Software-Defined Data Centers

Brighten Godfrey
CS 538 April 11, 2018

Multi-Tenant Data Centers: The Challenges

Key Needs

Agility

Strength

Constitution

Dexterity

Charisma

Key Needs

Agility

Location independent addressing

Performance uniformity

Security

Network semantics

Agility

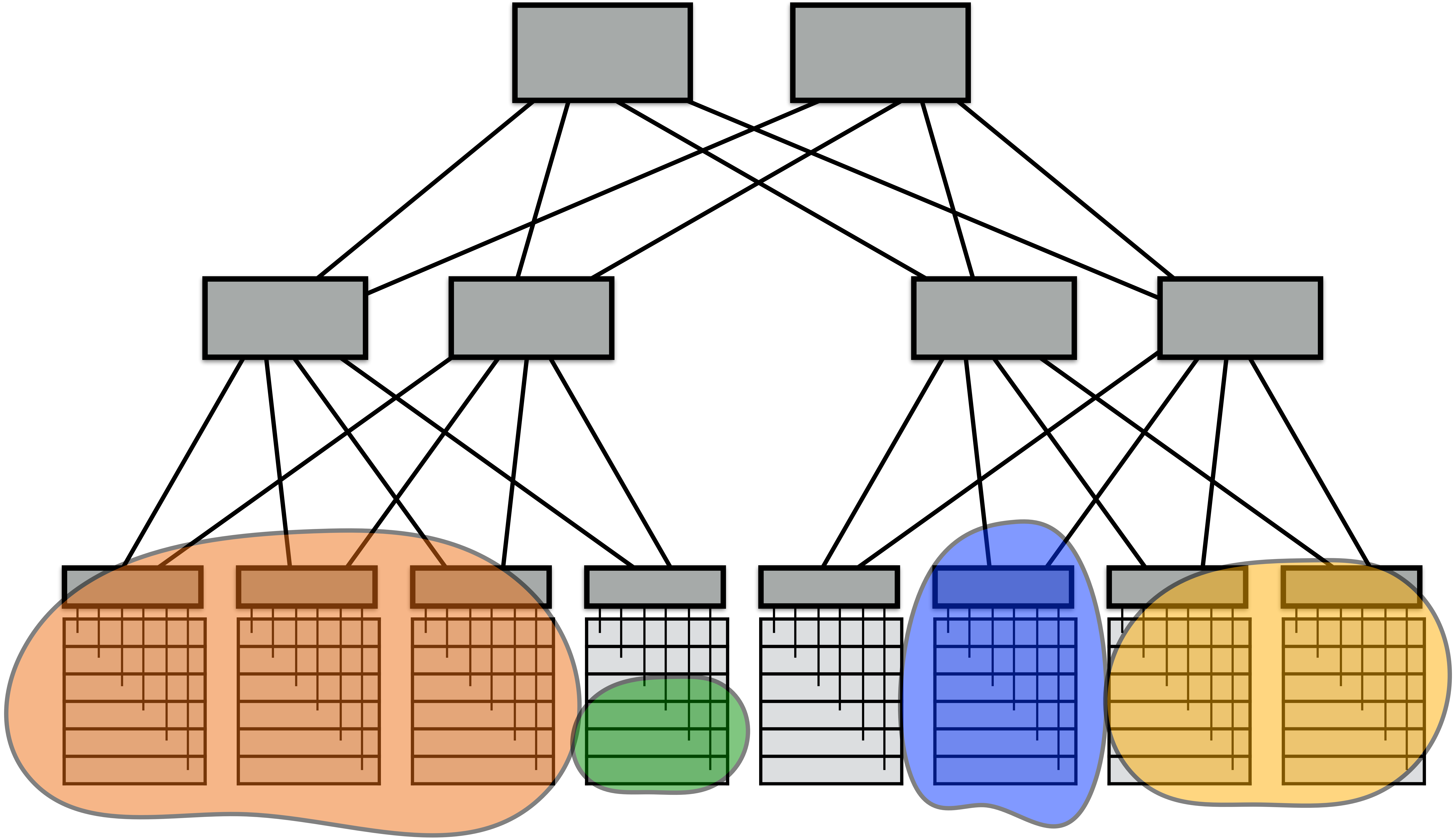
Agility: Use any server for any service at any time

- Better economy of scale through increased utilization
- Improved reliability

Service / tenant

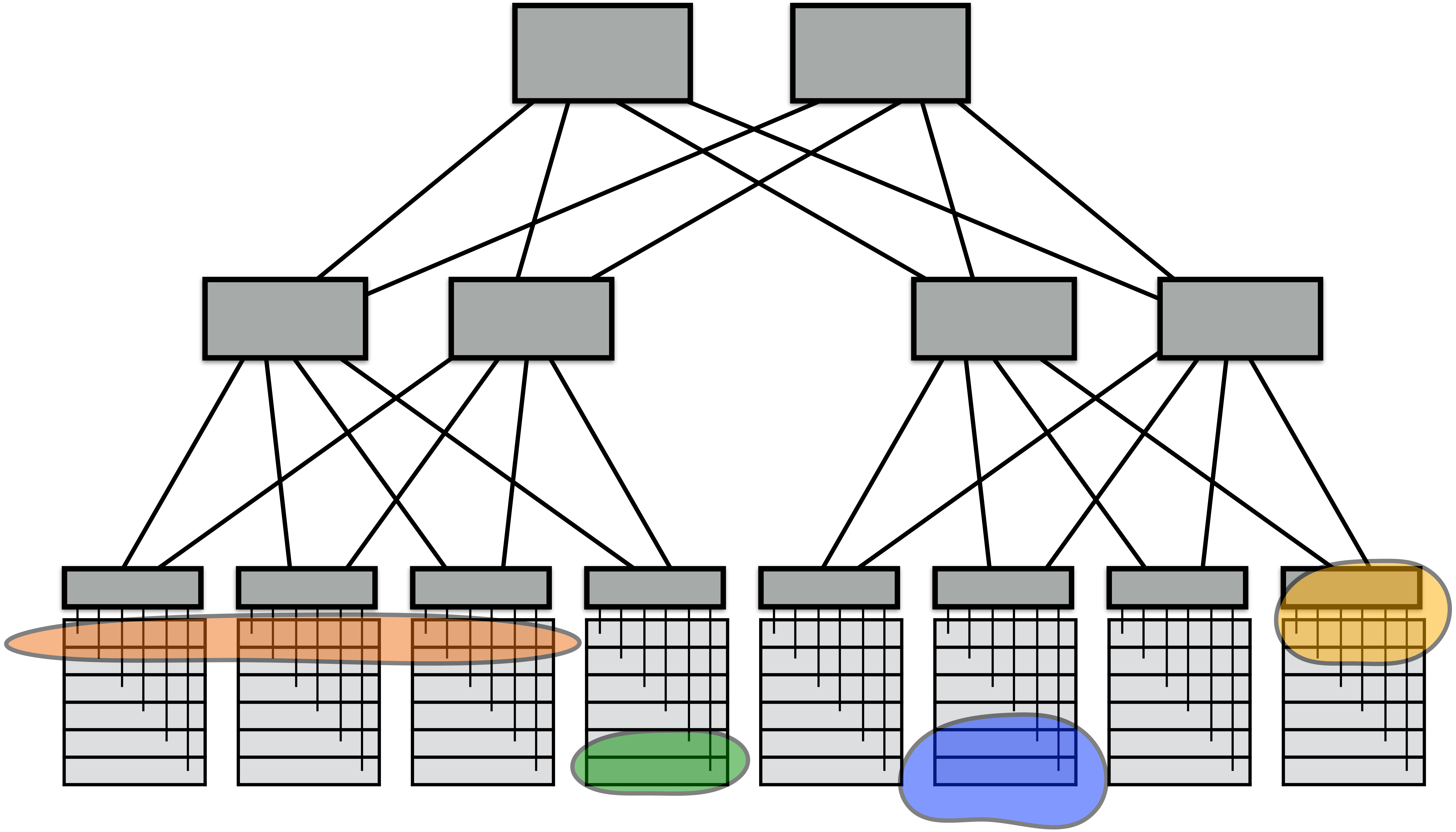
- Customer renting space in a public cloud
- Application or service in a private cloud (internal customer)

Lack of Agility in Traditional DCs



Tenants in “silos”: VLAN associated with a particular IP prefix

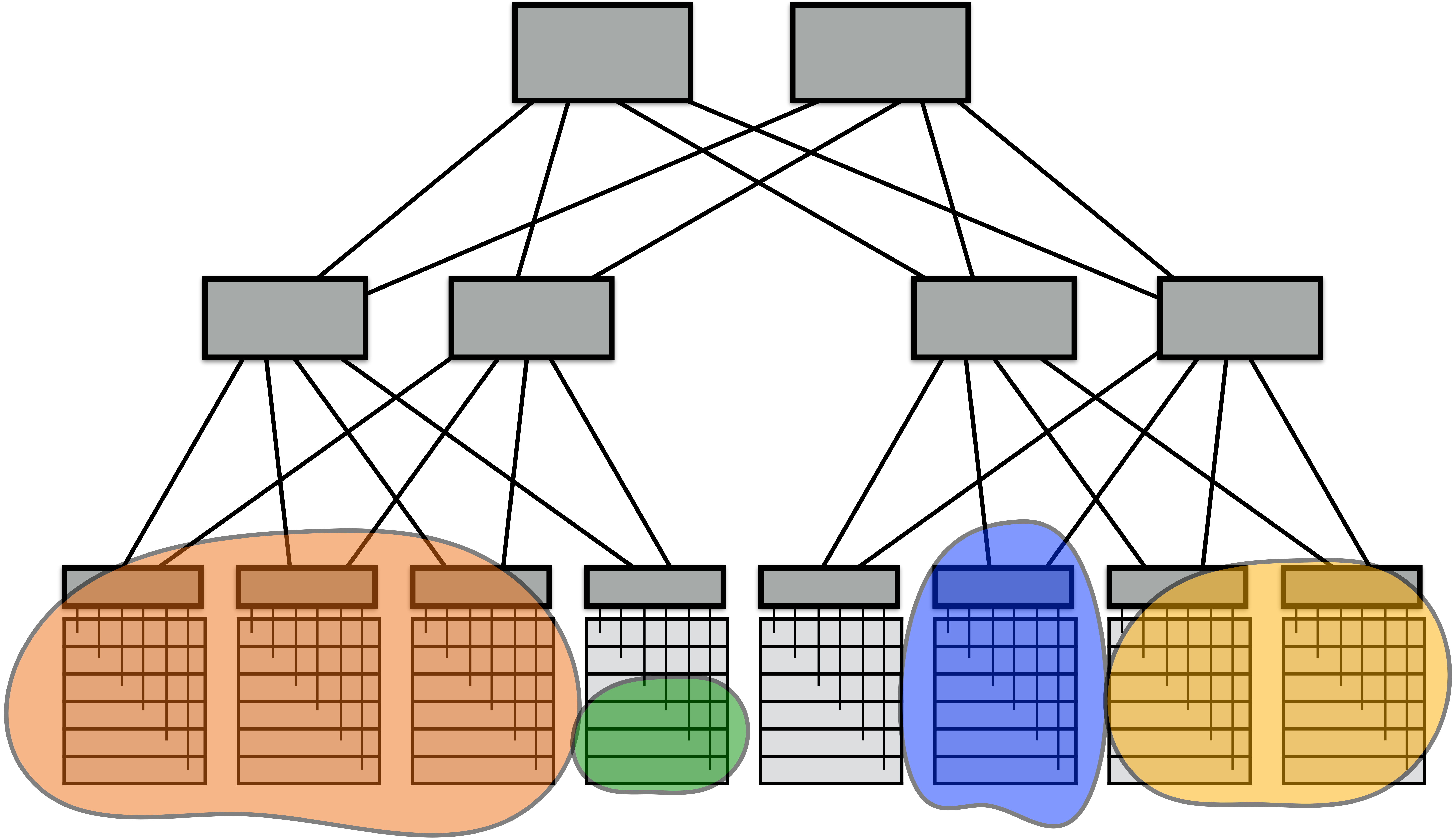
Lack of Agility in Traditional DCs



Tenants in “silos”

Poor utilization

Lack of Agility in Traditional DCs

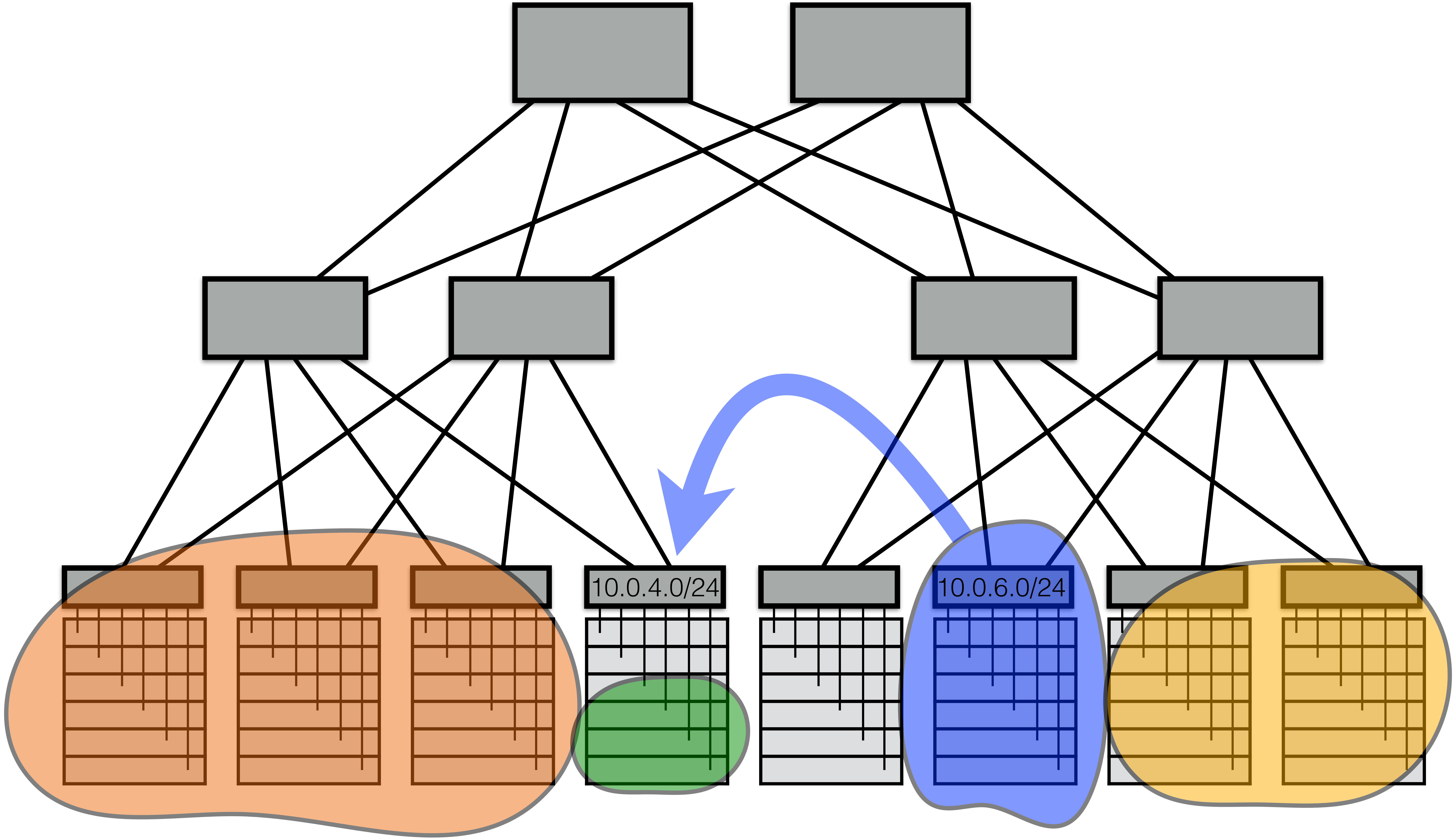


Tenants in “silos”

Poor utilization

Inability to expand

Lack of Agility in Traditional DCs



IP addresses locked to topological location!

