Cumulus® Linux® 2.5.3

What's New and Different since 2.5.0 (Technical)

Last Updated: June 18, 2015

What's New and Different in Cumulus Linux 2.5.3 vs. 2.5.0

- BFD additions
- BGP improvements
- Hardware watchdog support
- LACP bypass with multiple active interfaces
- Licensing change
- Resilient hashing
- `jdoo` replaces `monit`
- New hardware platforms

Cumulus® Linux®
Network OS

BFD Enhancements in Quagga

2.5.3 brings another way of enabling BFD

- Enable BFD in Quagga without a topology file
  (which was the only method until 2.5.3)
- ptmd is the underlying process in both methods

OSPF

```
quagga(config)# interface <interface name>
quagga(config-if)# ip ospf bfd
```

Simpler Configuration

BGP

```
quagga(config)# router bgp X
quagga(config-router)# neighbor <neighbor ip> bfd
```

New “afi” topology file parameter

- 3 supported values:

Value	Description
v4	session built only for IPv4 connected peer (default)
v6	session built only for IPv6 link-local connected peer
both	session built for both IPv4 and IPv6 link-local peers

Global and link-local IPv6 address peering supported using Quagga method

- Topology file method supports link-local address BFD peering but not global

BFD multihop enables BFD across arbitrary paths between two systems

- Multihop is supported via Quagga

  Topology file method doesn't support multihop
- Multihop sessions can use IPv4 or IPv6

ptmctl output has been enhanced

- Session type shows “multihop” or “singlehop”
- Source “local” address has been added to ptmctl outputs for multihop sessions

port	peer	state	local	type	diag	det mult	tx_timeout	rx_timeout	echo tx_timeout	echo rx_timeout	max hop_cnt	rx_ctrl	tx_ctrl	rx_echo	tx_echo
swp1	11.0.0.2	Up	N/A	singlehop	N/A	5	360	1750	0	0	N/A	1860	1805	0	0
N/A	12.12.12.1	Down	12.12.12.4	multihop	N/A	0	2000	1705	0	0	2	0	205	0	0

2.5.3 includes several BGP enhancements

- Internal/external syntax additions
- Unnumbered interfaces
  (extended next-hop encoding)
- New interface configuration

BGP — Internal/External Peering Syntax

Before 2.5.3, BGP peering established using

```
neighbor x.x.x.x remote-as <ASN>
```

New syntax

```
neighbor (ipv4 addr|ipv6 addr|word)
remote-as (<1-4294967295>|internal|external)
```

internal and external keywords enable simpler automation configurations

- No need to know peer's ASN

BGP unnumbered interfaces

- Enables BGP peering using interface names instead of IP addresses
- Works with both IPv4 and IPv6 prefixes
- Works with IPv4 interface address if it's a /30 or /31 address or over IPv6 link-local only
- Uses extended next-hop encoding (RFC 5549) to exchange IPv4 prefixes over IPv6 link-local session

BGP — Unnumbered Interfaces

New outputs

```
# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, A - Babel, T - Table,
       > - selected route, * - FIB route
K>* 0.0.0.0/0 via 192.168.0.2, eth0
C>* 6.0.0.5/32 is directly connected, lo
B>* 6.0.0.6/32 [20/0]     via fe80::202:ff:fe00:45, swp3, 00:46:12
  *                            via fe80::202:ff:fe00:35, swp1, 00:46:12
  *                            via fe80::202:ff:fe00:3d, swp2, 00:46:12
  *                            via fe80::202:ff:fe00:4d, swp4, 00:46:12
  *                            via fe80::202:ff:fe00:55, swp5, 00:46:12
  *                            via fe80::202:ff:fe00:5a, swp6, 00:46:12
```

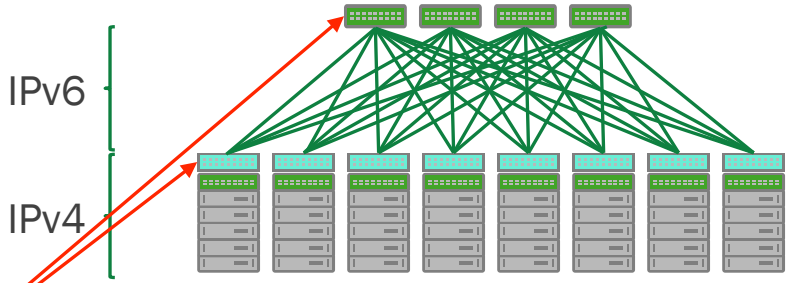
```
#  show ip bgp
BGP table version is 66, local router ID is 6.0.0.5
Status codes: s suppressed, d damped, h history, * valid, > best, = multipath,
              i internal, r RIB-failure, S Stale, R Removed
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric              LocPrf             Weight Path
*> 6.0.0.5/32       0.0.0.0      0                                32768  ?
*= 6.0.0.6/32       swp2         0            65534               64503  ?
*=                  swp6         0            65002               64503  ?
*=                        swp5         0            65001               64503  ?
*=                        swp1         0            65534               64503  ?
*=                        swp4         0            65534               64503  ?
*>                        swp3         0            65534               64503  ?
```

