Kathará

data centers' routing

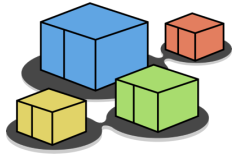| Version	3.1
Author(s)	L. Ariemma, G. Di Battista, M. Patrignani, M. Scazzariello, T. Caiazzi
E-mail	contact@kathara.org
Web	http://www.kathara.org/
Description	Data Centers' Routing: Fat-Trees, BGP

copyright notice

- all the pages/slides in this presentation, including but not limited to, images, photos, animations, videos, sounds, music, and text (hereby referred to as "material") are protected by copyright

- this material, with the exception of some multimedia elements licensed by other organizations, is property of the authors and/or organizations appearing in the first slide

- this material, or its parts, can be reproduced and used for didactical purposes within universities and schools, provided that this happens for non-profit purposes

- any other use is prohibited, unless explicitly authorized by the authors on the basis of an explicit agreement

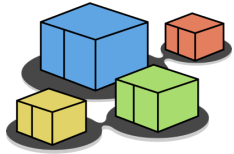- this copyright notice must always be redistributed together with the material, or its portions
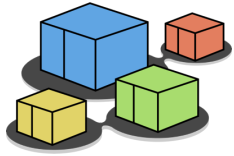
routing in Fat-Trees

why BGP?

overview – routing

- why routing and not switching?
- the choice of the routing protocols
- why using BGP in Fat-Tree data centers?
- inter-domain routing VS data center routing
- BGP in the data center

why routing and not switching?

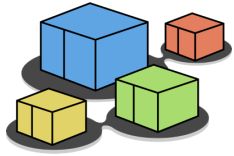|	Pros	Cons
Switching (L2)	No configuration needed	• STP protocol has no multipath support • Using VLANs for load balancing is tricky • Broadcast traffic flooding the network
Routing (L3)	Multipath support	• Need of routing protocols • Complex configuration • Need of automating the configuration

choosing a routing protocol

- there are many possibilities, among them
  - a classical IGP protocol (e.g., OSPF, IS-IS)
  - BGP (most used in Hyperscale data centers)
  - RIFT (Routing In Fat-Trees)
    - Under standardization at IETF
    - Designed for Fat-Trees topologies
  - OpenFabric
    - IS-IS variant created for Clos topologies
 - SDN Protocols
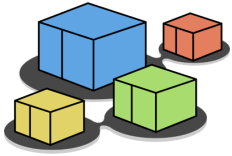
why using BGP in Fat-Trees?

- **BGP has several stable and robust implementations, even open-source**
  - e.g., FRRouting, Quagga
- **BGP generates less flooding than common IGPs (IS-IS, OSPF, etc.)**
  - if a received update does not change the best route, a BGP speaker does not propagate the update
- **BGP natively supports ECMP (Equal-Cost Multi-Path)**
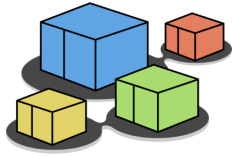 - Fat-Trees have many paths with the same length

why using eBGP? (and not iBGP)

- the most obvious choice would be to use iBGP since the data center networks is under the same administration

- however, eBGP is always used because
  - eBGP is easier to setup, no need for IGP
    - with iBGP, the IGP would compute routes
  - iBGP multipath support has some limitations
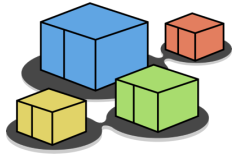   - can be overcame, but it is complex
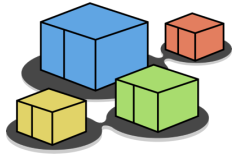
inter-domain routing VS DC routing

| inter-domain routing	data centers' routing
the internet has relatively sparse connectivity	data centers' networks have very dense connectivity
stability is preferred over quick convergence	quick convergence is preferred over stability
the aim is computing a single best path for each destination	the aim is computing multiple paths to each destination

BGP in the data center

- **BGP was primarily devised for inter-domain routing**
  - default configurations are not suitable in a data center
- **needs some tweaks described in RFC-7938**
  - AS numbers assignment
 - ECMP *policy relax*
  - timer adjustment
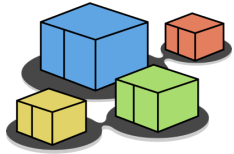
AS numbers assignment

- global ASNs are not used in the data center
  - they can be misleading
    - because operators associate them with names
  - they are dangerous
    - an accidental leakage of the internal BGP notifications to the internet may be disruptive
- private ASNs are generally used
  - the 2-byte ASNs allow only 1,023 private ASes in the range 64512–65534
  - the 4-byte ASNs support almost 95 million private ASes
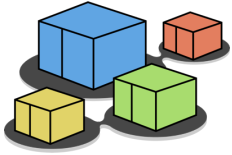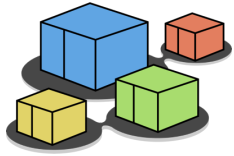
ASes and routers