Key Needs

Agility

Location independent addressing

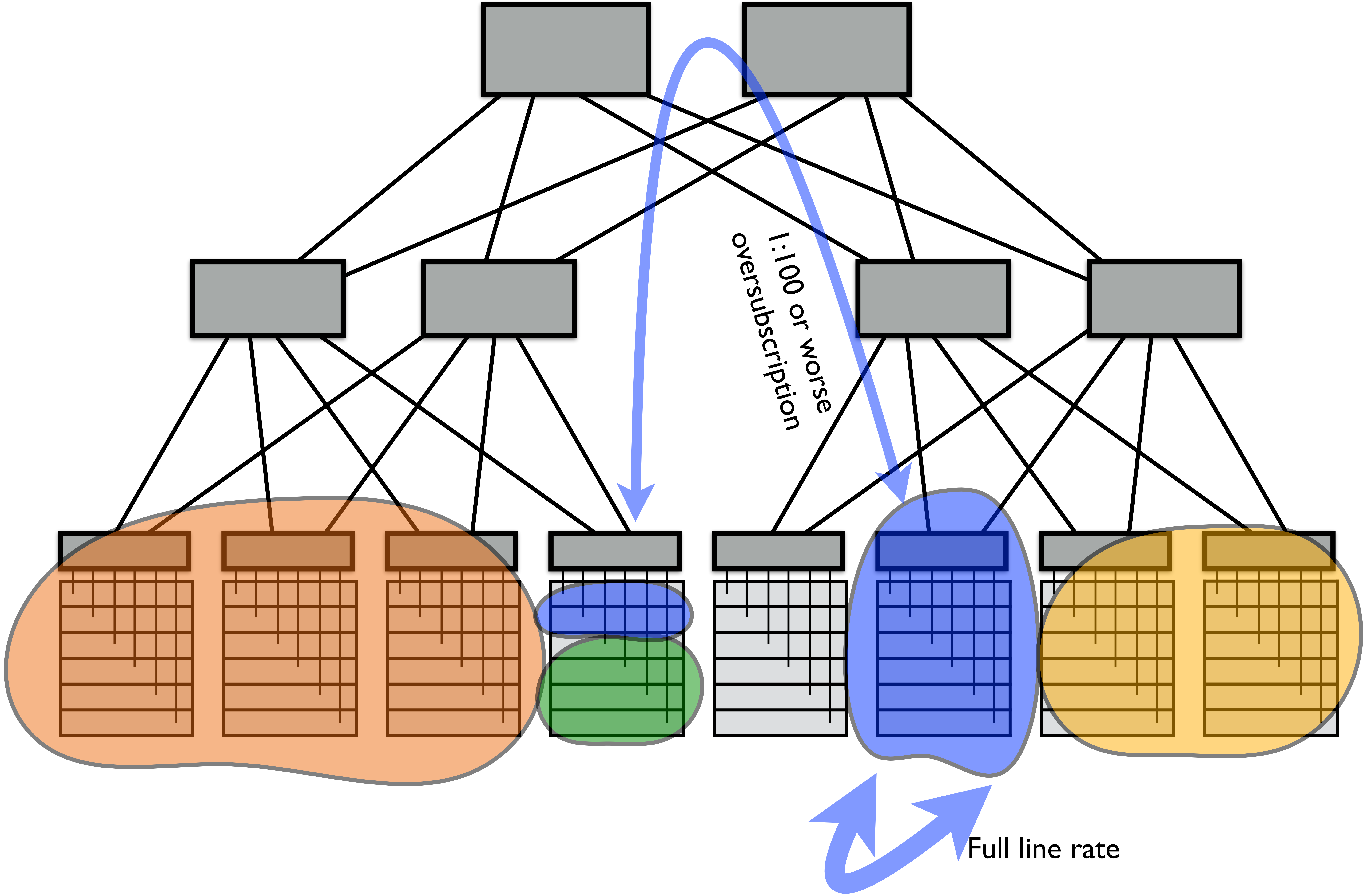
- Tenant's IP addresses should be portable anywhere

Performance uniformity

Security

Network semantics

Lack of Agility in Traditional DCs



Nonuniform performance

Key Needs

Agility

Location independent addressing

- Tenant's IP addresses can be taken anywhere

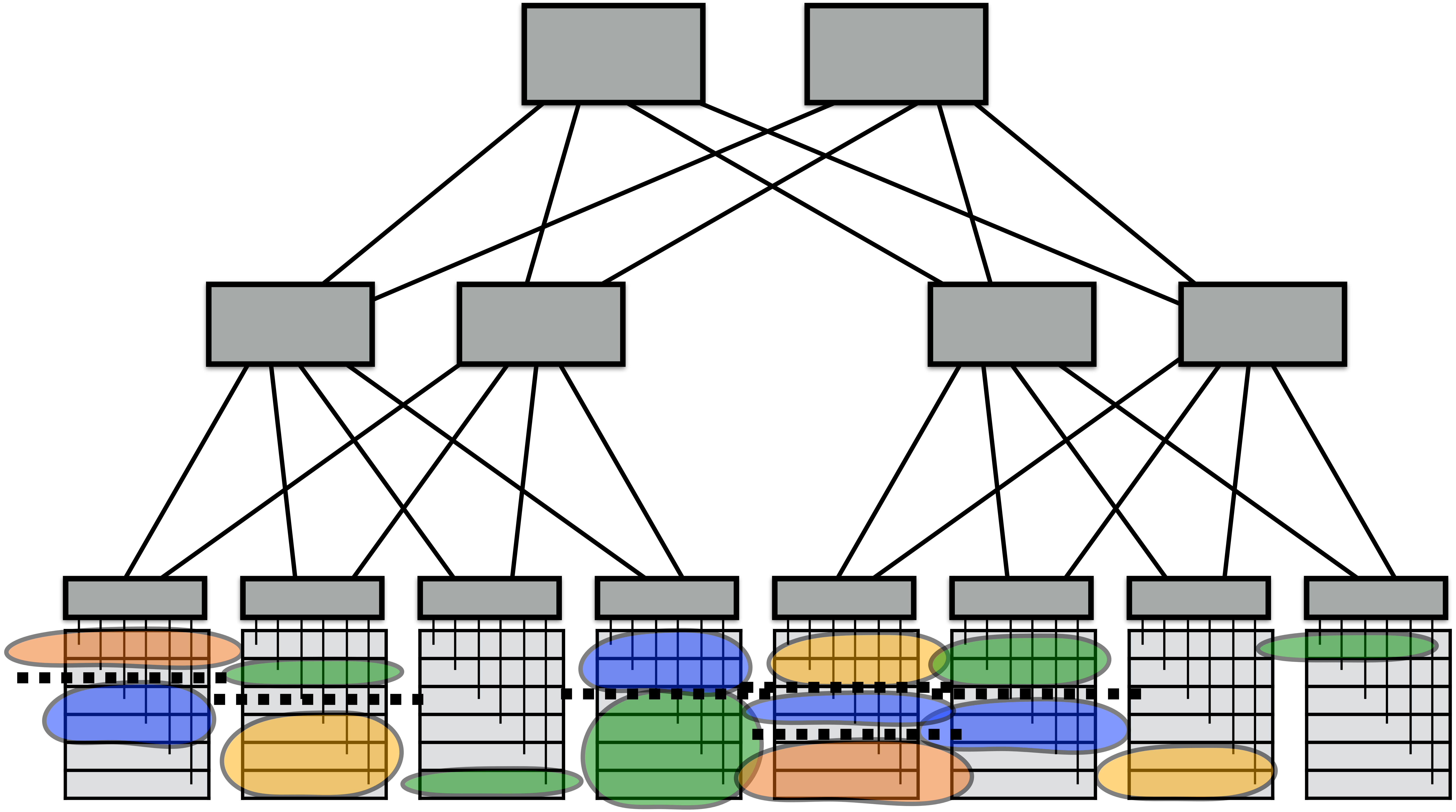
Performance uniformity

- VMs receive same throughput regardless of placement

Security

Network semantics

Lack of Agility in Traditional DCs



Untrusted
environment

Key Needs

Agility

Location independent addressing

- Tenant's IP addresses can be taken anywhere

Performance uniformity

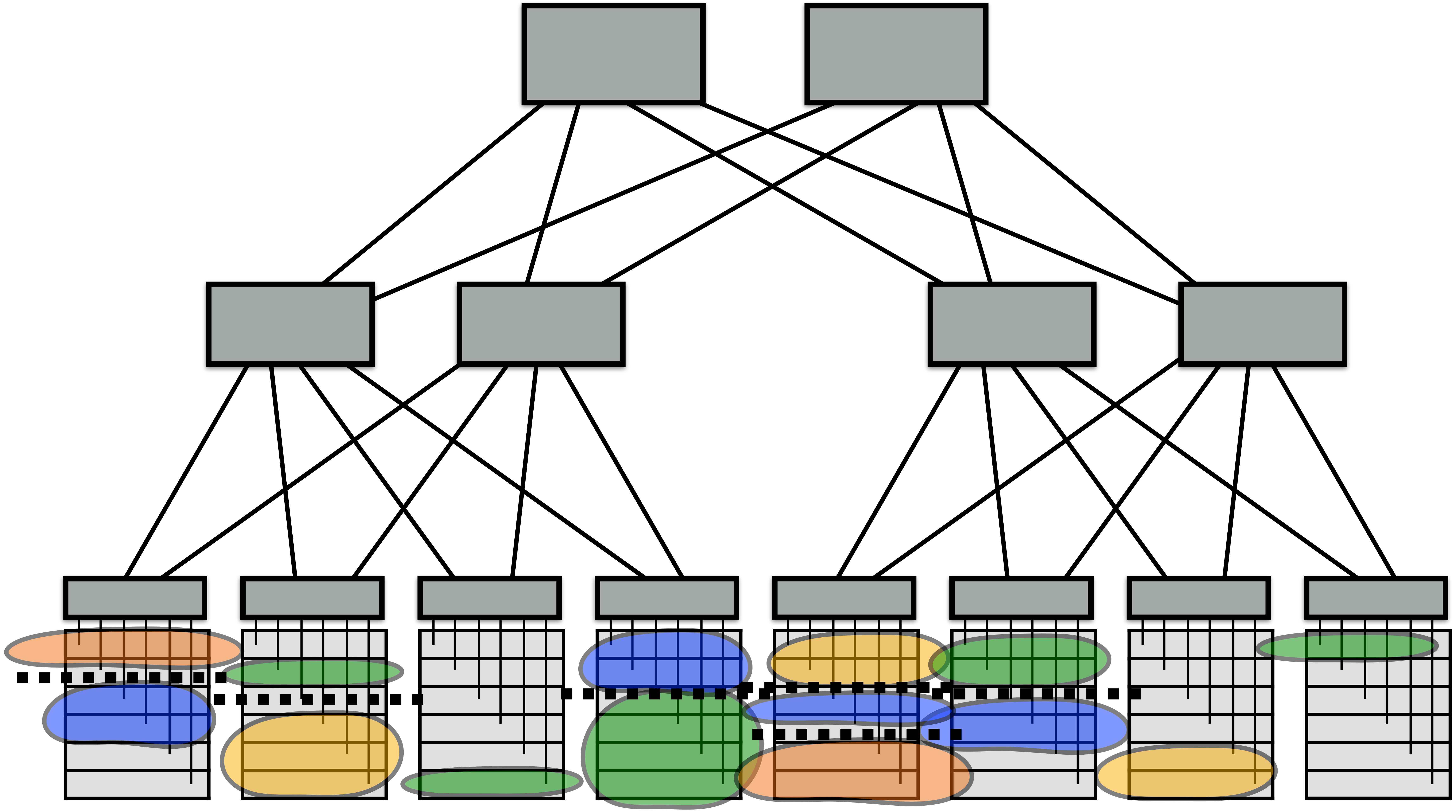
- VMs receive same throughput regardless of placement

Security

- Micro-segmentation: isolation at tenant or app granularity

Network semantics

Lack of Agility in Traditional DCs



x 1000s of legacy apps in a large enterprise…in a much messier topology

Key Needs

Agility

Location independent addressing

- Tenant's IP addresses can be taken anywhere

Performance uniformity

- VMs receive same throughput regardless of placement

Security

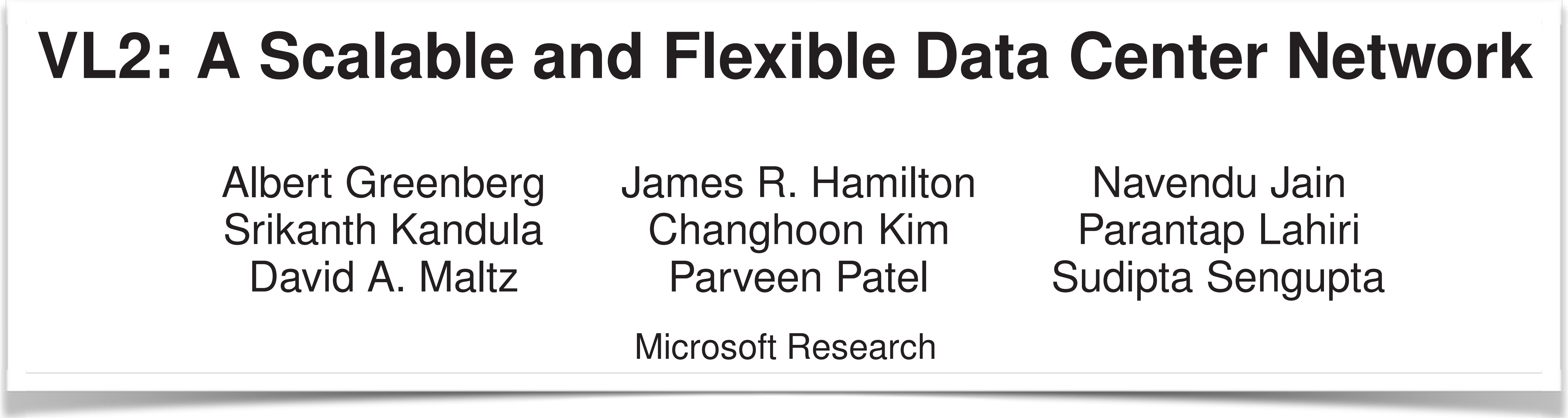
- Micro-segmentation: isolation at tenant granularity

