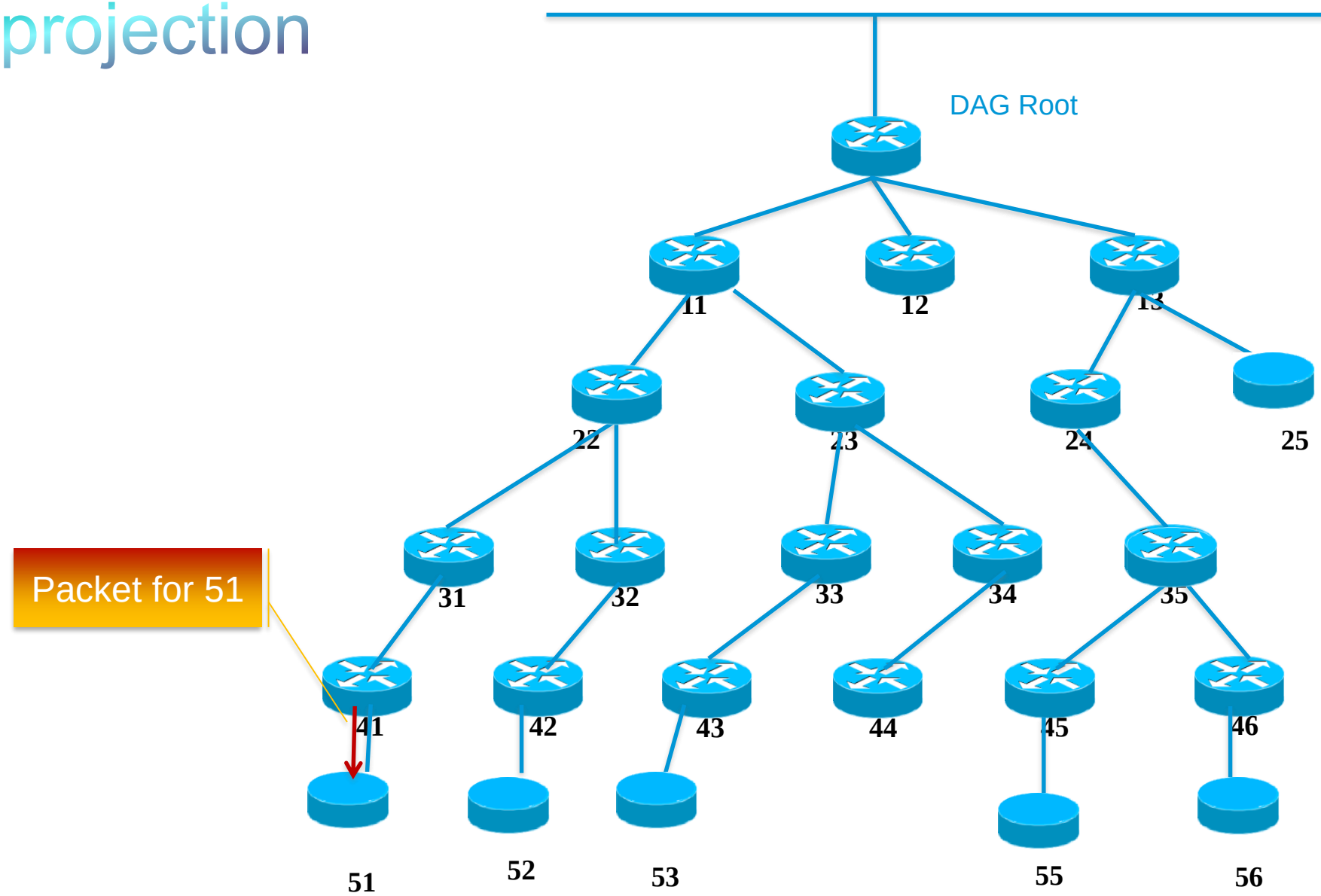
DAG Root

IP in IP encapsulation with SRH 42, 41

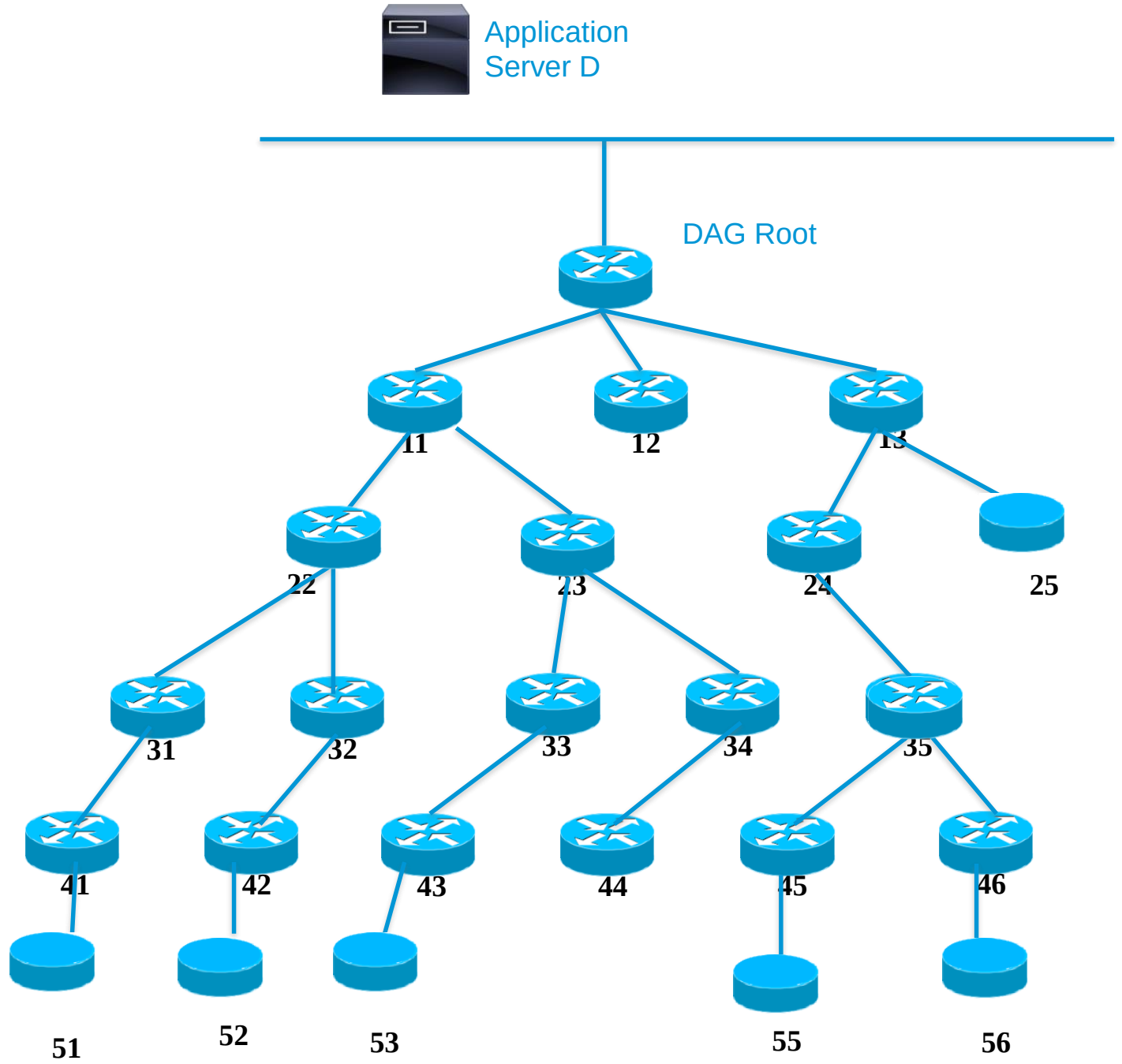


DAO projection

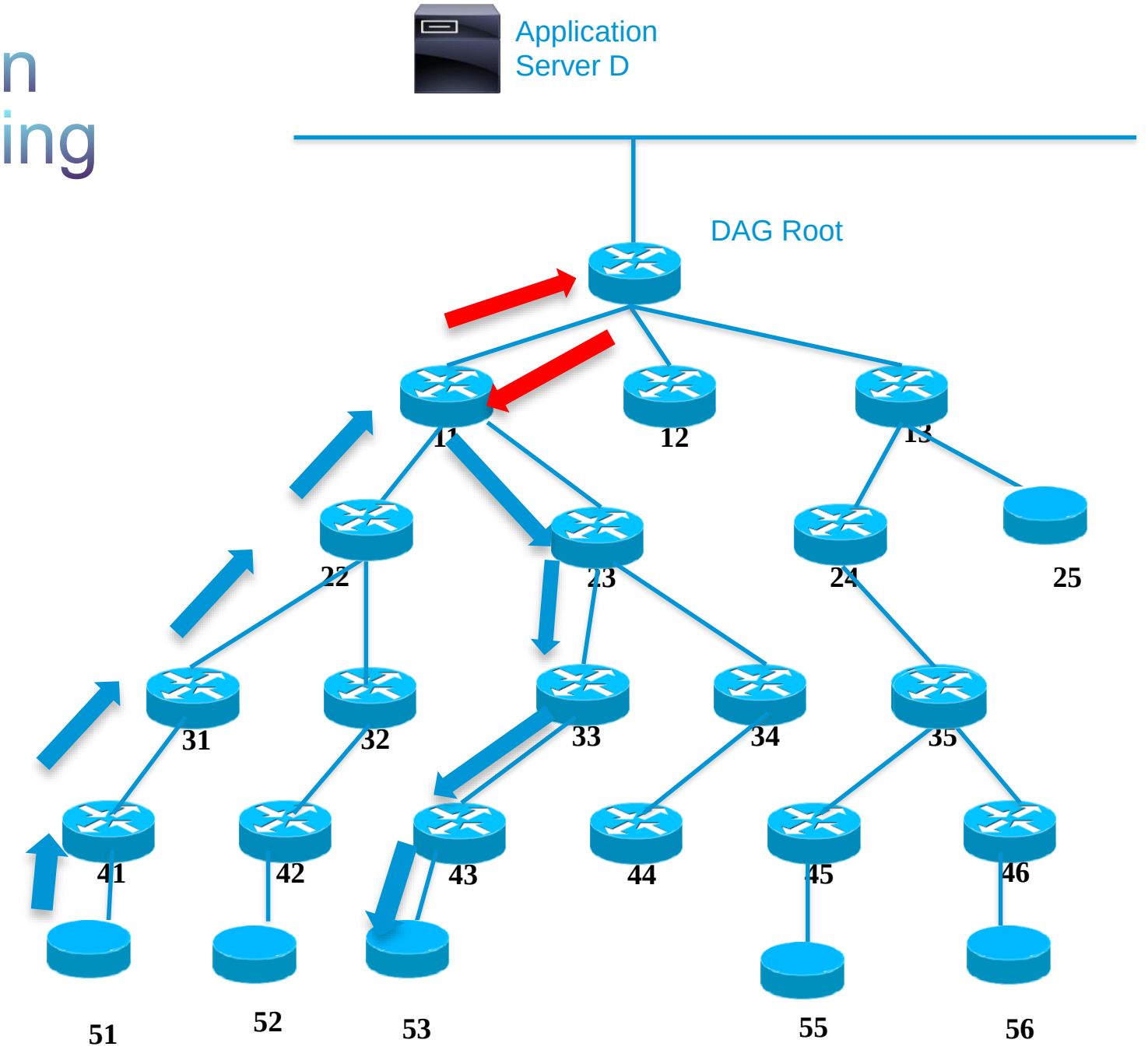
Application Server D



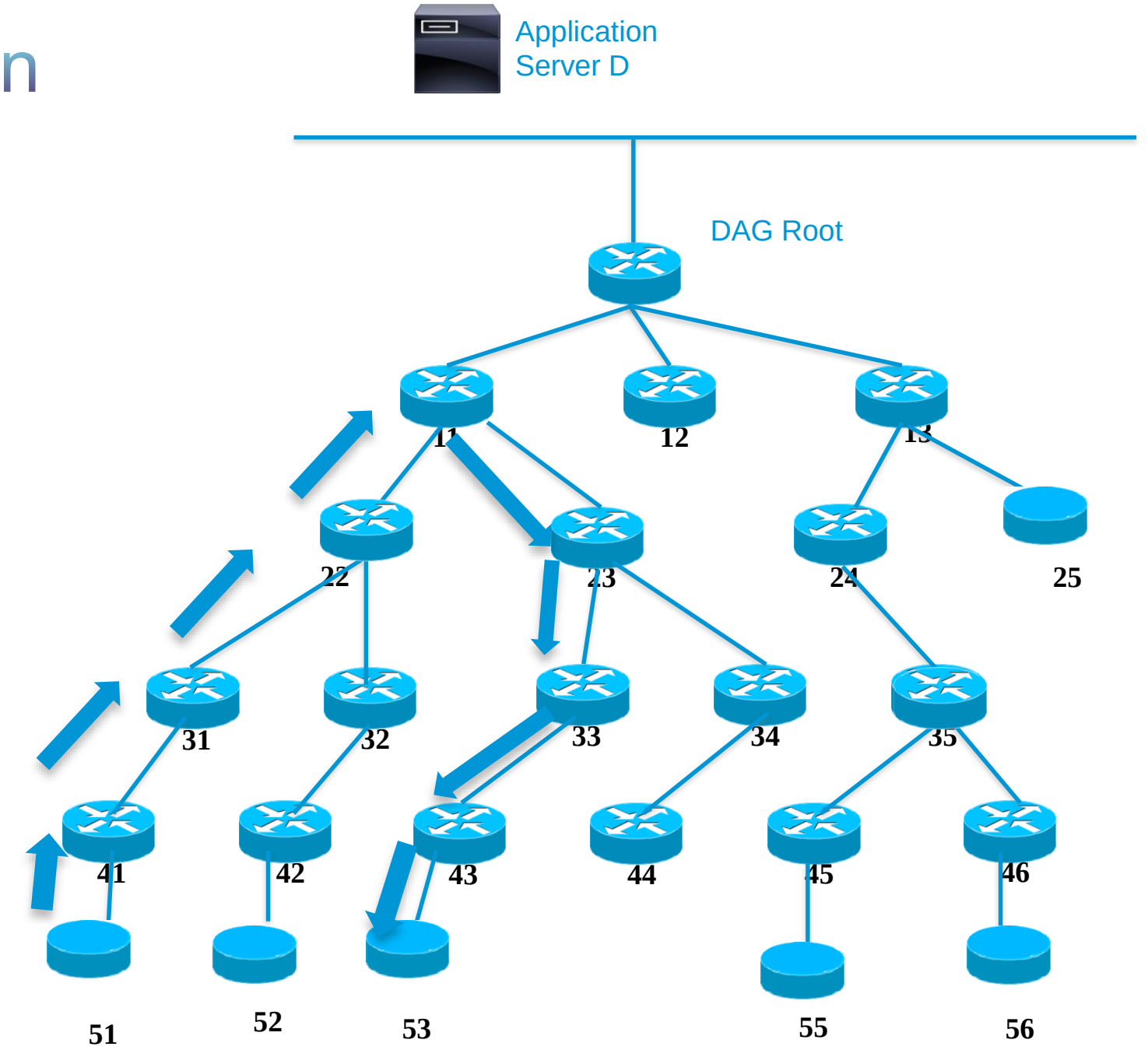
Storing mode transversal route


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