- the most obvious choice would be assigning a different ASN to each node

- however, this approach would lead to BGP *path exploration* issues
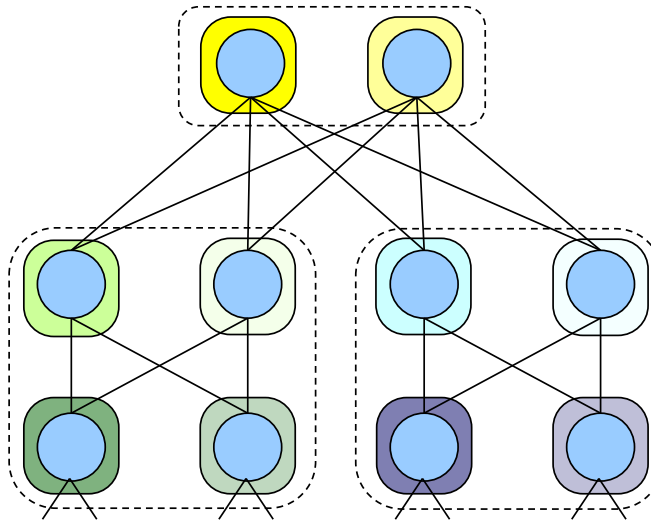
BGP path exploration

- **immediately after a fault there is a transient period when inconsistent AS-paths are propagated in the network**
  - routers have plenty of alternatives and jump from one to another before all alternatives are withdrawn
  - each best route change is propagated again by the routers
  - lots of useless BGP updates are transmitted
- **since data centers' topologies are dense and hence have a lot of cycles this problem has to be addressed**