Network semantics

- Layer 2 service discovery, multicast, broadcast, …

Network Virtualization Case Study: VL2

Case Study



VL2: A Scalable and Flexible Data Center Network

Albert Greenberg
Srikanth Kandula
David A. Maltz

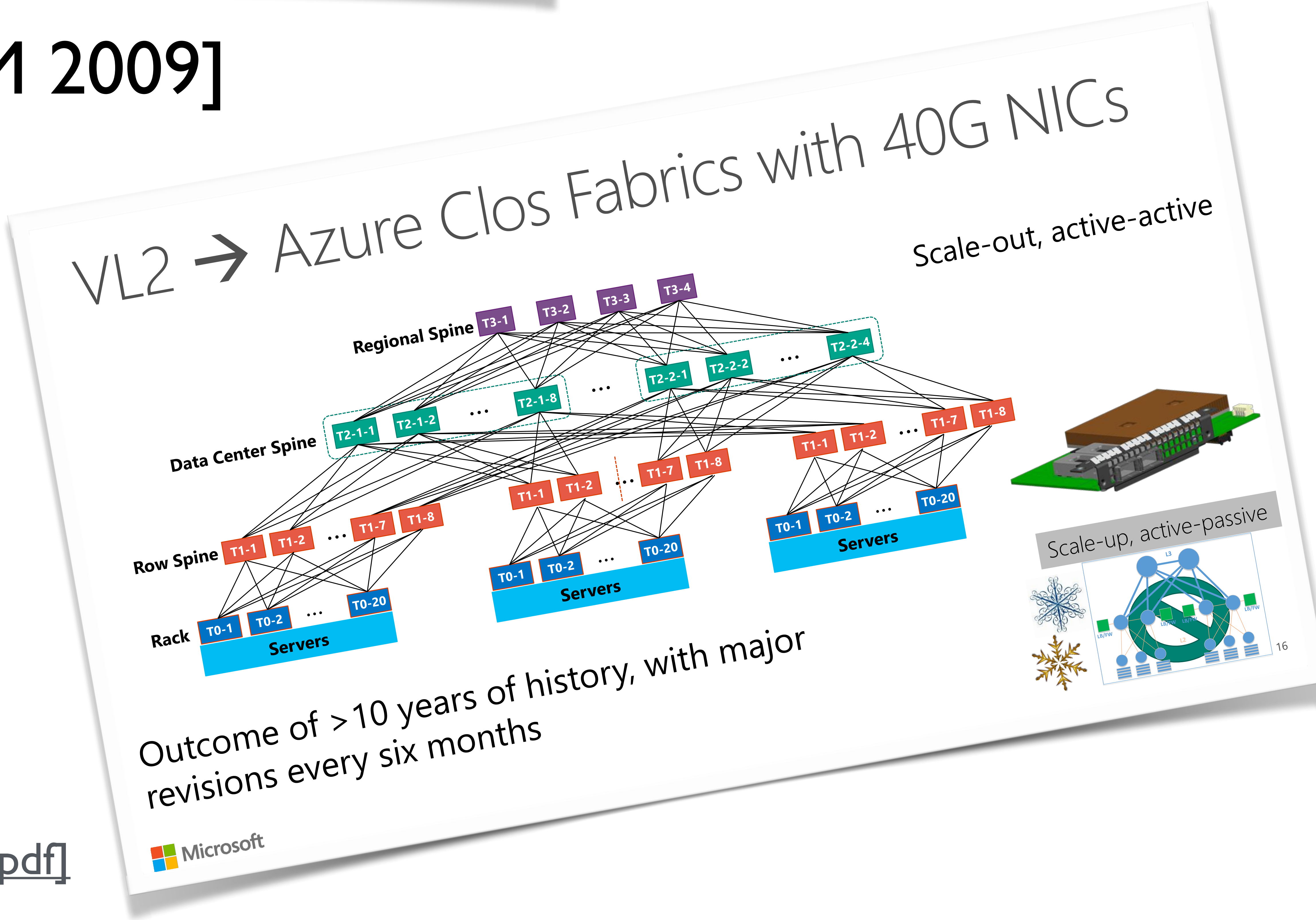
James R. Hamilton
Changhoon Kim
Parveen Patel

Navendu Jain
Parantap Lahiri
Sudipta Sengupta

Microsoft Research

[ACM SIGCOMM 2009]

Influenced architecture of Microsoft Azure



VL2 → Azure Clos Fabrics with 40G NICs

Scale-out, active-active

Scale-up, active-passive

Outcome of >10 years of history, with major revisions every six months

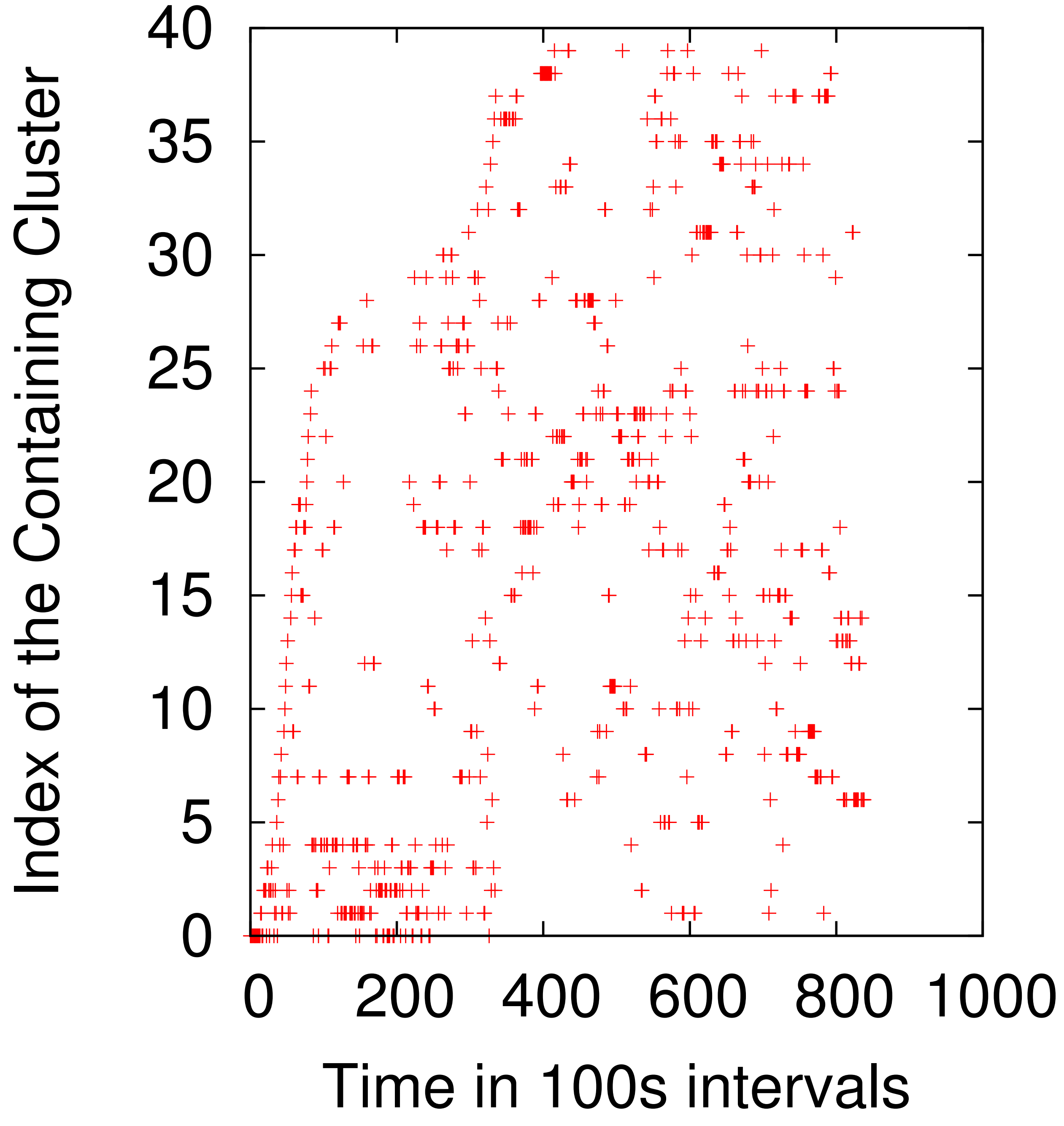
[From Albert Greenberg keynote at SIGCOMM 2015: http://conferences.sigcomm.org/sigcomm/2015/pdf/papers/keynote.pdf]

Motivating Environmental Characteristics

Increasing internal traffic is a bottleneck

- Traffic volume between servers is 4x external traffic

Unpredictable, rapidly-changing traffic matrices (TMs)



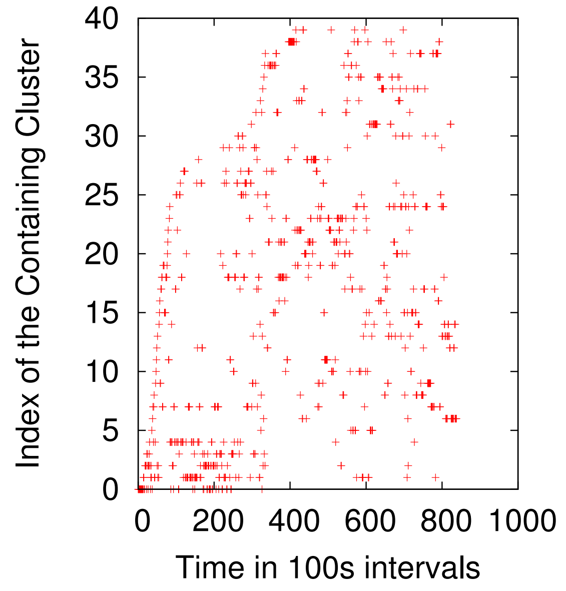
[Greenberg et al.]

Motivating Environmental Characteristics

Increasing internal traffic is a bottleneck

- Traffic volume between servers is 4x external traffic

Unpredictable, rapidly-changing traffic matrices (TMs)



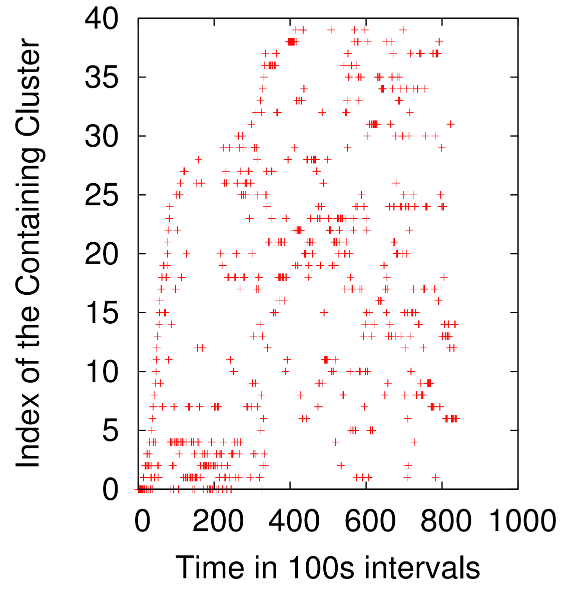
[Greenberg et al.]

Motivating Environmental Characteristics

Increasing internal traffic is a bottleneck

- Traffic volume between servers is 4x external traffic

Unpredictable, rapidly-changing traffic matrices (TMs)



[Greenberg et al.]

Design result: Nonblocking fabric

- High throughput for *any* TM that respects server NIC rates

Motivating Environmental Characteristics

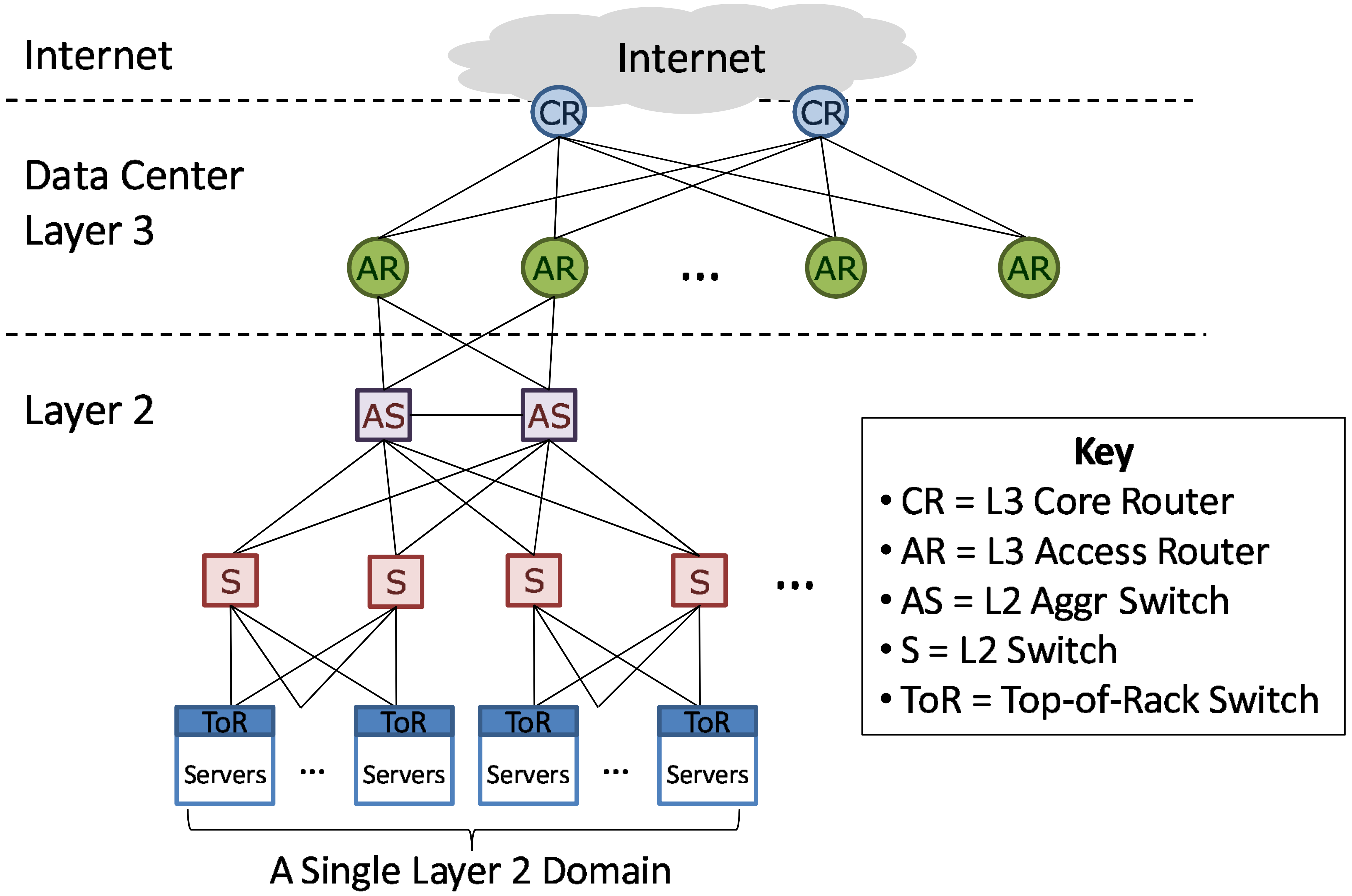
Failure characteristics

- Analyzed 300K alarm tickets, 36M error events
- 0.4% of failures were resolved in over one day
- 0.3% of failures eliminated all redundancy in a device group (e.g. both uplinks)

