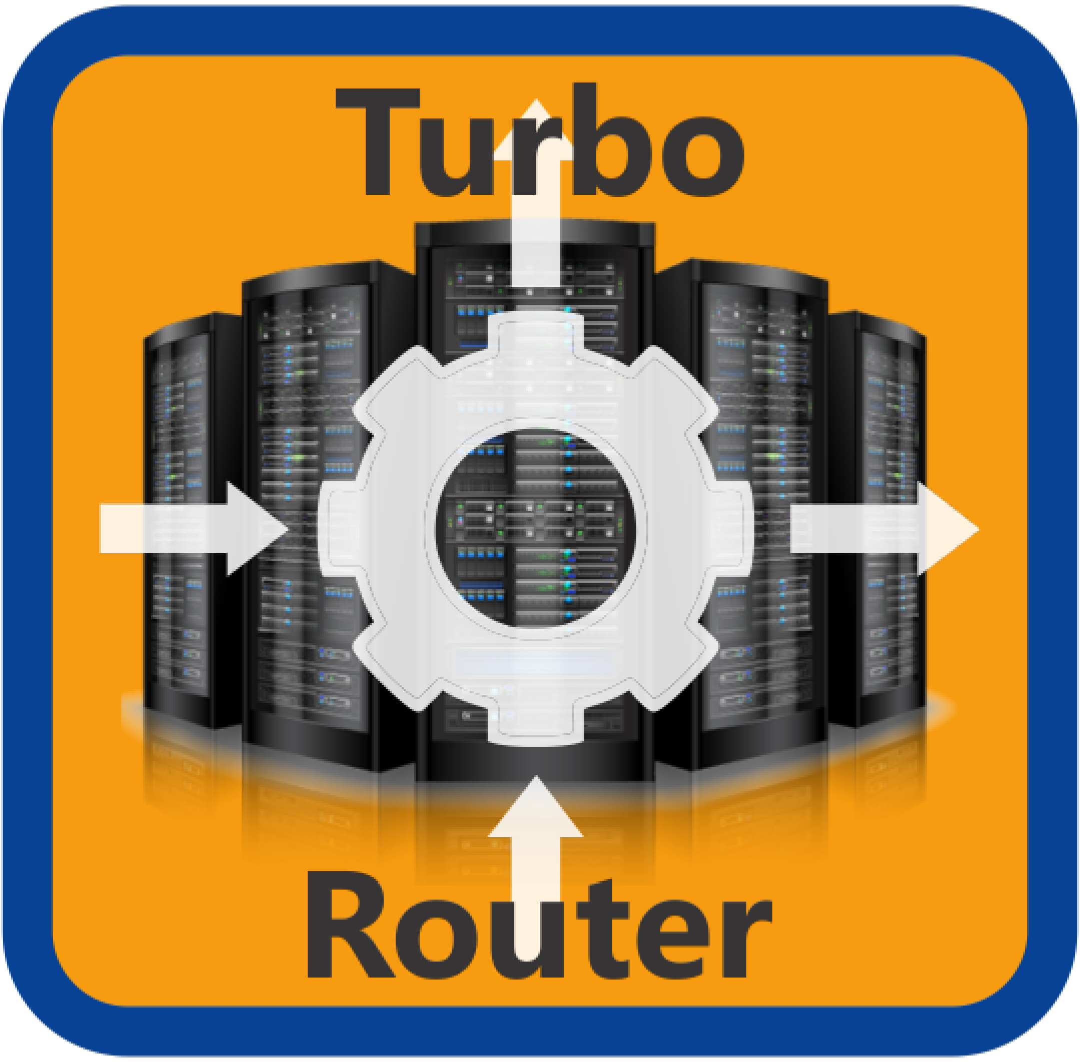


Border Router Deployment Guide

vRouter 2.1

6WIND S.A.
1, place Charles de Gaulle
78180 Montigny-le-Bretonneux
France
http://www.6wind.com

Notice

Contents

1. Overview

The purpose of this document is to guide the user in deploying the vRouter for a Border Router use case. It focuses on the concepts that are relevant to this specific use case, in order to provide a practical example. Exhaustive documentation of the vRouter features that are not covered in the use case can be found in the standard vRouter documentation.

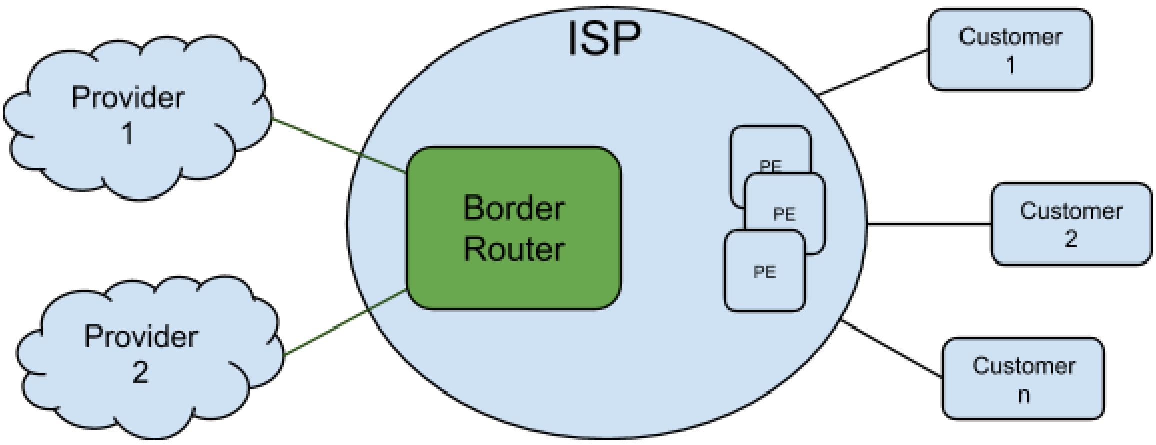
2. Installation and basic configuration

Follow the Getting Started guide to install the software in your environment and get a remote console with SSH.