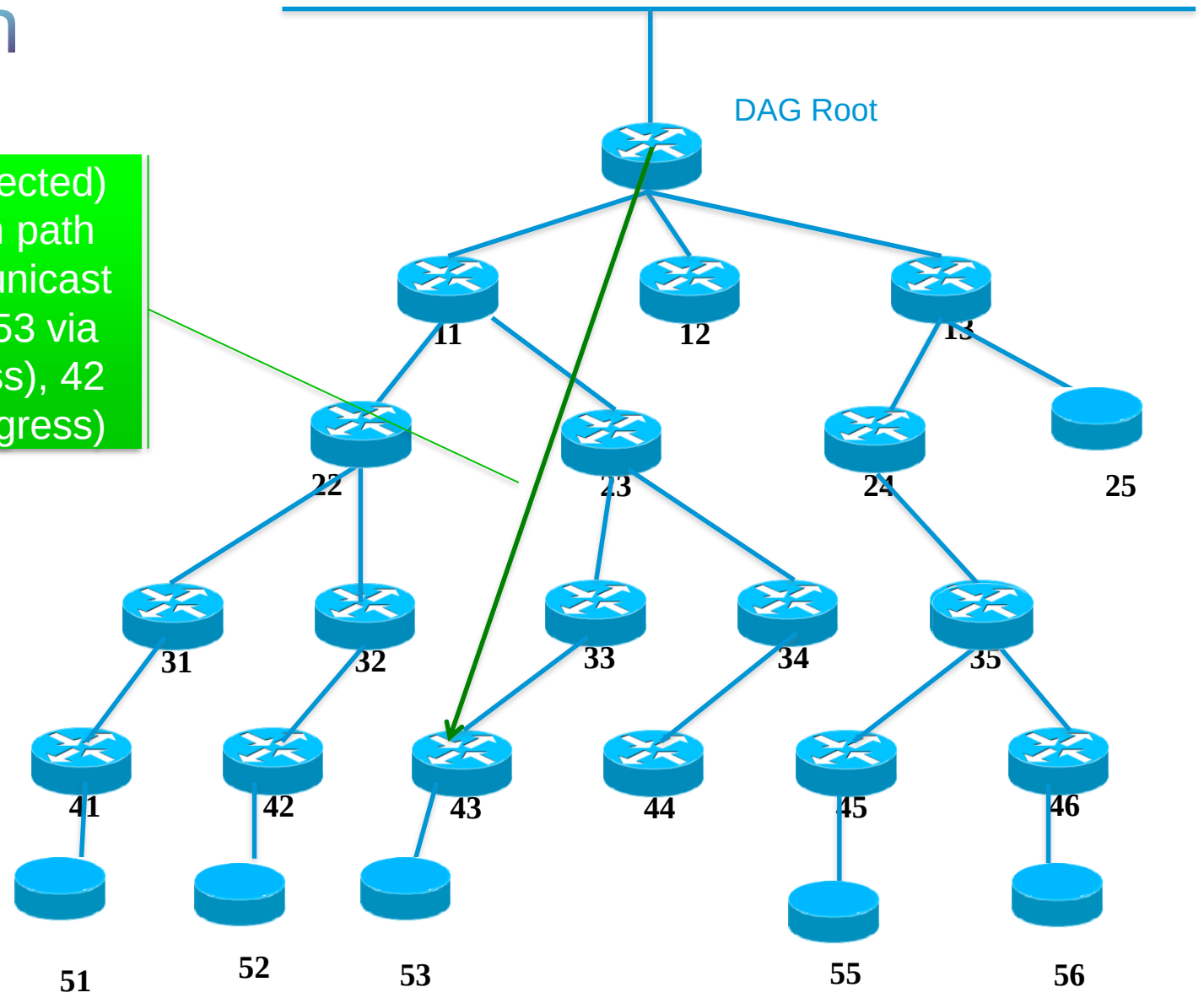
Stretch in storing mode



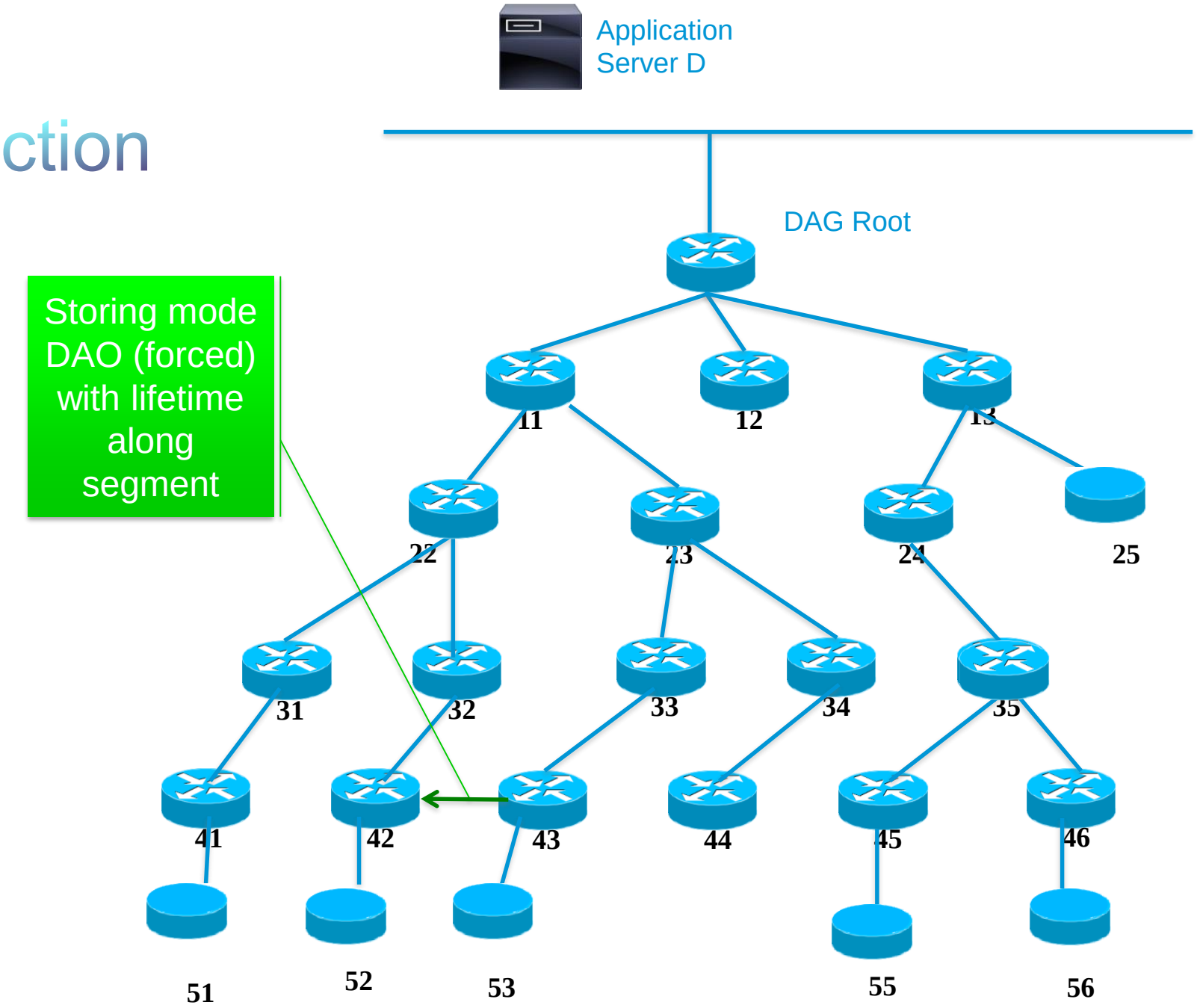
DAO projection

Application Server D

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



DAO projection

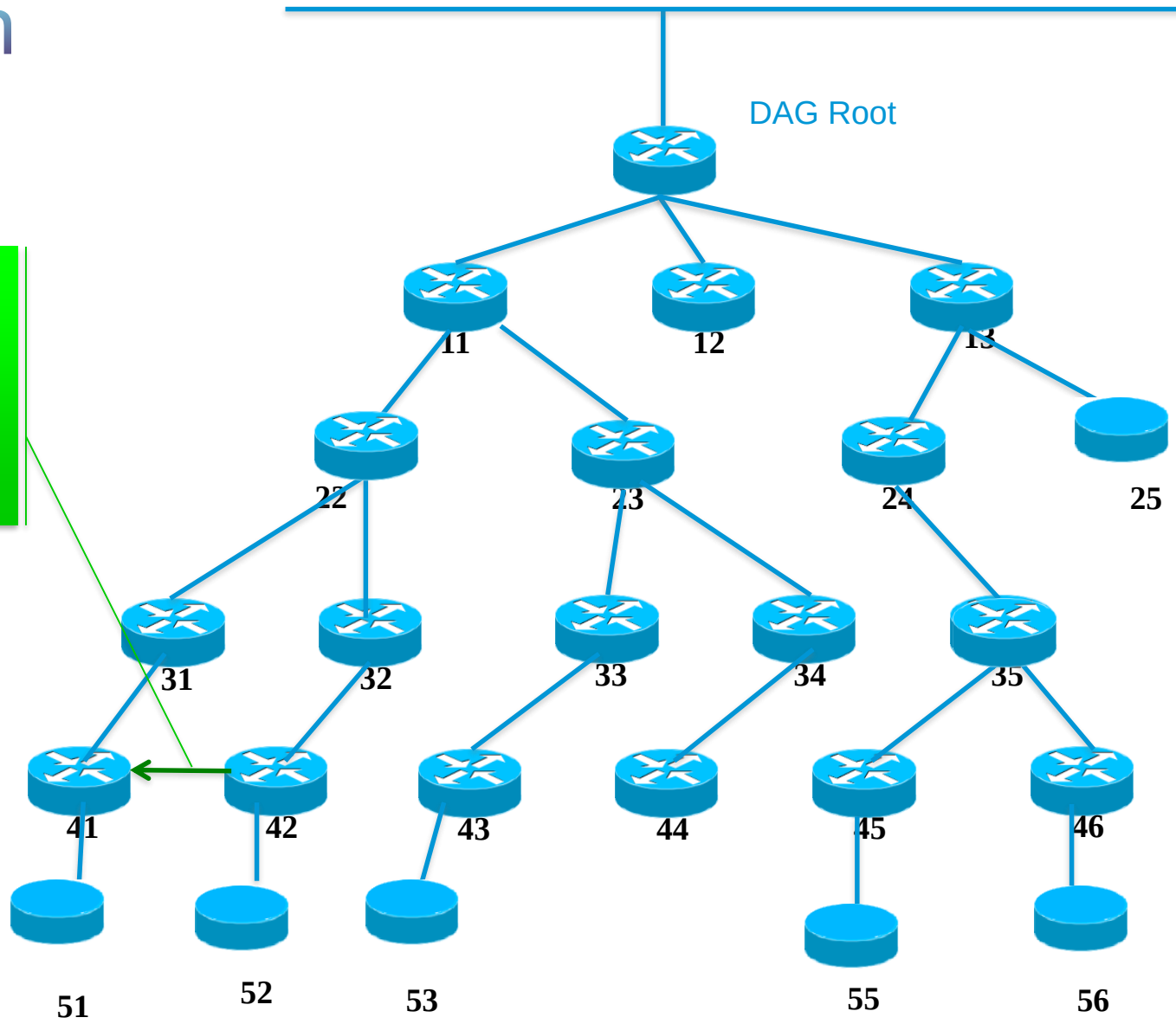


DAO projection