Design result: Clos topology

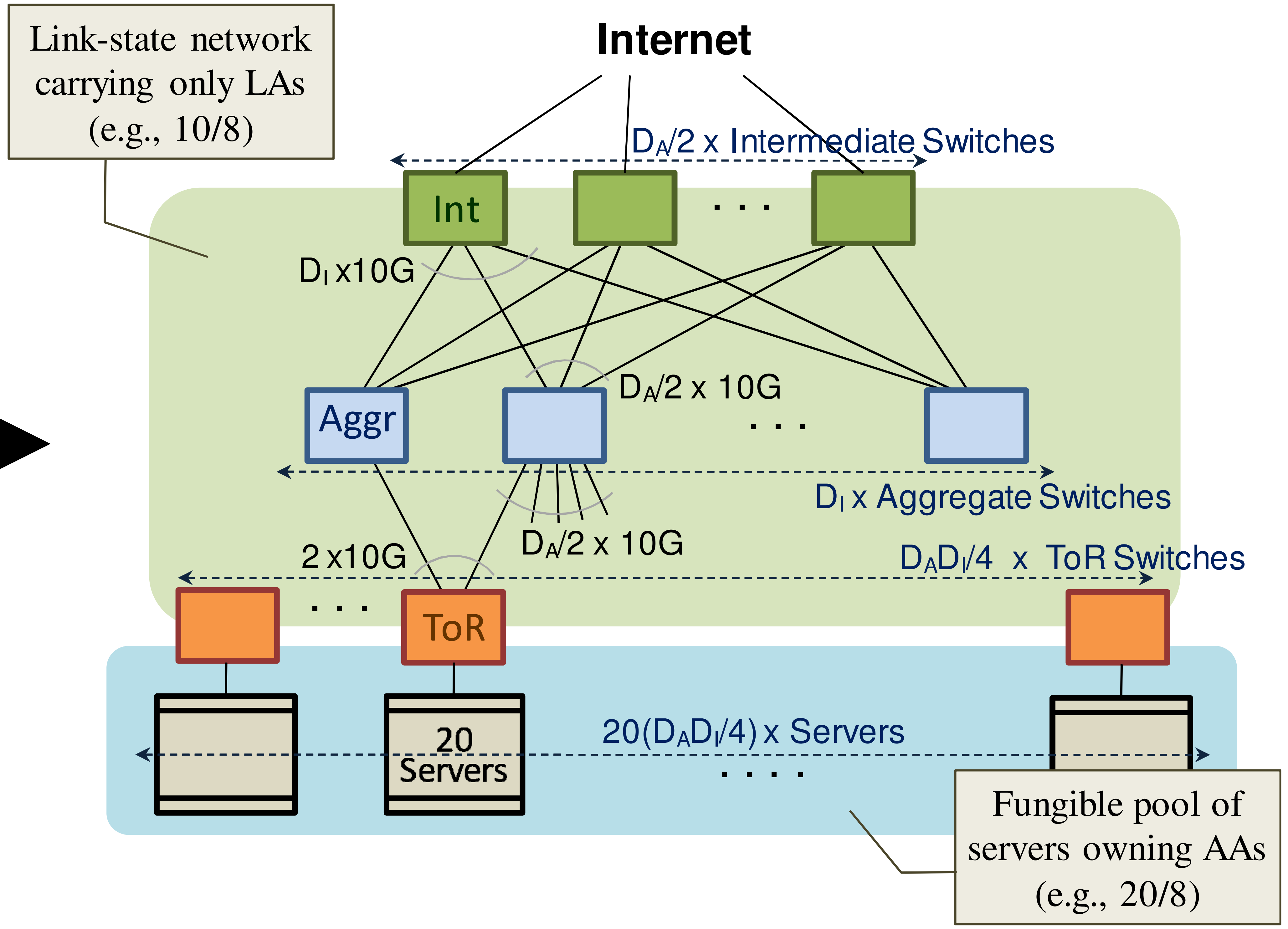
- "Scale out" instead of "scale up"

VL2 physical topology



Key
- CR = L3 Core Router
- AR = L3 Access Router
- AS = L2 Aggr Switch
- S = L2 Switch
- ToR = Top-of-Rack Switch

Traditional


Link-state network carrying only LAs (e.g., 10/8)
Internet
$D_A/2$ x Intermediate Switches
Int
. . .
D_I x10G
$D_A/2$ x 10G
Aggr
. . .
D_I x Aggregate Switches
2 x10G
$D_A/2$ x 10G
$D_AD_I/4$ x ToR Switches
. . .
ToR
20 Servers
$20(D_AD_I/4)$ x Servers
. . . .
Fungible pool of servers owning AAs (e.g., 20/8)

VL2

[Figures from Greenberg et al.]

Routing in VL2

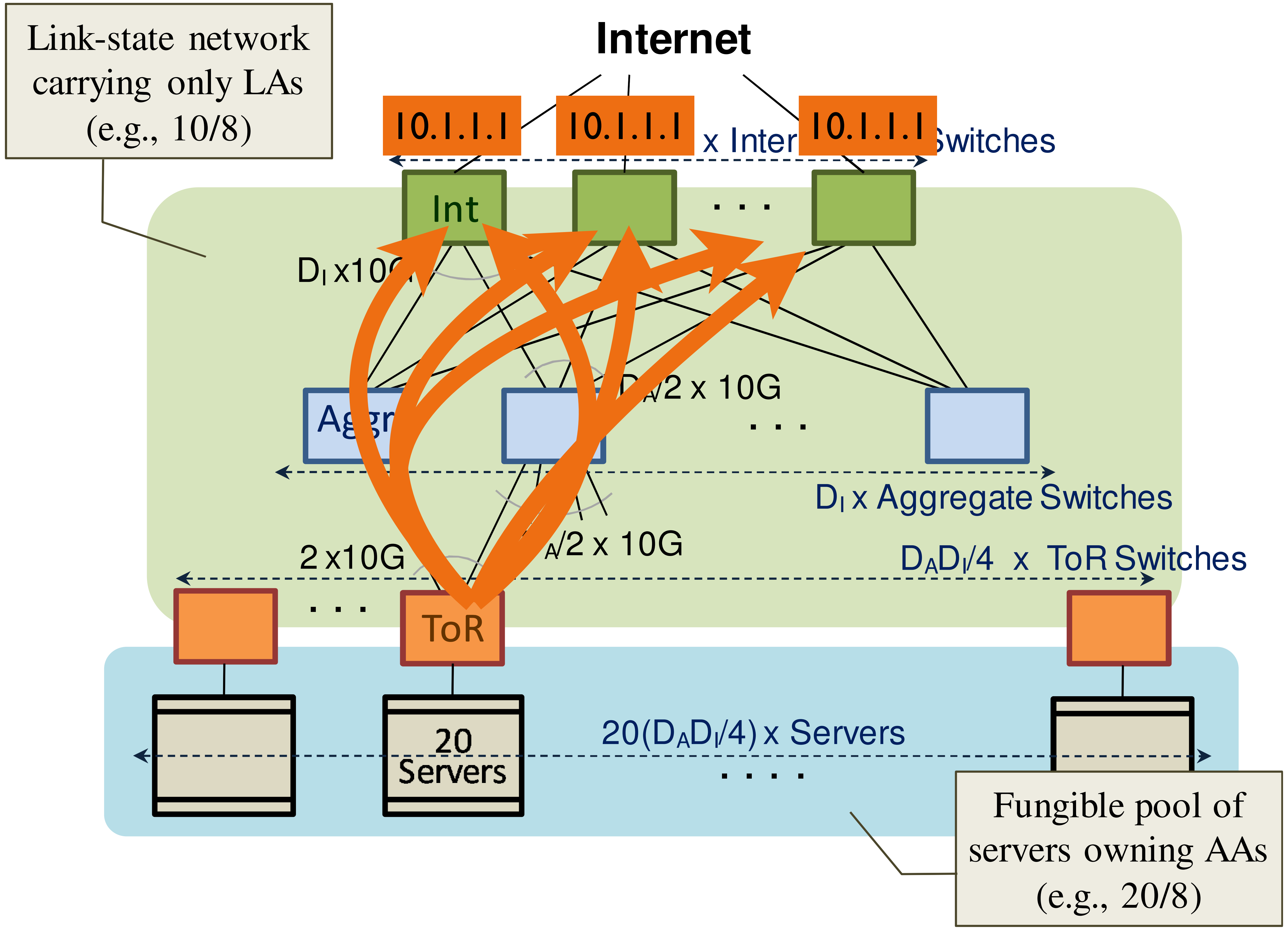
Unpredictable traffic

- Difficult to adapt

Design result: "Valiant Load Balancing" (at least as inspiration)

- Route traffic independent of current traffic matrix
- Spreads arbitrary traffic pattern so it's uniform among top layer switches

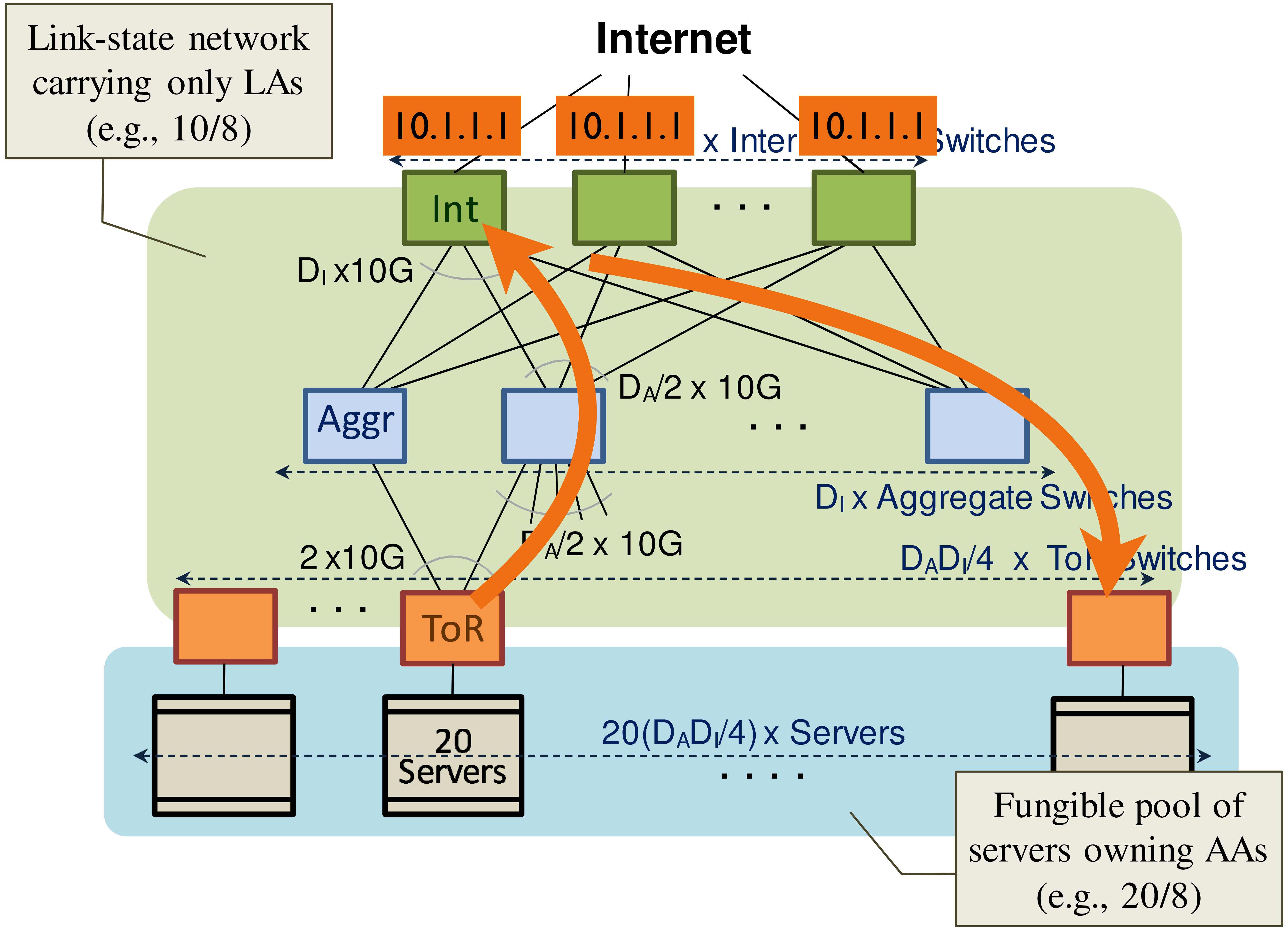
Routing Implementation



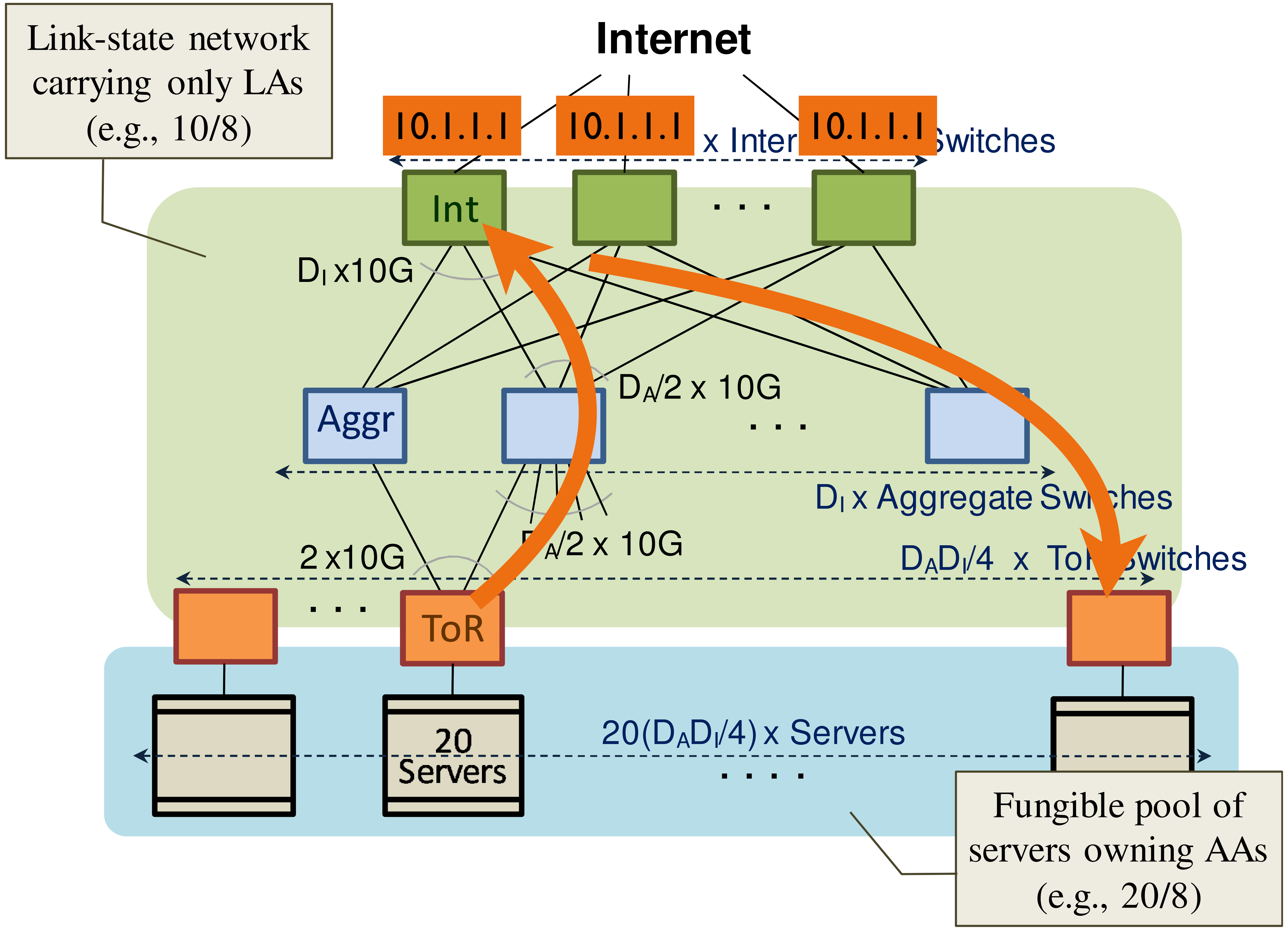
Link-state network carrying only LAs (e.g., 10/8)

Fungible pool of servers owning AAs (e.g., 20/8)

Routing Implementation



Routing Implementation



Link-state network carrying only LAs (e.g., 10/8)