3. Use case: dual multi-homed BGP peering

3.1. Overview

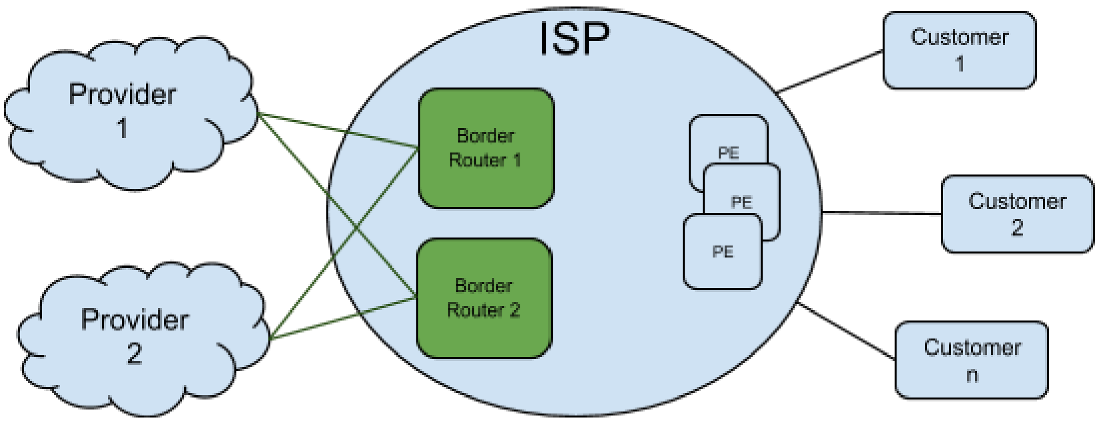
The following diagram depicts a fairly common use case where a border router provides connectivity to the Internet via several Peering and Transit Service Providers.



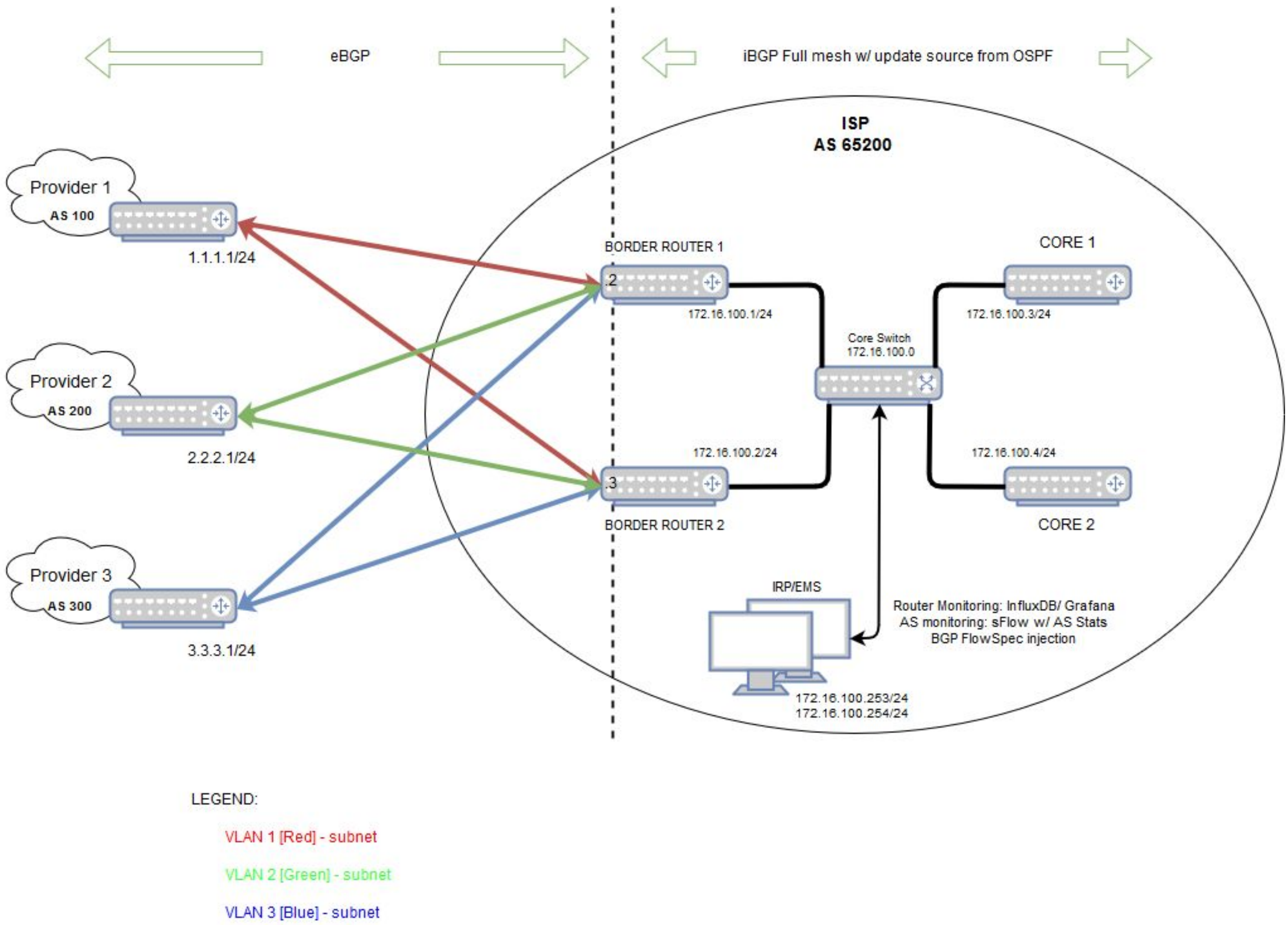
The outside network connectivity via eBGP provides service availability using different ASes. The inside network connectivity is established using an IGP (Internal Gateway Protocol) that usually relies on OSPF or iBGP or a combination of both. In the described case and for illustration purposes, we will be using iBGP and OSPFv2.

To avoid the border router being a single point of failure, we will describe how an ISP can provide a multi-homed BGP peering with several peering and transit providers.

