ONIC2022
Routing Platformのオープン化への Ciscoの取り組み

Teppei Kamata(tkamata@cisco.com)

Cisco Systems

Oct 2022

自己紹介

- 名前：鎌田徹平
- 所属：シスコシステムズ合同会社
- 役職：Systems Architect
- Service provider様向けのPre-sales SE職
- Interop Tokyo ShowNet NOC team member

Agenda

- はじめに
  Cisco Routing PlatformのOpen化への取り組み
- CiscoのCustom SiliconでSONiCが動く
  SONiC on SiliconOne
- ついにContainer化
  XRd vRouter
- IOS-XRのあまり知られていない使い方
  Service Layer API

はじめに
Cisco Routing PlatformのOpen化への取り組み

本日の内容

- CiscoのRouting platformはあまり知られていないですが、柔軟な使い方が選択できるようにOpen化へも取り組んでいます
- 本日はその取り組みについて紹介させていただきたいと思います

Simple

- Architecture
- Operations
- IOS-XR Install
- Software delivery
- Secure ZTP

Modern

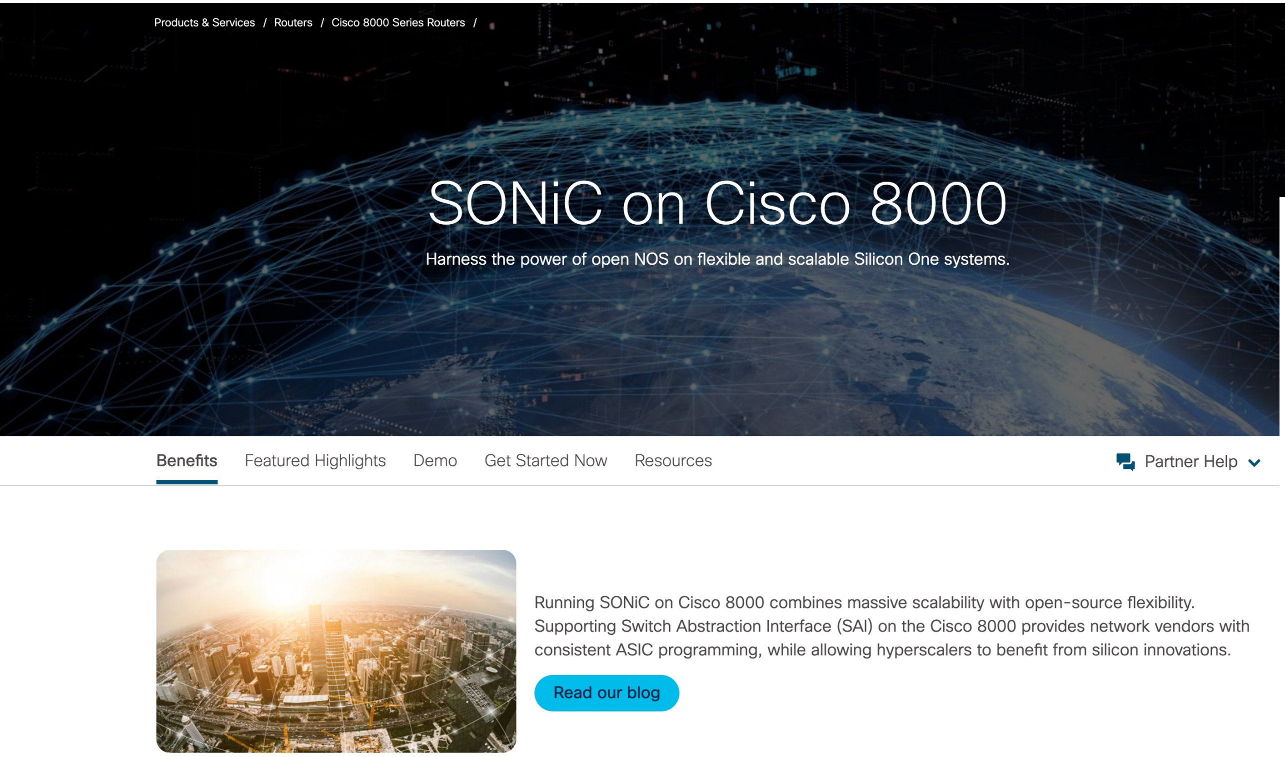
- ZTP API’s
- YANG Models – Device Mgmt
- Segment Routing, EVPN
- Streaming Telemetry
- Service-Layer, OFA