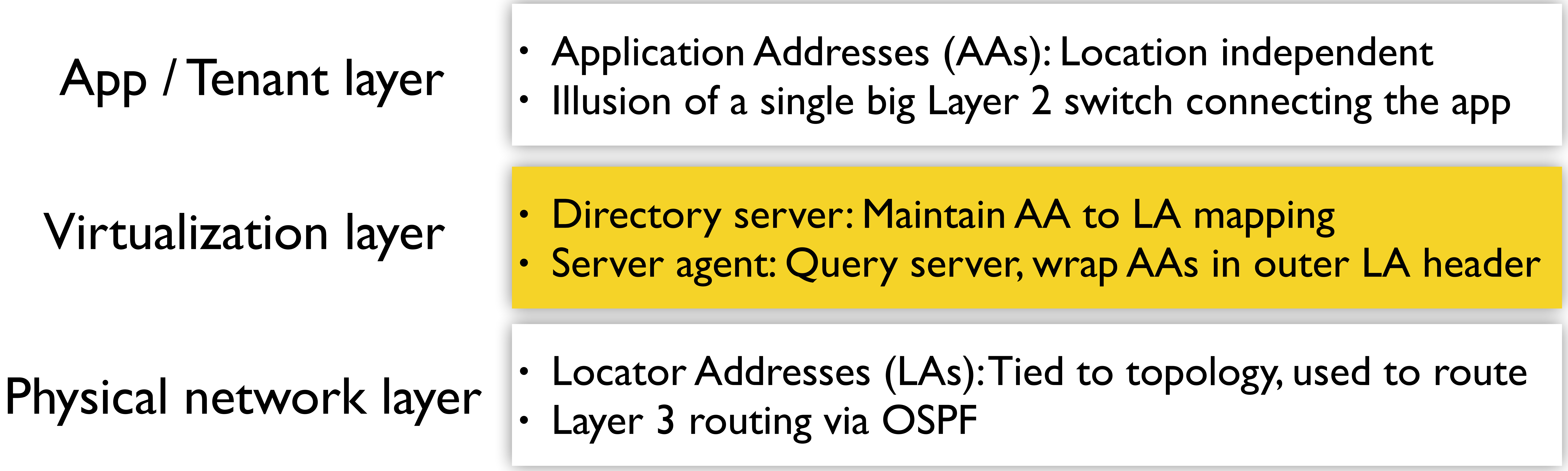
Fungible pool of servers owning AAs (e.g., 20/8)

Similar effect to ECMP to each rack

Smaller forwarding tables at most switches

Virtualization

> “All problems in computer science can be solved by another level of indirection.”
>
> – *David Wheeler*



App / Tenant layer

- Application Addresses (AAs): Location independent
- Illusion of a single big Layer 2 switch connecting the app

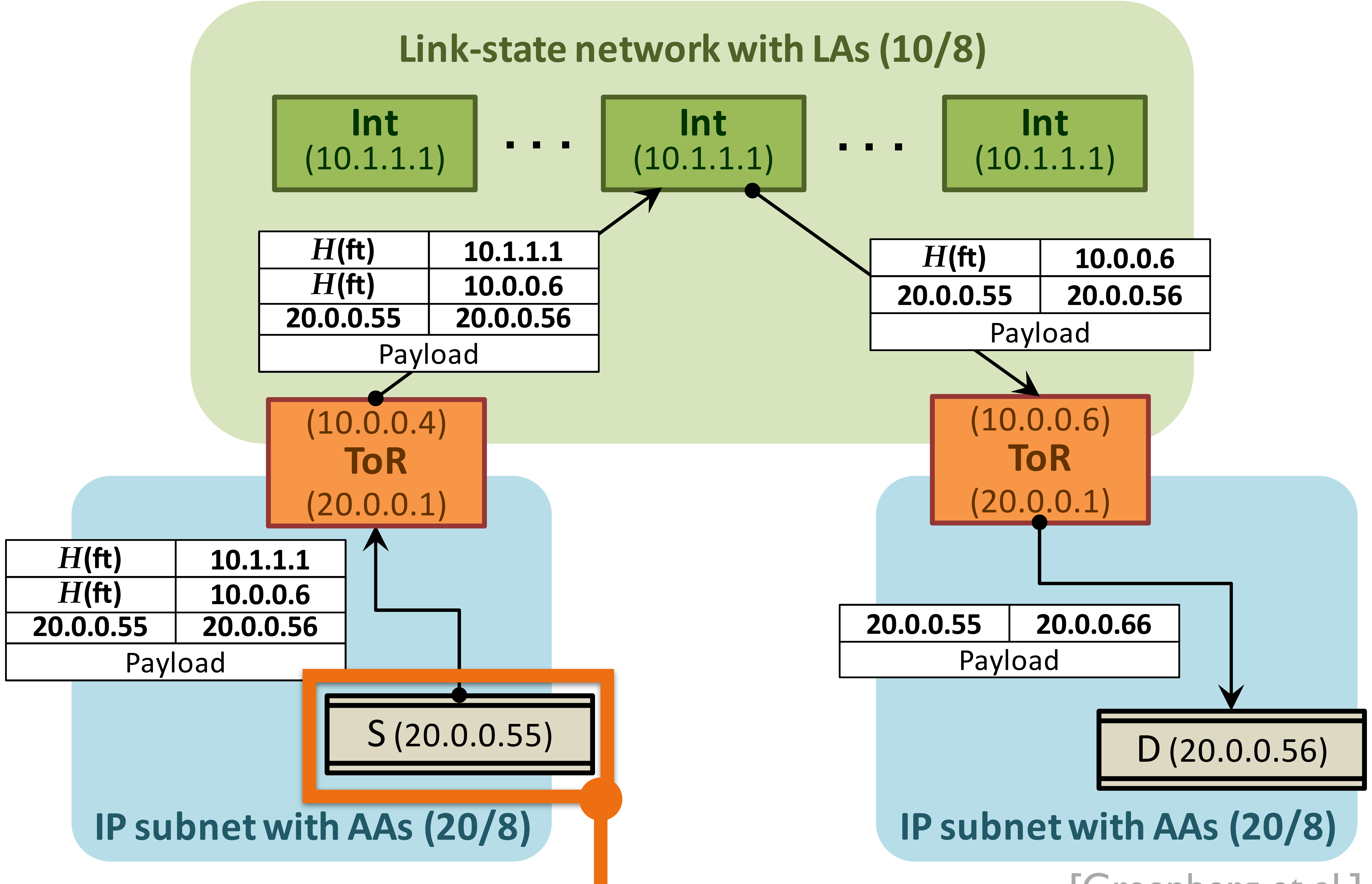
Virtualization layer

- Directory server: Maintain AA to LA mapping
- Server agent: Query server, wrap AAs in outer LA header

Physical network layer

- Locator Addresses (LAs): Tied to topology, used to route
- Layer 3 routing via OSPF

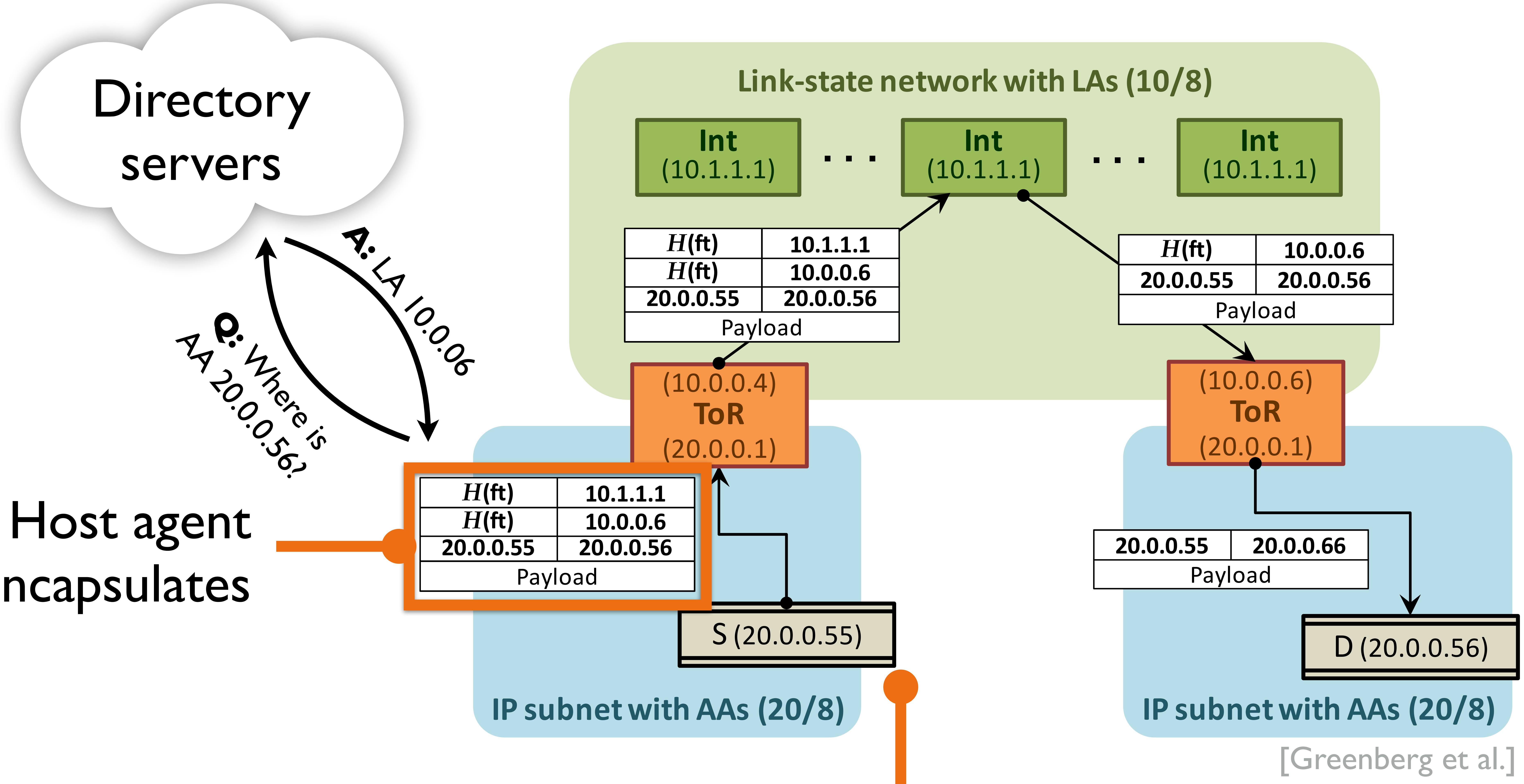
End-to-end example



[Greenberg et al.]

Application sends to AA 20.0.0.56

End-to-end example


Directory servers
A: LA 10.0.06
Q: Where is AA 20.0.0.56?
Link-state network with LAs (10/8)
Int (10.1.1.1)
. . .
Int (10.1.1.1)
. . .
Int (10.1.1.1)
(10.0.0.4) ToR (20.0.0.1)
(10.0.0.6) ToR (20.0.0.1)
S (20.0.0.55)
D (20.0.0.56)
IP subnet with AAs (20/8)
IP subnet with AAs (20/8)

H(ft)	10.1.1.1
H(ft)	10.0.0.6
20.0.0.55	20.0.0.56
Payload	

H(ft)	10.1.1.1
H(ft)	10.0.0.6
20.0.0.55	20.0.0.56
Payload	

H(ft)	10.0.0.6
20.0.0.55	20.0.0.56
Payload	

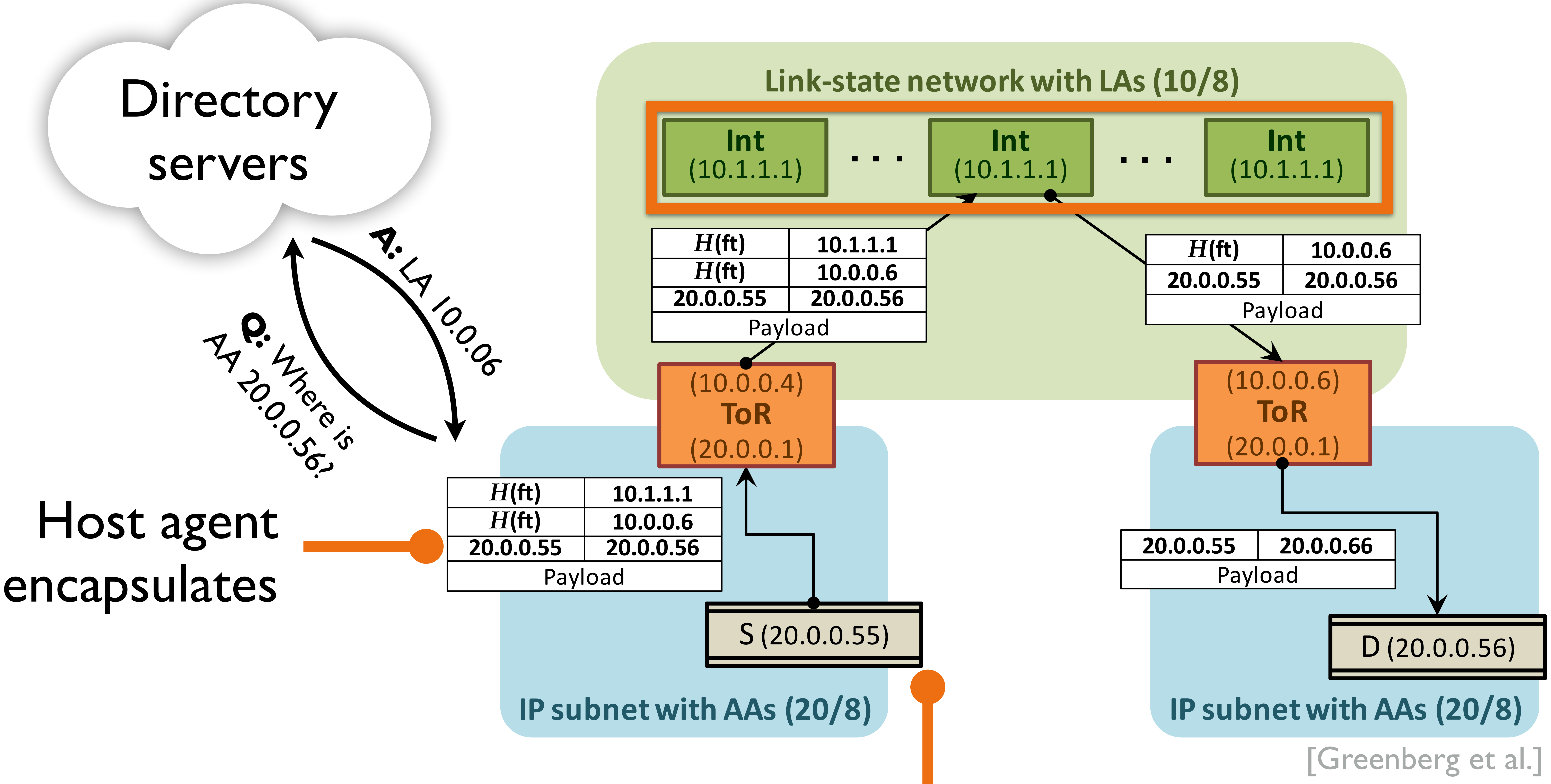
20.0.0.55	20.0.0.66
Payload	

Host agent encapsulates

Application sends to AA 20.0.0.56

[Greenberg et al.]