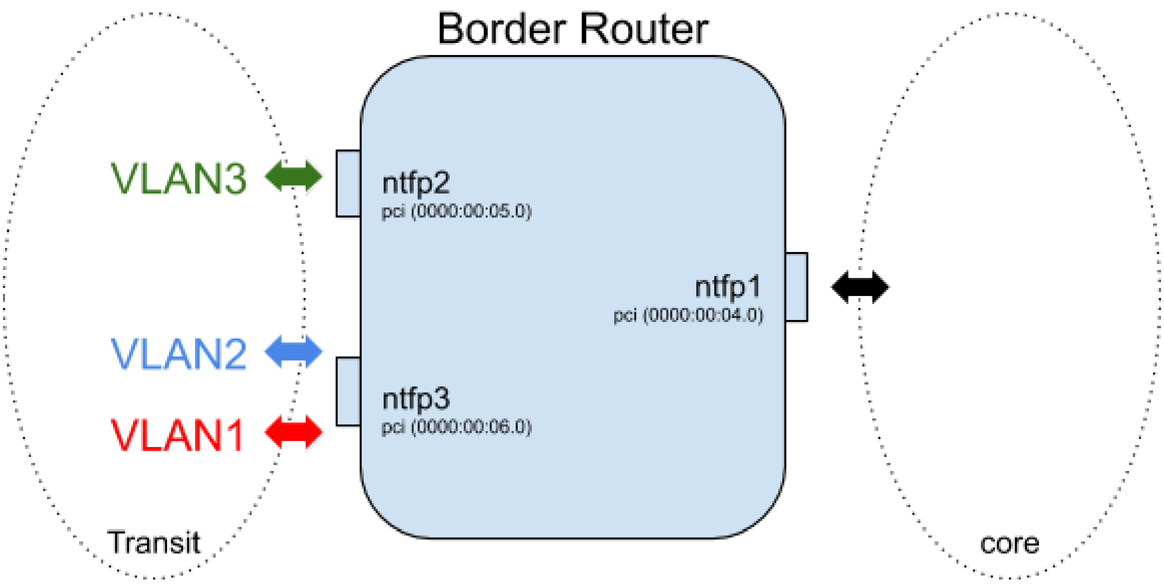
The setup we will be implementing in the following guide will be structured as follows:



3.2. Setup description



A number of VLANs are configured for ISP traffic separation in this dual-homed scenario.



In summary, the following router configuration items are covered using the 6WIND Turbo Router CLI:

- Interface configuration for physical, loopback & VLAN interfaces
- OSPF
- BGP, prefix-list, route-map & FlowSpec
- SNMP
- KPI
- sFlow

See Appendix A for the complete configuration of these items.

3.3. Network configuration

Using the vRouter CLI, let us start with setting hostname and then getting the interfaces configured.

To set the vRouter hostname proceed as follows:

```
vrouter> edit running
vrouter running config# system hostname border1
vrouter running config# commit
border1 running config#
```

Allocate the ports that will be involved in data plane processing into the fast path:

```
border1 > edit running
border1 running config# / system fast-path
border1 running fast-path#! port pci-b0s4
border1 running fast-path# port pci-b0s5
border1 running fast-path# port pci-b0s6
```

All physical and logical interfaces are configured under the ‘main’ VRF in this example.

```
border1 running fast-path# / vrf main
```

Create Ethernet interfaces and attach them to a port of a NIC:

```
border1 running vrf main# interface physical ntfp1
border1 running physical ntfp1#! port pci-b0s4
border1 running physical ntfp1# description “Border1_internal”
border1 running physical ntfp1# ipv4 address 172.16.100.1/24
border1 running physical ntfp1# ..

border1 running interface# physical ntfp2
border1 running physical ntfp2#! port pci-b0s5
border1 running physical ntfp2# ..

border1 running interface# physical ntfp3
border1 running physical ntfp3#! port pci-b0s6
border1 running physical ntfp3# ..
```

Add VLANs towards the ISP networks:

```
border1 running interface# vlan vlan1
border1 running vlan vlan1# description “Transit_1”
border1 running vlan vlan1# ipv4 address 1.1.1.2/24
border1 running vlan vlan1# vlan-id 1
border1 running vlan vlan1# link-interface ntfp3
border1 running vlan vlan1# ..

border1 running interface# vlan vlan2
border1 running vlan vlan2# description “Transit_2”
border1 running vlan vlan2# ipv4 address 2.2.2.2/24
border1 running vlan vlan2# vlan-id 2
border1 running vlan vlan2# link-interface ntfp3
border1 running vlan vlan2# ..

border1 running interface# vlan vlan3
border1 running vlan vlan3# description “Transit_3”
border1 running vlan vlan3# ipv4 address 3.3.3.2/24
border1 running vlan vlan3# vlan-id 3
border1 running vlan vlan3# link-interface ntfp2
border1 running vlan vlan3# ..
```

Add a loopback interface for OSPF to use as a BGP update-source:

```
border1 running interface# loopback loopback0
border1 running loopback loopback0# ipv4 address 172.16.200.1/32
border1 running loopback loopback0# ..
```

Review the configuration and commit it:

```
border1 running vrf main# show config nodefault /
interface
    physical ntfp1
        port pci-b0s4
[…]
border1 running vrf main# commit
Configuration committed.
```

Be sure other routers of the setup (PE1, PE2) are configured correctly, then check connectivity using the ping command. Here we will simply ping 172.16.100.4 (PE1) as an example:

```
border1 running loopback loopback0# cmd ping 172.16.100.4

PING 172.16.100.4 (172.16.100.4) 56(84) bytes of data:
64 bytes from 172.16.100.4: icmp_seq=1 ttl=64 time=0.396 ms
64 bytes from 172.16.100.4: icmp_seq=2 ttl=64 time=0.326 ms
64 bytes from 172.16.100.4: icmp_seq=3 ttl=64 time=0.346 ms
^C
--- 172.16.100.4 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 3136 ms
rtt min/avg/max/mdev = 0.326/0.356/0.396/0.0.036ms
```

See the User's Guide for more information regarding:

- CLI basics
- fast path configuration
- interfaces configuration
- ping command

3.3.3. OSPF

Next, we configure an interior routing protocol. This will be a very simple OSPF configuration with only neighbors on the 172.16.100.0 private network. We redistribute the loopback address configured earlier and make it a passive OSPF interface. The router-id will simply be set to the same loopback IPv4 address for ease of reading.

```
border1 running config# / vrf main routing ospf
border1 running ospf# router-id 172.16.200.1
border1 running ospf# abr-type standard
border1 running ospf# log-adjacency-changes detail
border1 running ospf# network 172.16.100.0/24 area 0
border1 running ospf# passive-interface loopback0
border1 running ospf# redistribute connected
border1 running ospf# commit
```

At this time it would be a good idea to check the OSPF adjacencies and routes. See the Troubleshooting section below.

See the User's Guide for more information regarding:

- OSPF

3.3.4. BGP

The key configuration item of this and most border routers is the exterior routing protocol BGP.

This configuration example will have iBGP/eBGP ipv4-unicast neighbors. Route-reflectors could easily have been used, but in this example we choose to create a full internal mesh using OSPF.

We will anchor the update-sources using the loopback addresses that was redistributed via OSPF in section 3.2.2.

3.3.4.1. iBGP

First we will peer with the internal network, i.e. the other border router and PEs. We start out by defining our own AS and router-id.

