NTT

Innovative R&D by NTT

Beluganos

“Beluganos”:
Open Network OS for White-box Switches

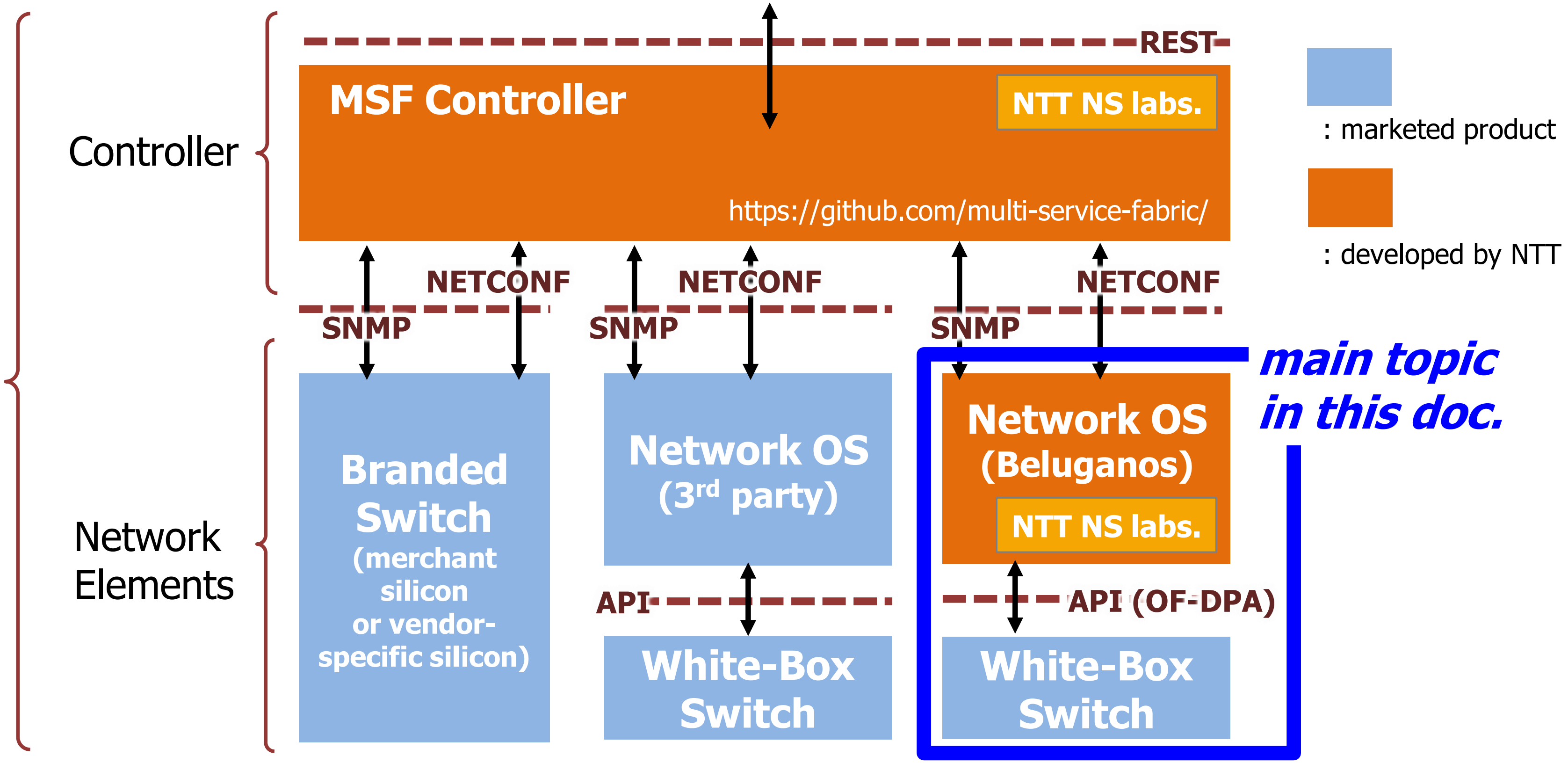
Jun. 2019

NTT Network Service Systems Laboratories

Introduction

MSF（Multi-Service Fabric）

- **MSF（Multi-Service Fabric）** is the architectural design of carrier-grade SDN that provides wide logical network slices with various type routers/switches.
  - □ The targeted routers or switches of MSF are not only branded switches but also white-box switches. Moreover, network OS on the white-box switches is also developed by NTT, which called “**Beluganos**”.



REST

Controller

MSF Controller

NTT NS labs.

https://github.com/multi-service-fabric/

NETCONF

SNMP

NETCONF

SNMP

NETCONF

SNMP

Network Elements

Branded Switch (merchant silicon or vendor-specific silicon)

Network OS (3rd party)

API

White-Box Switch

Network OS (Beluganos)

NTT NS labs.

API (OF-DPA)

White-Box Switch

main topic in this doc.