End-to-end example

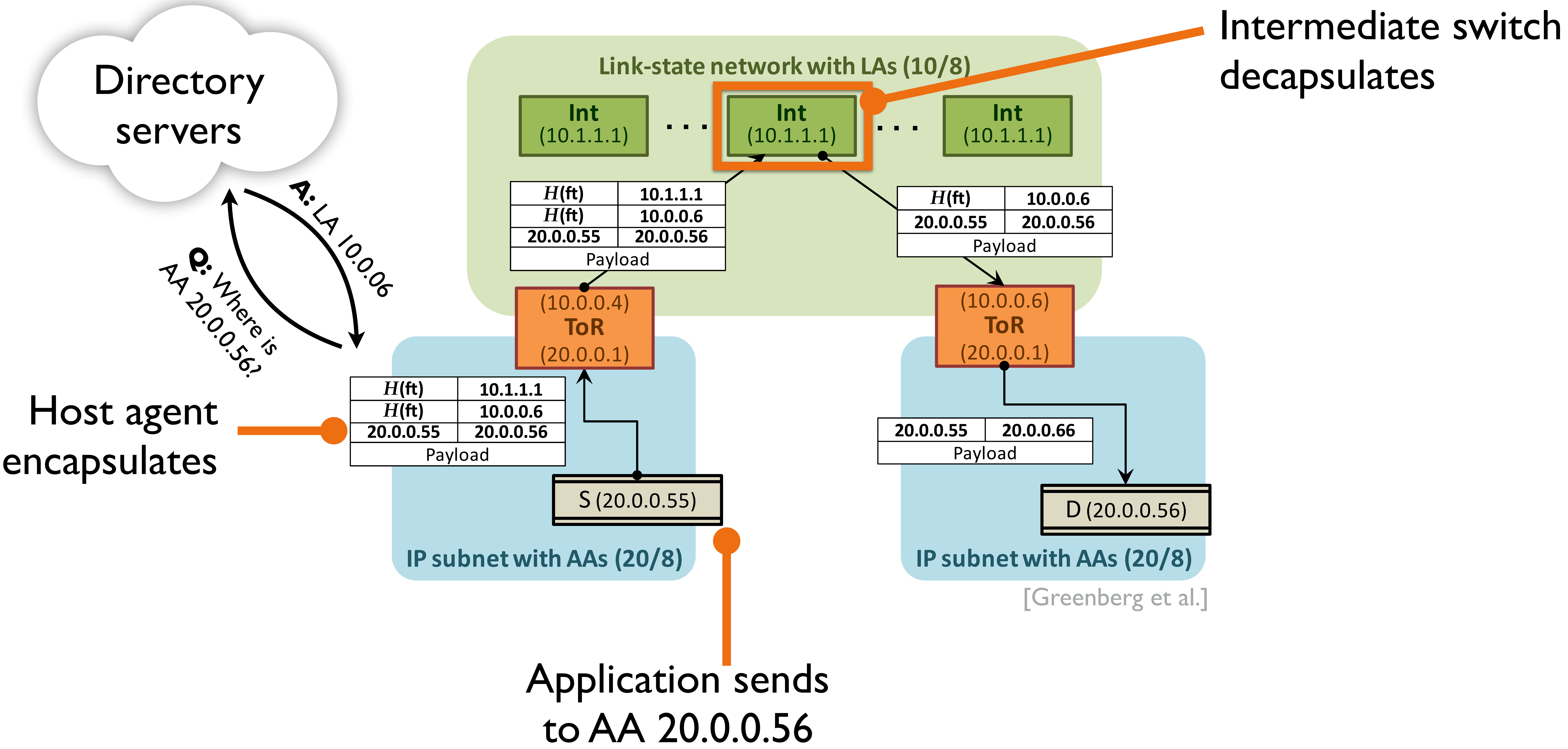


Host agent encapsulates

Application sends to AA 20.0.0.56

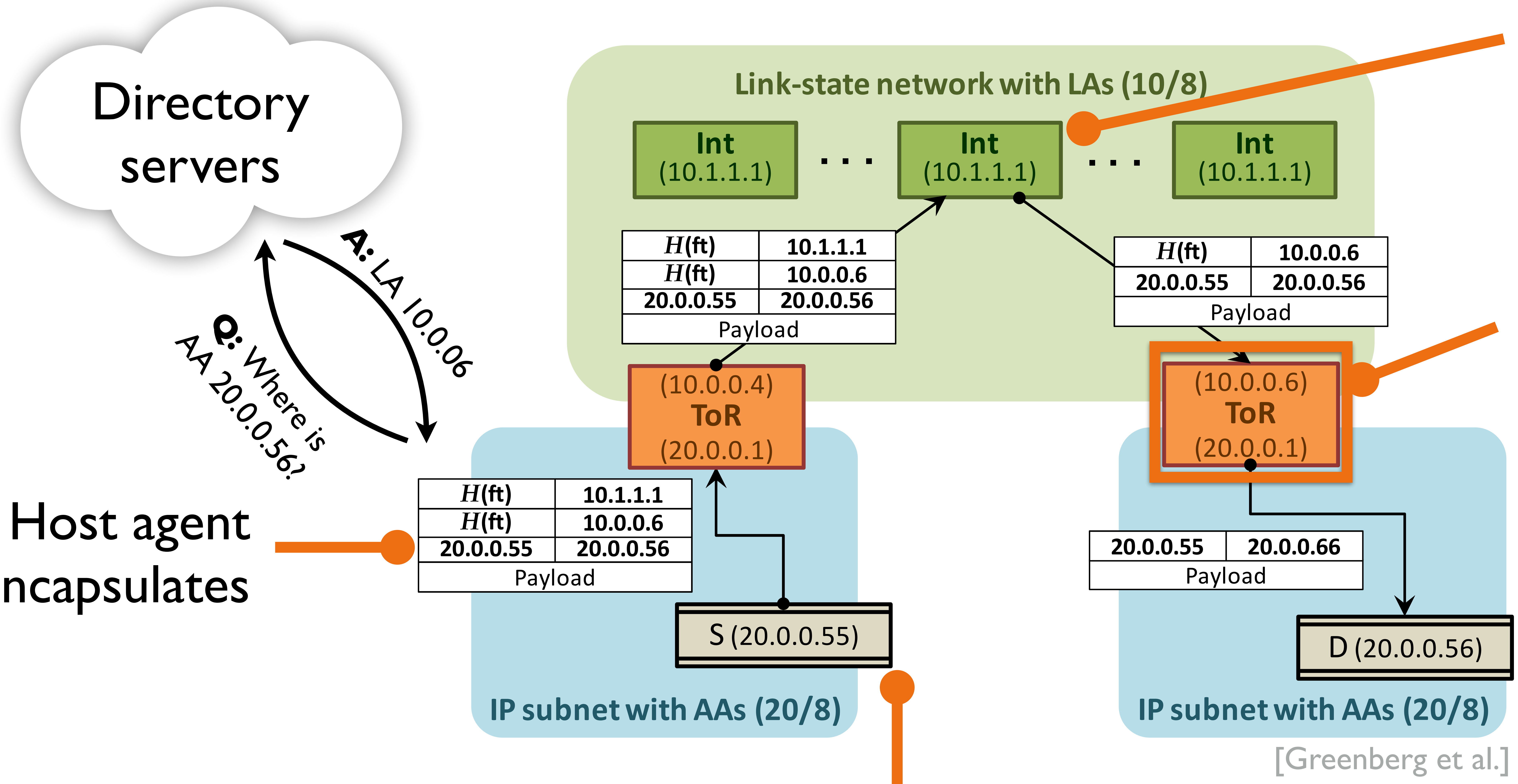
[Greenberg et al.]

End-to-end example



[Greenberg et al.]

End-to-end example



[Greenberg et al.]

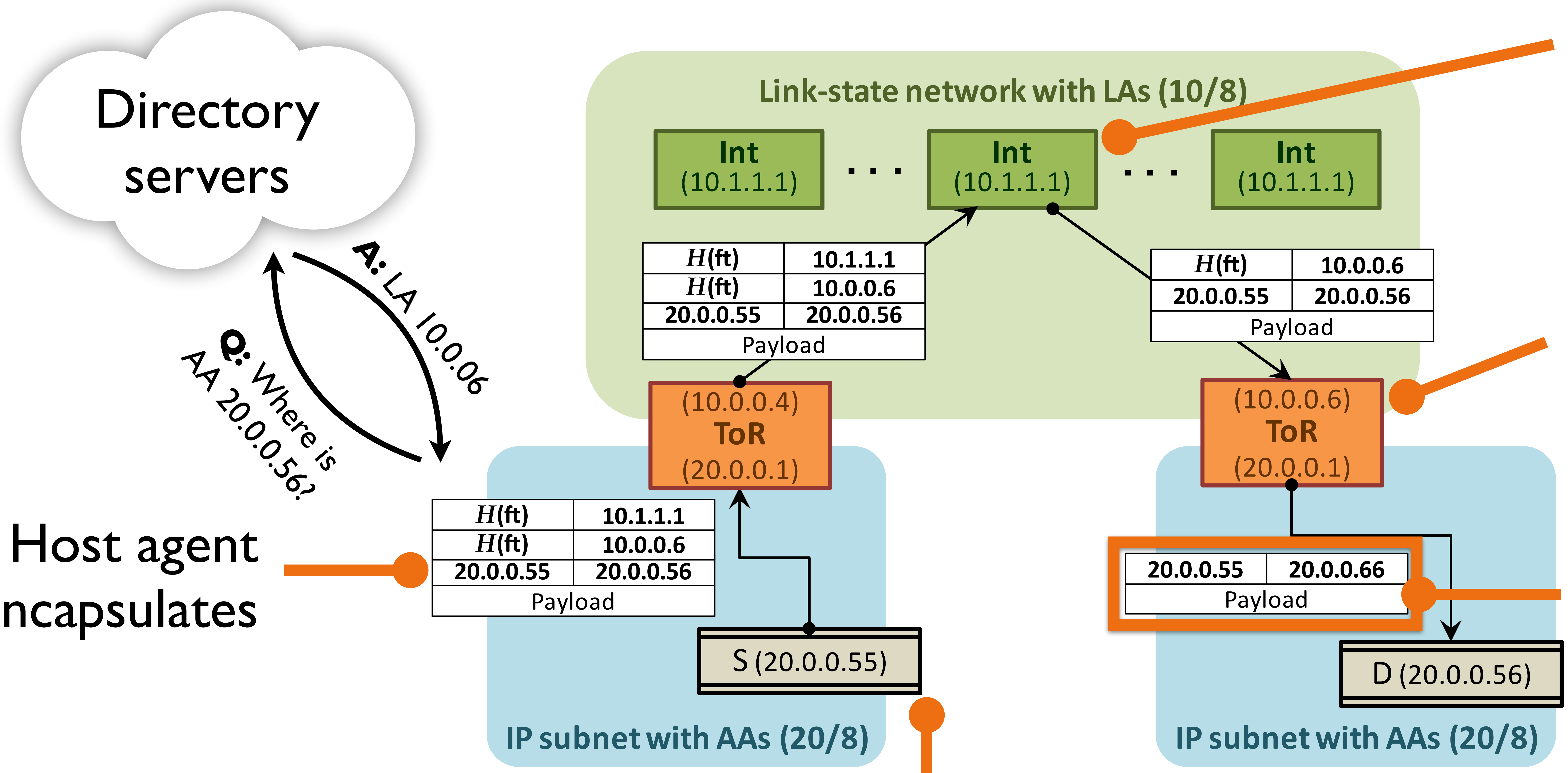
Intermediate switch decapsulates

Destination ToR decapsulates again

Host agent encapsulates

Application sends to AA 20.0.0.56

End-to-end example



H(ft)	10.1.1.1
H(ft)	10.0.0.6
20.0.0.55	20.0.0.56
Payload	

H(ft)	10.1.1.1
H(ft)	10.0.0.6
20.0.0.55	20.0.0.56
Payload	

H(ft)	10.0.0.6
20.0.0.55	20.0.0.56
Payload	

20.0.0.55	20.0.0.66
Payload	

[Greenberg et al.]

Application sends to AA 20.0.0.56

Host agent encapsulates

Intermediate switch decapsulates

Destination ToR decapsulates again

Host agent delivers

Did we achieve agility?

Location independent addressing

- AAs are location independent

L2 network semantics

- Agent intercepts and handles L2 broadcast, multicast

- Both of the above require "layer 2.5" shim agent running on host; but, concept transfers to hypervisor-based virtual switch

Did we achieve agility?

Performance uniformity

- Clos network is nonblocking (non-oversubscribed)
- Uniform capacity everywhere
- ECMP provides decent (but far from perfect) load balance
- But, performance isolation among tenants depends on TCP backing off to rate destination can receive
- Leaves open the possibility of better load balancing

Security

- Directory system can allow/deny connections by choosing whether to resolve an AA to a LA
- But, segmentation not explicitly enforced at hosts

Where's the SDN?

Directory servers: Logically centralized control

- Orchestrate application locations
- Control communication policy

Host agents: dynamic "programming" of data path

VL2 Enduring Take-Aways

Scale-out nonblocking Clos network

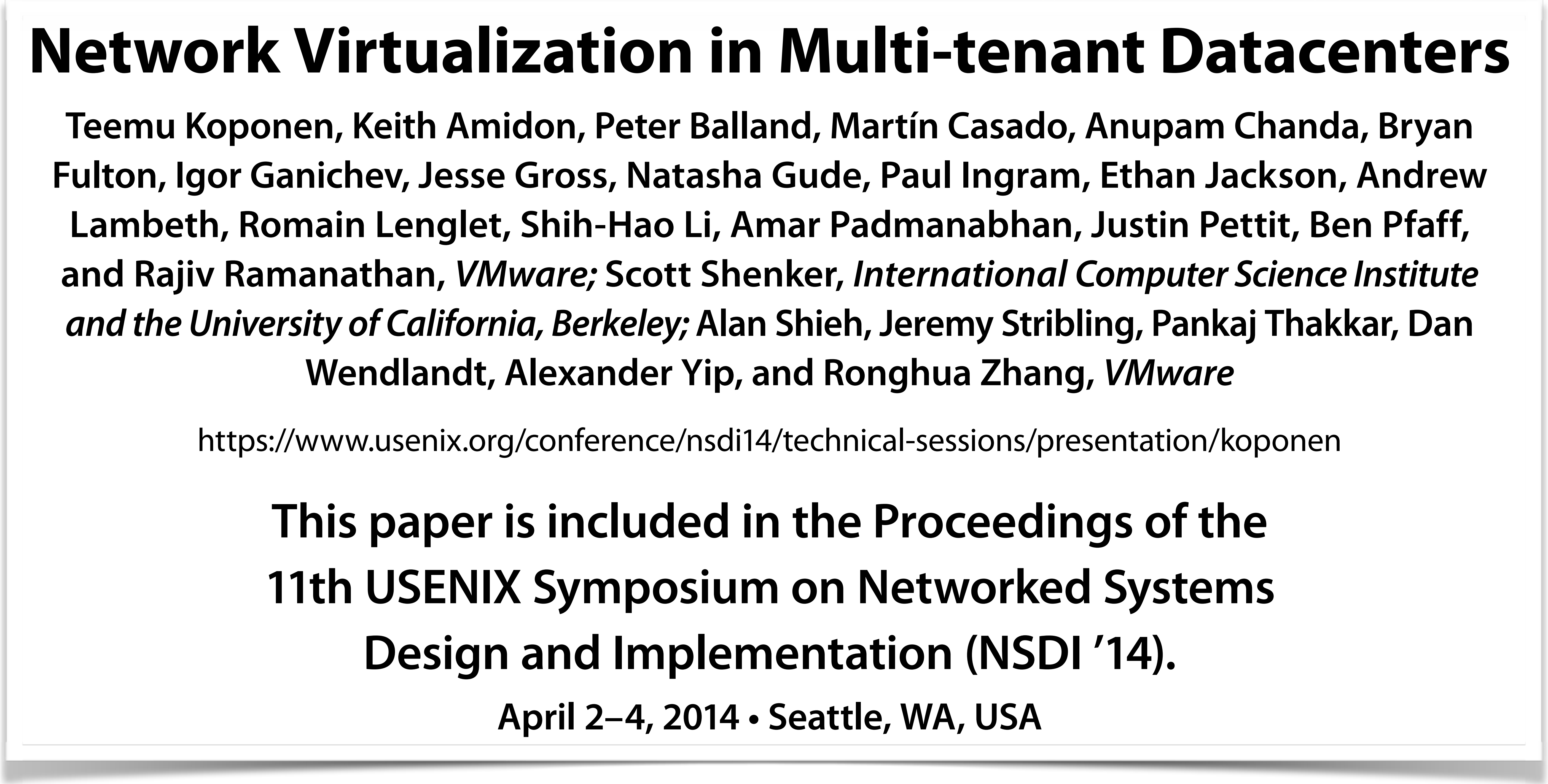
ECMP for traffic-oblivious routing

Separation of virtual and physical addresses

Centralized control plane

Network Virtualization
Case Study: NVP

Case Study: NVP



Network Virtualization in Multi-tenant Datacenters

Teemu Koponen, Keith Amidon, Peter Balland, Martín Casado, Anupam Chanda, Bryan Fulton, Igor Ganichev, Jesse Gross, Natasha Gude, Paul Ingram, Ethan Jackson, Andrew Lambeth, Romain Lenglet, Shih-Hao Li, Amar Padmanabhan, Justin Pettit, Ben Pfaff, and Rajiv Ramanathan, *VMware;* Scott Shenker, *International Computer Science Institute and the University of California, Berkeley;* Alan Shieh, Jeremy Stribling, Pankaj Thakkar, Dan Wendlandt, Alexander Yip, and Ronghua Zhang, *VMware*