Example and config

- BGP interface configuration requires enabling IPv6 ND router-advertisement

```
#New Configuration
router bgp 500
 bgp router-id 10.0.0.1
 neighbor EBGP peer-group
 neighbor EBGP remote-as external
 neighbor swp1 interface
 neighbor swp1 peer-group EBGP
 neighbor swp1 capability extended-nexthop

interface swp1
 no ipv6 nd suppress-ra
 ipv6 nd ra-interval
```



BGP — Interface Configuration

Allows for peering without using peer's IP in configuration

With BGP unnumbered interfaces, enables generic BGP configs instead of configuring BGP on node-by-node basis

- IPv6 link-local address-based peering for both IPv4 and IPv6 prefixes
  No IPv4 addresses can be assigned
- IPv4 interface address-based peering for IPv4 prefixes and IPv6 link-local address for IPv6 prefixes
  Simplified peer configuration is available when using /30 or /31 IPv4 subnets

BGP — Interface Configuration with IPv6 Link-local Peering

Configuration	Details
Without an IPv4 or IPv6 global address configured on the peering interface	• IPv6 prefixes will use IPv6 link-local next-hops • IPv4 prefixes will use BGP unnumbered interfaces if configured
IPv4 addresses configured on peering interface	• IPv4 prefixes will use IPv4 next-hops
IPv6 global address configured on peering interface	• IPv6 prefixes can use either global or link-local next-hops • IPv4 prefixes will use an IPv4 next-hop address unless IPv4 is not configured on the interface, then IPv4 prefixes will use IPv6 link-local next-hops

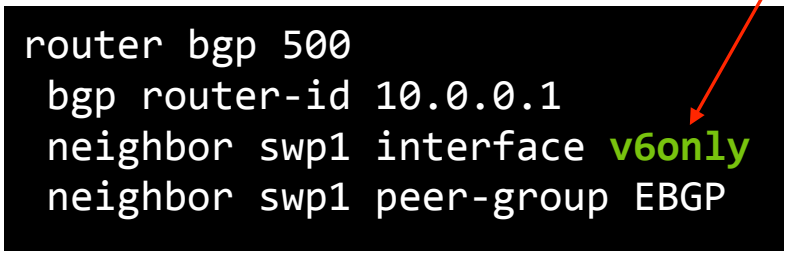
cumulus

Interface-based peering logic

Configuration	Details
With a /30 or /31 subnet IPv4 address configured	• Peering is done using IPv4 addresses • IPv4 prefixes exchanged using IPv4 address and IPv6 prefixes exchanged using the IPv6 addresses (link-local if no v6 address configured)
With no IPv4 address configured and **extended next-hop encoding** not enabled	• IPv6 link-local address-based peering is established • Only IPv6 prefixes are exchanged
If no IPv4 address configured but extended nexthop capability is enabled and supported by the peer	• IPv6 link-local address-based peering is established • IPv4 and IPv6 prefixes are exchanged

BGP — Interface Configuration with IPv4 Addresses

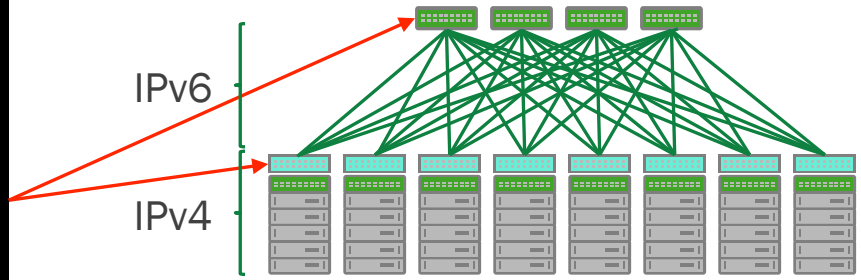
To configure an interface IPv4 address and still utilize interface-based peering based on IPv6 link-local address, use the *v6only* option



Example config

- Much simpler config for automating.
- No need to configure neighbor's ASN or IP!