🟦 : marketed product

🟧 : developed by NTT

Beluganos

Innovative R&D by NTT

Carrier-grade open network OS designed for white-box switch

https://github.com/beluganos/beluganos

Interoperability

- Autonomous MPLS-VPN router
- Interoperability for current IP/MPLS router

Full hardware processing

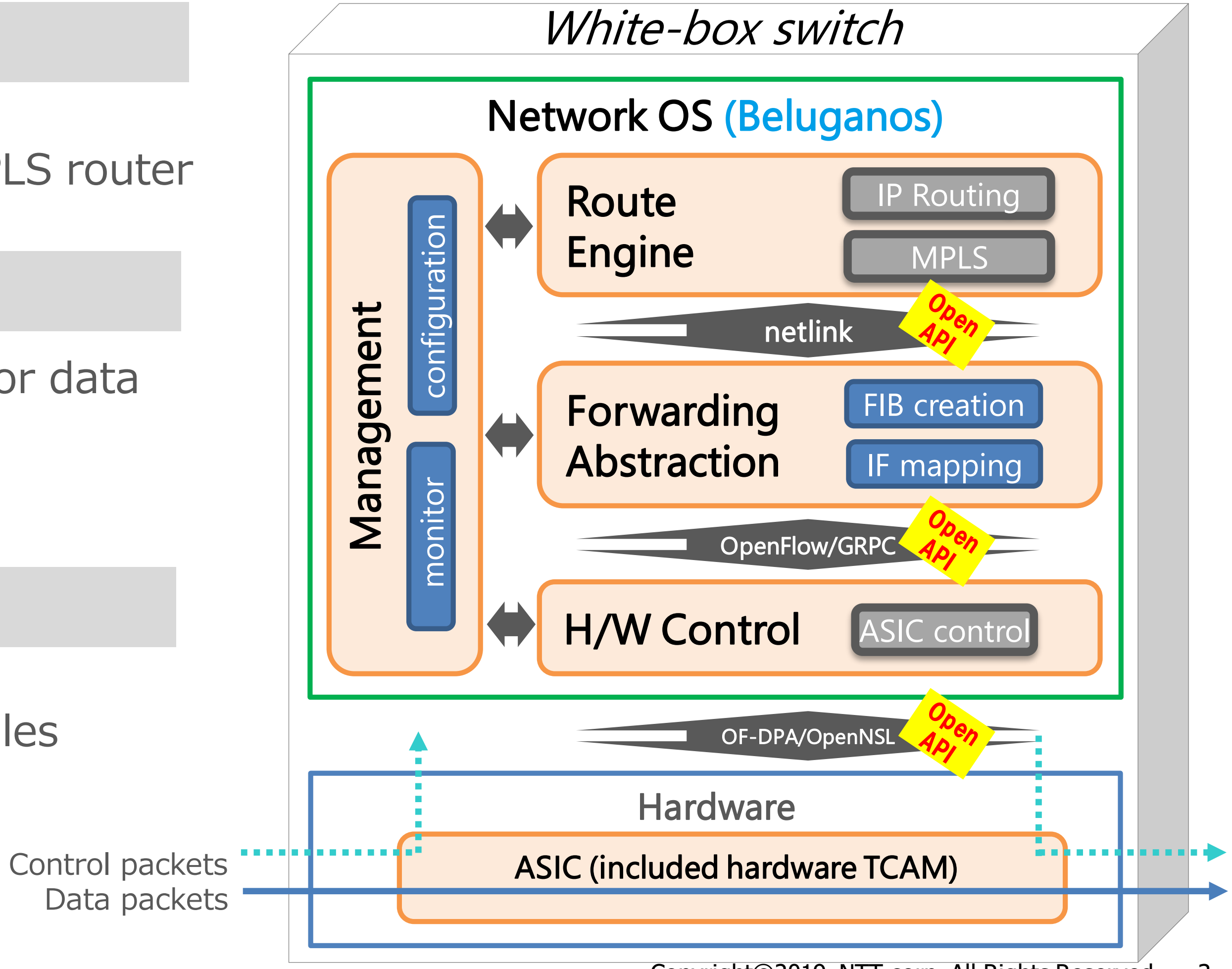
- Full hardware (ASIC) processing for data packets
- hardware TCAM

"White-box software"

- Open API (netlink/OF-DPA/OpenNSL)
- Use of existing open source modules

OSS

NTT original codes



White-box switch

Network OS (Beluganos)

Management

configuration

monitor

Route Engine

IP Routing

MPLS

netlink

Open API

Forwarding Abstraction

FIB creation

IF mapping

OpenFlow/GRPC

Open API

H/W Control

ASIC control

OF-DPA/OpenNSL

Open API

Hardware

ASIC (included hardware TCAM)