Application Server D

DAG Root

Storing mode DAO (forced) with lifetime along segment

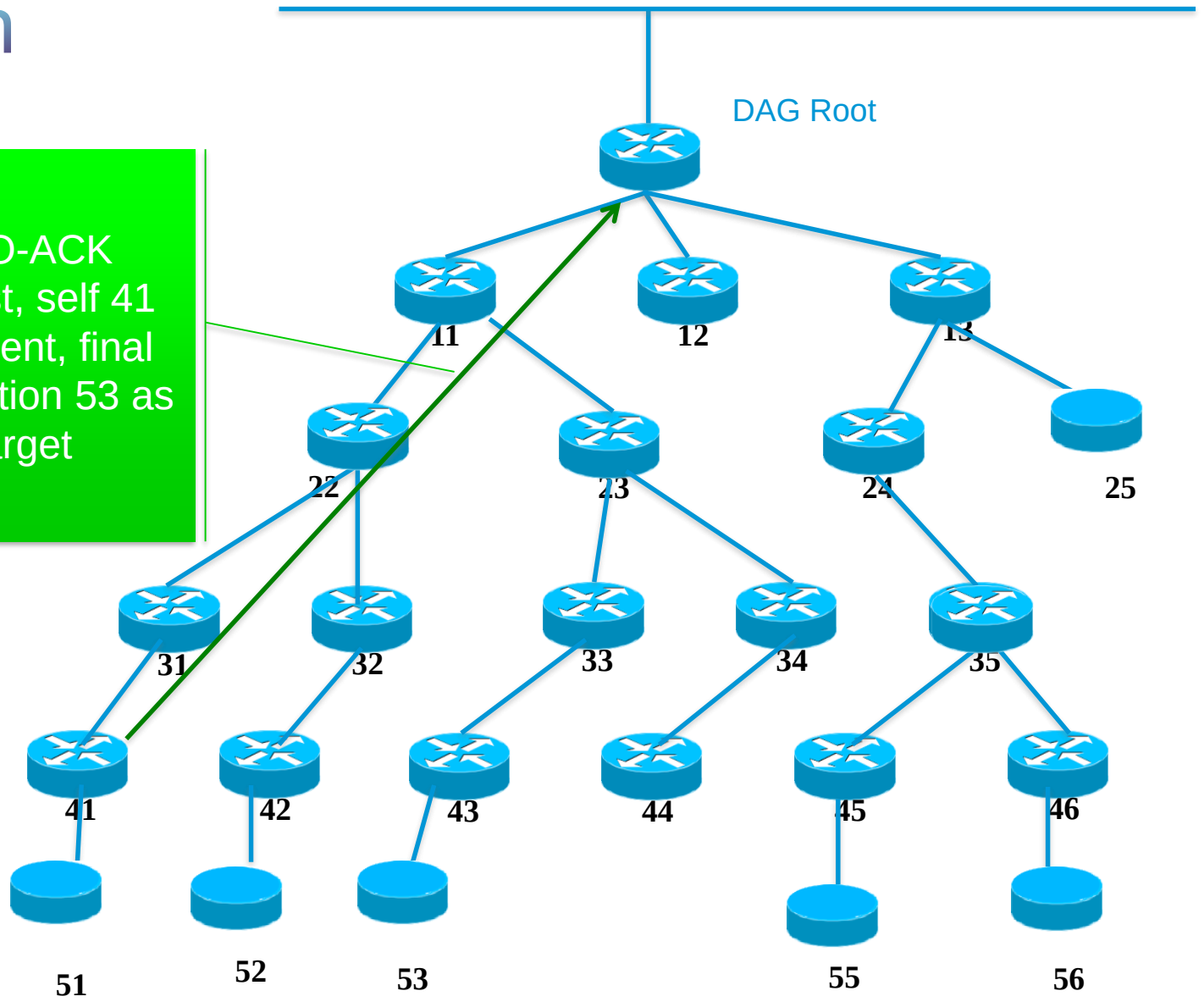


DAO projection

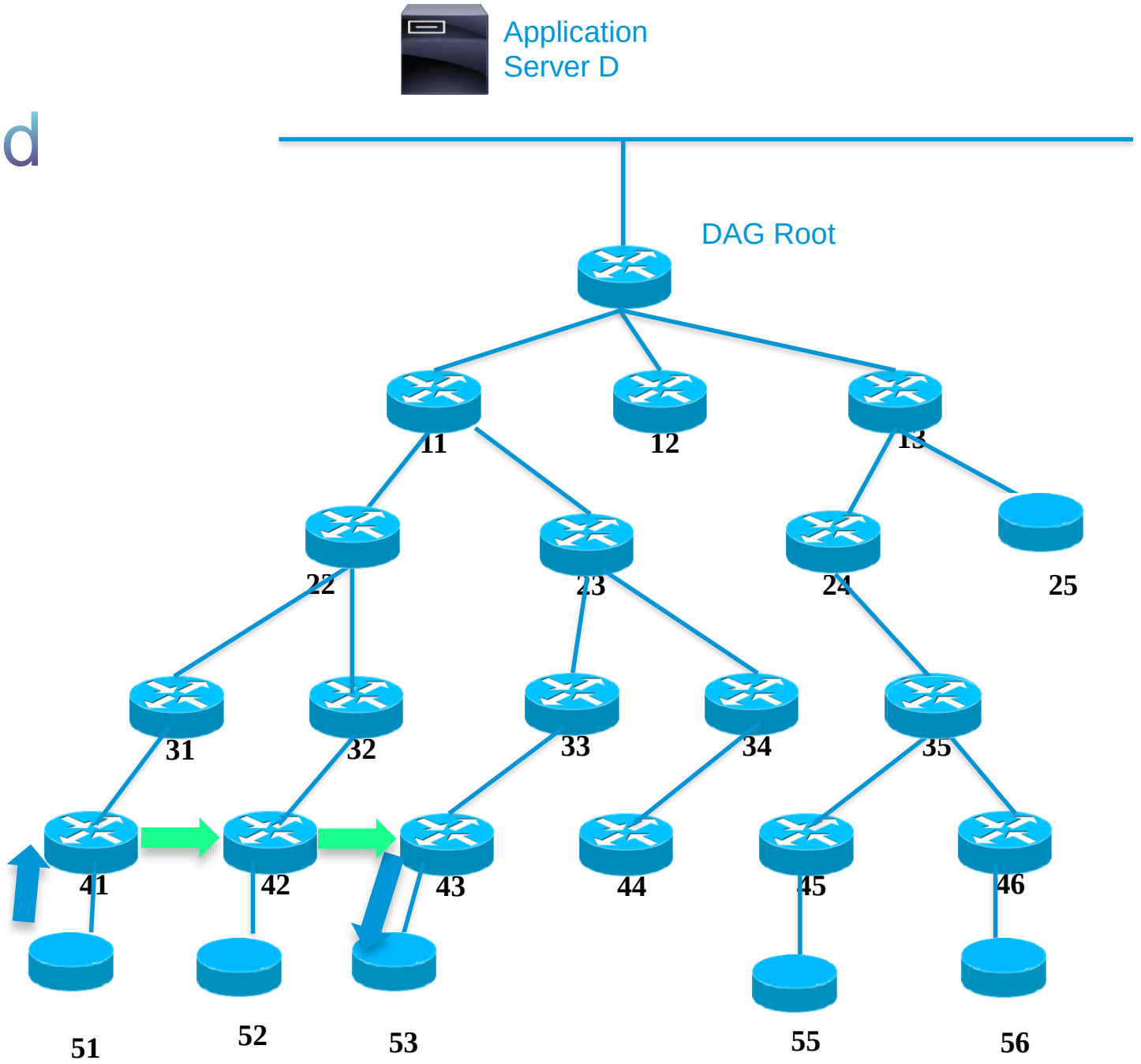
Application Server D

DAG Root

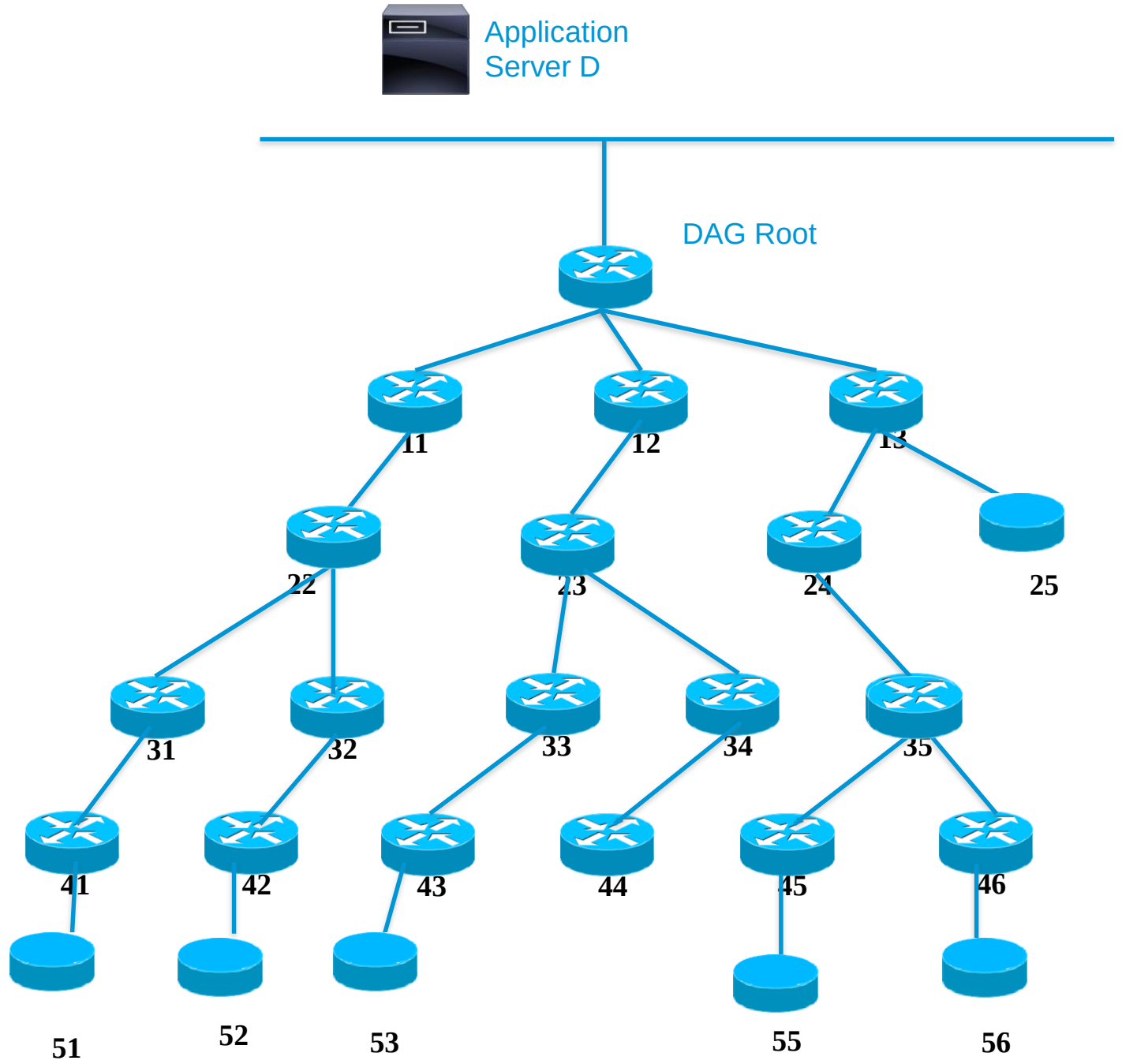
DAO-ACK unicast, self 41 as parent, final destination 53 as target