Flexible

- VNF
- Merchant Silicon
- 3rd party Qualified Hardware
- Custom Silicon
- Containerized

CiscoのCustom SiliconでSONiCが動く
SONiC on SiliconOne

Cisco8000ルータがSONiC対応しました

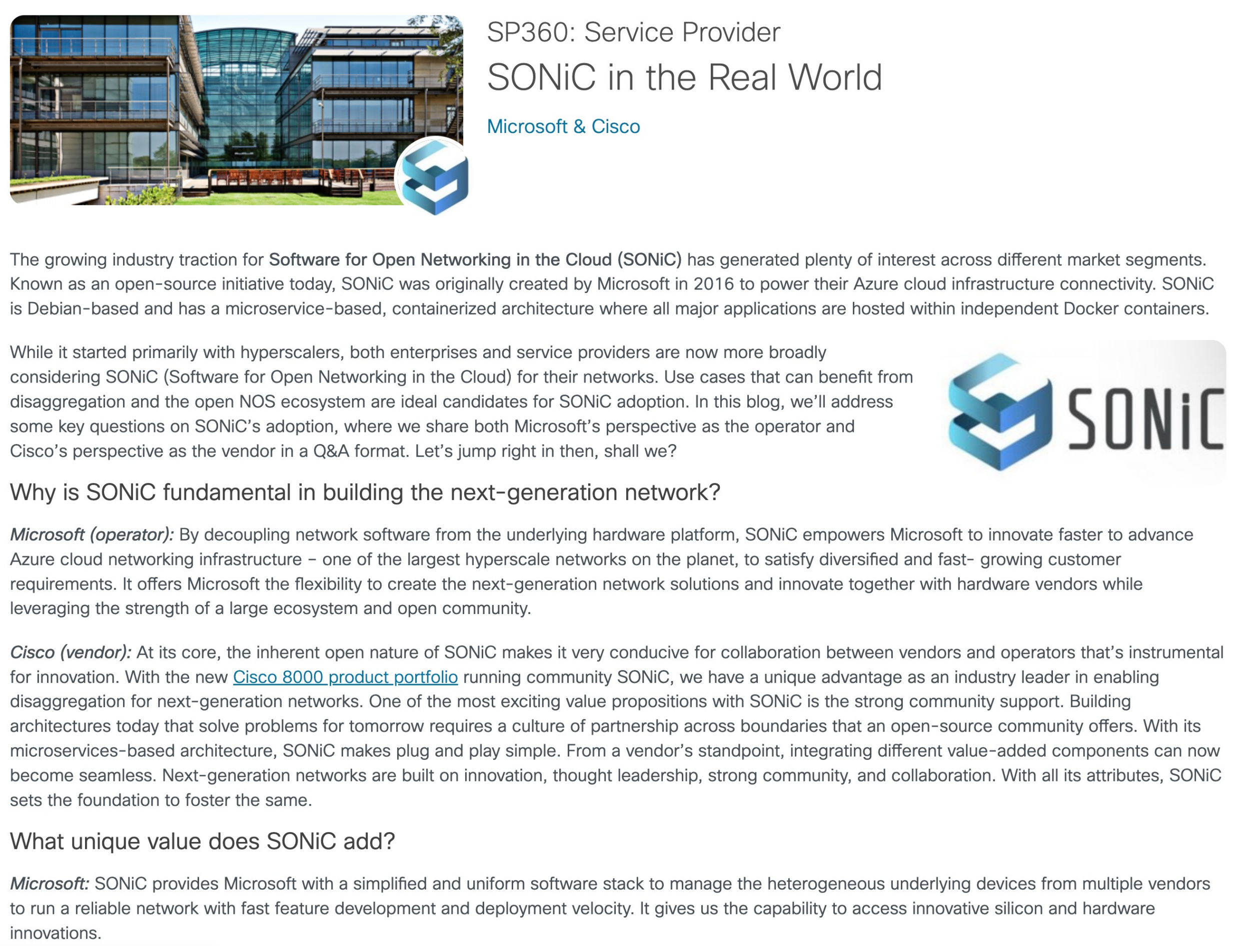


Products & Services / Routers / Cisco 8000 Series Routers /

SONiC on Cisco 8000

Harness the power of open NOS on flexible and scalable Silicon One systems.