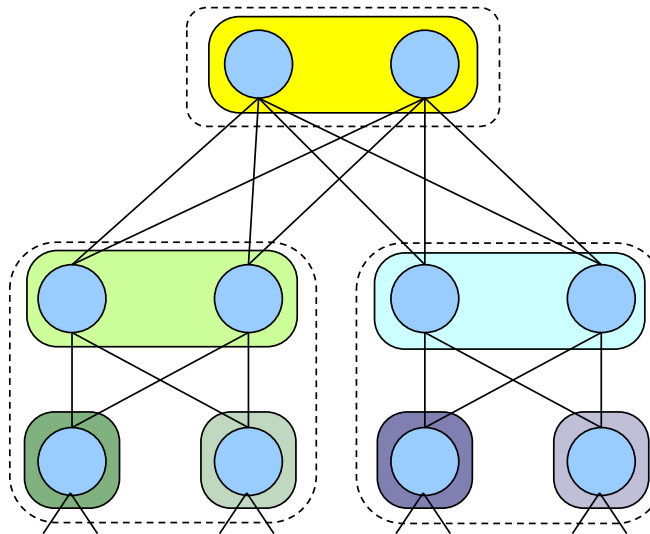# the problem of BGP path exploration

- because of the adopted AS scheme several fake routes could be possible during path exploration
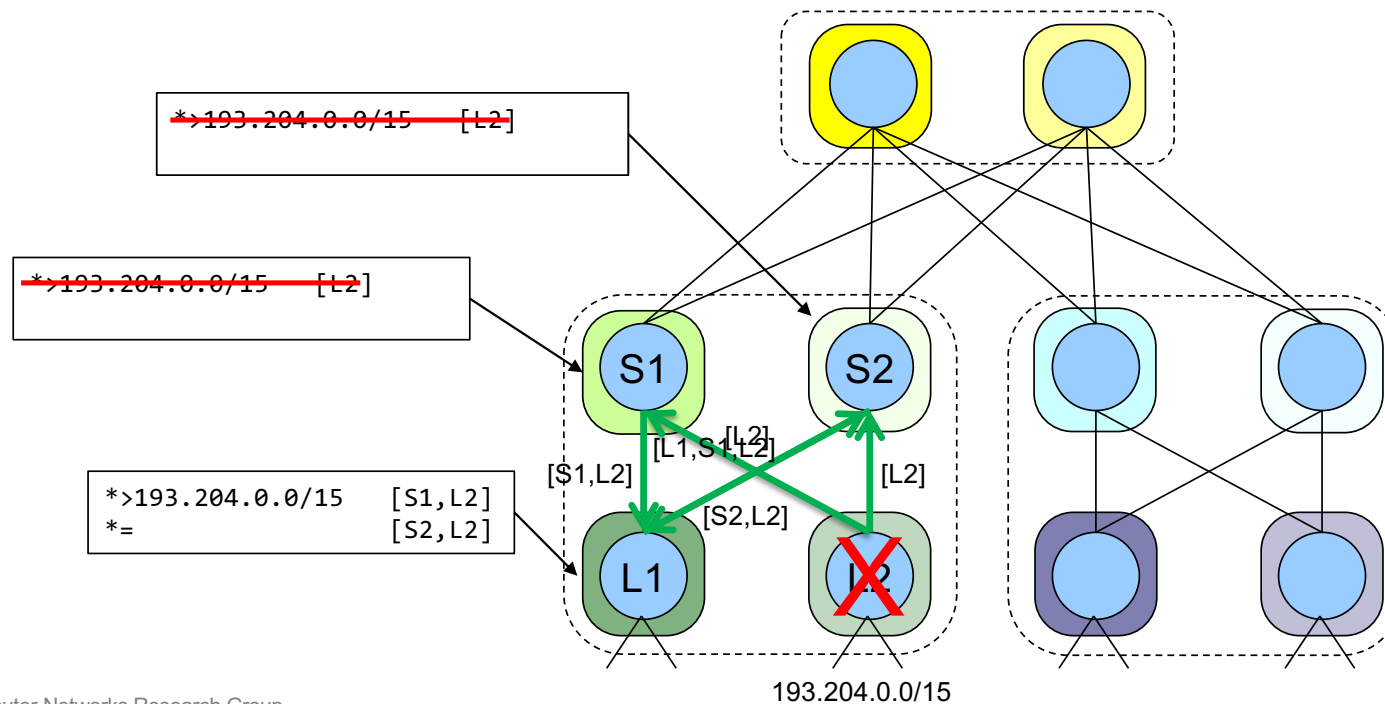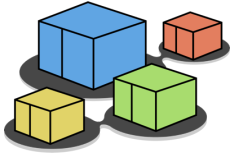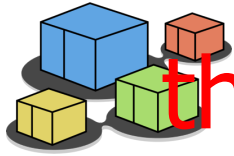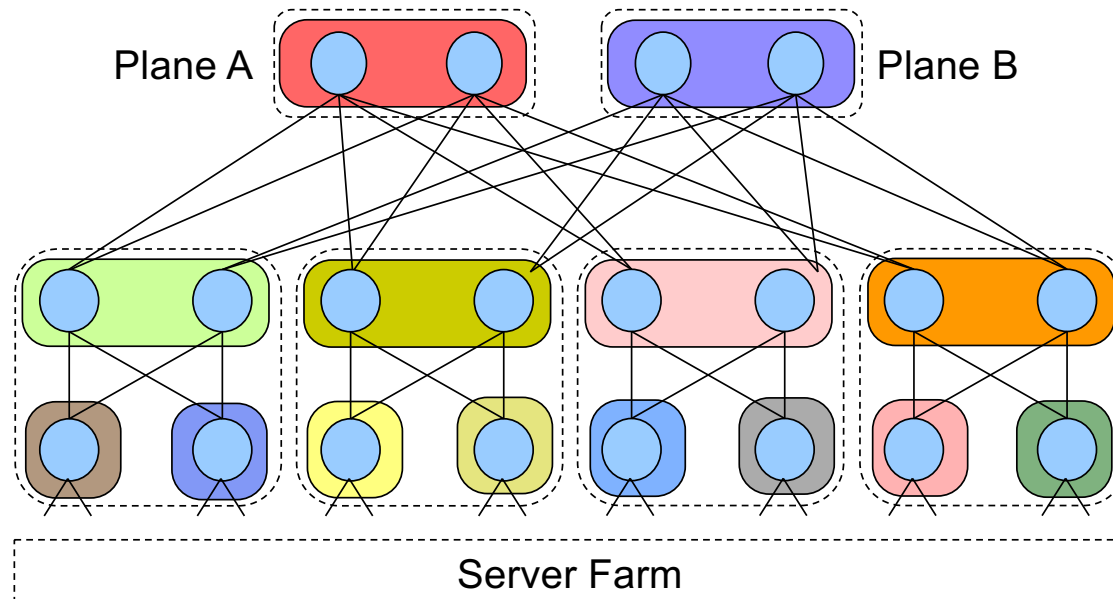
the ASes scheme of choice

- each Leaf is a different AS

- the Spine nodes of each PoD belong to the same AS

- the ToF nodes of the same plane belong to the same AS

BEWARE: there are NO iBGP peerings

the ASes scheme of choice – multi-plane
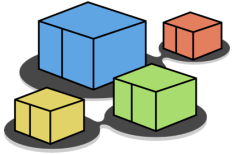
Plane A

Plane B

Server Farm

multiple paths between Leaves

- the adopted AS scheme allows for multiple paths between pairs of Leaves
    - in a FT(K=2,R=2) only two paths are possible between two Leaves of the same PoD
        - they don't leave the PoD