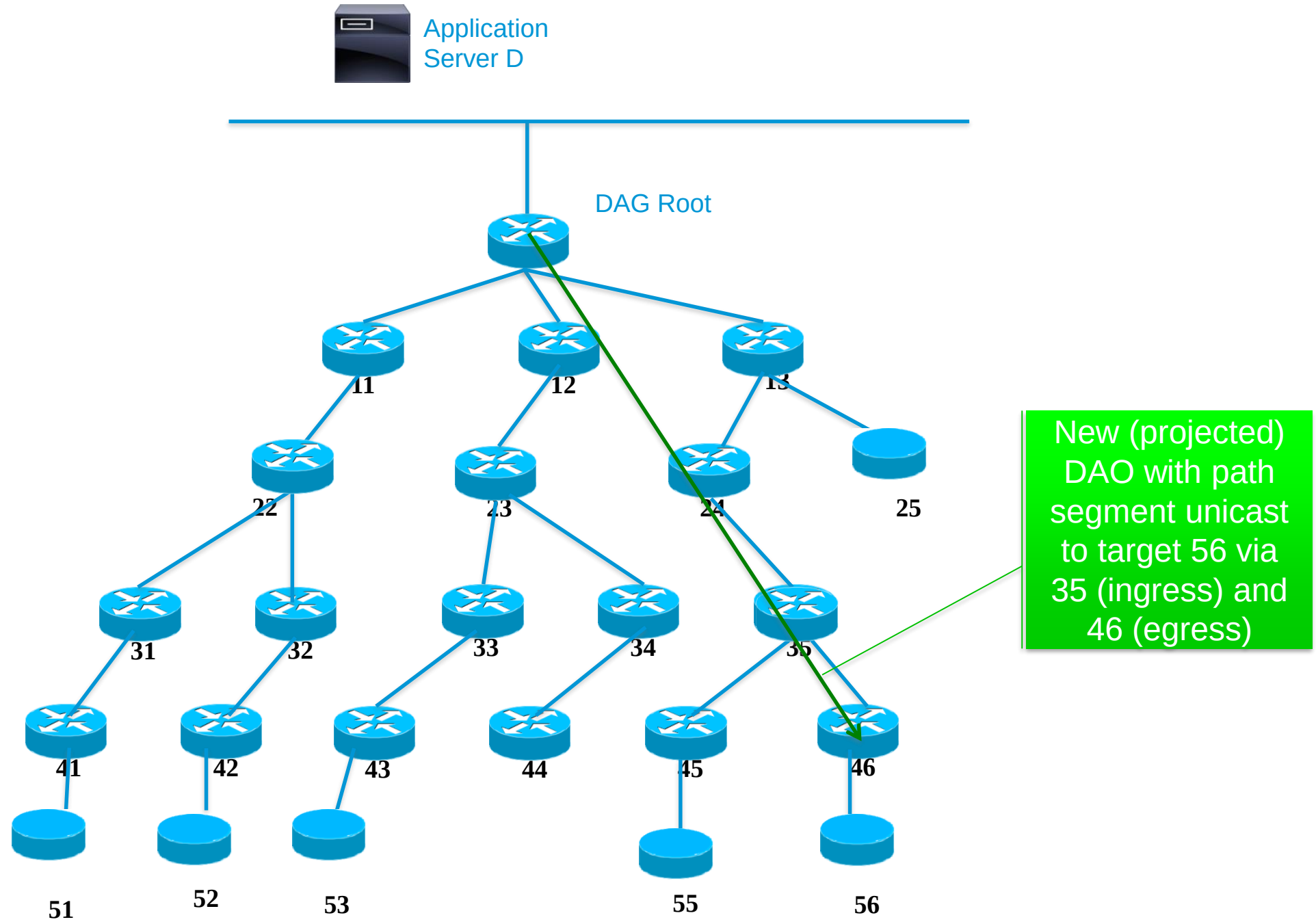


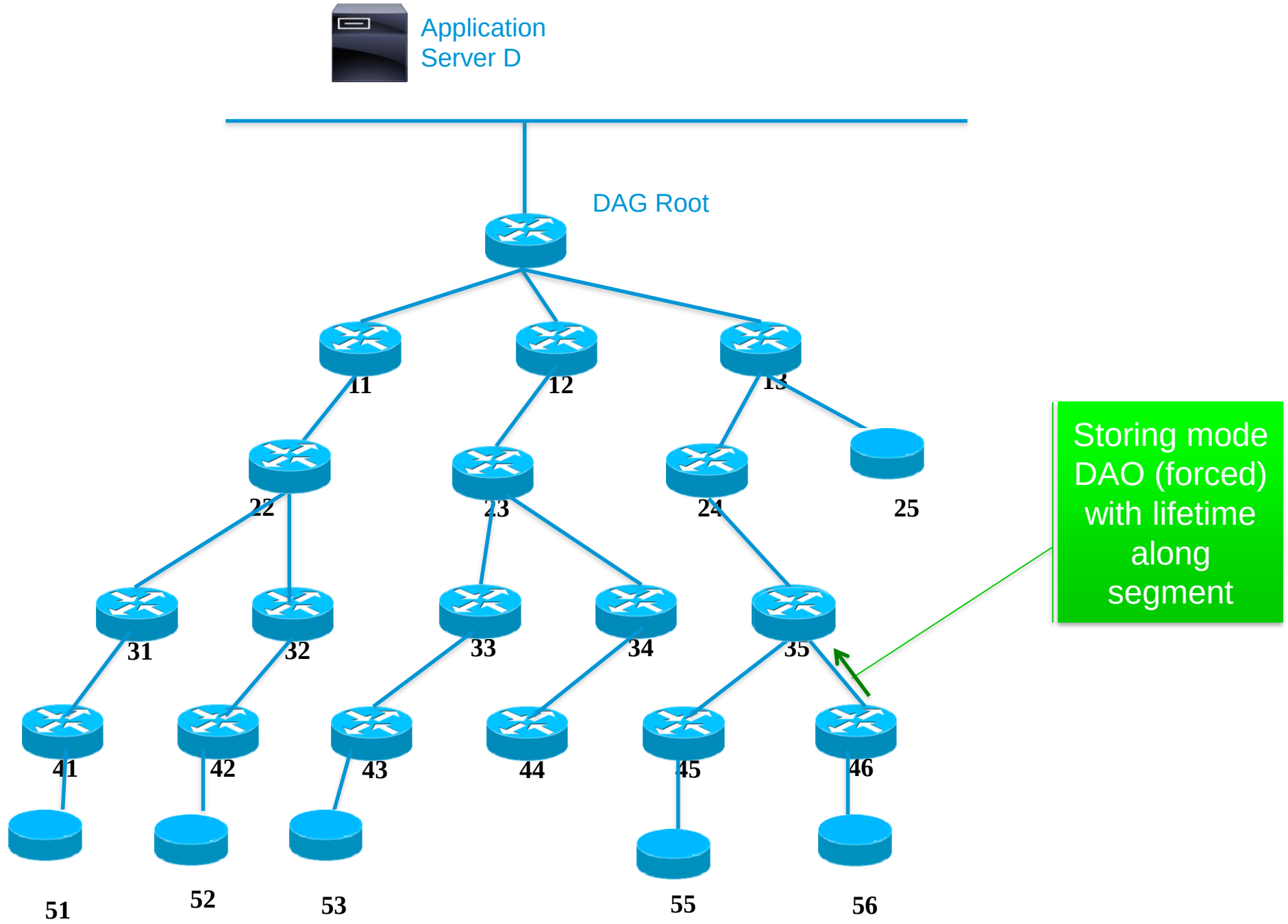
Optimized Path

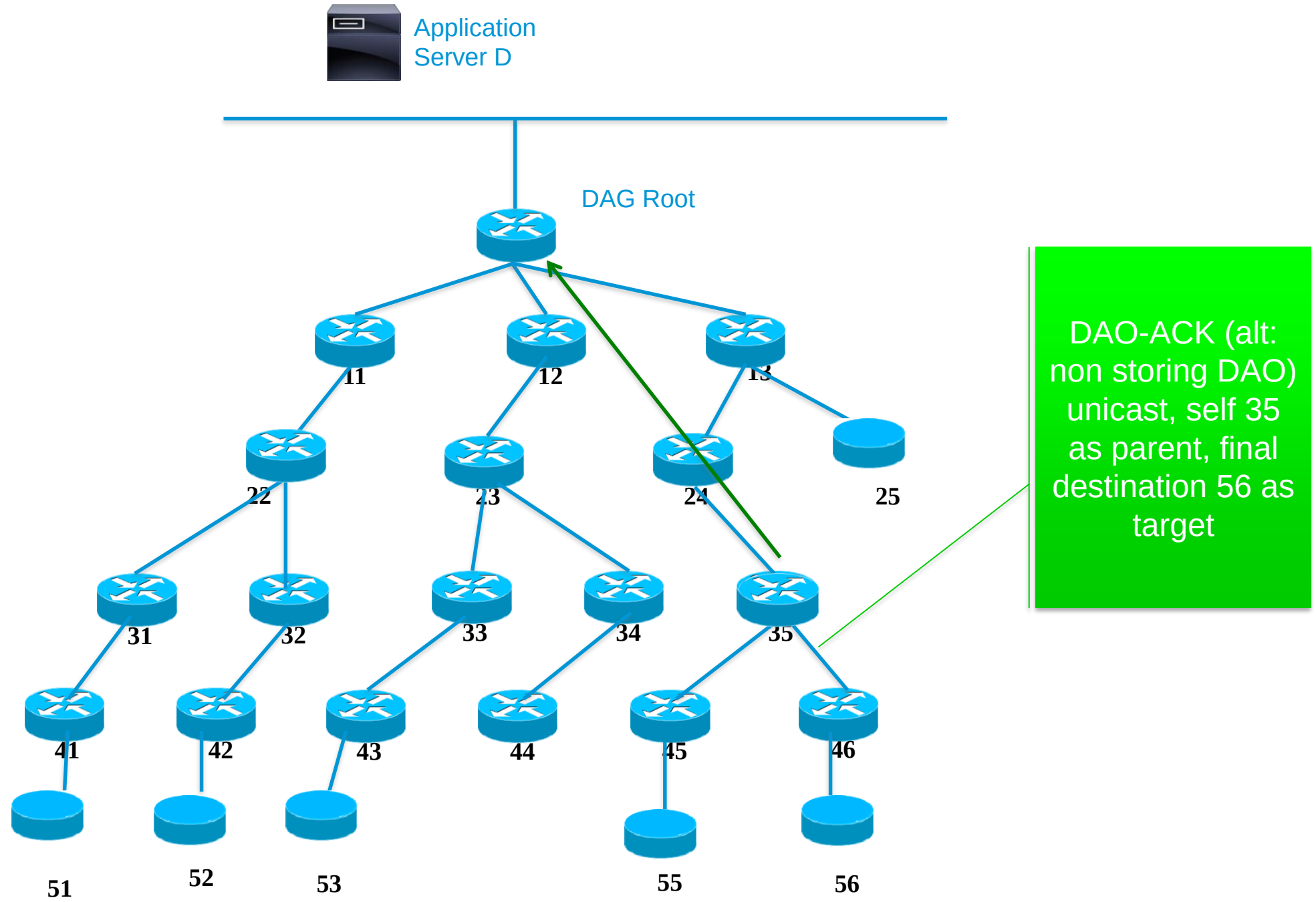


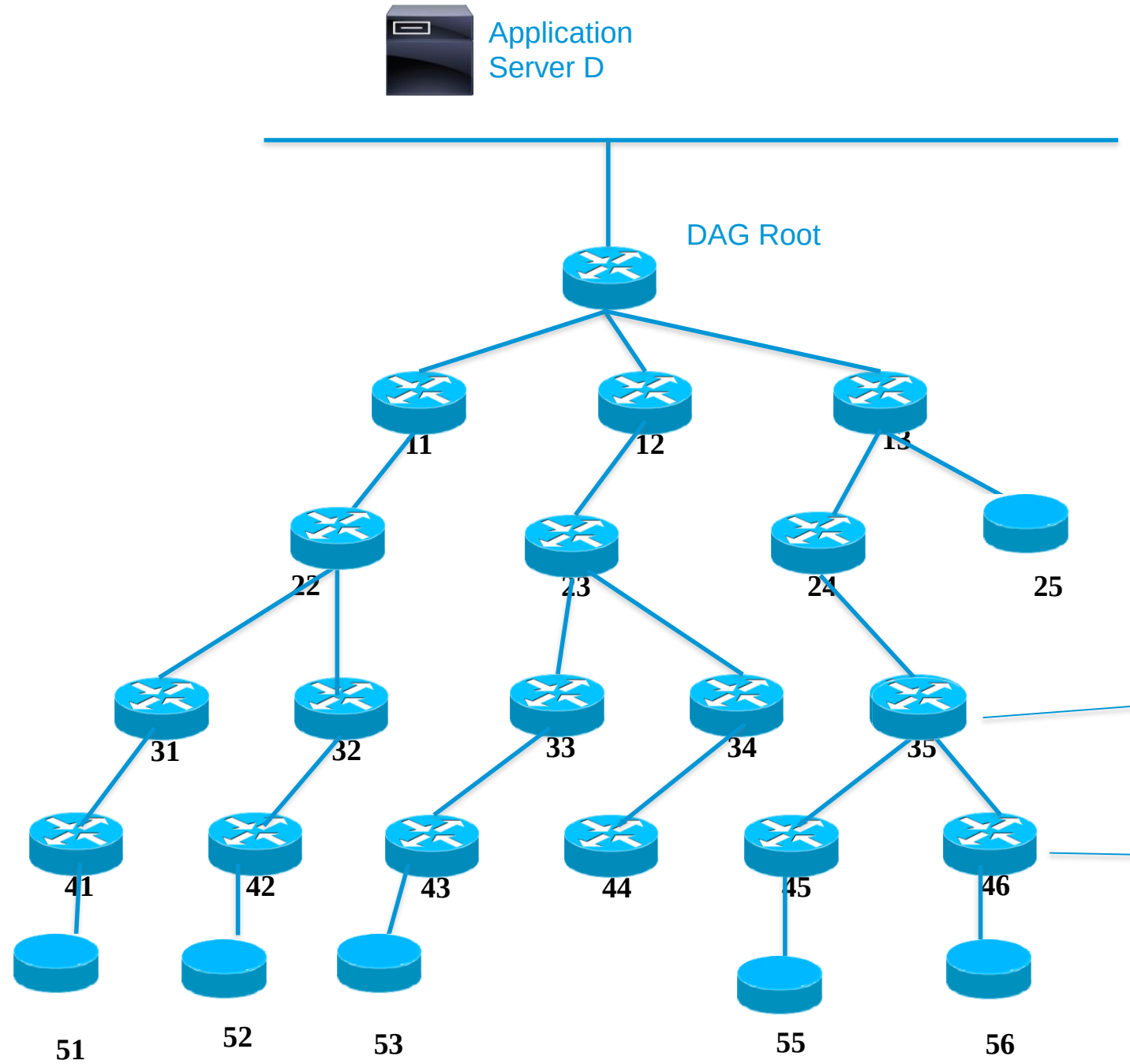
Existing non storing optimization


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

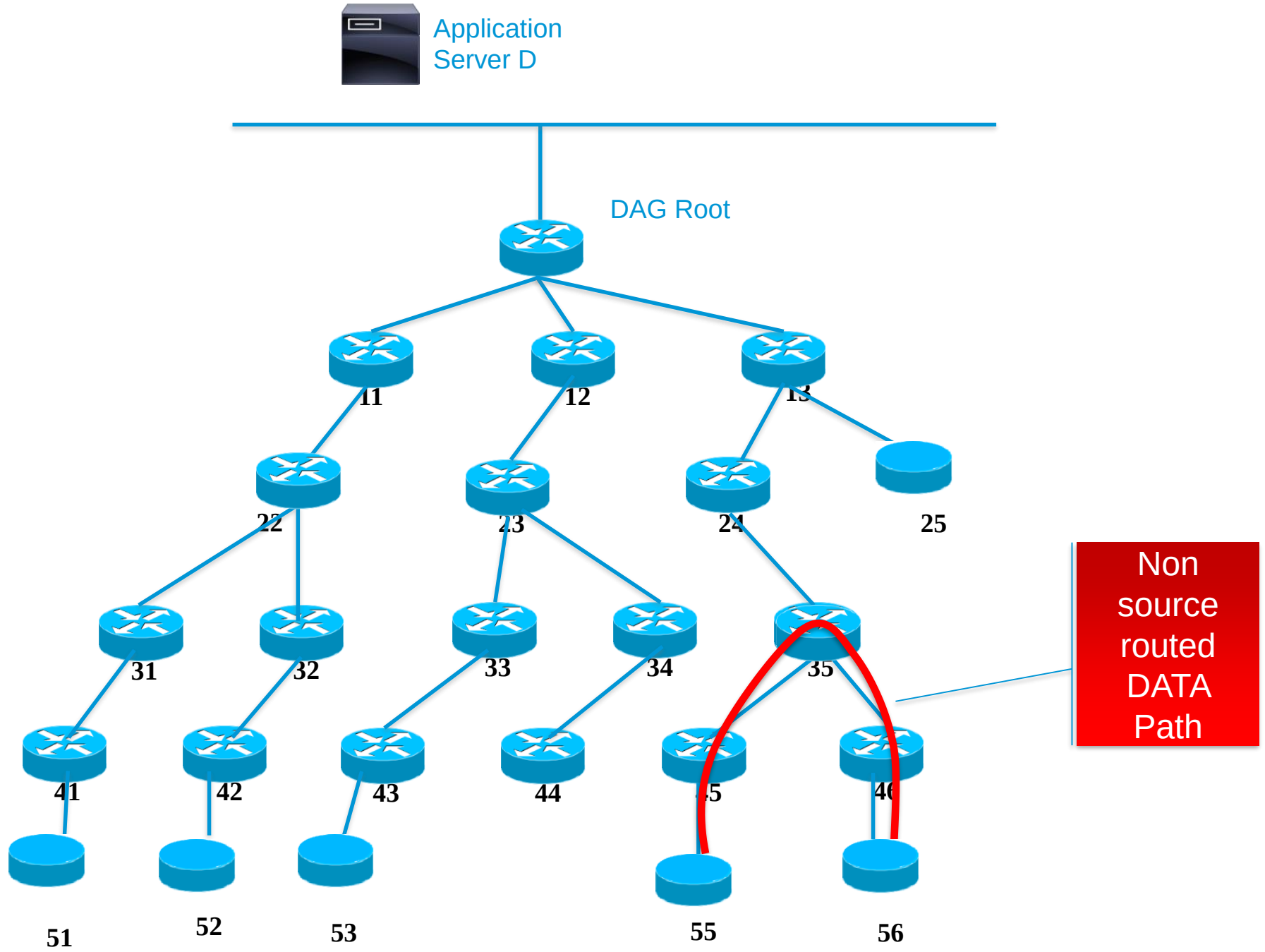

Application
Server D
DAG Root
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56
DAO-ACK (alt: non storing DAO) unicast, self 35 as parent, final destination 56 as target