Configuring the local AS and router-id:

```
border1 running vrf main# routing bgp
border1 running bgp#! as 65200
border1 running bgp# router-id 172.16.200.1
```

Next we configure BGP peering with the second Border Router:

```
border1 running bgp# neighbor 172.16.200.2
border1 running neighbor 172.16.200.2#! remote-as 65200
border1 running neighbor 172.16.200.2# neighbor-description border2
border1 running neighbor 172.16.200.2# update-source loopback0
border1 running neighbor 172.16.200.2# address-family ipv4-unicast soft-reconfiguration-inbound true
border1 running neighbor 172.16.200.2# ..
border1 running bgp# commit
```

Then finally, as part of our iBGP configuration, we configure BGP peering with the internal routers PE1 & PE2:

```
border1 running bgp# neighbor 172.16.200.3
border1 running neighbor 172.16.200.3#! remote-as 65200
border1 running neighbor 172.16.200.3# neighbor-description PE1
border1 running neighbor 172.16.200.3# update-source loopback0
border1 running neighbor 172.16.200.3# address-family ipv4-unicast nexthop-self force true
border1 running neighbor 172.16.200.3# address-family ipv4-unicast soft-reconfiguration-inbound true
border1 running neighbor 172.16.200.3# ..

border1 running bgp# neighbor 172.16.200.4
border1 running neighbor 172.16.200.4#! remote-as 65200
border1 running neighbor 172.16.200.4# neighbor-description PE2
border1 running neighbor 172.16.200.4# update-source loopback0
border1 running neighbor 172.16.200.4# address-family ipv4-unicast nexthop-self force true
border1 running neighbor 172.16.200.4# address-family ipv4-unicast soft-reconfiguration-inbound true
border1 running neighbor 172.16.200.4# ..
```

3.3.4.2. eBGP

Configure peering with ISPs:

```
border1 running bgp# neighbor 1.1.1.1
border1 running neighbor 1.1.1.1#! remote-as 100
border1 running neighbor 1.1.1.1# neighbor-description Transit1-IPv4
border1 running neighbor 1.1.1.1# update-source loopback0
border1 running neighbor 1.1.1.1# address-family ipv4-unicast soft-reconfiguration-inbound true
border1 running neighbor 1.1.1.1# ..

border1 running bgp# neighbor 2.2.2.1
border1 running neighbor 2.2.2.1#! remote-as 200
border1 running neighbor 2.2.2.1# neighbor-description Transit2-IPv4
border1 running neighbor 2.2.2.1# update-source loopback0
border1 running neighbor 2.2.2.1# address-family ipv4-unicast soft-reconfiguration-inbound true
border1 running neighbor 2.2.2.1# ..

border1 running bgp# neighbor 3.3.3.1
border1 running neighbor 3.3.3.1#! remote-as 300
border1 running neighbor 3.3.3.1# neighbor-description Transit3-IPv4
border1 running neighbor 3.3.3.1# update-source loopback0
border1 running neighbor 3.3.3.1# address-family ipv4-unicast soft-reconfiguration-inbound true
border1 running neighbor 3.3.3.1# ..
```

In order to direct traffic across a specific border router, we will change the metric of the locally originated prefixes by way of a route-map:

```
border1 running config# / routing
border1 running routing# ipv4-prefix-list prefix-local-origin
border1 running ipv4-prefix-list prefix-local-origin# seq 10 address 200.200.208.0/20 policy permit le 32
border1 running routing# / vrf main routing route-map TRANSIT-OUT
border1 running route-map TRANSIT_OUT# seq 1 match ip address prefix-list prefix-local-origin
border1 running route-map TRANSIT_OUT# seq 1 set metric 100
border1 running route-map TRANSIT_OUT# .. bgp
border1 running bgp# neighbor 1.1.1.1 address-family ipv4-unicast route-map out route-map-name TRANSIT-OUT
border1 running bgp# neighbor 2.2.2.1 address-family ipv4-unicast route-map out route-map-name TRANSIT-OUT
border1 running bgp# neighbor 3.3.3.1 address-family ipv4-unicast route-map out route-map-name TRANSIT-OUT
border1 running bgp# commit
```

We can optimize the configuration further by filtering out possible bogus IP addresses we could receive:

```
border1 running config# / routing ipv4-prefix-list filter-bogons
border1 running ipv4-prefix-list filter-bogons# seq 5 address 0.0.0.0/8 policy deny le 32
border1 running ipv4-prefix-list filter-bogons# seq 10 address 10.0.0.0/8 policy deny le 32
border1 running ipv4-prefix-list filter-bogons# seq 15 address 127.0.0.0/8 policy deny le 32
border1 running ipv4-prefix-list filter-bogons# seq 20 address 169.254.0.0/16 policy deny le 32
border1 running ipv4-prefix-list filter-bogons# seq 25 address 172.16.0.0/12 policy deny le 32
border1 running ipv4-prefix-list filter-bogons# seq 30 address 192.168.0.0/16 policy deny le 32
border1 running ipv4-prefix-list filter-bogons# seq 35 address 224.0.0.0/3 policy deny le 32
border1 running ipv4-prefix-list filter-bogons# seq 40 address 0.0.0.0/0 policy deny ge 25
border1 running ipv4-prefix-list filter-bogons# seq 45 address 0.0.0.0/0 policy permit le 32

border1 running ipv4-prefix-list filter-bogons# / vrf main routing bgp
border1 running bgp# neighbor 1.1.1.1 prefix-list in prefix-list-name filter-bogons
border1 running bgp# neighbor 2.2.2.1 prefix-list in prefix-list-name filter-bogons
border1 running bgp# neighbor 3.3.3.1 prefix-list in prefix-list-name filter-bogons
border1 running bgp# commit
```

See the User’s Guide for more information regarding:

- BGP

3.3.5. Route optimization through BGP FlowSpec and sFlow

The IRP monitoring station runs route optimization software that relies on sFlow for collecting traffic statistics from the border router and on BGP Flowspec to inject Policy-Based Routing rules to redirect a specific traffic through a transit router or another.

This section details the sFlow and BGP configuration on the border router for this purpose.