multiple paths between Leaves

- the adopted AS scheme allows for multiple paths between pairs of Leaves
  - in a FT(K=2,R=2) eight paths are possible between two Leaves of different PoDs
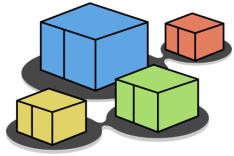
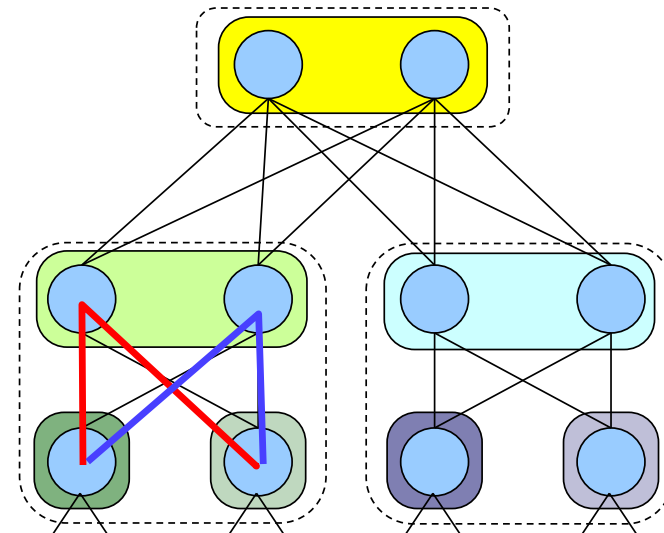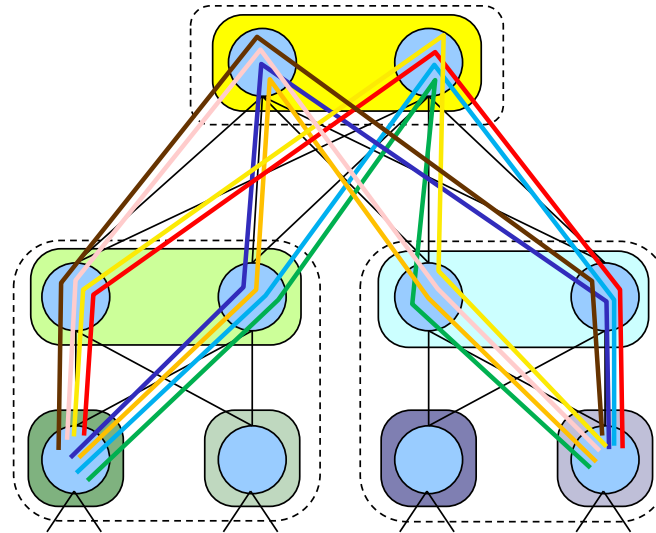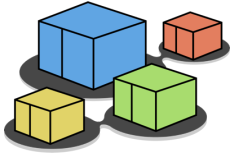# BGP and ECMP

- **BGP natively supports ECMP**

- **by default, BGP considers two announcement of the same prefix "equal" if they are equal in each best-path selection criterion except for the last one**
  - lowest router-id of the announcing peer

- **to enable multi-path, BGP requires that the AS-paths selected for multi-path match exactly**
  - not just they have equal-length but equal AS numbers inside

BGP multi-path relax

- when the AS_PATH lengths of different announcements for the same prefix are the same, the best-path algorithm skips checking for exact match of the AS numbers

- this modification is often called "as-path multipath-relax"
  - different vendors may use different names

- really needed for using dual attached servers and multi-plane Fat-Trees

tuning BGP timers

- there are four main timers that are responsible for BGP behaviour
  - advertisement interval timer
  - keepalive timer
  - hold timer
  - connect timer
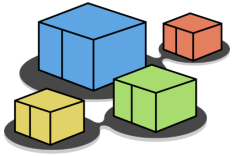
advertisement interval timer

- announcements that need to be sent to a neighbor are bunched together and sent together only when the interval expires
  - then the timer is reset for that neighbor
- the default value for eBGP is 30s
  - in interdomain routing this improves stability and reduces the number of multiple updates for the same prefix
- in data centers is set to 0s
  - it is required for fast convergence
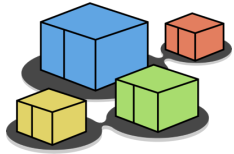
keepalive and hold timers

- each BGP peer sends periodic keepalive messages to its neighbors according to the keepalive timer
- when a peer doesn't receive a keepalive for a period greater than the hold timer
  - the connection is dropped
  - all the announcements received are considered invalid
  - the peer tries to re-establish the connection
- by default, the keepalive timer is 60s and the hold timer is 180s
- in data centers timers of 3s and 9s are used, respectively

connect timer

- when a connection to a peer fails, BGP waits for the connect timer expiration before attempting to reconnect
    - the connect timer by default is 60s
    - this can delay session re-establishment when a link recovers from a failure or a node powers up
- in data centers it is set to 10s
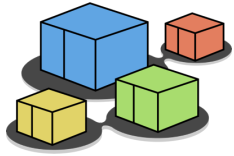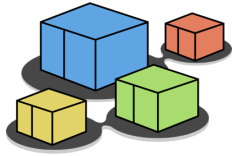
automating the configuration

- **unnumbered interfaces**
  - used to establish peerings specifying the interface name rather than the IP address and the remote AS number