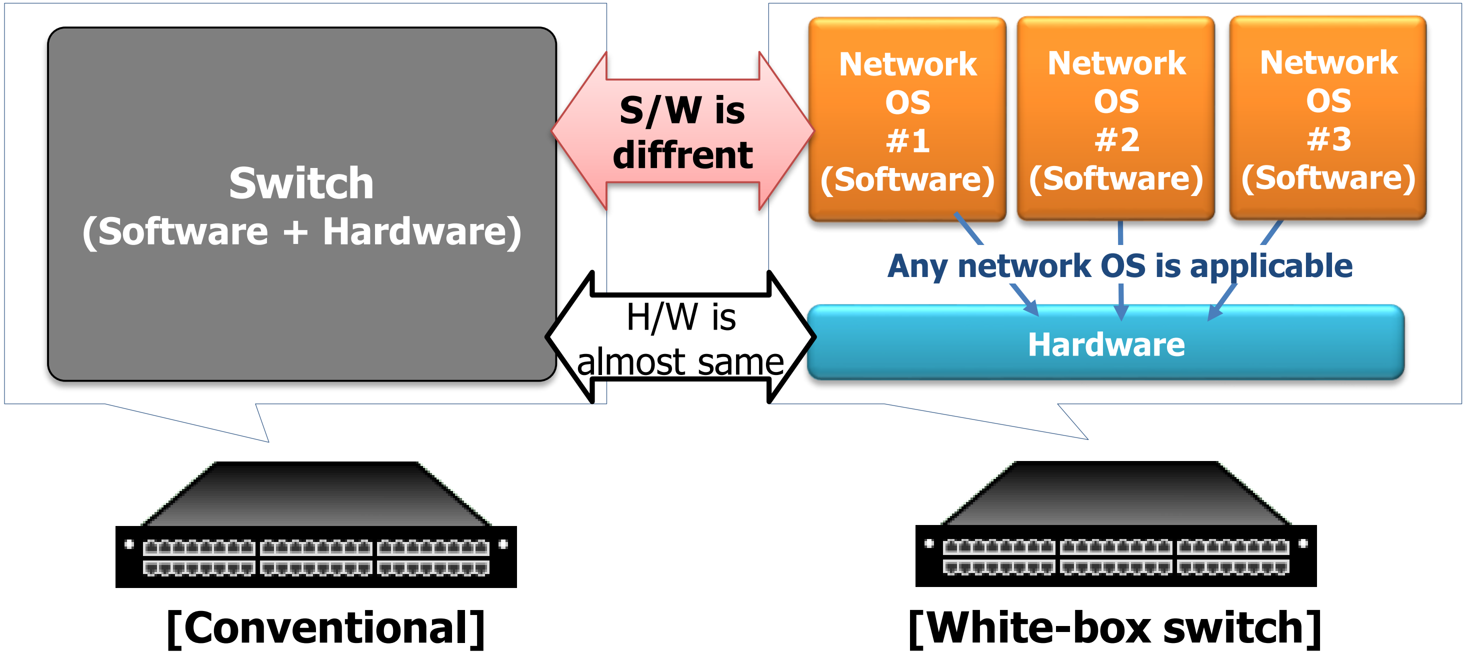
Control packets

Data packets

NTT

White-box switches

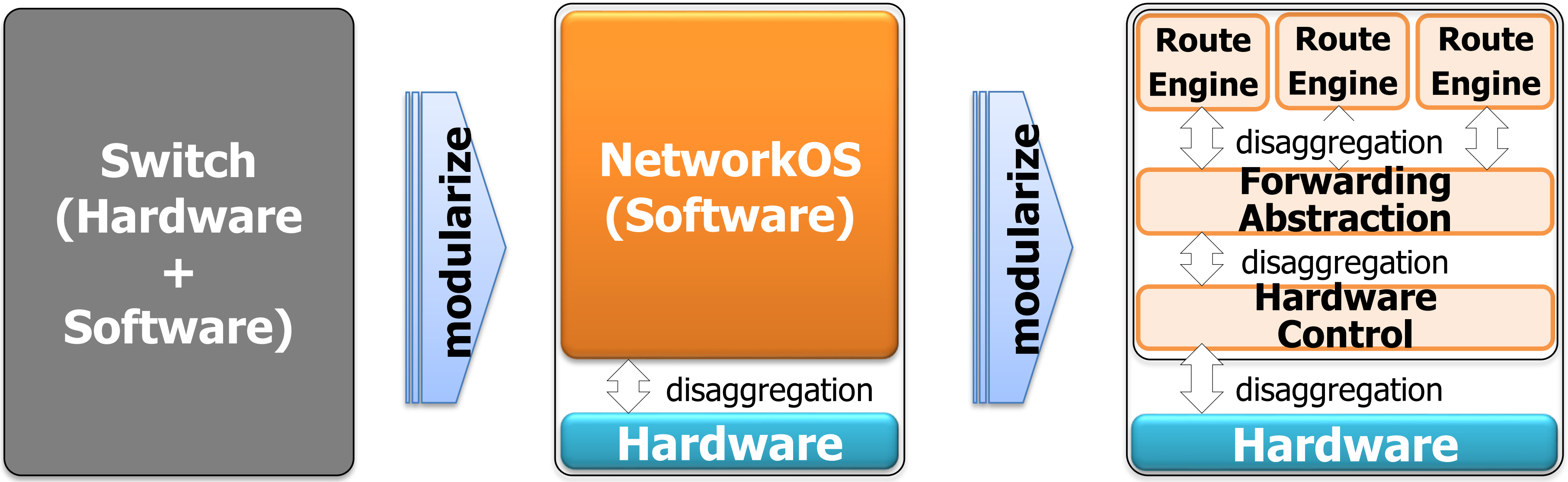
- Only hardware and boot loader. No network OS.
  - Low power consumption, space-saving design.



Beluganos may not pay attention to the difference of switches.

- The abstraction of ASIC : **OpenNSL/OF-DPA**
- The abstraction of H/W : **Open Network Linux**

White-box “software”



[Conventional]

Switch (Hardware + Software)

modularize

[White-box switch]

NetworkOS (Software)

disaggregation

Hardware

modularize

[Beluganos]

Route Engine | Route Engine | Route Engine

disaggregation

Forwarding Abstraction

disaggregation

Hardware Control

disaggregation

Hardware

Vendor-branded, conventional switches

☹ Inflexibility (especially about new H/W or new ASIC)

Disaggregating between hardware and software

☺ Flexibility for new H/W or new ASIC

☹ Remaining inflexibility of new functions to the software

Further disaggregation in terms of software (white-box software)