Running SONiC on Cisco 8000 combines massive scalability with open-source flexibility. Supporting Switch Abstraction Interface (SAI) on the Cisco 8000 provides network vendors with consistent ASIC programming, while allowing hyperscalers to benefit from silicon innovations.



SP360: Service Provider

SONiC in the Real World

Microsoft & Cisco

The growing industry traction for **Software for Open Networking in the Cloud (SONiC)** has generated plenty of interest across different market segments. Known as an open-source initiative today, SONiC was originally created by Microsoft in 2016 to power their Azure cloud infrastructure connectivity. SONiC is Debian-based and has a microservice-based, containerized architecture where all major applications are hosted within independent Docker containers.

While it started primarily with hyperscalers, both enterprises and service providers are now more broadly considering SONiC (Software for Open Networking in the Cloud) for their networks. Use cases that can benefit from disaggregation and the open NOS ecosystem are ideal candidates for SONiC adoption. In this blog, we'll address some key questions on SONiC's adoption, where we share both Microsoft's perspective as the operator and Cisco's perspective as the vendor in a Q&A format. Let's jump right in then, shall we?

Why is SONiC fundamental in building the next-generation network?

Microsoft (operator): By decoupling network software from the underlying hardware platform, SONiC empowers Microsoft to innovate faster to advance Azure cloud networking infrastructure – one of the largest hyperscale networks on the planet, to satisfy diversified and fast- growing customer requirements. It offers Microsoft the flexibility to create the next-generation network solutions and innovate together with hardware vendors while leveraging the strength of a large ecosystem and open community.

Cisco (vendor): At its core, the inherent open nature of SONiC makes it very conducive for collaboration between vendors and operators that's instrumental for innovation. With the new Cisco 8000 product portfolio running community SONiC, we have a unique advantage as an industry leader in enabling disaggregation for next-generation networks. One of the most exciting value propositions with SONiC is the strong community support. Building architectures today that solve problems for tomorrow requires a culture of partnership across boundaries that an open-source community offers. With its microservices-based architecture, SONiC makes plug and play simple. From a vendor's standpoint, integrating different value-added components can now become seamless. Next-generation networks are built on innovation, thought leadership, strong community, and collaboration. With all its attributes, SONiC sets the foundation to foster the same.

What unique value does SONiC add?

Microsoft: SONiC provides Microsoft with a simplified and uniform software stack to manage the heterogeneous underlying devices from multiple vendors to run a reliable network with fast feature development and deployment velocity. It gives us the capability to access innovative silicon and hardware innovations.

※ Commercial SONiC Distributionではなく、
C8k用のCisco SONiC Distributionです。

https://www.cisco.com/c/en/us/products/routers/8000-series-routers/sonic.html
https://blogs.cisco.com/sp/sonic-in-the-real-world

なんでCiscoがSONiC対応するの?