Configure sFlow on the loopback interface, reporting information from the VLAN interfaces connected to the transit routers:

```
border1 running config# / vrf main sflow
border1 running sflow# agent-interface loopback0
border1 running sflow# sflow-collector 172.16.100.253
border1 running sflow# sflow-collector 172.16.100.254
border1 running sflow# sflow-interface vlan1
border1 running sflow# sflow-interface vlan2
border1 running sflow# sflow-interface vlan3
border1 running sflow# sflow-sampling speed 40G
border1 running sflow# sflow-sampling speed 10G rate 10000
border1 running sflow# /
border1 running config# commit
```

See the User's Guide for more information regarding:

- sFlow

Add the IRP monitoring station as a BGP Flowspec peer:

```
border1 running config# / vrf main routing bgp
border1 running bgp# neighbor 172.16.100.253
border1 running neighbor 172.16.100.253#! remote-as 65200
border1 running neighbor 172.16.100.253# neighbor-description IRP
border1 running neighbor 172.16.100.253# address-family ipv4-unicast soft-reconfiguration-inbound true
border1 running neighbor 172.16.100.253# address-family ipv4-unicast route-reflector-client true
border1 running neighbor 172.16.100.253# address-family ipv4-flowspec soft-reconfiguration-inbound true
border1 running neighbor 172.16.100.253# address-family ipv4-flowspec route-reflector-client true
border1 running neighbor 172.16.100.253# commit
```

See the User's Guide for more information regarding:

- Flowspec

3.4. Monitoring

For remote monitoring, the vRouter supports:

- SNMP
- Exporting KPIs to a time-series database for viewing system counters (CPU usage, IP statistics and many more). These are displayable via a graphical dashboard (for instance Grafana) for a very convenient remote view of the router health & status.
- sFlow for statistical sampling on selected interfaces

3.4.1. SNMP

The following example shows a minimal SNMP setup:

```
border1> edit running
border1 running config# / vrf main snmp
border1 running snmp# static-info contact "noc@6wind.com"
border1 running snmp# static-info location "paris"
border1 running snmp# community local authorization read-only
border1 running snmp# community local source 127.0.0.1
border1 running snmp# community ems authorization read-only
border1 running snmp# community ems source 172.16.100.254
border1 running snmp# /
border1 running vrf main# commit
```

See the User's Guide for more information regarding:

- SNMP

3.4.2. KPI and dash-board

Here we will show how to export KPIs to a time-series database which can then be used with a graphical tool like Grafana.

```
border1> edit running
border1 running config# system kpi enabled true
border1 running config# vrf main kpi
border1 running kpi# interface ntfp1
border1 running kpi# interface ntfp2
border1 running kpi# interface ntfp3
border1 running kpi# telegraf influxdb-output url http://172.16.100.254:8086 database telegraf
border1 running kpi# /
border1 running config# commit
```


See the User's Guide for more information regarding:

- KPI

For information on installing Grafana, please visit:

- 6WIND Grafana Setup on github

3.5. Troubleshooting

3.5.1. CLI show commands

The CLI incorporates a number of show commands of which a few are shown here.

Showing the current basic state of an interface (add a command qualifier for more detail) :

```
border1> show interface name ntfp1
10: ntfp1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP mode DEFAULT group default qlen
1000
        link/ether de:ed:01:29:7e:0e brd ff:ff:ff:ff:ff:ff
```

Basic interface UDP traffic dump example:

```
border1> cmd show-traffic ntfp1 filter udp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ntfp1, link-type EN10MB (Ethernet), capture size 262144 bytes
18:38:47.221472 de:ed:01:e3:55:78 > de:ed:01:07:da:e2, ethertype IPv4 (0x0800), length 746:
172.16.100.2.45791 > 172.16.100.254.6343: sFlowv5, IPv4 agent 172.16.200.2, agent-id 100000, length 704
18:38:47.221482 de:ed:01:e3:55:78 > de:ed:01:07:da:e2, ethertype IPv4 (0x0800), length 746:
172.16.100.2.45791 > 172.16.100.254.6343: sFlowv5, IPv4 agent 172.16.200.2, agent-id 100000, length 704
18:38:47.221484 de:ed:01:e3:55:78 > de:ed:01:1b:a5:56, ethertype IPv4 (0x0800), length 746:
172.16.100.2.45791 > 172.16.100.253.6343: sFlowv5, IPv4 agent 172.16.200.2, agent-id 100000, length 704
18:38:47.221485 de:ed:01:e3:55:78 > de:ed:01:1b:a5:56, ethertype IPv4 (0x0800), length 746:
172.16.100.2.45791 > 172.16.100.253.6343: sFlowv5, IPv4 agent 172.16.200.2, agent-id 100000, length 704
^C
4 packets captured
4 packets received by filter
0 packets dropped by kernel
```

See the User's Guide for more information regarding:

- Show Traffic

The first obvious choice to troubleshoot connectivity problems is to verify that all the routes are in the routing table using the following command:

```
border1> show ipv4-routes
Codes: K - kernel route, C - connected, S - static, R - RIP,
        O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
        T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
        F - PBR,
        > - selected route, * - FIB route

K>* 0.0.0.0/0 [0/0] via 10.0.2.2, ens3, 00:28:35
O   1.1.1.0/24 [110/20] via 172.16.100.2, ntfp1, 00:27:22
C>* 1.1.1.0/24 is directly connected, vlan1, 00:28:19
O   2.2.2.0/24 [110/20] via 172.16.100.2, ntfp1, 00:27:22
C>* 2.2.2.0/24 is directly connected, vlan2, 00:28:19
O   3.3.3.0/24 [110/20] via 172.16.100.2, ntfp1, 00:27:22
C>* 3.3.3.0/24 is directly connected, vlan3, 00:28:18
B   10.0.2.0/24 [200/0] via 172.16.200.3, 00:27:21
                        via 172.16.200.4, 00:27:21
O   10.0.2.0/24 [110/20] via 172.16.100.2, ntfp1, 00:27:22
```

```
                        via 172.16.100.3, ntfp1, 00:27:22
                        via 172.16.100.4, ntfp1, 00:27:22
C>* 10.0.2.0/24 is directly connected, ens3, 00:28:35
B   172.16.100.0/24 [200/0] via 172.16.200.3, 00:27:21
                                via 172.16.200.4, 00:27:21
O   172.16.100.0/24 [110/10] is directly connected, ntfp1, 00:27:23
C>* 172.16.100.0/24 is directly connected, ntfp1, 00:28:19
C>* 172.16.200.1/32 is directly connected, loopback0, 00:28:34
O>* 172.16.200.2/32 [110/20] via 172.16.100.2, ntfp1, 00:27:22
B   172.16.200.3/32 [200/0] via 172.16.200.3, 00:27:21
O>* 172.16.200.3/32 [110/20] via 172.16.100.3, ntfp1, 00:27:22
B   172.16.200.4/32 [200/0] via 172.16.200.4, 00:27:21
O>* 172.16.200.4/32 [110/20] via 172.16.100.4, ntfp1, 00:27:22
B   200.200.210.0/24 [200/0] via 172.16.200.3, 00:27:21
O>* 200.200.210.0/24 [110/20] via 172.16.100.3, ntfp1, 00:27:22
B   200.200.220.0/24 [200/0] via 172.16.200.4, 00:27:21
O>* 200.200.220.0/24 [110/20] via 172.16.100.4, ntfp1, 00:27:22
B>* 217.151.210.0/24 [200/0] via 1.1.1.1, vlan1, 00:27:20
B>* 217.151.211.0/24 [200/0] via 2.2.2.1, vlan2, 00:27:20
B>* 217.151.212.0/24 [20/0] via 3.3.3.1, vlan3, 00:28:13
```