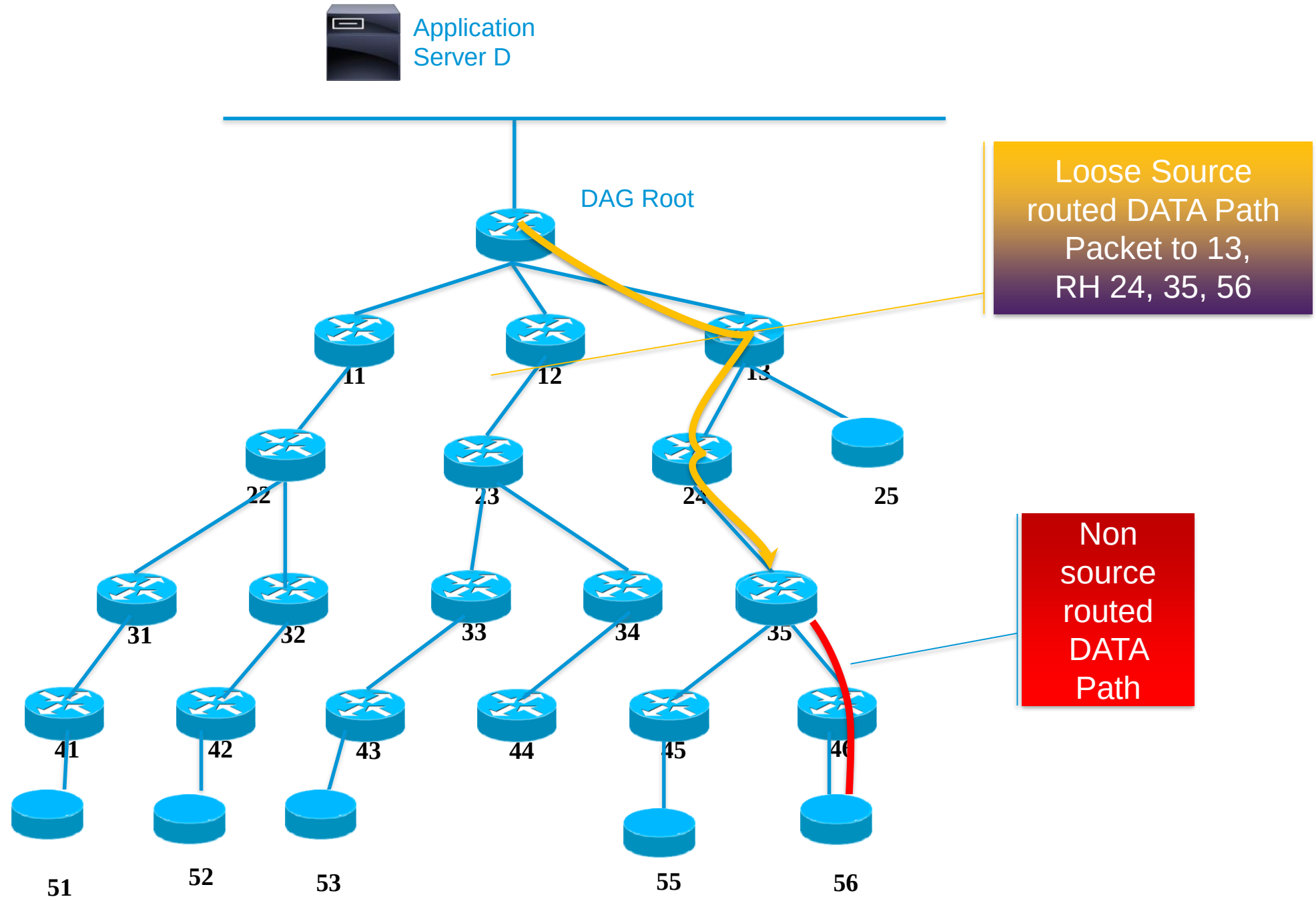


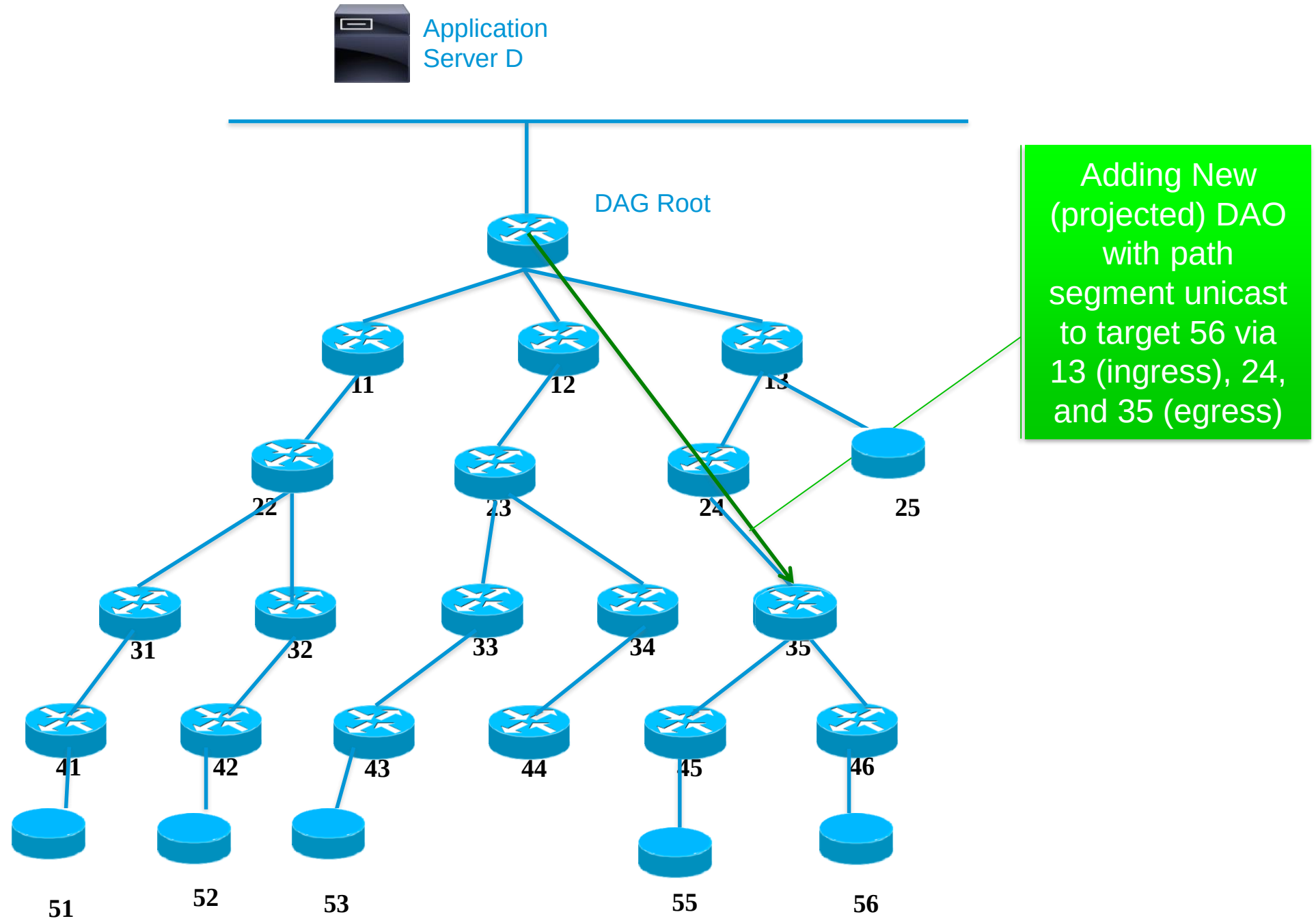
DAO from 46 installs a route to 56 in 35
(all nodes in projected route from ingress included to egress excluded)
=> egress should already have a route to target