☺ **Disaggregation**

☺ **Commoditization**

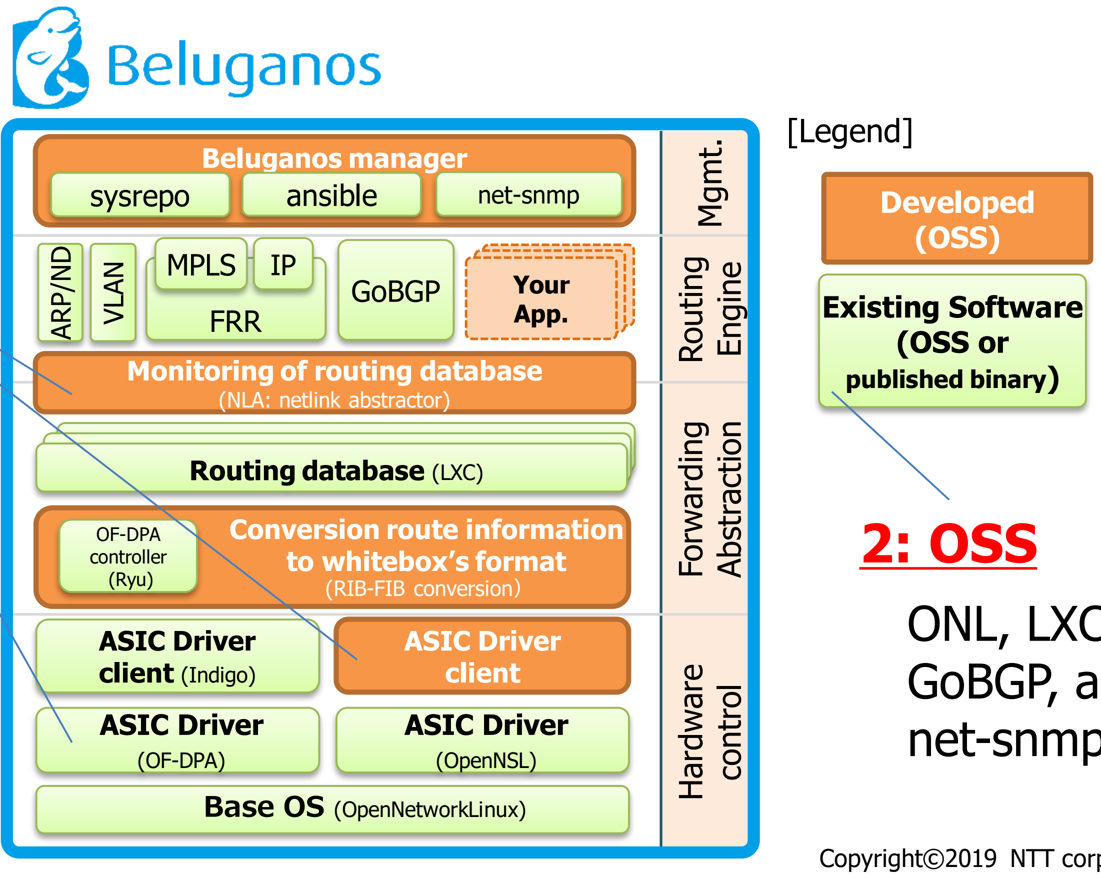
Key technology

Beluganos which supports ASIC packet processing is achieved by

- ✓ 1: **Open API**
- ✓ 2: **Open Source Software (OSS)**

1: OpenAPI

(ASIC)
OpenNSL,
OF-DPA
(Inter-process)
netlink, etc



2: OSS

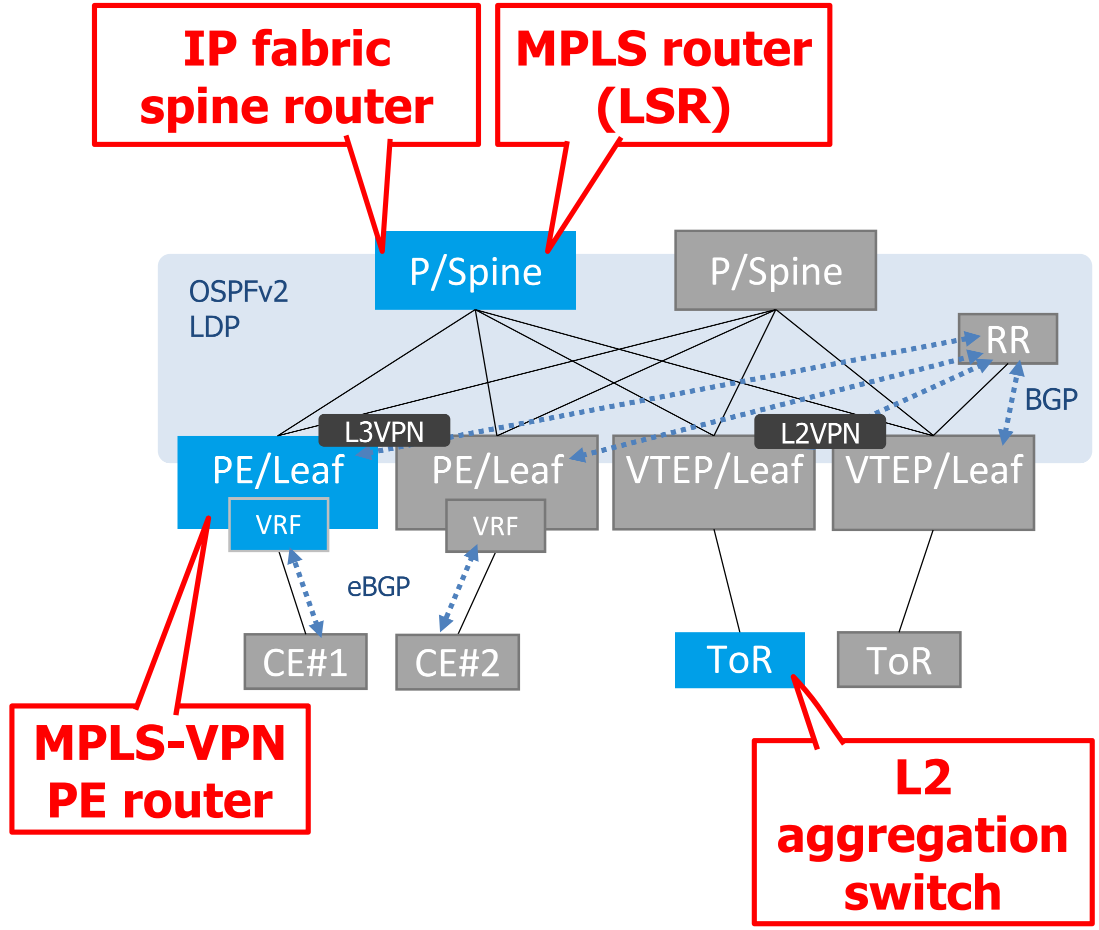
ONL, LXC, FRRouting,
GoBGP, ansible,
net-snmp, etc

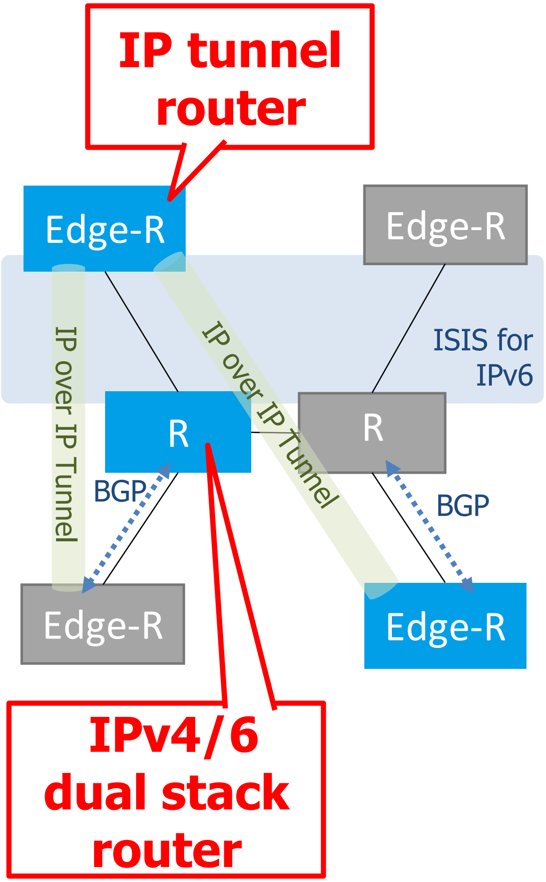
Features

Use case

[Legend] White-box with Beluganos

Conventional router





Features in detail

Note: this plan is subject to change.