Refining the show command, we can first look at the OSPF routes:

```
border1> show ospf route
VRF Name: default
============ OSPF network routing table ============
N       172.16.100.0/24         [10] area: 0.0.0.0
                        directly attached to ntfp1

============ OSPF router routing table =============
R       172.16.200.2            [10] area: 0.0.0.0, ASBR
                        via 172.16.100.2, ntfp1
R       172.16.200.3            [10] area: 0.0.0.0, ASBR
                        via 172.16.100.3, ntfp1
R       172.16.200.4            [10] area: 0.0.0.0, ASBR
                        via 172.16.100.4, ntfp1

============ OSPF external routing table ===========
N E2 1.1.1.0/24         [10/20] tag: 0
                        via 172.16.100.2, ntfp1
N E2 2.2.2.0/24         [10/20] tag: 0
                        via 172.16.100.2, ntfp1
N E2 3.3.3.0/24         [10/20] tag: 0
                        via 172.16.100.2, ntfp1
N E2 10.0.2.0/24        [10/20] tag: 0
                        via 172.16.100.2, ntfp1
                        via 172.16.100.3, ntfp1
                        via 172.16.100.4, ntfp1
N E2 172.16.200.2/32    [10/20] tag: 0
                        via 172.16.100.2, ntfp1
N E2 172.16.200.3/32    [10/20] tag: 0
                        via 172.16.100.3, ntfp1
N E2 172.16.200.4/32    [10/20] tag: 0
                        via 172.16.100.4, ntfp1
N E2 200.200.210.0/24   [10/20] tag: 0
                        via 172.16.100.3, ntfp1
N E2 200.200.220.0/24   [10/20] tag: 0
                        via 172.16.100.4, ntfp1
```

If OSPF routes seem to be missing, try verifying that OSPF has formed the correct neighbor relationships:

```
border1> show ospf neighbor
VRF Name: default

Neighbor ID     Pri State           Dead Time Address         Interface            RXmtL RqstL
DBsmL
172.16.200.2      1 2-Way/DROther     36.233s 172.16.100.2    ntfp1:172.16.100.1       0     0
        0
172.16.200.3      1 Full/Backup       34.142s 172.16.100.3    ntfp1:172.16.100.1       0     0
        0
172.16.200.4      1 Full/DR           33.873s 172.16.100.4    ntfp1:172.16.100.1       0     0
```

```
        0
```

And we can also verify the OSPF topology database:

```
border1> show ospf database
VRF Name: default

       OSPF Router with ID (172.16.200.1)

                Router Link States (Area 0.0.0.0)

Link ID         ADV Router      Age  Seq#       CkSum  Link count
172.16.200.1    172.16.200.1     682 0x80000008 0x5e8e 1
172.16.200.2    172.16.200.2     684 0x80000008 0x5c8d 1
172.16.200.3    172.16.200.3     675 0x80000009 0xb2d8 1
172.16.200.4    172.16.200.4     796 0x80000008 0xb2d6 1

                Net Link States (Area 0.0.0.0)

Link ID         ADV Router      Age  Seq#       CkSum
172.16.100.4    172.16.200.4     736 0x80000006 0xbbc2

                AS External Link States

Link ID         ADV Router      Age  Seq#       CkSum  Route
1.1.1.0         172.16.200.1     722 0x80000006 0x67c8 E2 1.1.1.0/24 [0x0]
1.1.1.0         172.16.200.2     724 0x80000006 0x61cd E2 1.1.1.0/24 [0x0]
2.2.2.0         172.16.200.1     752 0x80000006 0x43e9 E2 2.2.2.0/24 [0x0]
2.2.2.0         172.16.200.2     754 0x80000006 0x3dee E2 2.2.2.0/24 [0x0]
3.3.3.0         172.16.200.1     742 0x80000006 0x1f0b E2 3.3.3.0/24 [0x0]
3.3.3.0         172.16.200.2     744 0x80000006 0x1910 E2 3.3.3.0/24 [0x0]
10.0.2.0        172.16.200.1     702 0x80000006 0xf234 E2 10.0.2.0/24 [0x0]
10.0.2.0        172.16.200.2     704 0x80000006 0xec39 E2 10.0.2.0/24 [0x0]
10.0.2.0        172.16.200.3     715 0x80000007 0xe43f E2 10.0.2.0/24 [0x0]
10.0.2.0        172.16.200.4     706 0x80000007 0xde44 E2 10.0.2.0/24 [0x0]
172.16.200.1    172.16.200.1     762 0x80000006 0x5b51 E2 172.16.200.1/32 [0x0]
172.16.200.2    172.16.200.2     764 0x80000006 0x4b5f E2 172.16.200.2/32 [0x0]
172.16.200.3    172.16.200.3     755 0x80000007 0x396e E2 172.16.200.3/32 [0x0]
172.16.200.4    172.16.200.4     696 0x80000007 0x297c E2 172.16.200.4/32 [0x0]
200.200.210.0   172.16.200.3     735 0x80000007 0xd2f8 E2 200.200.210.0/24 [0x0]
200.200.220.0   172.16.200.4     766 0x80000007 0x5e62 E2 200.200.220.0/24 [0x0]
```

If 2-way and FULL states have not been established between the OSPF neighbors, check that all OSPF interface settings are correct. All usual OSPF neighborship requirements must be fulfilled.

The next step would be to enable OSPF logging as shown under the CLI log commands section.

Now, let's check BGP.