- **peer groups**
  - used to specify policies for groups of peers

connecting the servers

each Leaf announces its server's prefix

Prefixes are assigned directly to the servers

Containers or VMs

Servers
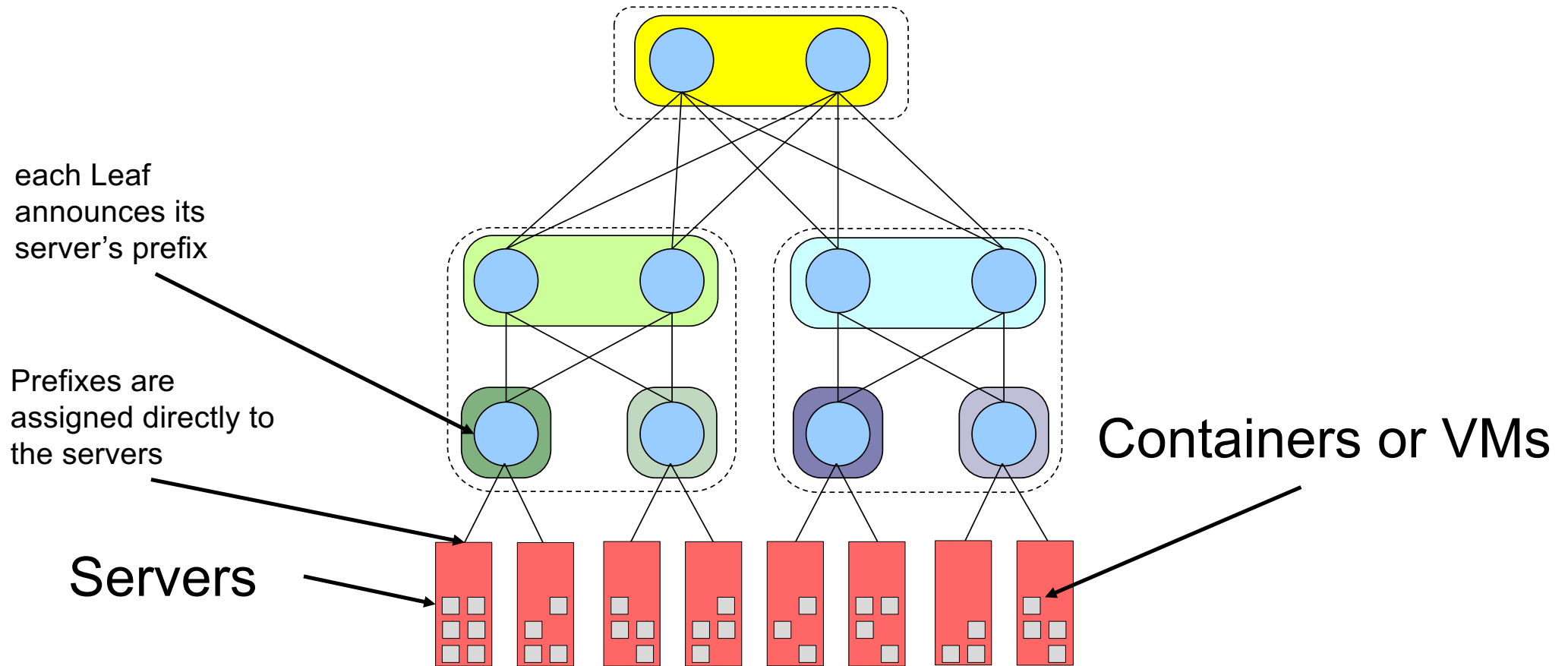
basic Fat-Tree lab

hands on Kathará

lab pre-conditions

- **Linux and macOS**
  - no specific requirement
- **Windows**
  - WSL 2 does not support Multi-Path
  - need to fallback to Hyper-V Docker backend
    - open Docker Desktop and go to Settings (cog in the top-right corner)
    - unselect the "Use the WSL 2 based engine" checkbox
    - click "Apply & restart"
  - or create a VM with a Linux distribution

lab pre-conditions

- **FRR handles unnumbered interface only with IPv6 link-local addresses**
  - all the routers in the fabric must have IPv6 enabled in order to work
    - to enable IPv6 for machine "router", add router[ipv6]=True to the lab.conf file

naming convention

- **tof_x_y_z**
  - x: plane number
  - y: level, always 2
  - z: ToF number
- **spine_x_y_z**
  - x: PoD number
  - y: level, always 1
  - z: Spine number

- **leaf_x_y_z**
  - x: PoD number
  - y: level, always 0
  - z: Leaf number
- **server_x_y_z**
  - x: PoD number
  - y: corresponding Leaf number
  - z: server number