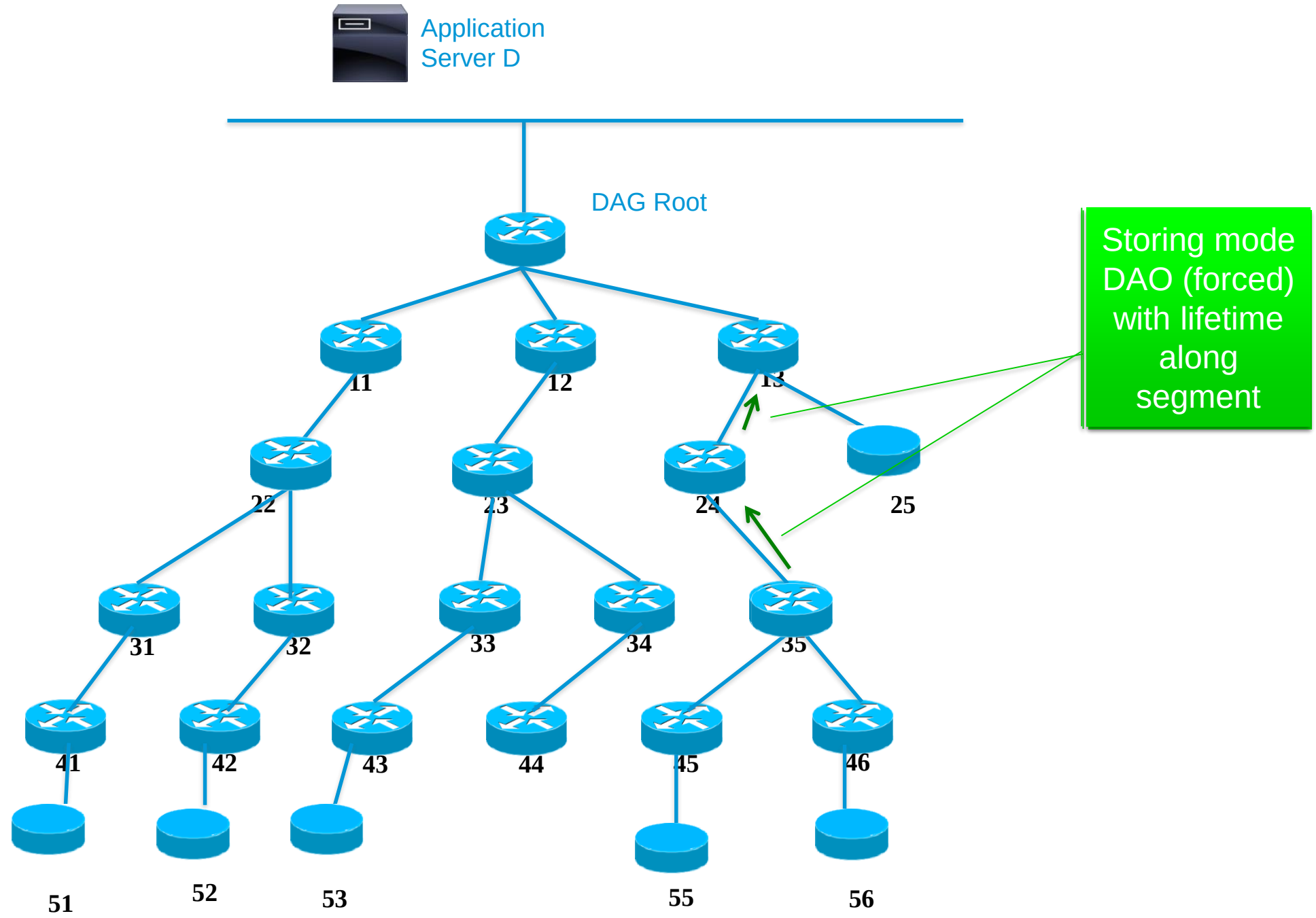
56 via 46

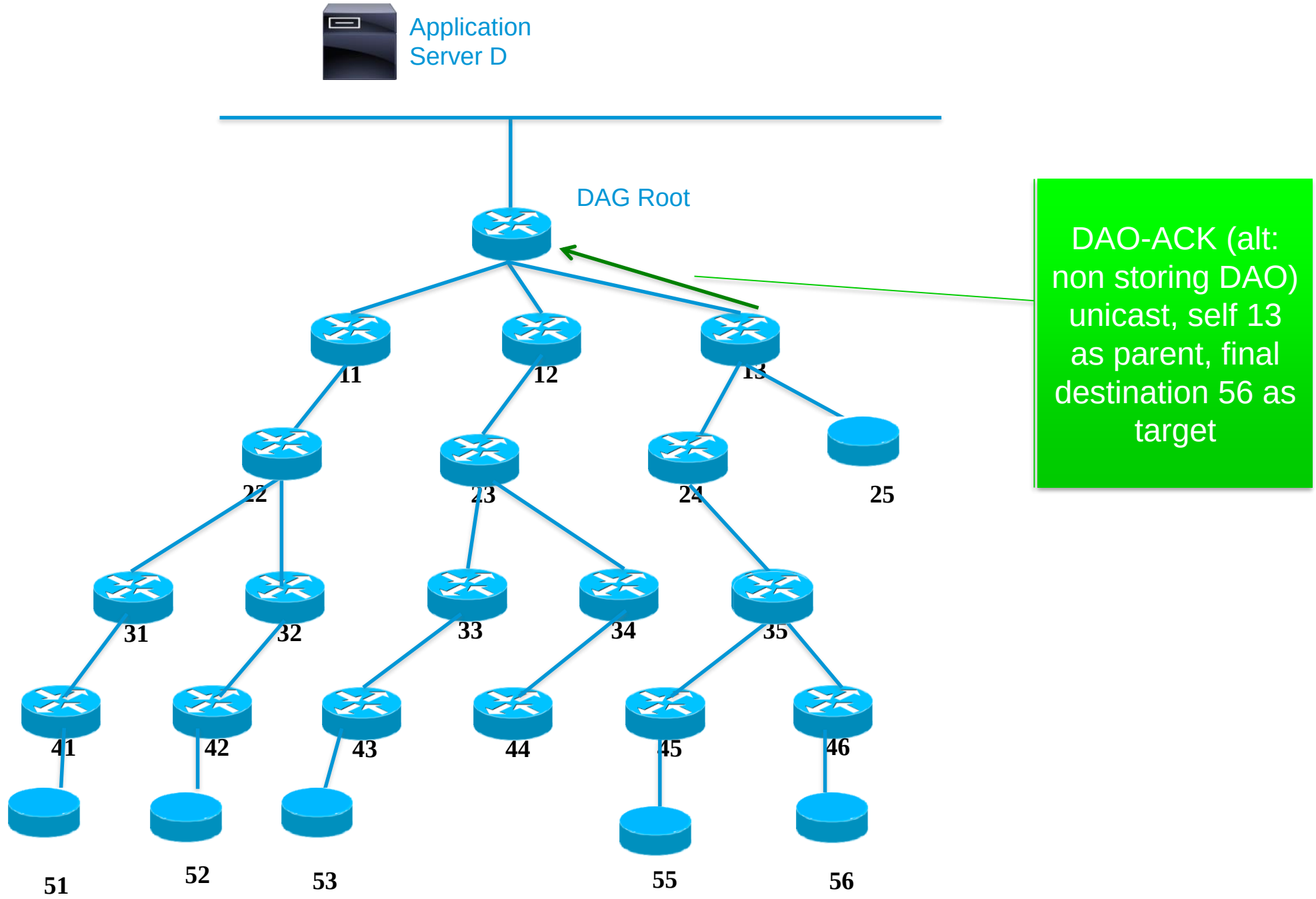
Preexisting connected route to 56

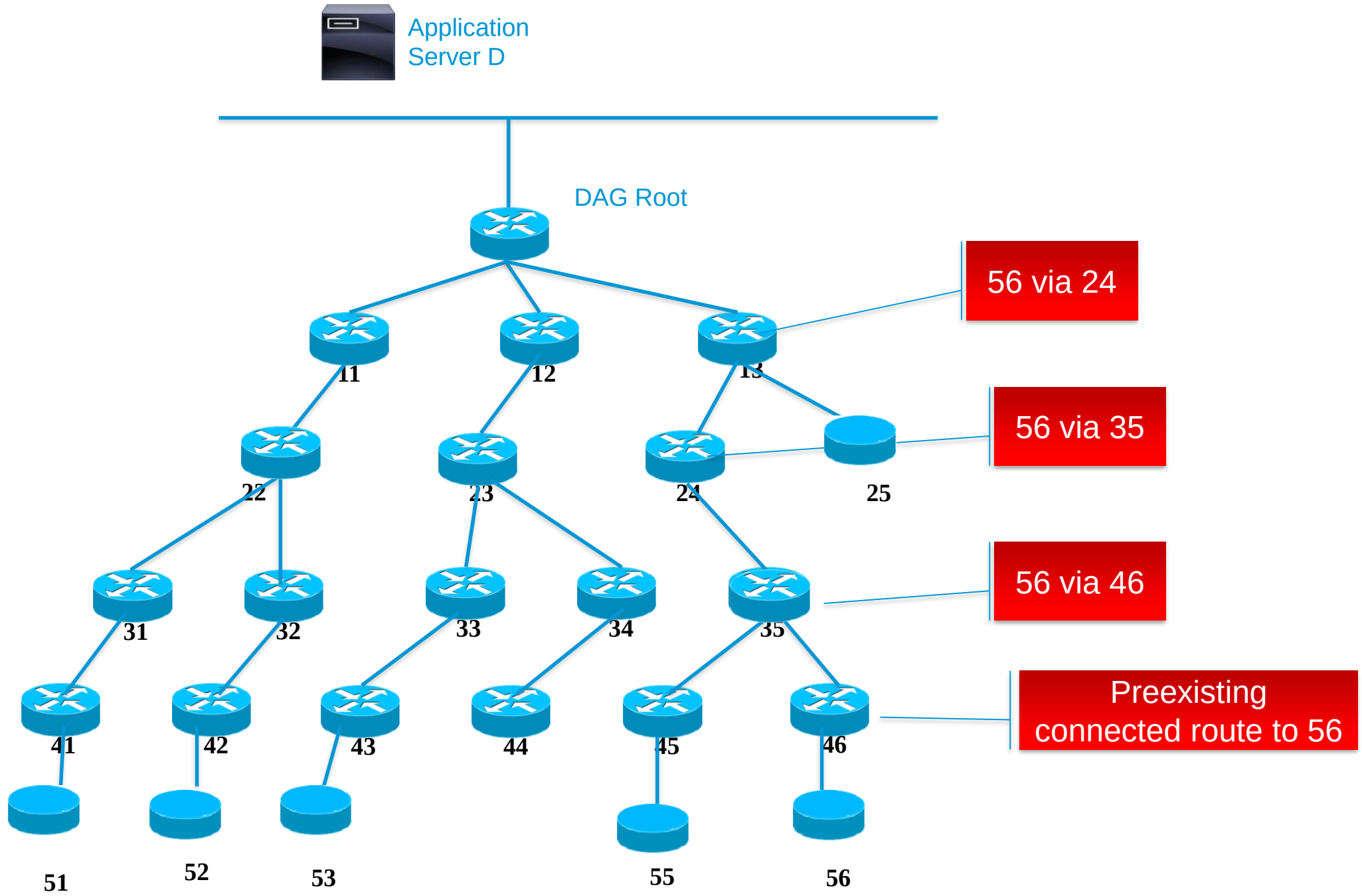

Application
Server D
DAG Root
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56
Non source routed DATA Path