Open source

- Vendor independence
- Feature velocity & community support

Disaggregation

- Modular components
- Decoupling sw functions

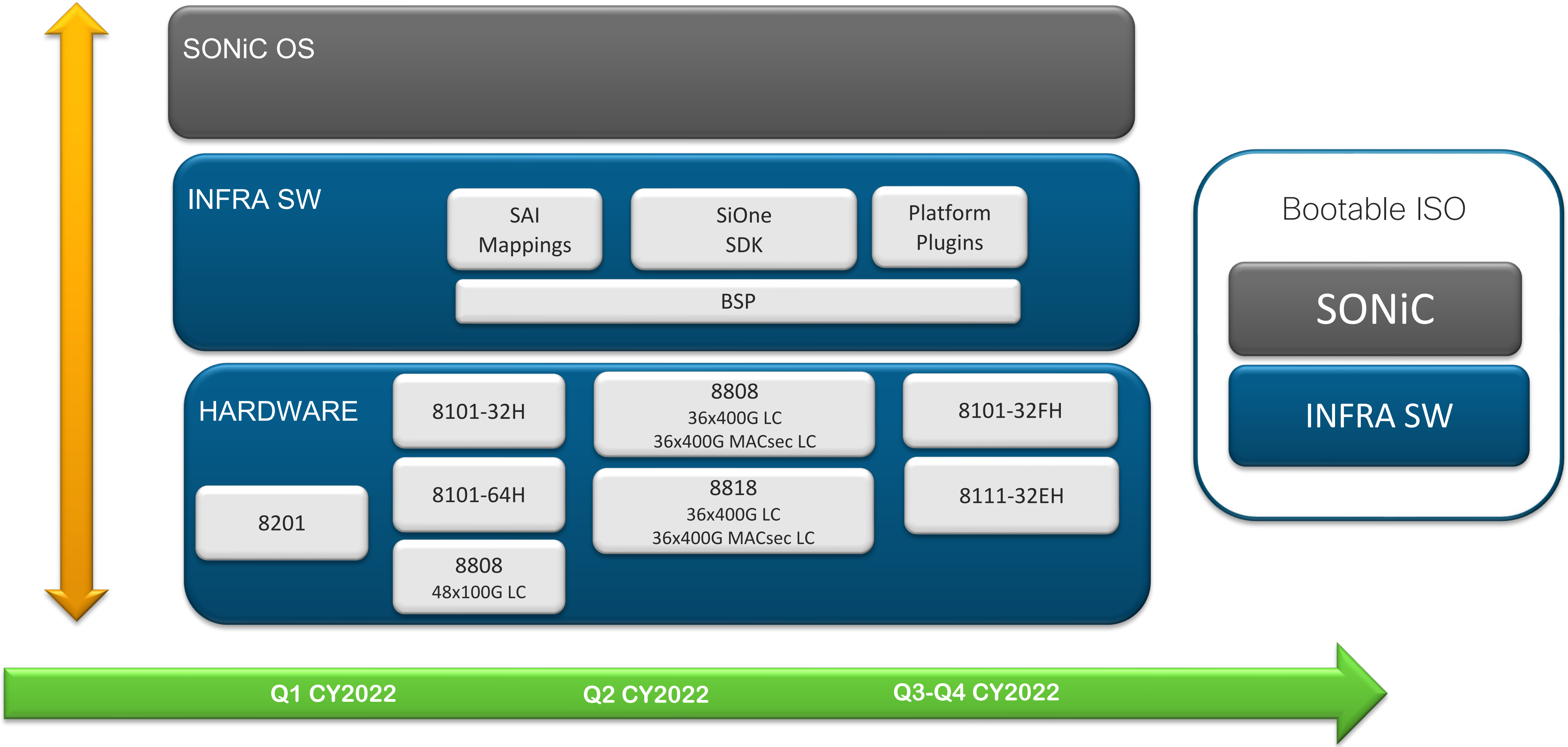
Vendor agnostic

- Normalize different vendor hardware
- SAI API

DevOps

- Containerized – ease of automation & orchestration
- Microservices – ease of upgrades