Innovative R&D by NTT

Features		Support status		備考
Category	Items	OF-DPA	OpenNSL	
Basic	L2 switching	—	Ready	
	IPv4 unicast routing / ARP	Ready	Ready	
	IPv6 unicast routing / ND	—	Ready	
	IPv4 muticast routing / IGMP	—	TBD	
	IPv6 multicast routing / MLD	—	TBD	
	IP/MPLS forwarding	Ready	Planned	
L2	VLAN（802.1q）	Ready	Ready	Including sub IF for L3 routed port
	Link Aggregation（802.3ad LACP）	Ready	Ready	
	STP、PVST	—	Planned	
	Loop avoidance（storm-control）	—	TBD	
L3 Unicast	Static Routing （IPv4/IPv6）	Ready（IPv4 only）	Ready	
	BGP （IPv4/IPv6）	Ready（IPv4 only）	Ready	GoBGP or FRRouting
	OSPFv2/OSPFv3	Ready（IPv4 only）	Ready	FRRouting
	VRRP（IPv4/IPv6）	TBD	TBD	
	ISIS（IPv4/IPv6）	Ready	Ready	FRRouting
L3 Multicast	PIM-SM/SSM	—	TBD	
L3 MPLS	LDP	Ready	TBD	
	RSVP-TE	TBD	TBD	
	Segment Routing with OSPFv2	Planned	Planned	
VPN	VXLAN（EVPN）	—	Planned	
	IP tunneling（IP-IP）	—	Ready	
	IP-VPN（RFC4364）	Ready	TBD	CE connection is limited at connected/static/bgp
Management/OAM	SSH login	Ready	Ready	
	ICMP（ping, traceroute）	Ready	Ready	
	SNMP (MIB, Trap)	Ready	Ready	IF counter, IF status
	NETCONF/YANG	Ready	Ready	OpenConfig (beluganos/netconf/etc/openconfig)
	SPAN (port mirroring)	Planned	Planned	
	Syslog	Ready	Ready	
Other	Load-balancing（ECMP）	Planned	Planned	
	Access list (ACL)	Planned	Planned	
	Policing	—	Planned	
	Scheduling, Diffserv	—	Planned	
	Logical node（vChassis）	—	Planned	Beyond Pbps

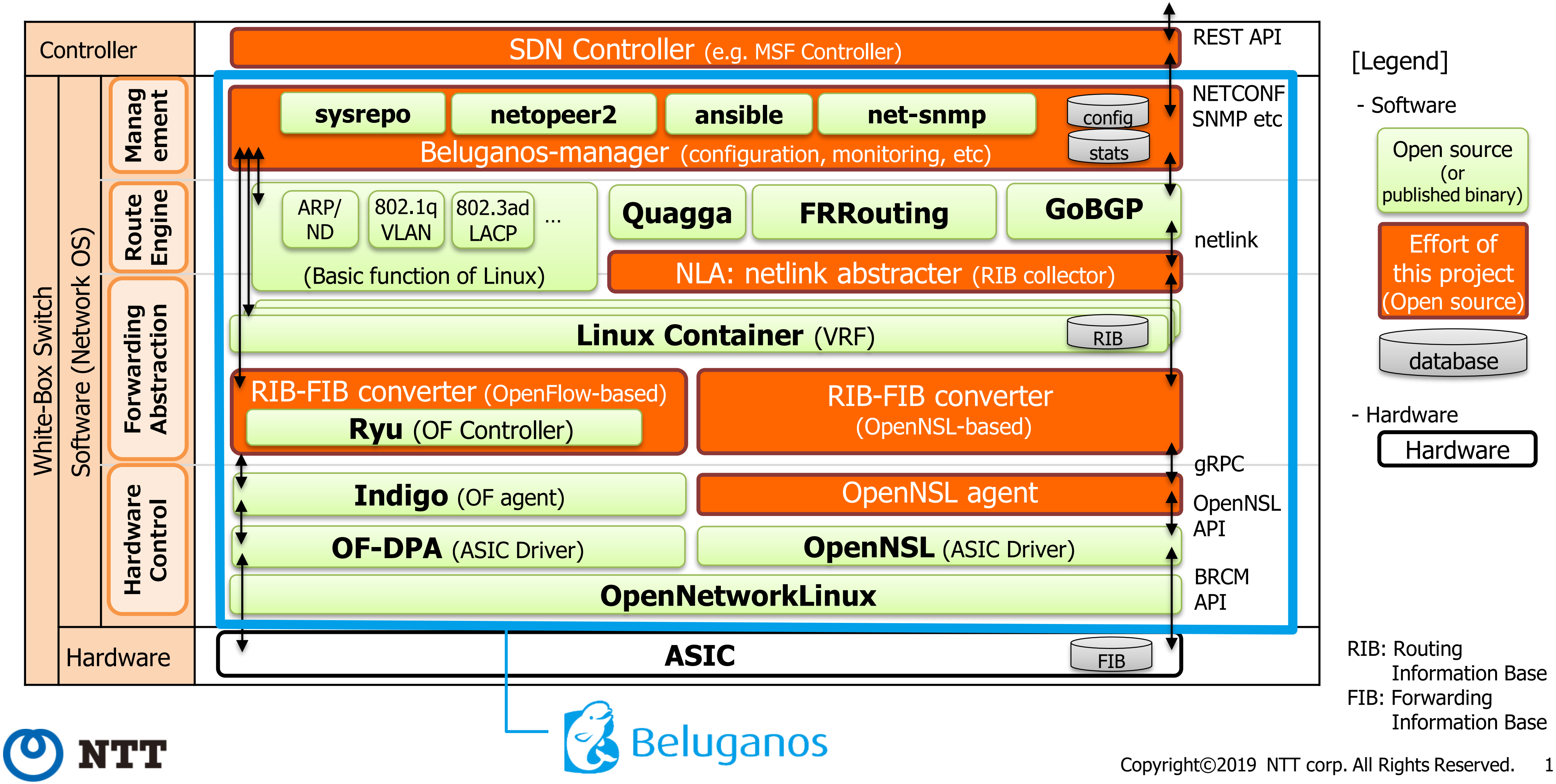
Information about features also described here.
https://github.com/beluganos/beluganos/blob/master/doc/function.md