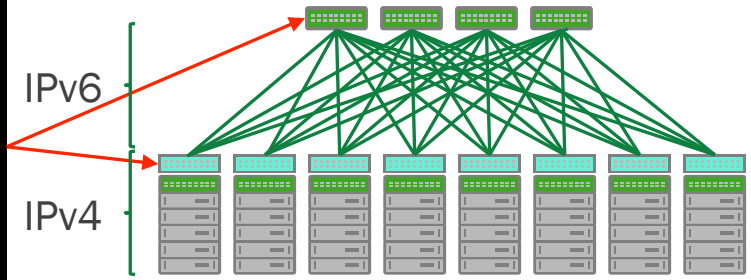
```
#New in 2.5.3
router bgp 500
 bgp router-id 10.0.0.1
 neighbor EBGP peer-group
 neighbor EBGP remote-as external
 neighbor swp1 interface
 neighbor swp1 peer-group EBGP
```



Combined cleaner config

```
#New Configuration
router bgp 500
 bgp router-id 10.0.0.1
 neighbor EBGP peer-group
 neighbor EBGP remote-as external
 neighbor swp1 interface
 neighbor swp1 peer-group EBGP
 neighbor swp1 capability extended-nexthop

interface swp1
    no ipv6 nd suppress-ra
    ipv6 nd ra-interval
```



Platform Watchdog

A hardware monitor is needed to validate a switch so it doesn't become brain dead

Cumulus Linux includes a simplified version of the `wd_keepalive` daemon from the standard Debian package `watchdog`

How it's used

- A daemon writes to /dev/watchdog at least once per minute
- If there is not a write within one minute, the switch resets

Using the Watchdog Feature

To enable the hardware watchdog:

- Disabled by default on all platforms except QuantaMesh BMS T1048-LB9
- Edit `/etc/watchdog.d/<your_platform>`
- Set `run_watchdog = 1`

To disable the watchdog:

- Edit `/etc/watchdog.d/<your_platform>`
- Set `run_watchdog = 0`

Then stop/start or restart the daemon:

- `cumulus@switch:~$ sudo service wd_keepalive stop/start | restart`

LACP bypass

- Allows a bond to become active and forward traffic without an LACP partner
- Extensions to LACP bypass allow multiple links active in a LACP bond in bypass mode

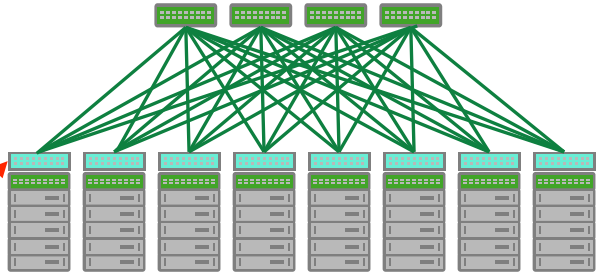
During PXE boot of a new server

- LACP bypass allows a link to pass traffic (become active) before the server can form an LACP bond
- Some boot systems require all links to be up during bypass, and if both links are not active, PXE will not execute

Example and config

```
auto esx-server3
iface esx-server3
bond-slaves swp11 swp12
bond-mode 802.3ad
bond-miimon 100
bond-lacp-rate 1
bond-lacp-bypass-allow 1
bond-lacp-bypass-all-active 1
mtu 9000
bond-xmit-hash-policy layer3+4

bond-lacp-bypass-allow 1
bond-lacp-bypass-all-active 1
```



Enable LACP bypass

Allow all links of bond to “bypass”

New license format for 2.5.3 and later

How it's used

- Single line license file
- New location: `/etc/cumulus/.license`

What does this mean?

- Old licenses will still work with 2.5.3
- New license incompatible with versions older than 2.5.3

Resilient ECMP Hashing

Ensures minimal disruption in flows when adding or removing a link

- Achieved by keeping a flow mapped to its current link if any new member is added or removed
- To configure resilient hashing:

  Edit **/etc/cumulus/datapath/traffic.conf**

  Enable by setting

  ```
  resilient_hash_enable = TRUE
  ```

  Restart switchd

  ```
  sudo service switchd restart
  ```

To verify the configuration change:

```
cumulus@switch:~$ sudo cl-cfg -a switchd | grep resilient
traffic.resilient_hash_entries_ecmp = 256
traffic.resilient_hash_enable = TRUE
```

cumulus

Supported only under certain circumstances

- Trident II chipsets

  Check cumulusnetworks.com/hcl/ or run `netshow system`
- Unicast traffic
- ECMP groups
- IPv4 routes

jdoo replaces monit

Starting in 2.5.2 monit was replaced with jdoo

- jdoo forked from monit
- jdoo is a monitoring utility used to validate things like CPU, memory and uptime for services
- jdoo can execute causal actions for error situations, restart a process that does not respond, if it uses too many resources
- Files, directories and file systems can all be monitored for changes to existing timestamps, checksum, etc.

Change made because of licensing conflicts with monit

Thank You!