lab topology

© Computer Networks Research Group
Roma Tre University
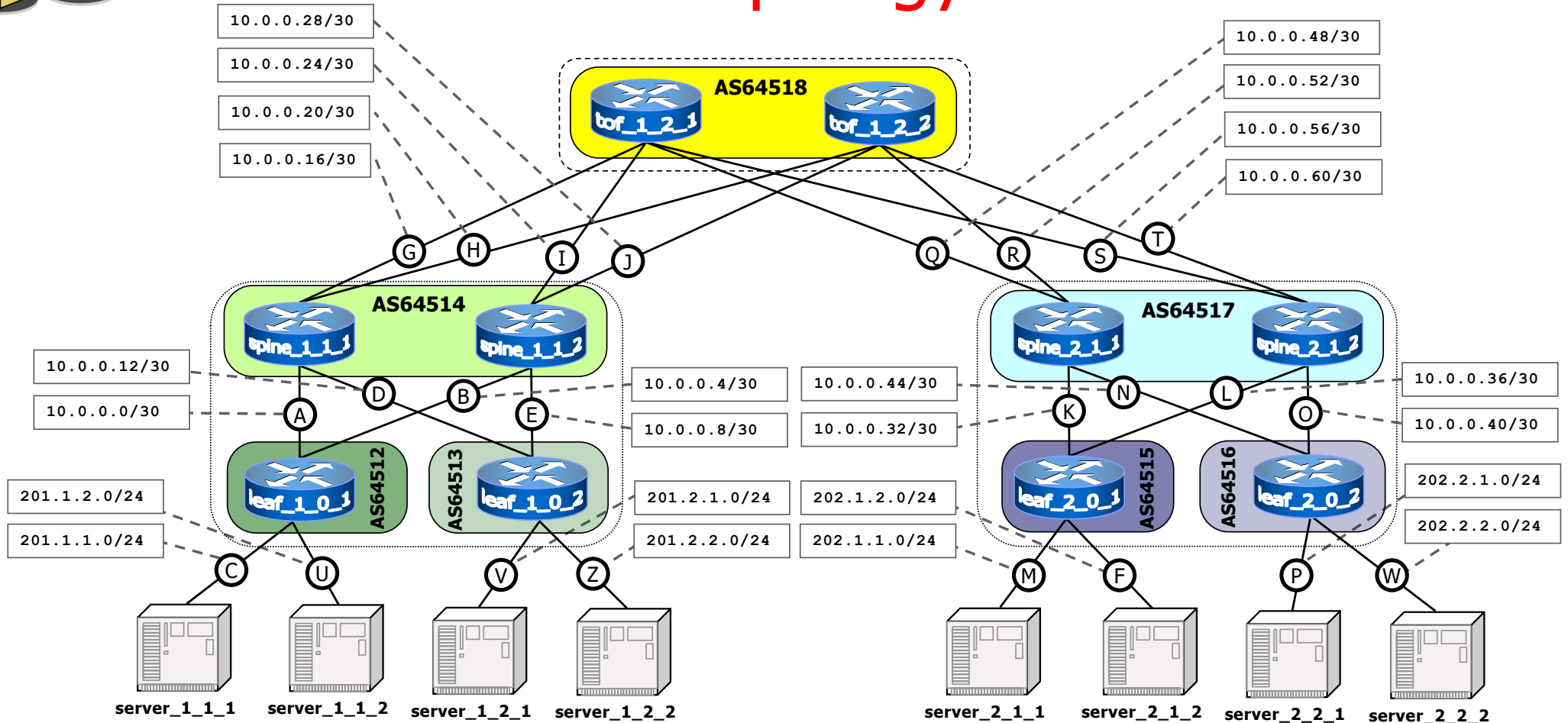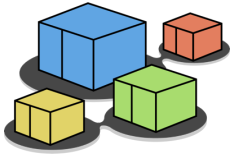
ToF configuration example

```
bgpd.conf
router bgp 64518
 timers bgp 3 9
 bgp router-id 192.168.0.14
 no bgp ebgp-requires-policy
 bgp bestpath as-path multipath-relax

neighbor fabric peer-group
 neighbor fabric remote-as external
 neighbor fabric advertisement-interval 0
 neighbor fabric timers connect 10
 neighbor eth0 interface peer-group fabric
 neighbor eth1 interface peer-group fabric
 neighbor eth2 interface peer-group fabric
 neighbor eth3 interface peer-group fabric

address-family ipv4 unicast
   neighbor fabric activate
   maximum-paths 64
exit-address-family
```
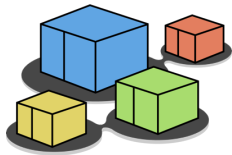
set keepalive and hold timers

BGP multi-path relax

create a peer-group

all peers of the group are eBGP. The remote ASN is inferred

set advertisement interval timer

set connect timer

add interfaces to peer-group for unnumbered peering

enable A.F. to peer-group

maximum Multi-Path possibilities
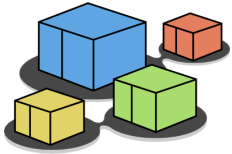
Spine configuration example

bgpd.conf – part 1

```
router bgp 64514
 timers bgp 3 9
 bgp router-id 192.168.0.5
 no bgp ebgp-requires-policy
 bgp bestpath as-path multipath-relax

neighbor TOR peer-group
 neighbor TOR remote-as external
 neighbor TOR advertisement-interval 0
 neighbor TOR timers connect 10
 neighbor eth0 interface peer-group TOR
 neighbor eth1 interface peer-group TOR

neighbor fabric peer-group
 neighbor fabric remote-as external
 neighbor fabric advertisement-interval 0
 neighbor fabric timers connect 10
 neighbor eth2 interface peer-group fabric
 neighbor eth3 interface peer-group fabric
```

bgpd.conf – part 2

```
address-family ipv4 unicast
  neighbor fabric activate
  neighbor TOR activate
  maximum-paths 64
exit-address-family
```