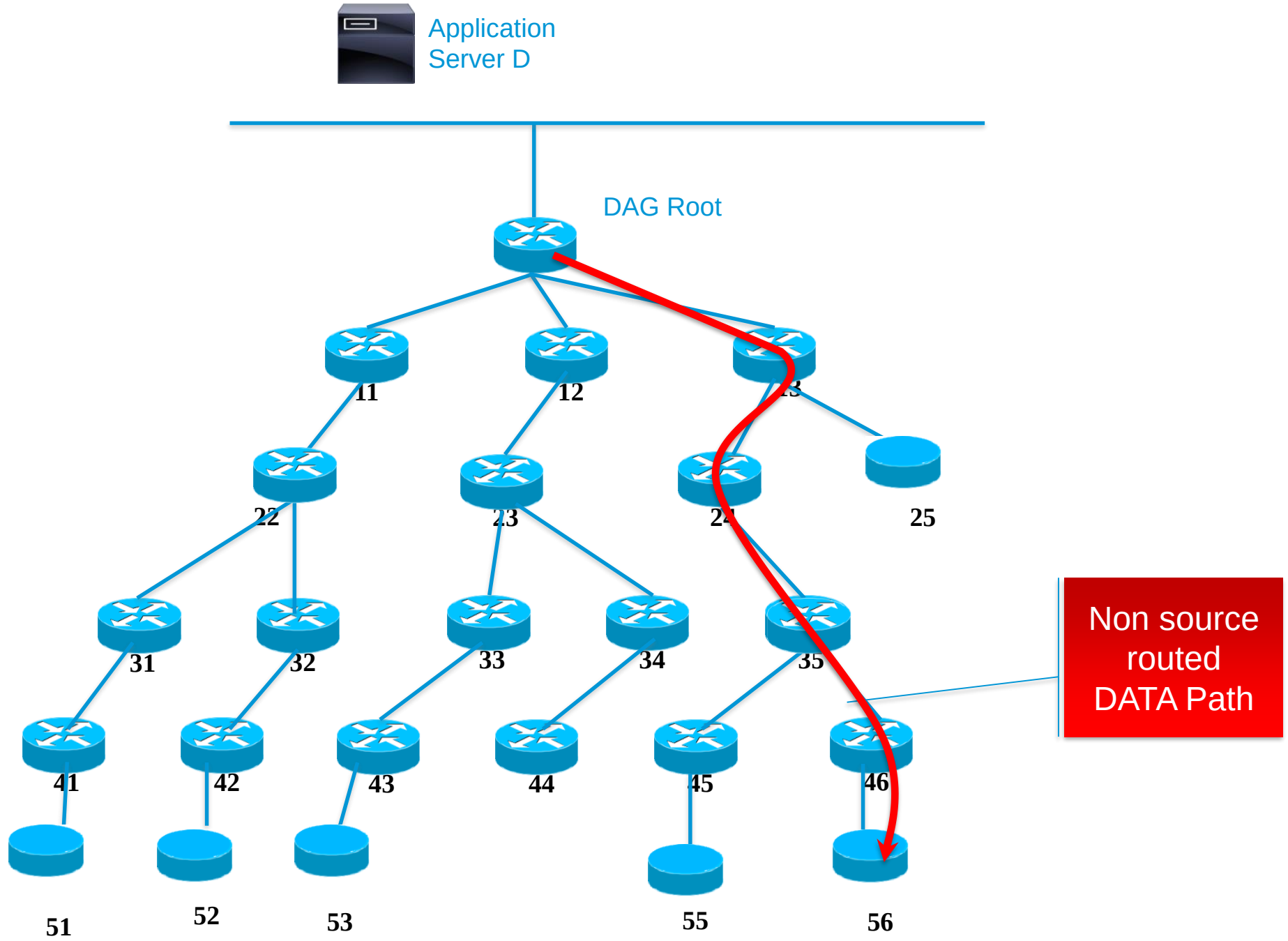

Application
Server D
DAG Root
Loose Source routed DATA Path Packet to 13, RH 24, 35, 56
Non source routed DATA Path
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56


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

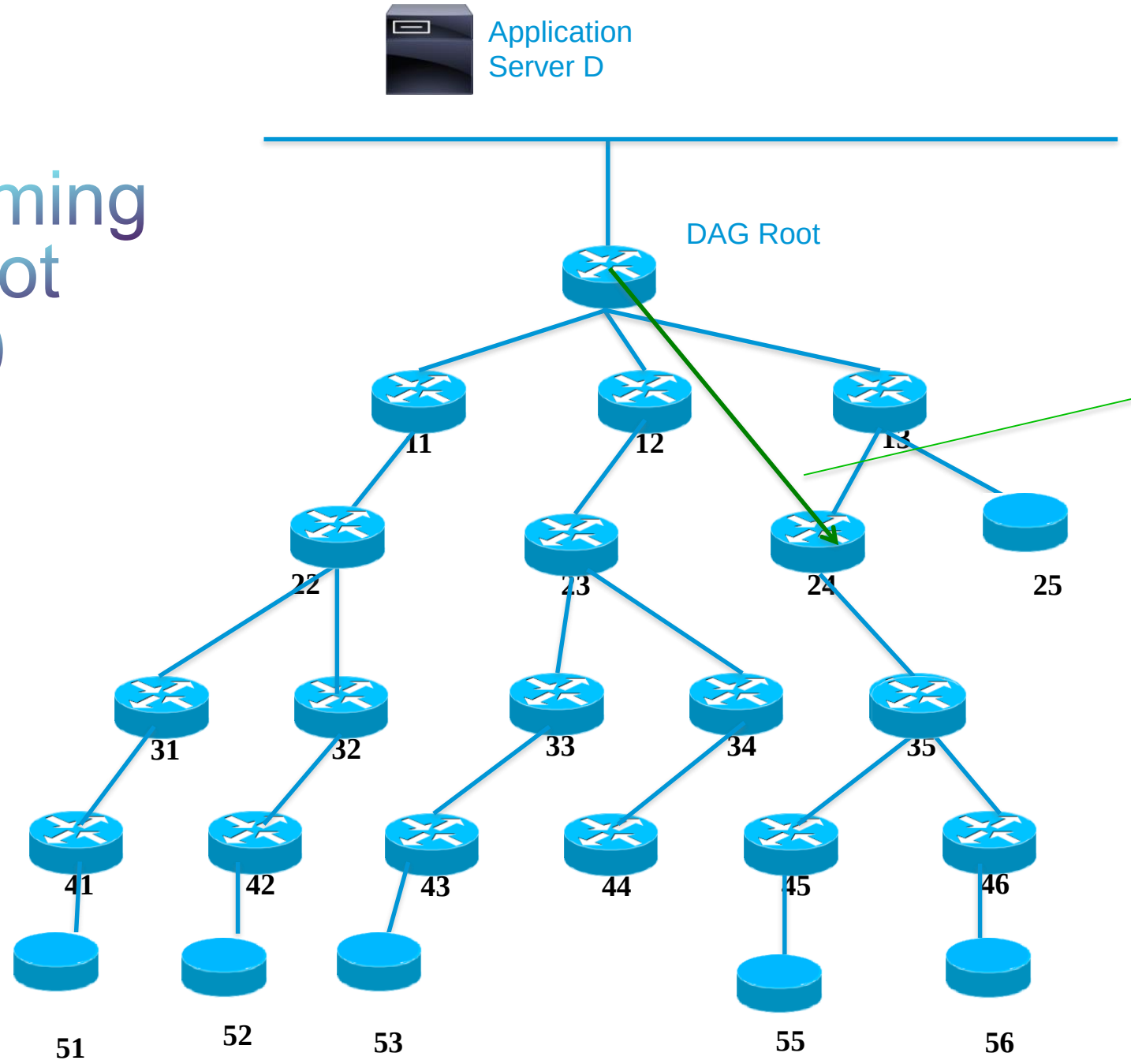

Application
Server D
DAG Root
Storing mode
DAO (forced)
with lifetime
along
segment
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56


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

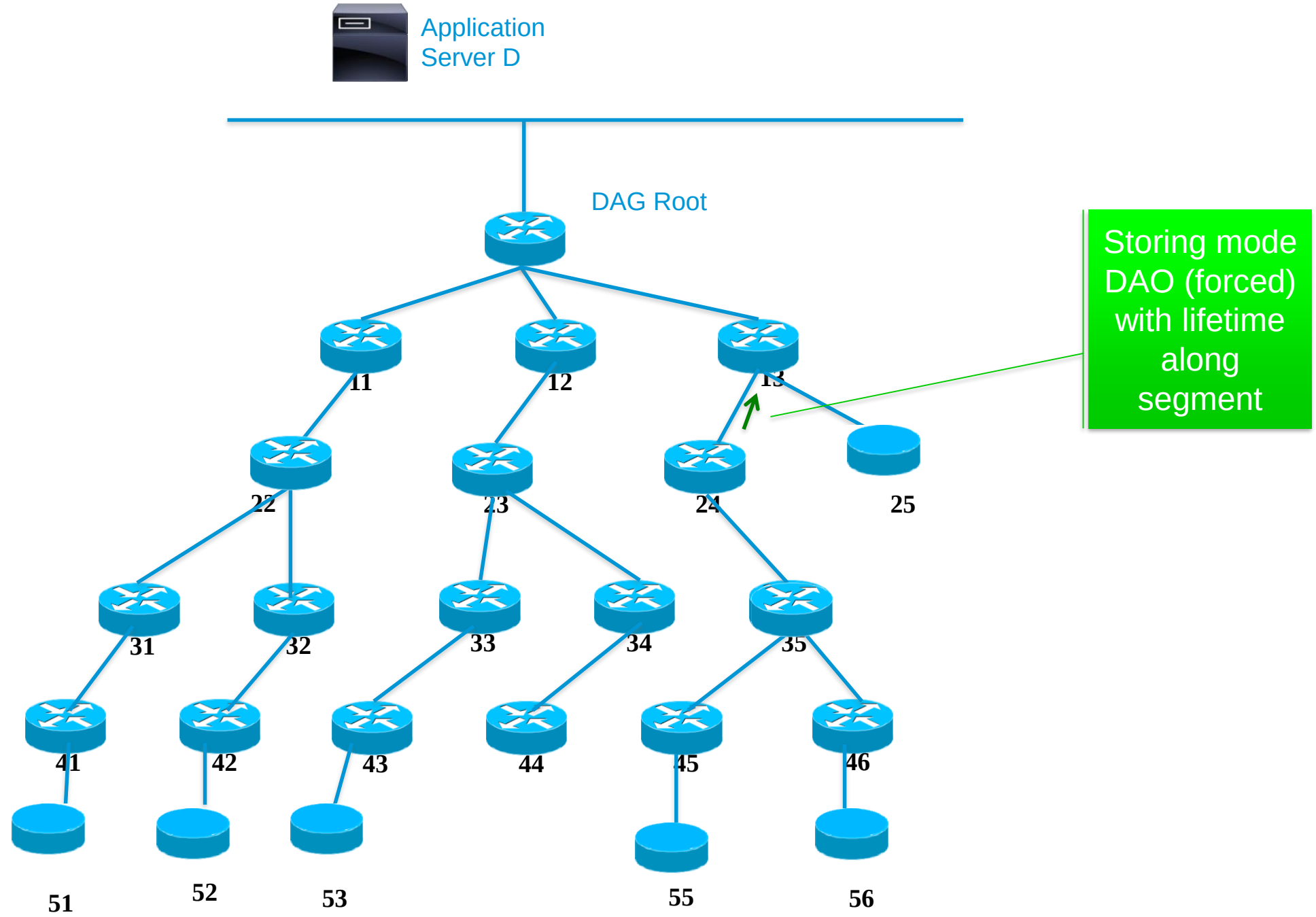

Application
Server D
DAG Root
56 via 24
56 via 35
56 via 46
Preexisting
connected route to 56
11
12
13
22
23
24
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56