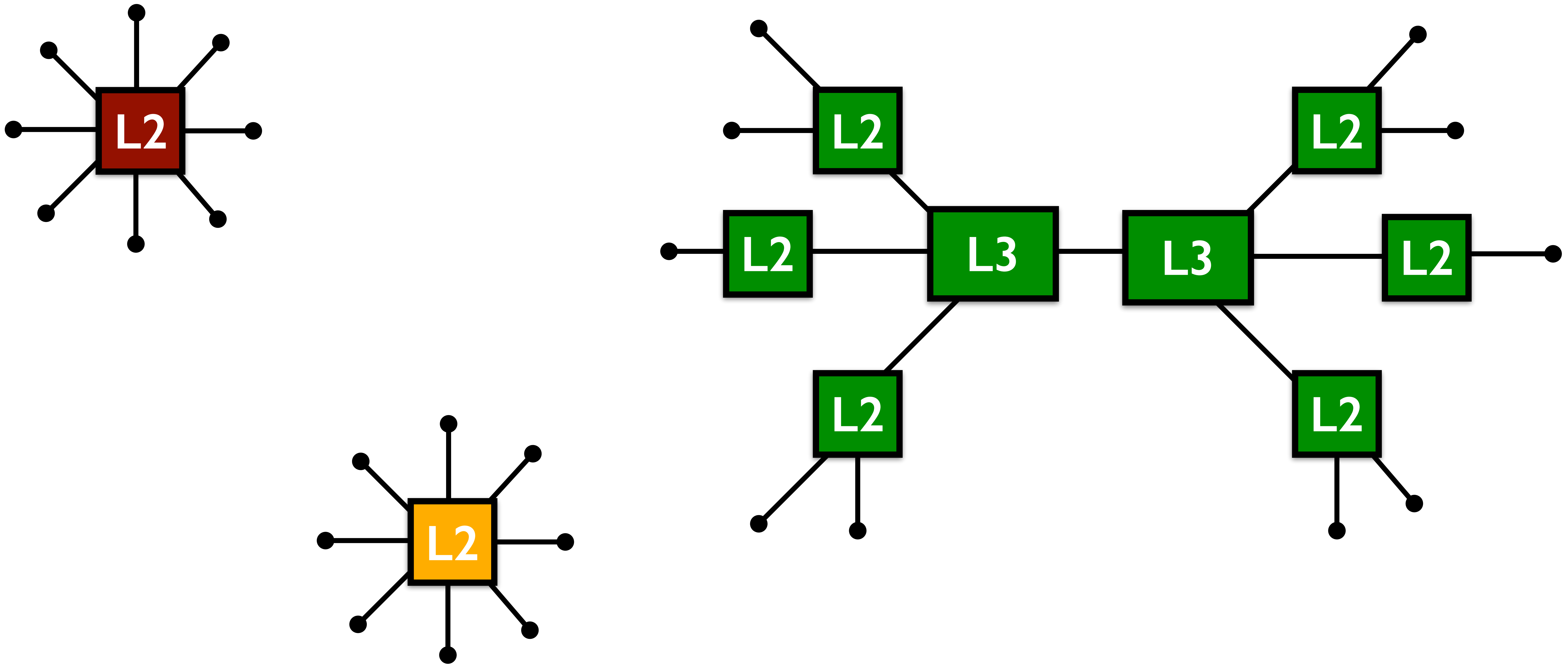
https://www.usenix.org/conference/nsdi14/technical-sessions/presentation/koponen

This paper is included in the Proceedings of the
11th USENIX Symposium on Networked Systems
Design and Implementation (NSDI '14).

April 2–4, 2014 • Seattle, WA, USA

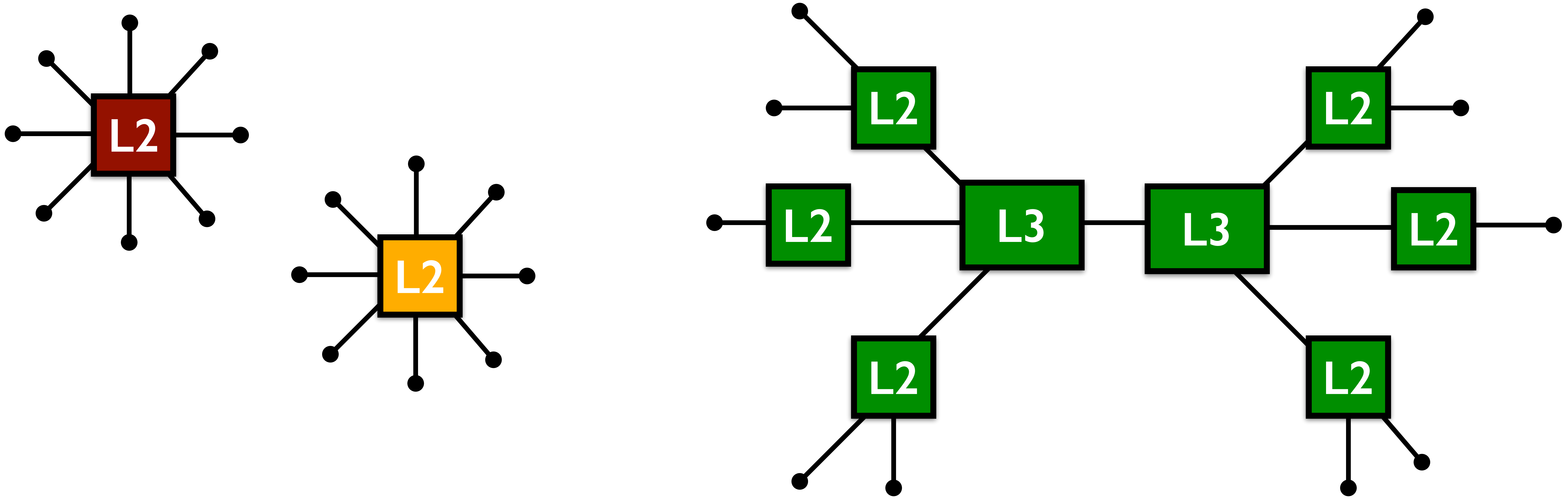
NVP Approach to Virtualization

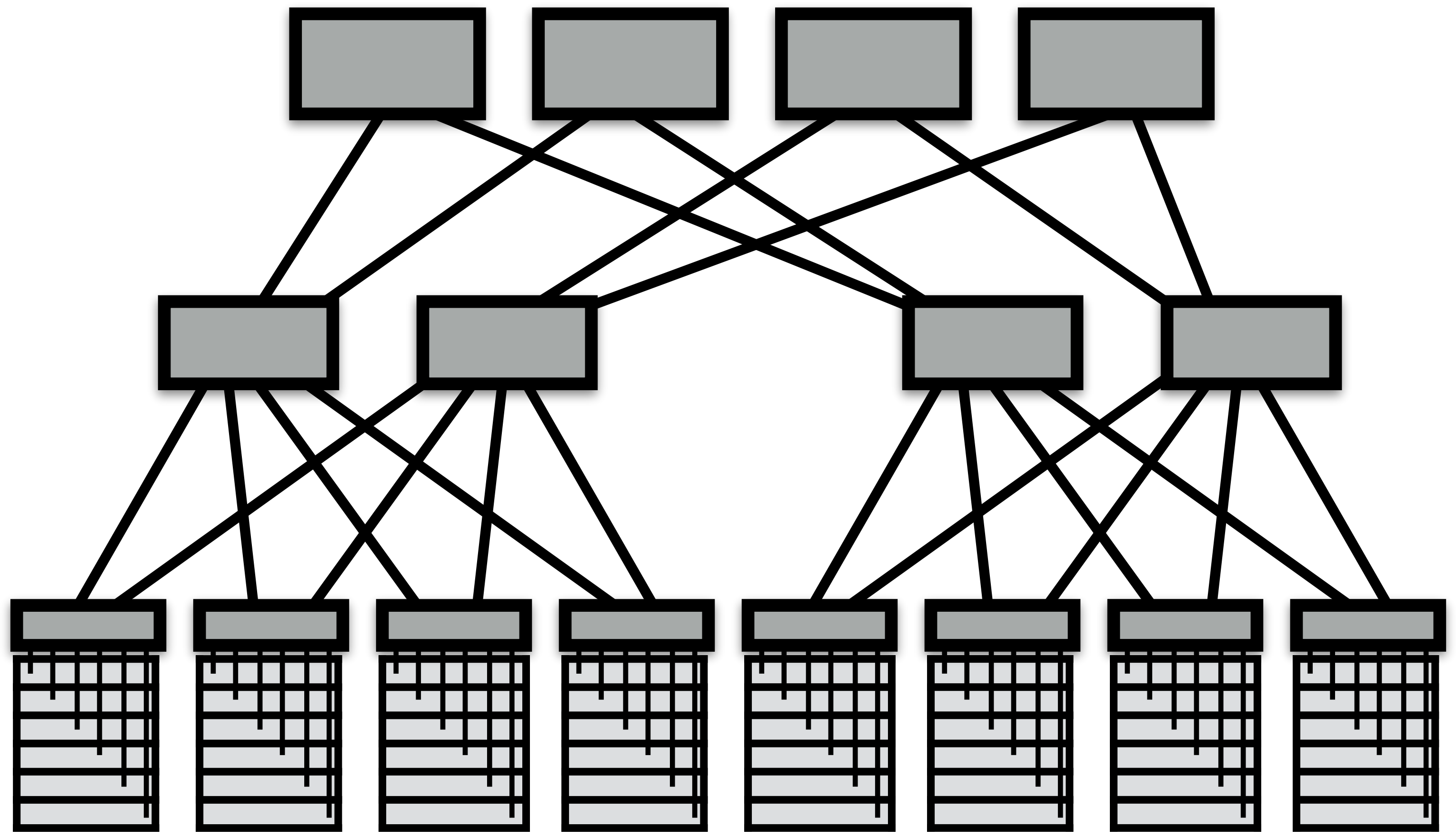
1. Service: Arbitrary network topology



NVP Approach to Virtualization

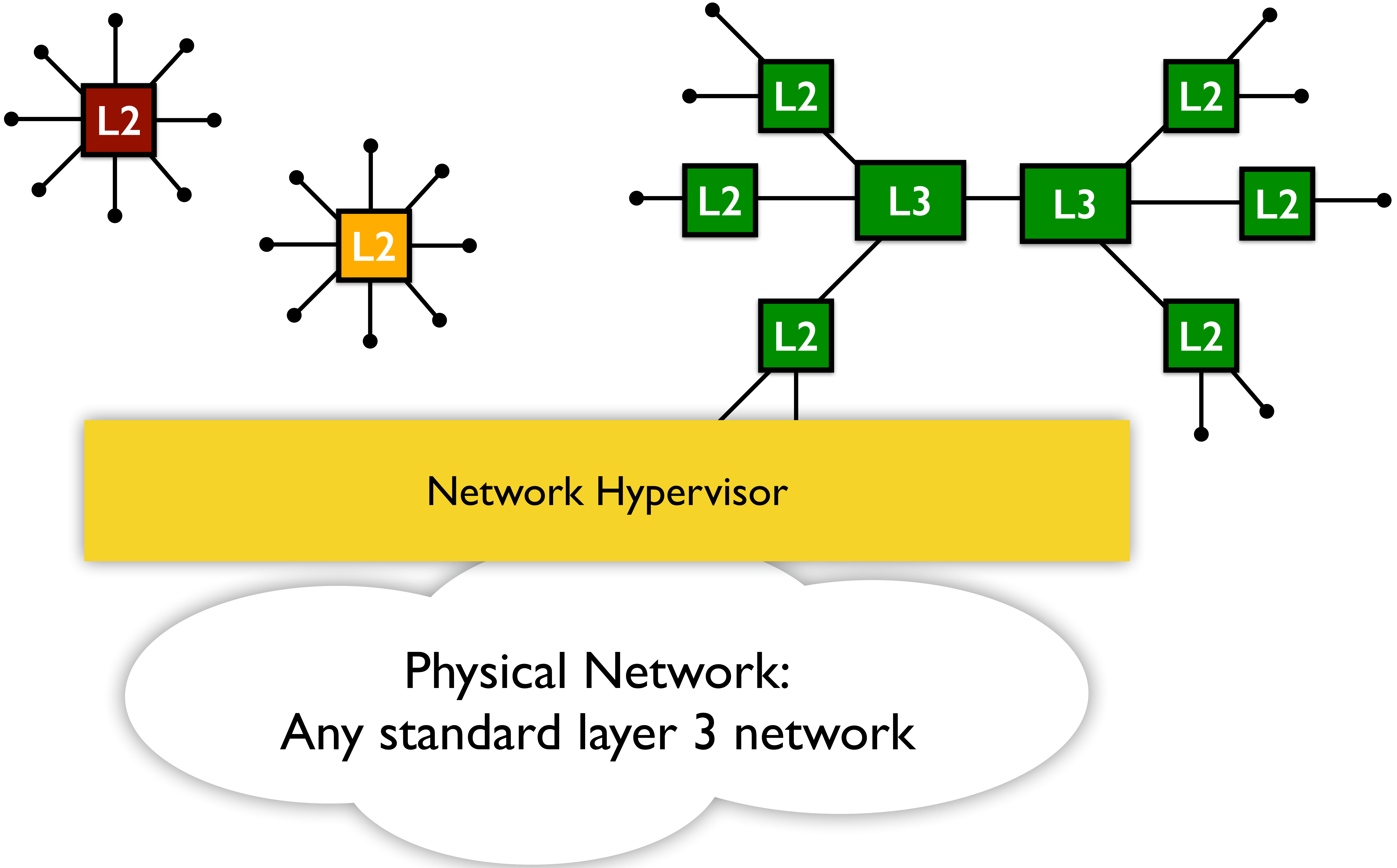
1. Service: Arbitrary network topology





NVP Approach to Virtualization

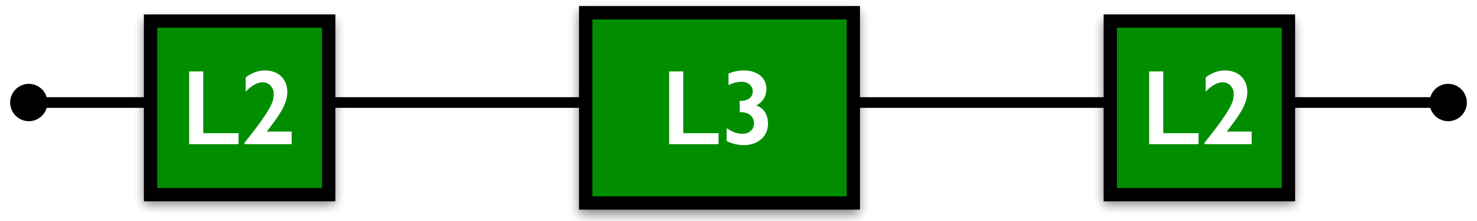
Service: Arbitrary network topology



Network Hypervisor

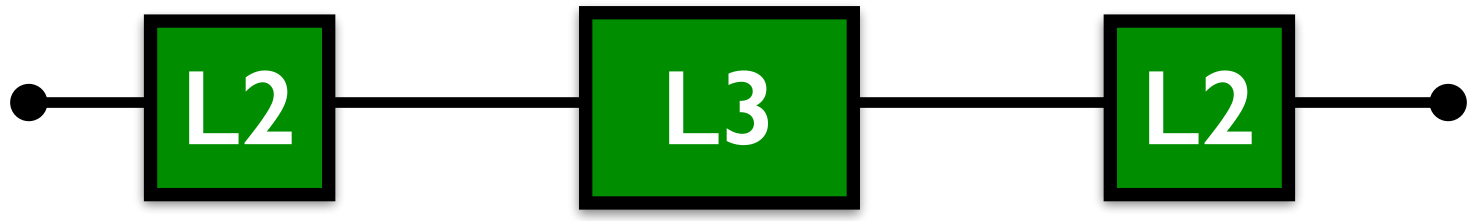
Physical Network:
Any standard layer 3 network