Leaf configuration example

```
bgpd.conf

router bgp 64512
 timers bgp 3 9
 bgp router-id 192.168.0.1
 no bgp ebgp-requires-policy
 bgp bestpath as-path multipath-relax

neighbor TOR peer-group
 neighbor TOR remote-as external
 neighbor TOR advertisement-interval 0
 neighbor TOR timers connect 10
 neighbor eth0 interface peer-group TOR
 neighbor eth1 interface peer-group TOR

address-family ipv4 unicast
  neighbor TOR activate
  network 201.1.1.0/24
  network 201.1.2.0/24
  maximum-paths 64
exit-address-family
```
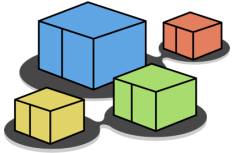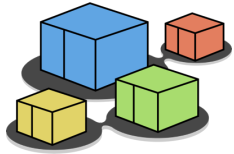
announce the server prefixes

data plane

```
root@spine_1_1_1:/# ip route
10.0.0.0/30 dev eth0 proto kernel scope link src 10.0.0.2
10.0.0.8/30 dev eth1 proto kernel scope link src 10.0.0.10
10.0.0.16/30 dev eth2 proto kernel scope link src 10.0.0.17
10.0.0.20/30 dev eth3 proto kernel scope link src 10.0.0.21
201.1.1.0/24 nhid 12 via 10.0.0.1 dev eth0 proto bgp metric 20
201.1.2.0/24 nhid 12 via 10.0.0.1 dev eth0 proto bgp metric 20
201.2.1.0/24 nhid 11 via 10.0.0.9 dev eth1 proto bgp metric 20
201.2.2.0/24 nhid 11 via 10.0.0.9 dev eth1 proto bgp metric 20
202.1.1.0/24 nhid 16 proto bgp metric 20
        nexthop via 10.0.0.18 dev eth2 weight 1
        nexthop via 10.0.0.22 dev eth3 weight 1
202.1.2.0/24 nhid 16 proto bgp metric 20
        nexthop via 10.0.0.18 dev eth2 weight 1
        nexthop via 10.0.0.22 dev eth3 weight 1
202.2.1.0/24 nhid 16 proto bgp metric 20
        nexthop via 10.0.0.18 dev eth2 weight 1
        nexthop via 10.0.0.22 dev eth3 weight 1
202.2.2.0/24 nhid 16 proto bgp metric 20
        nexthop via 10.0.0.18 dev eth2 weight 1
        nexthop via 10.0.0.22 dev eth3 weight 1
```

multiple next-hop for the same prefix

control plane

```
spine_1_1_1# show ip bgp
BGP table version is 8, local router ID is 192.168.0.6, vrf id 0
Default local pref 100, local AS 64514
Status codes:  s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 201.1.1.0/24     10.0.0.5                 0             0 64512 i
*> 201.1.2.0/24     10.0.0.5                 0             0 64512 i
*> 201.2.1.0/24     10.0.0.13                0             0 64513 i
*> 201.2.2.0/24     10.0.0.13                0             0 64513 i
*> 202.1.1.0/24     10.0.0.26                              0 64518 64517 64515 i
*=                  10.0.0.30                              0 64518 64517 64515 i
*> 202.1.2.0/24     10.0.0.26                              0 64518 64517 64515 i
*=                  10.0.0.30                              0 64518 64517 64515 i
*> 202.2.1.0/24     10.0.0.26                              0 64518 64517 64516 i
*=                  10.0.0.30                              0 64518 64517 64516 i
*> 202.2.2.0/24     10.0.0.26                              0 64518 64517 64516 i
*=                  10.0.0.30                              0 64518 64517 64516 i

Displayed  8 routes and 12 total paths
```
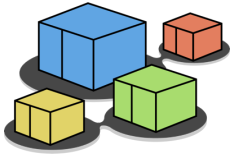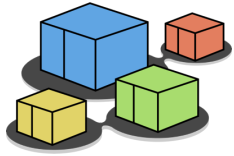
Multi-Path traceroute