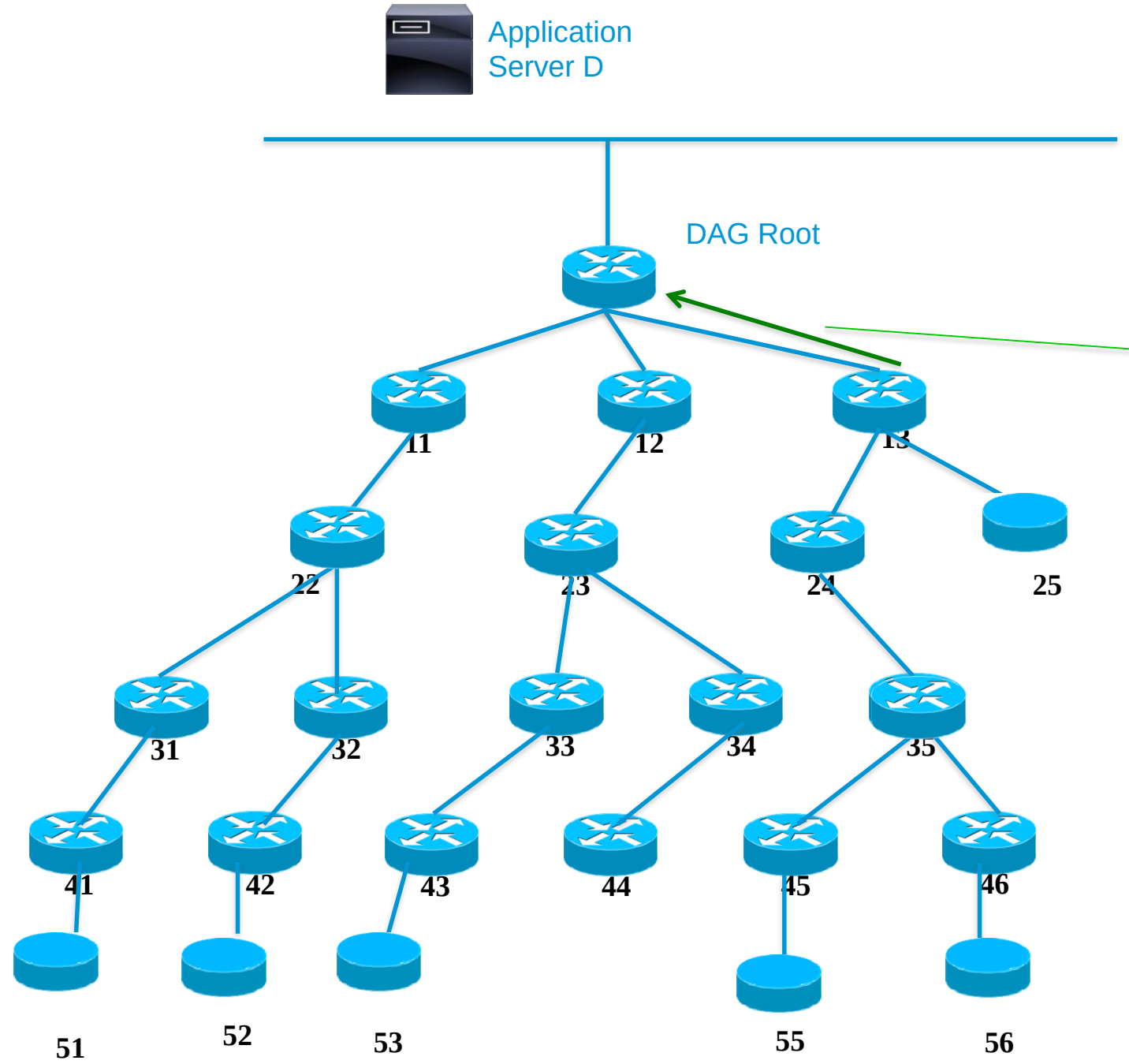

Application
Server D
DAG Root
11
12
22
23
25
31
32
33
34
35
41
42
43
44
45
46
51
52
53
55
56
Non source
routed
DATA Path

Alternate Programming By the root (Michael)

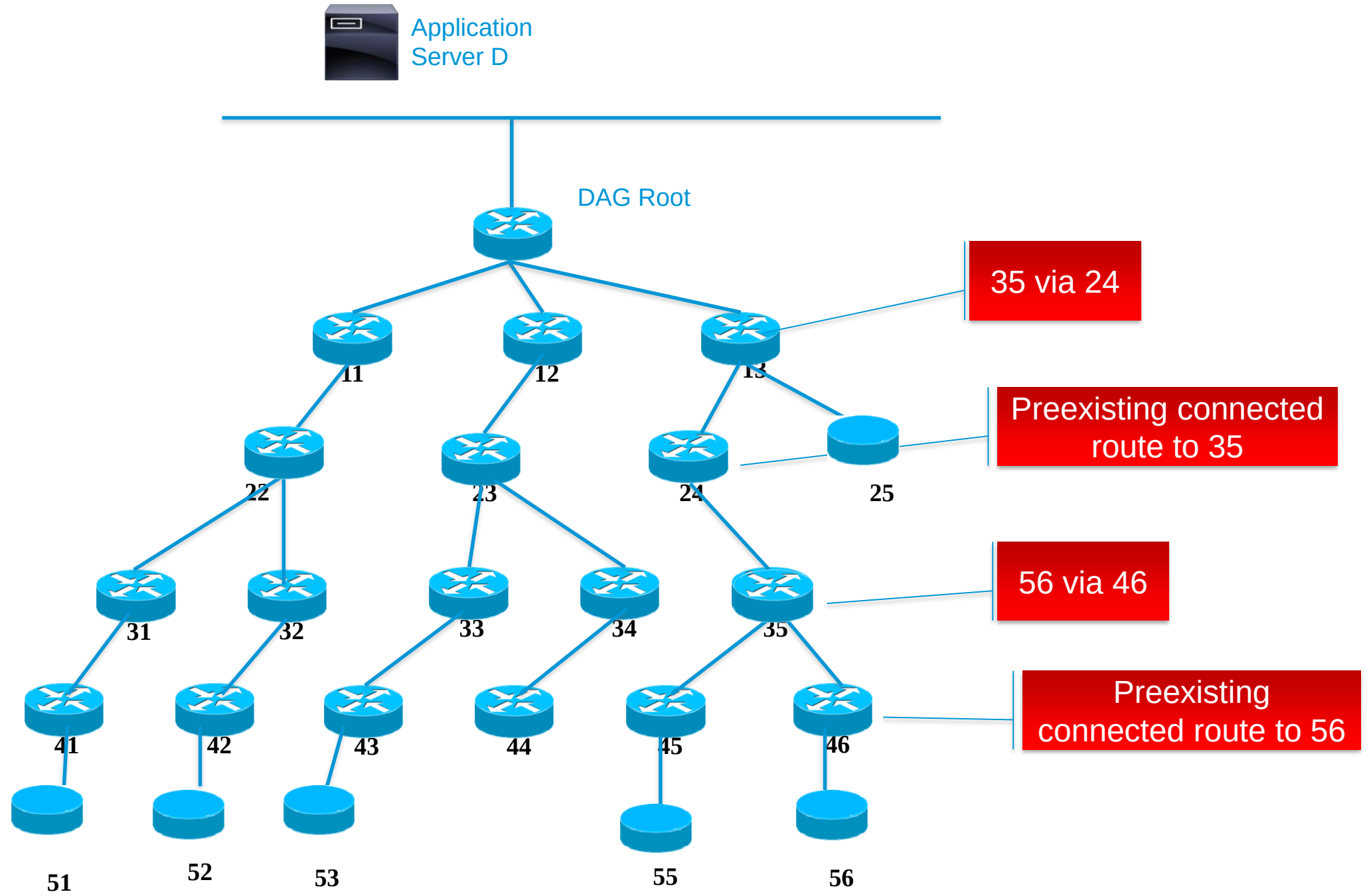


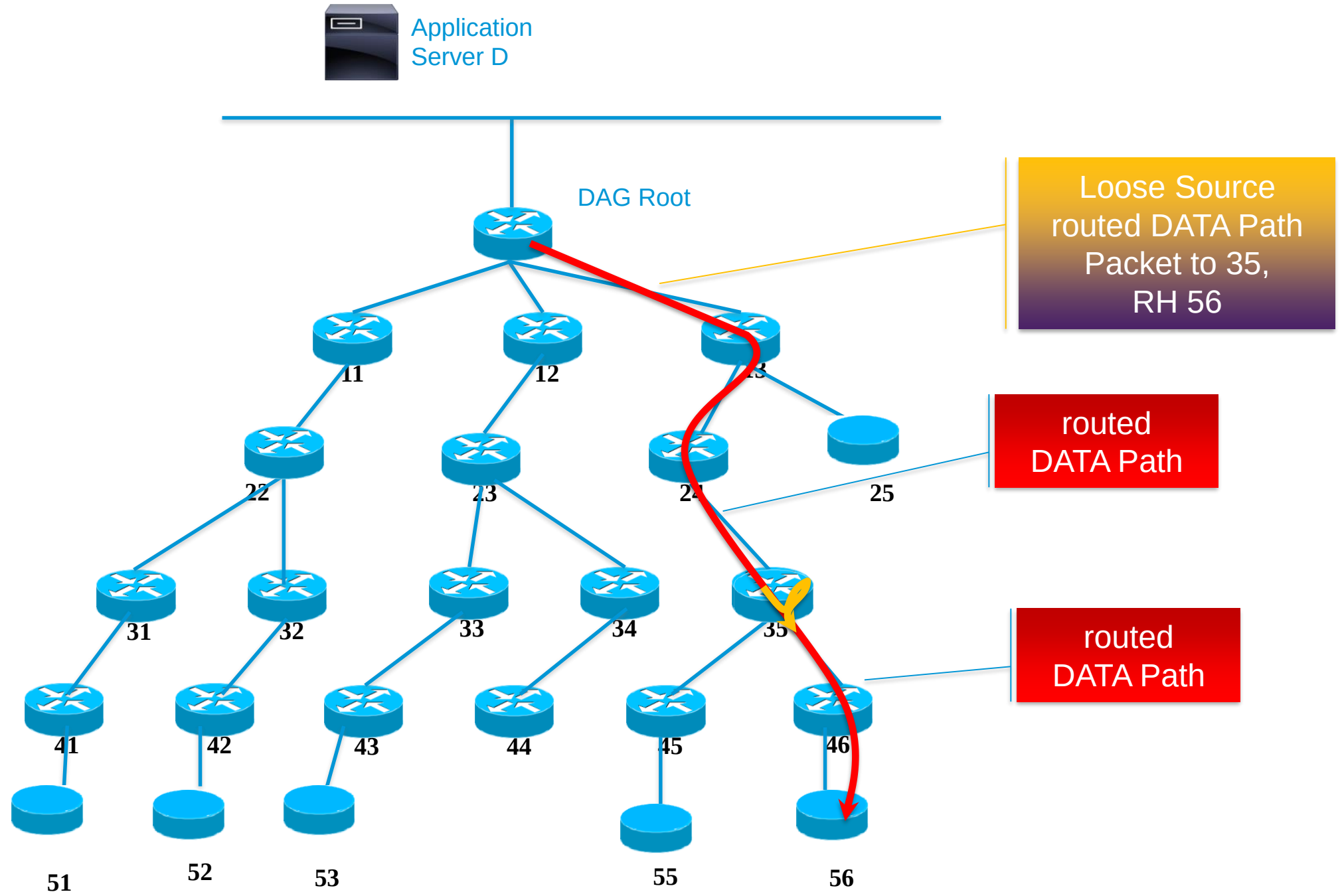
ALT: Adding New (projected) DAO with path segment unicast to target 35 via 13 (ingress) and 24 (egress)


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



DAO-ACK (alt: non storing DAO) unicast, self 13 as parent, final destination 56 as target


Application
Server D
DAG Root
35 via 24
Preexisting connected
route to 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
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