Technical Details

Feb. 2019 Update

Architecture at a glance

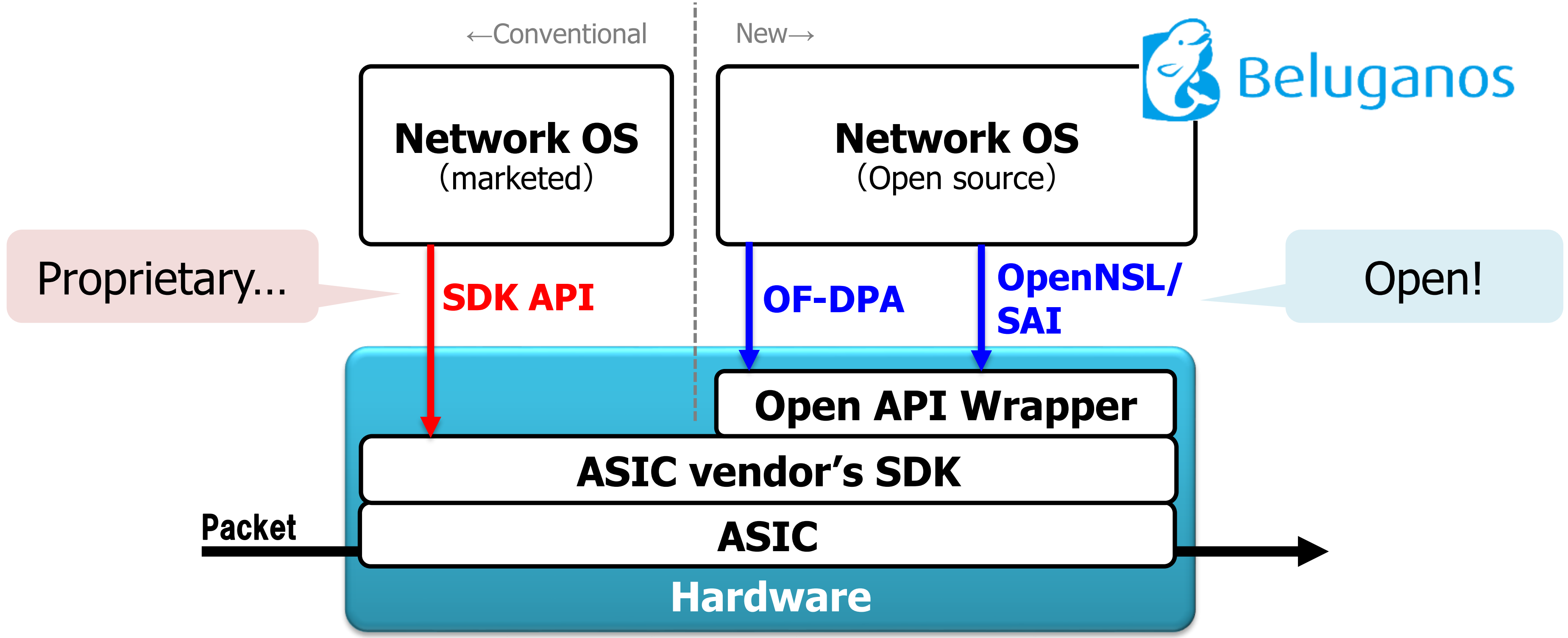
- **Fully opened architecture** which also supports ASIC (H/W) processing
  - □ Point 1 : Beluganos uses only **Open API** (OF-DPA/OpenNSL).
  - □ Point 2 : Beluganos integrates **existed OSS** and our original source code.



RIB: Routing Information Base
FIB: Forwarding Information Base

Architecture:
How to write FIB

- ◆ Calling API of ASIC realizes H/W packet processing.
- ◆ Thanks to white-box evolution, **Open APIs** are prepared to control ASIC.
  → The specification of API is published on Web site.

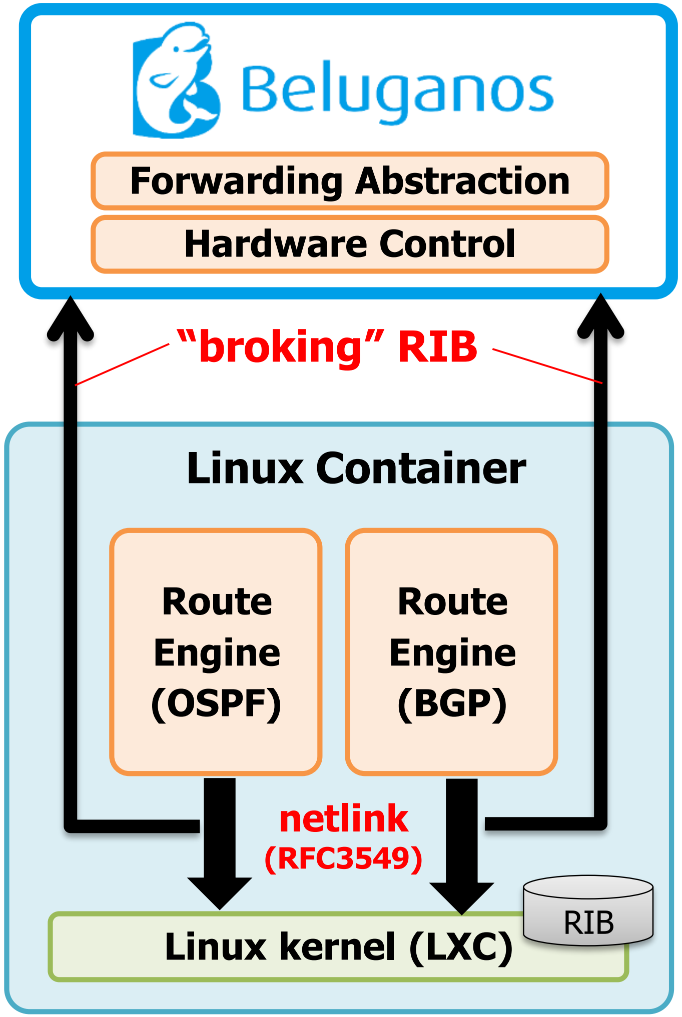


Architecture:

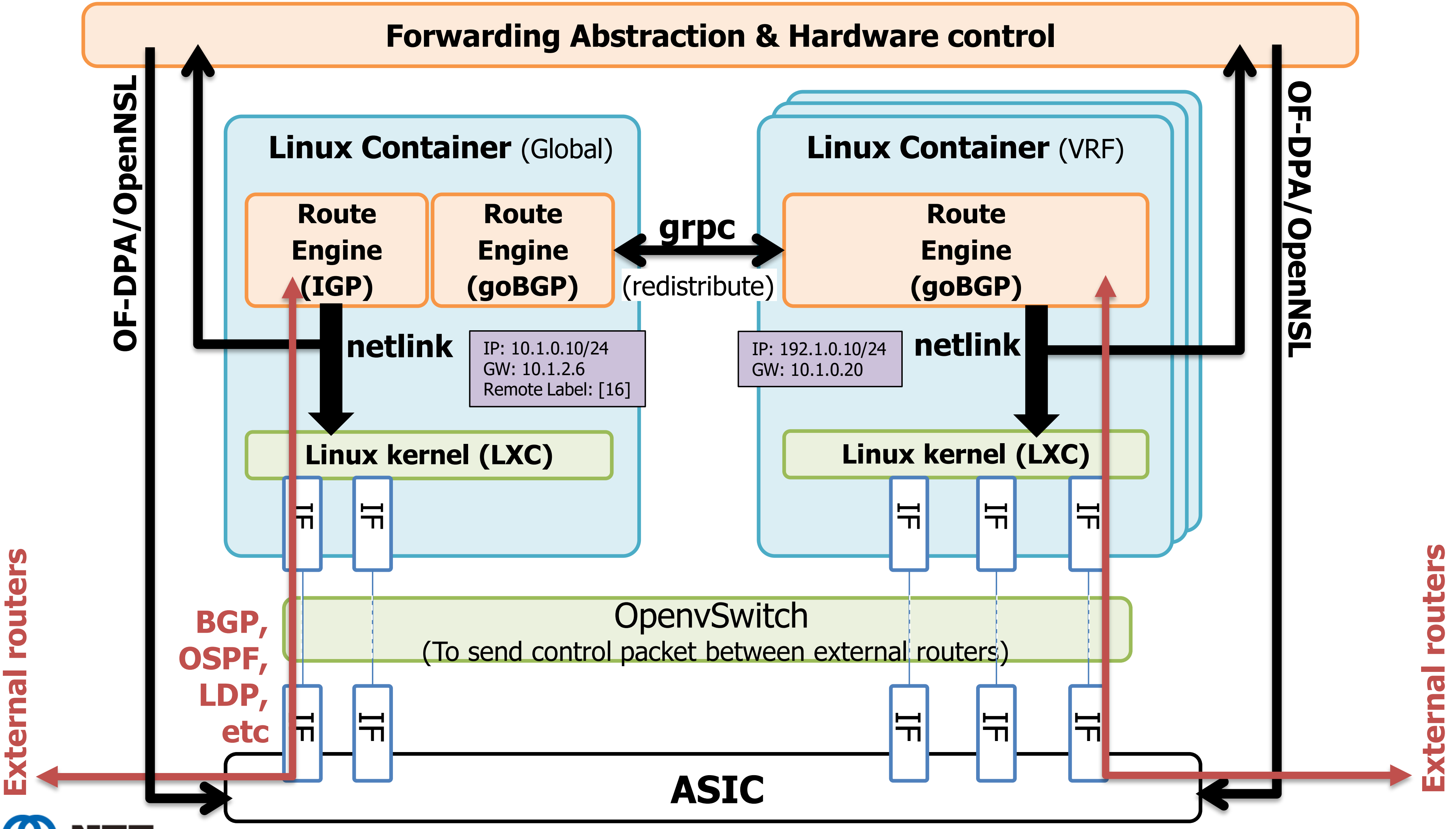
How to calculate RIB

- **netlink** (RFC3549)
  - □ Linux's standard protocol for networking between Linux kernel space and user land.
    - ➢ After Linux 4.1, not only IP networking but also **MPLS networking** is supported.
  - □ In Beluganos, netlink is used for broking RIB information from Linux protocol stack (FRRouting/GoBGP).

- **Linux Container** (LXC)
  - □ Virtualization technology for Linux OS.
  - □ In Beluganos, the **routing table of Linux kernel** is used as RIB. Routing calculation is operated at global LXC.
    - ➢ Multiple LXC is only used for **VRF implementation** (Beluganos also implements route redistribution).

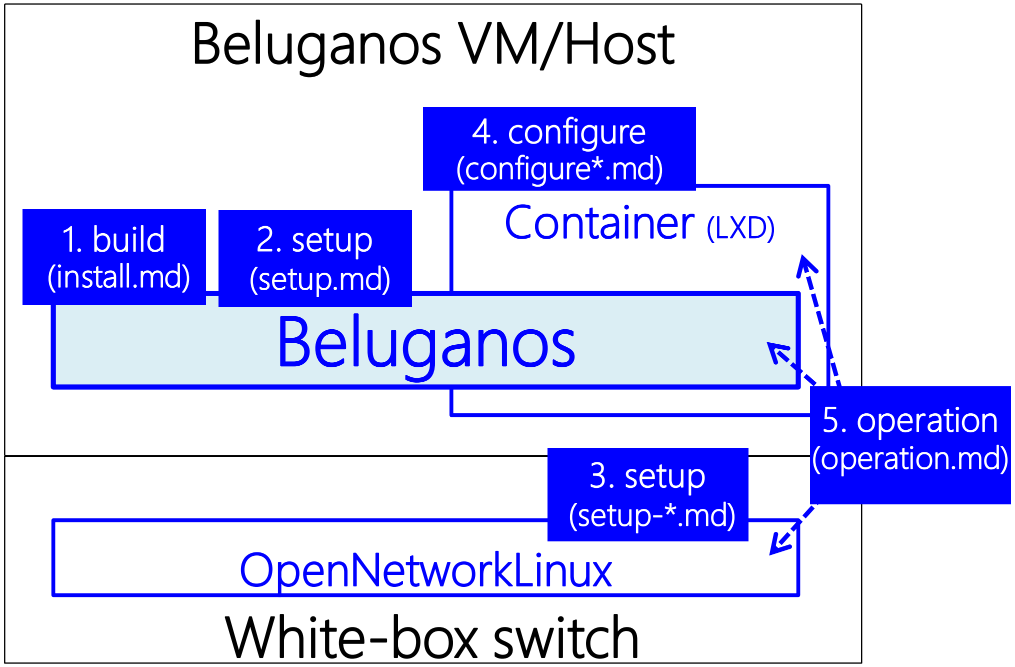


Architecture:
Life of routing info.



Getting Started

Task list to start Beluganos



[Task list]

First time

*Required internet access

1. Build (install.md)
2. Setup to register hardware (setup.md)
3. Setup hardware (setup-*.md)
4. Configure (configure*.md)
5. Start (operation.md)

Once install finished

*Not required internet access

4. Configure (configure*.md)
5. Start (operation.md)

The basic idea of network configuration

Almost all network configuration (IP address, routing protocol, etc.) **should be set to Linux container**, not Linux host OS.