- traditional traceroute may provide hard-to-interpret or even misleading results when used in presence of ECMP
- Multi-Path traceroute tools generate packet header contents to obtain a more precise picture of the actual routes of packets
  - allow all probes towards a destination to follow the same path in the presence of per-flow load balancing
  - allow a user to distinguish between the presence of per-flow load balancing and per-packet load balancing
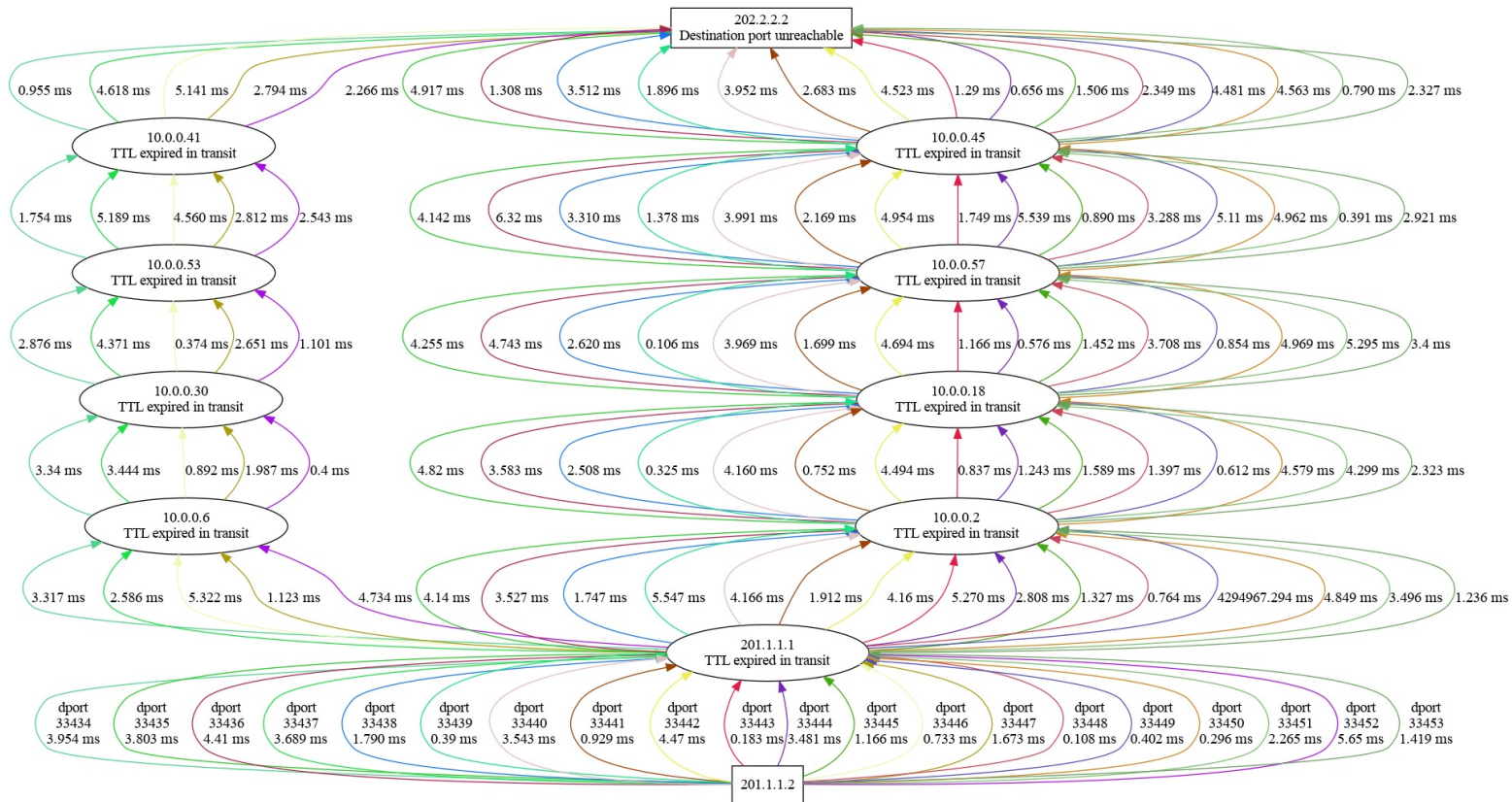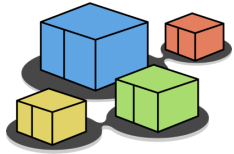
Multi-Path traceroute

- two different tools:
  - `paris-traceroute`
    - traceroute designed to work in presence of Multi-Path and load balancers
  - `dublin-traceroute`
    - based on the `paris-traceroute`
    - adds a NAT detection technique
    - introduces visualization and analysis tools
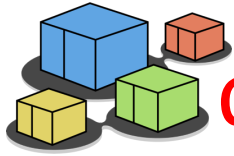
dublin-traceroute example output

bibliography and further readings

- [Caiazzi '22] Caiazzi, Scazzariello, Alberro, Ariemma, Castro, Grampin, Di Battista, "Sibyl: a Framework for Evaluating the Implementation of Routing Protocols in Fat-Trees", NOMS 2022

- [Caiazzi '21] Caiazzi, Scazzariello, Ariemma, "VFTGen: a Tool to Perform Experiments in Virtual Fat Tree Topologies", IM 2021

- [Caiazzi '19] Caiazzi, "Software Defined Data Centers: methods and tools for routing protocol verification and comparison", Ms. Thesis, Roma Tre University, 2019

- [Dutt '17] Dutt, "BGP in the Data Center", O'Reilly, 2017

- [RFC-7938] Lapukhov, Premji, "Use of BGP for Routing in Large-Scale Data Centers" Internet Engineering Task Force (IETF) Request for Comments: 7938