Verify the BGP routes:

```
border1> show bgp ipv4
BGP table version is 11, local router ID is 172.16.200.1, vrf id 0
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
*=i10.0.2.0/24      172.16.200.4        0       100      0 ?
*>i                 172.16.200.3        0       100      0 ?
*=i172.16.100.0/24  172.16.200.4        0       100      0 ?
*>i                 172.16.200.3        0       100      0 ?
*>i172.16.200.3/32  172.16.200.3        0       100      0 ?
```

```
*>i172.16.200.4/32  172.16.200.4             0       100     0 ?
*>i200.200.210.0        172.16.200.3           0       100     0 ?
*>i200.200.220.0        172.16.200.4           0       100     0 ?
*>i217.151.210.0        1.1.1.1                0       100     0 100 i
*>i217.151.211.0        2.2.2.1                0       100     0 200 i
* i217.151.212.0        3.3.3.1                0       100     0 300 i
*>                      3.3.3.1                0               0 300 i

Displayed  9 routes and 12 total paths
```

Let's check BGP neighbors; in this example just the Transit_3 neighbor for brevity:

```
border1> show bgp neighbor 3.3.3.1
 BGP neighbor is 3.3.3.1, remote AS 300, local AS 65200, external link
 Description: Transit3-IPv4
Hostname: transit3-vm
  BGP version 4, remote router ID 7.7.7.7
  BGP state = Established, up for 00:30:02
  Last read 00:00:02, Last write 00:00:02
  Hold time is 180, keepalive interval is 60 seconds
  Neighbor capabilities:
        4 Byte AS: advertised and received
        AddPath:
        IPv4 Unicast: RX advertised IPv4 Unicast and received
        Route refresh: advertised and received(old & new)
        Address Family IPv4 Unicast: advertised and received
        Address Family IPv6 Unicast: received
        Hostname Capability: advertised (name: border1,domain name: n/a) received (name:
transit3-vm,domain name: n/a)
        Graceful Restart Capabilty: advertised and received
        Remote Restart timer is 120 seconds
        Address families by peer:
        none
  Graceful restart informations:
        End-of-RIB send: IPv4 Unicast
        End-of-RIB received: IPv4 Unicast
  Message statistics:
        Inq depth is 0
        Outq depth is 0
                        Sent    Rcvd
        Opens:                  1       1
        Notifications:          0       0
        Updates:                3       4
        Keepalives:             31      31
        Route Refresh:          0       0
        Capability:             0       0
        Total:                  35      36
  Minimum time between advertisement runs is 0 seconds

 For address family: IPv4 Unicast
  Update group 1, subgroup 1
  Packet Queue length 0
  Inbound soft reconfiguration allowed
  Community attribute sent to this neighbor(all)
  Inbound path policy configured
  Outbound path policy configured
  Incoming update prefix filter list is *filter-bogons
  Route map for outgoing advertisements is *TRANSIT-OUT
  1 accepted prefixes

  Connections established 1; dropped 0
  Last reset never
Local host: 3.3.3.2, Local port: 40048
Foreign host: 3.3.3.1, Foreign port: 179
Nexthop: 3.3.3.2
Nexthop global: fe80::dced:1ff:fed8:6d1c
Nexthop local: fe80::dced:1ff:fed8:6d1c
BGP connection: shared network
BGP Connect Retry Timer in Seconds: 120
Read thread: on  Write thread: on
```

Verify BGP flowspec (so far in this case nothing to show):

```
border1> show bgp ipv4 flowspec
No BGP prefixes displayed, 0 exist
```

Many more show commands are available, please check in the User's Guide as appropriate.

3.5.2. CLI log commands

To display the system log locally (kernel logs in this case):

```
border1> show log kernel
-- Logs begin at Tue 2019-07-09 14:37:46 UTC, end at Tue 2019-07-09 21:03:52 UTC. --
Jul 09 14:40:24 border1 kernel: Silicon Labs C2 port support v. 0.51.0 - (C) 2007 Rodolfo Giometti
Jul 09 14:40:31 border1 kernel: VFIO - User Level meta-driver version: 0.3
Jul 09 14:40:32 border1 kernel: iommu: Adding device 0000:00:04.0 to group 0
Jul 09 14:40:32 border1 kernel: vfio-pci 0000:00:04.0: Adding kernel taint for vfio-noiommu group on
device
Jul 09 14:40:32 border1 kernel: iommu: Adding device 0000:00:05.0 to group 1
Jul 09 14:40:32 border1 kernel: vfio-pci 0000:00:05.0: Adding kernel taint for vfio-noiommu group on
device
Jul 09 14:40:32 border1 kernel: iommu: Adding device 0000:00:06.0 to group 2
Jul 09 14:40:32 border1 kernel: vfio-pci 0000:00:06.0: Adding kernel taint for vfio-noiommu group on
device
Jul 09 14:40:33 border1 kernel: dpvi: loading out-of-tree module taints kernel.
Jul 09 14:40:33 border1 kernel: dpvi: module verification failed: signature and/or required key missing
- tainting kernel
Jul 09 14:40:33 border1 kernel: dpvi_shmem: dpvi_shmem module initialized 00000000bfa363e7
```

To specifically look at routing system (bgp, ospf,..) events:

```
border1> show log service routing
-- Logs begin at Fri 2019-07-26 09:16:24 UTC, end at Fri 2019-07-26 09:47:01 UTC. --
Jul 26 09:18:54 border1 systemd[1]: Started zebra.
Jul 26 09:19:13 border1 systemd[1]: Started bgpd.
Jul 26 09:19:13 border1 systemd[1]: Started ospfd.
```

Logging of BGP neighbor changes:

```
border1> edit running
border1 running config# / vrf main routing bgp
border1 running bgp# log-neighbor-changes true
```

A per VRF remote logging capability can be enabled for the system log:

```
border1> edit running
border1 running config# / vrf main logging syslog
border1 running syslog#! remote-server 172.16.100.253 protocol tcp port 514
border1 running remote-server 172.16.100.253# commit
```

For more detail on logging, please refer to

- System logging

For more details on troubleshooting, refer to:

- Troubleshooting guide

3.6. Optimizing performance

The default limit of 1 Million IPv4 routes may not be sufficient for a border router receiving several full BGP tables. The following example shows how to increase this to 3 Million.

```
border1> edit running
border1 running config# system fast-path limits ip4-max-route 3000000
border1 running config# commit
```

For more details about tuning system capabilities, see:

- Fast path limits configuration

In case the router is overloaded and control packets are lost, the amount of CPU dedicated to prioritizing control plane vs. data plane traffic can be increased using the following command (default is 10%):

```
border1> edit running
border1 running config# system fast-path cp-protection budget 20
border1 running config# commit
```

For more details, see:

- Control plane protection

3.7. Appendix

Listed here is the CLI configuration for the configuration discussed in this use case.

```
border1 running config# show config nodefault
vrf main
    routing
        bgp
            as 65200
            bestpath
                as-path
                    ..
                ..
            client-to-client
                ..
            listen
                ..
            log-neighbor-changes true
            packet-rw-quantum
                ..
            router-id 172.16.200.1
            neighbor 172.16.200.2
                remote-as 65200
                neighbor-description border2
                update-source loopback0
                address-family
                    ipv4-unicast
                        addpath
                            ..
                        soft-reconfiguration-inbound true
                        ..
                    ..
                ..
            neighbor 172.16.200.3
                remote-as 65200
                neighbor-description PE1
                update-source loopback0
                address-family
                    ipv4-unicast
                        addpath
```



```
                    ..
                nexthop-self
                    force true
                    ..
                soft-reconfiguration-inbound true
                ..
            ..
        ..
    neighbor 172.16.200.4
        remote-as 65200
        neighbor-description PE2
        update-source loopback0
        address-family
            ipv4-unicast
                addpath
                    ..
                nexthop-self
                    force true
                    ..
                soft-reconfiguration-inbound true
                ..
            ..
        ..
    neighbor 1.1.1.1
        remote-as 100
        neighbor-description Transit1-IPv4
        address-family
            ipv4-unicast
                addpath
                    ..
                prefix-list in prefix-list-name filter-bogons
                soft-reconfiguration-inbound true
                route-map out route-map-name TRANSIT-OUT
                ..
            ..
        ..
    neighbor 2.2.2.1
        remote-as 200
        neighbor-description Transit2-IPv4
        address-family
            ipv4-unicast
                addpath
                    ..
                prefix-list in prefix-list-name filter-bogons
                soft-reconfiguration-inbound true
                route-map out route-map-name TRANSIT-OUT
                ..
            ..
        ..
    neighbor 3.3.3.1
        remote-as 300
        neighbor-description Transit3-IPv4
        address-family
            ipv4-unicast
                addpath
                    ..
                prefix-list in prefix-list-name filter-bogons
                soft-reconfiguration-inbound true
                route-map out route-map-name TRANSIT-OUT
                ..
            ..
        ..
    neighbor 172.16.100.253
        remote-as 65200
        neighbor-description IRP
        address-family
            ipv4-unicast
                addpath
                    ..
                soft-reconfiguration-inbound true
                route-reflector-client true
                ..
            ipv4-flowspec
                soft-reconfiguration-inbound true
                route-reflector-client true
                ..
```

```
                ..
            ..
        ..
    ospf
        router-id 172.16.200.1
        abr-type standard
        log-adjacency-changes detail
        network 172.16.100.0/24 area 0
        passive-interface loopback0
        redistribute connected
        ..
    ..
interface
    physical ntfp1
        port pci-b0s4
        description Border1_internal
                rx-cp-protection true
                tx-cp-protection true
        ipv4
            address 172.16.100.1/24
            ..
        ipv6
            ..
        ethernet
            auto-negotiate true
            ..
        ..
    physical ntfp2
        port pci-b0s5
        rx-cp-protection true
        tx-cp-protection true
        ipv4
            ..
        ipv6
            ..
        ethernet
            auto-negotiate true
            ..
        ..
    physical ntfp3
        port pci-b0s6
        rx-cp-protection true
        tx-cp-protection true
        ipv4
            ..
        ipv6
            ..
        ethernet
            auto-negotiate true
            ..
        ..
    loopback loopback0
        ipv4
            address 172.16.200.1/32
            ..
        ipv6
            ..
        ..
    vlan vlan1
        description Transit_1
          ipv4
            address 1.1.1.2/24
            ..
        ipv6
            ..
        vlan-id 1
        link-interface ntfp3
        ..
    vlan vlan2
        description Transit_2
         ipv4
            address 2.2.2.2/24
            ..
        ipv6
            ..
        vlan-id 2
```



```
        link-interface ntfp3
        ..
    vlan vlan3
        description Transit_3
         ipv4
            address 3.3.3.2
            ..
        ipv6
            ..
        vlan-id 3
        link-interface ntfp2
        ..
    ..
kpi
    enabled true
    service fp-bridge-stats
    service fp-context-switch-stats
    service fp-cp-protect-stats
    service fp-cpu-usage
    service fp-dpvi-stats
    service fp-ebtables-stats
    service fp-exception-queue-stats
    service fp-exceptions-stats
    service fp-filling
    service fp-global-stats
    service fp-gre-stats
    service fp-gro-stats
    service fp-ip-stats
    service fp-ip6-stats
    service fp-ipsec-stats
    service fp-ipsec6-stats
    service fp-npf-stats
    service fp-ports-stats
    service fp-status
    service fp-vlan-stats
    service fp-vxlan-stats
    service network-nic-eth-stats
    service network-nic-hw-info
    service network-nic-traffic-stats
    service product-license
    service product-version
    service system-cpu-usage
    service system-disk-usage
    service system-memory
    service system-numa-stats
    service system-processes
    service system-soft-interrupts-stats
    service system-uptime
    service system-user-count
    service system-users
    interface ntfp1
    interface ntfp2
    interface ntfp3
    telegraf
        influxdb-output url http://172.16.100.254:8086 database telegraf
        ..
    ..
sflow
    agent-interface loopback0
    sflow-collector 172.16.100.253
    sflow-collector 172.16.100.254
    sflow-interface vlan1
    sflow-interface vlan2
    sflow-interface vlan3
    sflow-sampling speed 40G
    sflow-sampling speed 10G rate 10000
    ..
snmp
    static-info
        location paris
        contact noc@6wind.com
        ..
    community local
        authorization read-only
        source 127.0.0.1
        ..
```

```
        community ems
            authorization read-only
            source 172.16.100.254
            ..
        ..
    ..
system
    hostname border1
    fast-path
        port pci-b0s4
        port pci-b0s5
        port pci-b0s6
        cp-protection
            budget 20
                    ..
        limits
                    ip4-max-route 3000000
                    ..
        linux-sync
            ..
        ..
    ..
routing
    ipv4-prefix-list prefixes-local-originated
        seq 10 address 200.200.208.0/20 policy permit le 32
        ..
    ipv4-prefix-list filter-bogons
        seq 5 address 0.0.0.0/8 policy deny le 32
        seq 10 address 10.0.0.0/8 policy deny le 32
        seq 15 address 127.0.0.0/8 policy deny le 32
        seq 20 address 169.254.0.0/16 policy deny le 32
        seq 25 address 172.16.0.0/12 policy deny le 32
        seq 35 address 192.168.0.0/16 policy deny le 32
        seq 40 address 224.0.0.0/3 policy deny le 32
        seq 45 address 0.0.0.0/0 policy deny ge 25
        seq 50 address 0.0.0.0/0 policy permit le 32
        ..
    route-map TRANSIT-OUT
        seq 1
            policy permit
            match
                ip
                    address
                        prefix-list prefixes-local-originated
                        ..
                    ..
                ..
            set
                metric 100
                ..
            ..
        ..
    bgp
        ..
    ..
```