[Method 1]
Directly configuration

Legacy, but all settings can be changed

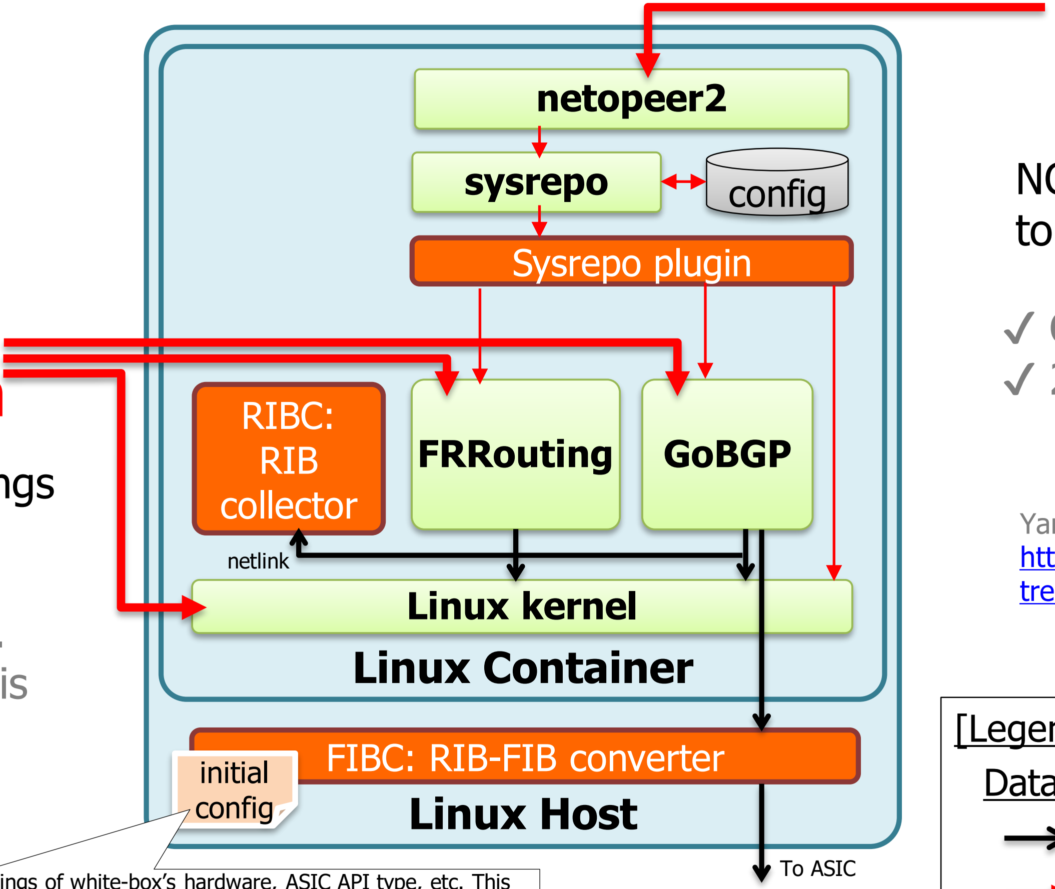
- ✓ vtysh, ip link, etc.
- ✓ ansible playbook is also prepared

[Method 2]
NETCONF via sysrepo

NOT all feature is finished to implement yet

- ✓ Openconfig based yang
- ✓ 2 data store (running, candidate)

Yang files:
https://github.com/beluganos/netconf/tree/master/etc/openconfig



Note: settings of white-box's hardware, ASIC API type, etc. This configuration can be changed by ansible playbook, but you don't have to change it once you set up for the first time.

[Legend]

Datapath
→ RIB info.
→ Config info.

Component
OSS
Beluganos's component (OSS)

Quick start

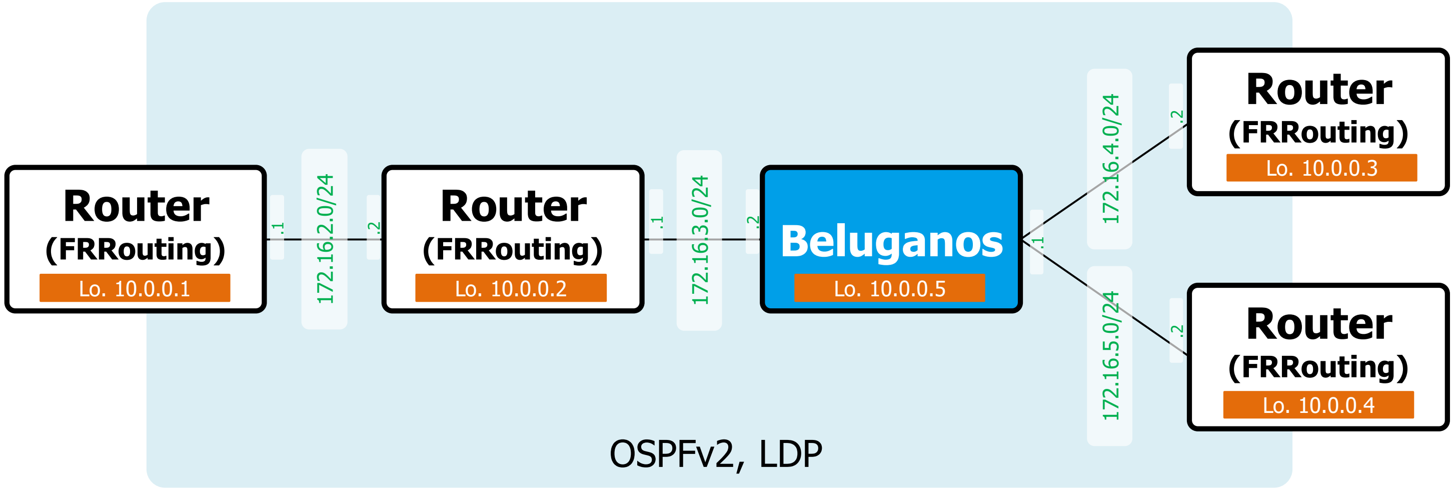
Example case can install and configure automatically **not only Beluganos but also other routers** to connect with Beluganos.

https://github.com/beluganos/beluganos/blob/master/doc/example/case1/case1.md

Note: You do **NOT required** to prepare white-box switches in this case.
OVS is used instead of white-box switches.

Note: You do **NOT required** to check other document like "install-guide.md" in this case.
All required process is described at "case1.md".

[Case 1]



Beluganos