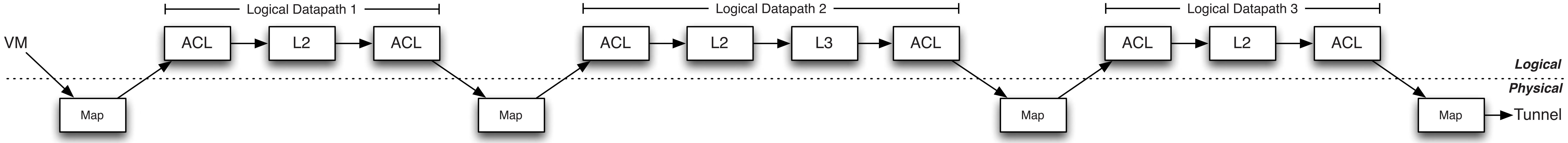
Virtual network service



[Figure: Koponen et al.]

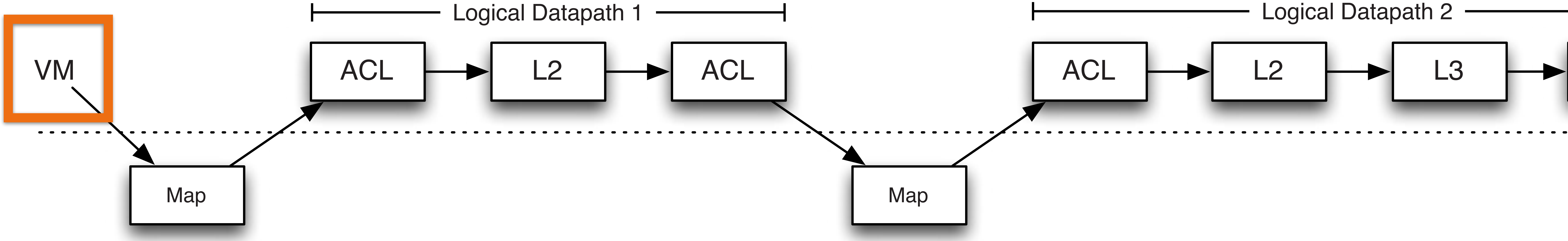
Virtual network service



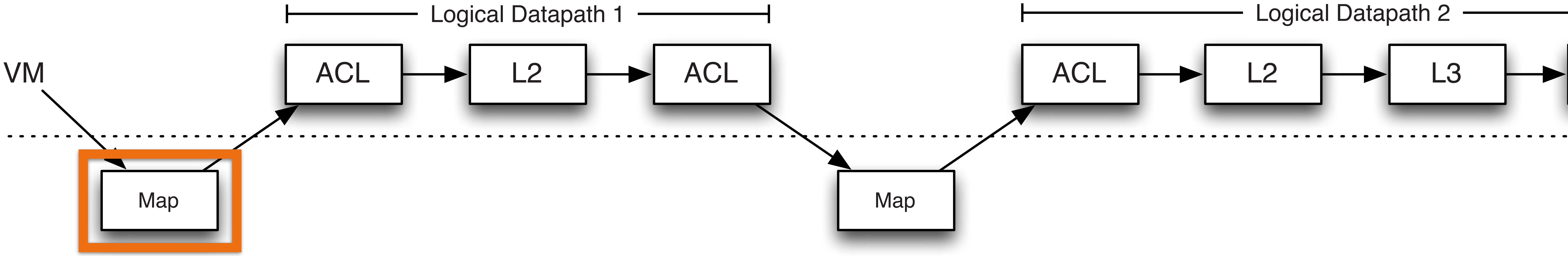


[Figure: Koponen et al.]

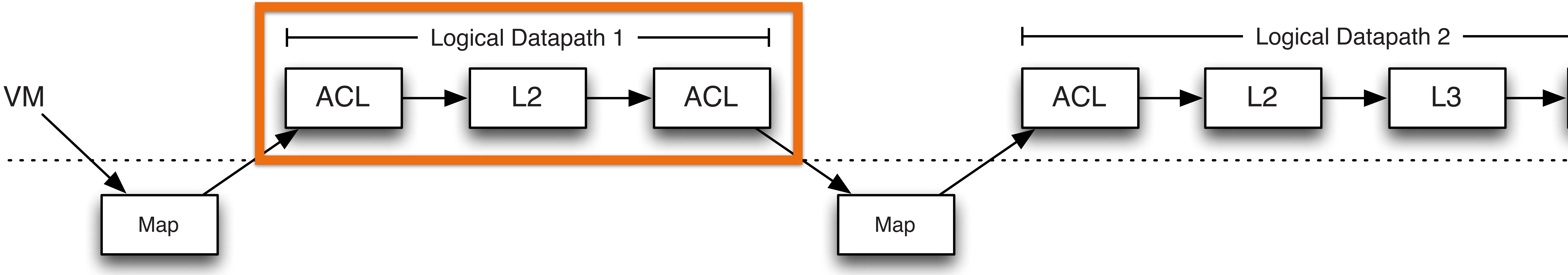
Virtual network service



Virtual network service



Virtual network service



Virtual network service

Logical Datapath 1
Logical Datapath 2
VM
ACL
L2
ACL
ACL
L2
L3
ACL
Map
Map

Virtual network service

Logical Datapath 1
ACL
L2
ACL
Logical Datapath 2
ACL
L2
L3
ACL
ap
Map
Map

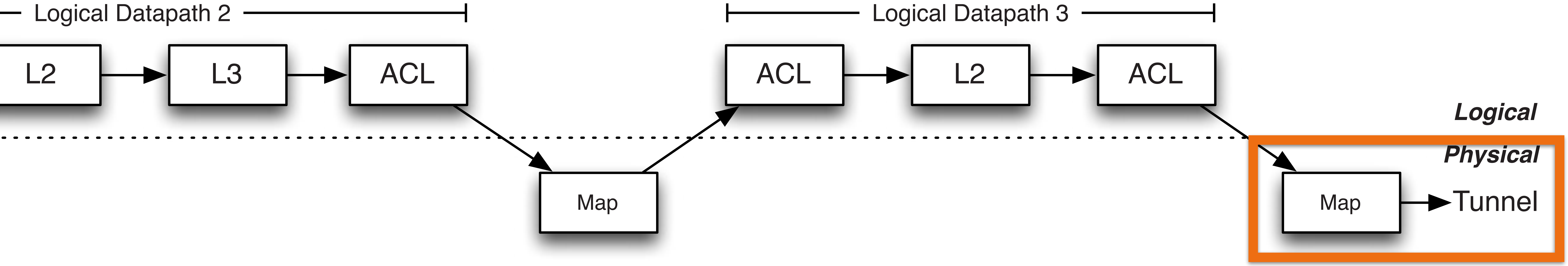
Virtual network service

Logical Datapath 2
Logical Datapath 3
ACL
L2
L3
ACL
ACL
L2
ACL
Map
Map

Virtual network service

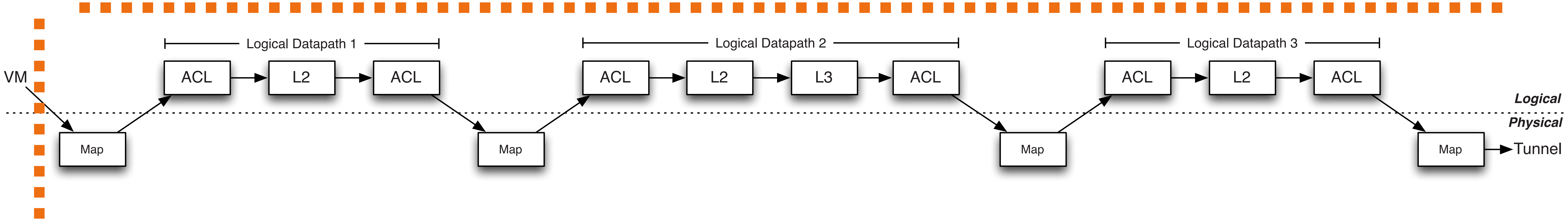
Logical Datapath 2
ACL
L2
L3
ACL
Map
Logical Datapath 3
ACL
L2
ACL
Logical
Physical
Map
Tunnel

Virtual network service

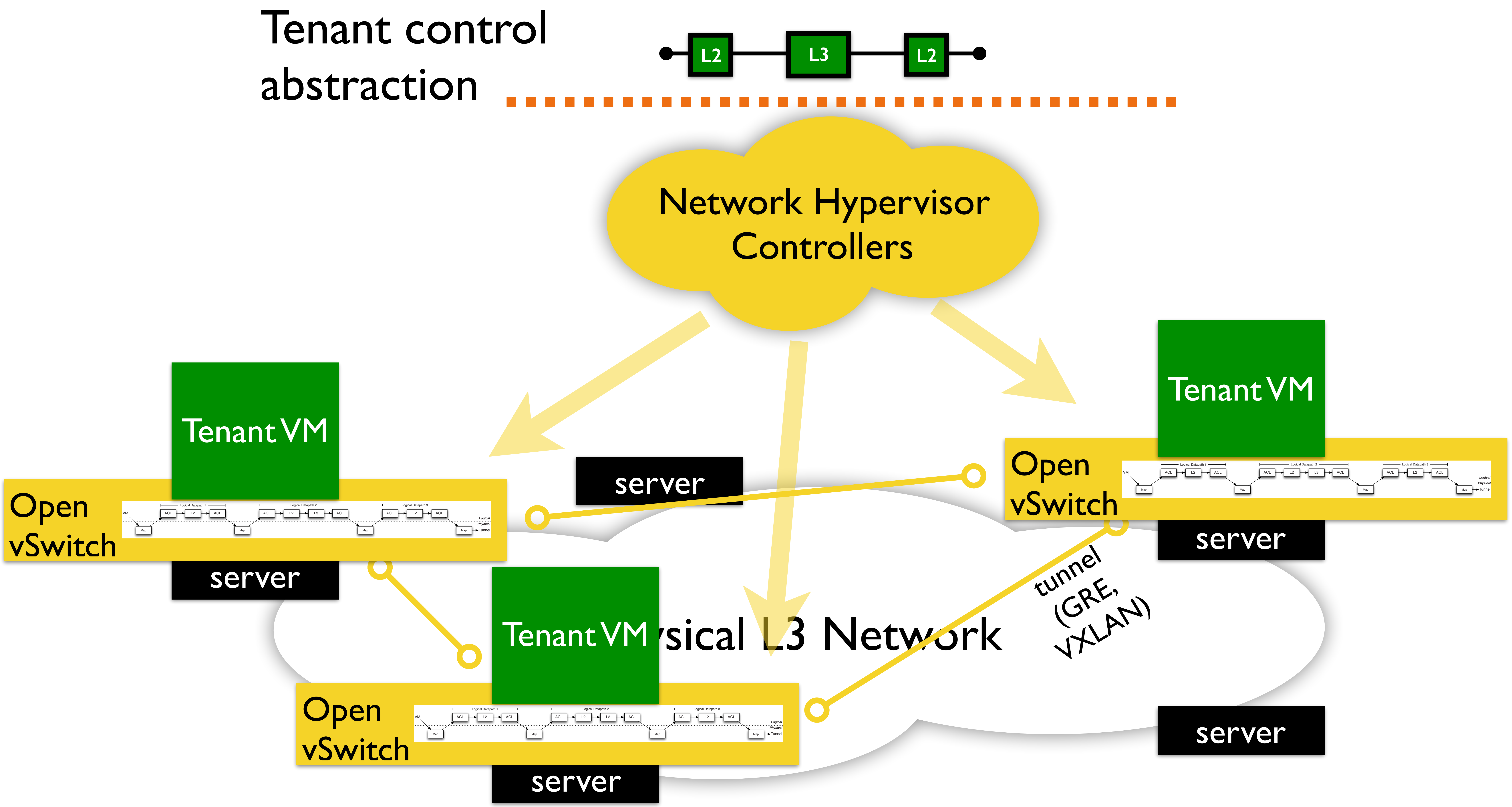


Virtual network service

Control abstraction
(sequence of OpenFlow flow tables)



Packet
abstraction


Tenant control abstraction
L2
L3
L2
Network Hypervisor Controllers
Tenant VM
Open vSwitch
server
Tenant VM
Open vSwitch
server
server
Tenant VM
Open vSwitch
server
ysical L3 Network
tunnel (GRE, VXLAN)
server

Challenge: Performance

Large amount of state to compute

- Full virtual network state at every host with a tenant VM!
- O(n^2) tunnels for tenant with *n* VMs
- Solution 1: Automated incremental state computation with *nlog* declarative language
- Solution 2: Logical controller computes single set of universal flows for a tenant, translated more locally by "physical controllers"

Challenge: Performance

Pipeline processing in virtual switch can be slow

- Solution: Send first packet of a flow through the full pipeline; thereafter, put an exact-match packet entry in the kernel

Tunneling interferes with TCP Segmentation Offload (TSO)

- NIC can't see TCP outer header
- Solution: STT tunnels adds "fake" outer TCP header

Discussion

Where's the SDN?

- API to data plane
- centralized controller
- control abstractions

Why was micro-segmentation a "killer app" for SDN?

- Needed to automate control of a dynamic, virtualized environment, not suited to manual solutions

How does it compare to wide-area control in B4?

Industry Impact

Multiple vendors with software-defined data center "micro-segmentation" products

- VMware's NSX
  - VMware claims more than 2,400 customers, $1B/yr sales
- Cisco's ACI
- Startups vArmour, Illumio

Next time

- Programmable switches