The “Product” – SONiC on Cisco 8000



SONiC OS

INFRA SW

SAI Mappings

SiOne SDK

Platform Plugins

BSP

HARDWARE

8201

8101-32H

8101-64H

8808
48x100G LC

8808
36x400G LC
36x400G MACsec LC

8818
36x400G LC
36x400G MACsec LC

8101-32FH

8111-32EH

Bootable ISO

SONiC

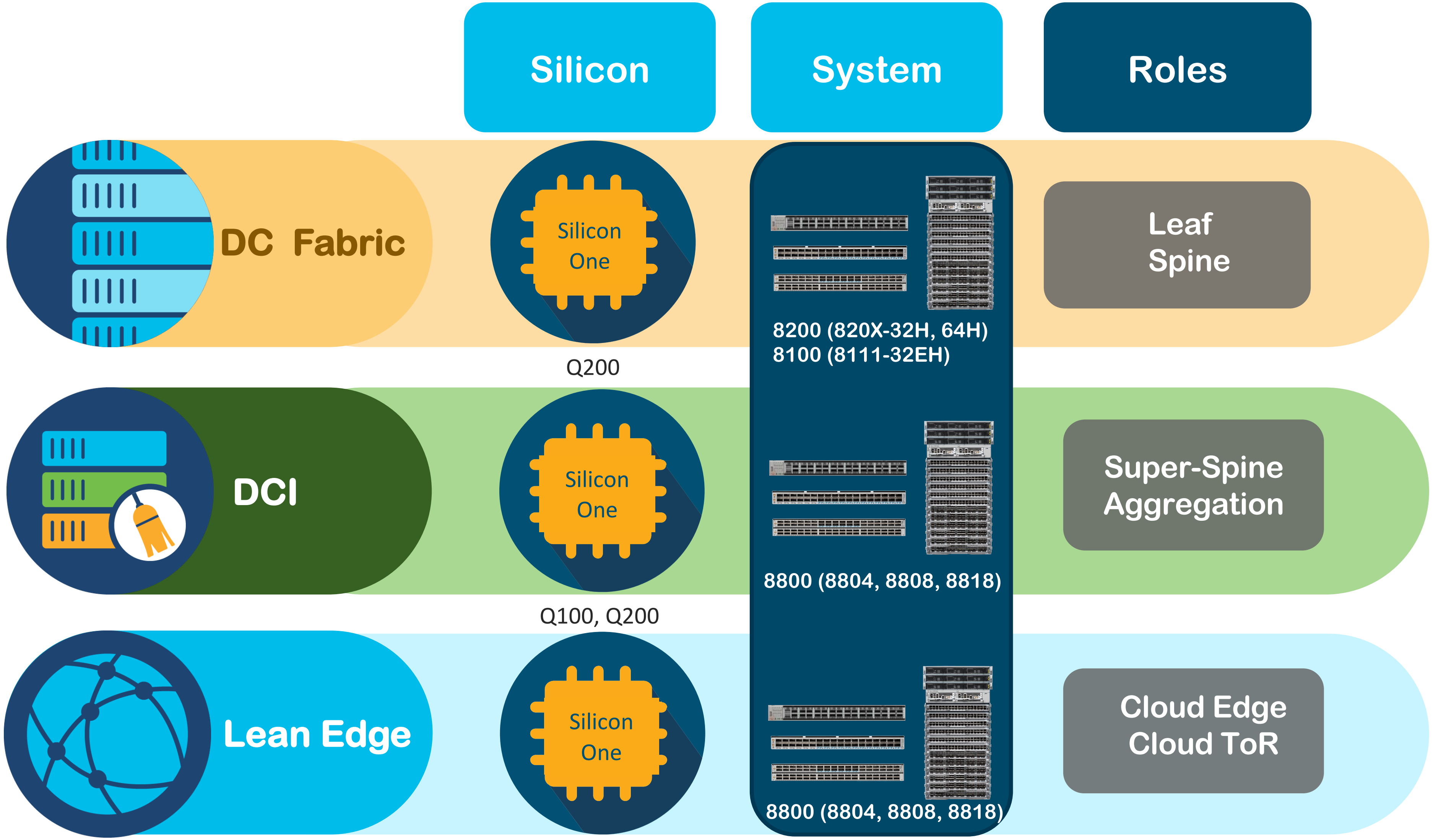
INFRA SW

Q1 CY2022

Q2 CY2022

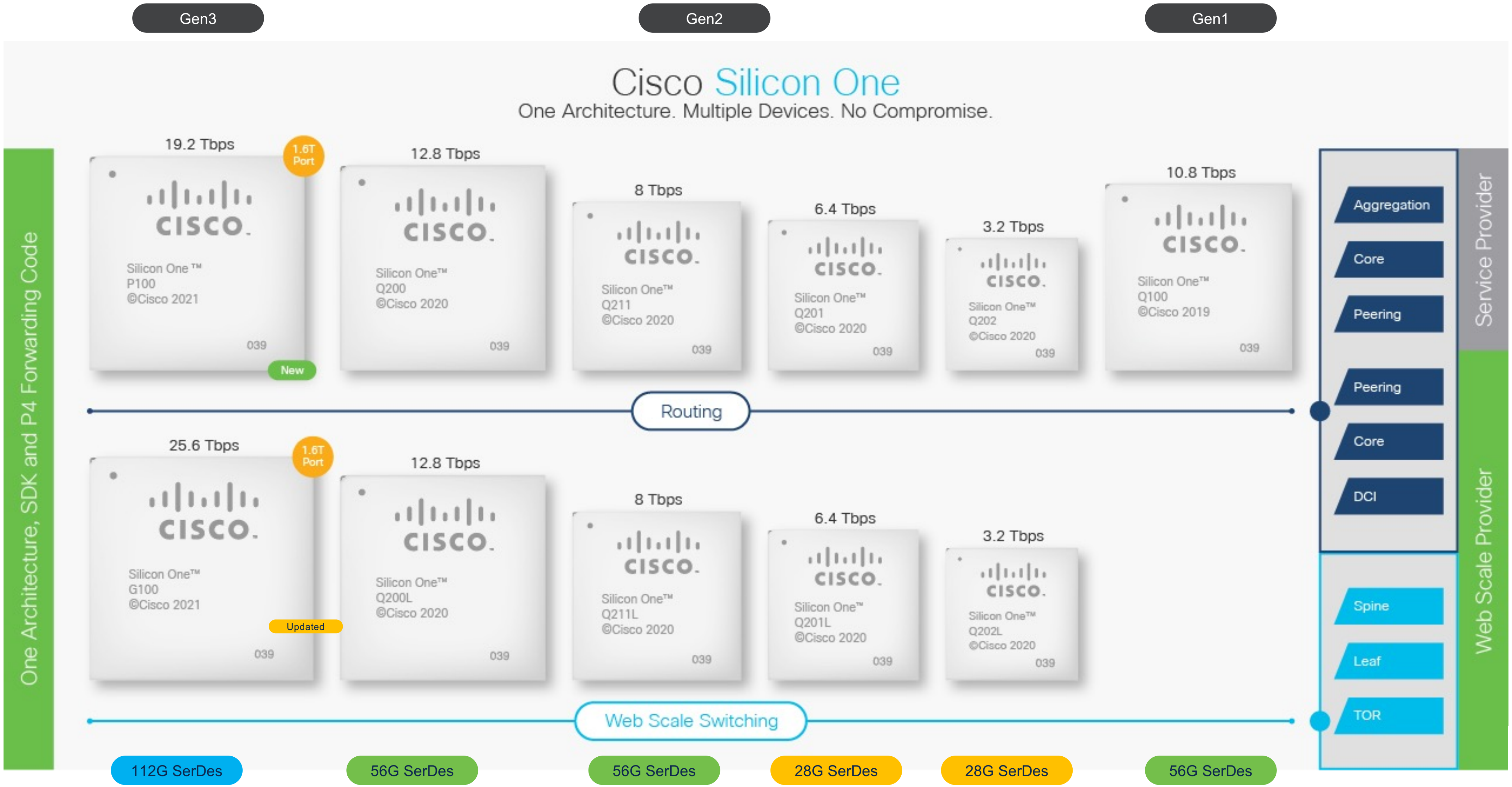
Q3-Q4 CY2022

SONiC & Silicon One



	Silicon	System	Roles
DC Fabric	Silicon One Q200	8200 (820X-32H, 64H) 8100 (8111-32EH)	Leaf Spine
DCI	Silicon One Q100, Q200	8800 (8804, 8808, 8818)	Super-Spine Aggregation
Lean Edge	Silicon One Q100, Q200	8800 (8804, 8808, 8818)	Cloud Edge Cloud ToR
		8100, 8200, 8800	

Cisco Silicon One


Gen3
Gen2
Gen1
Cisco Silicon One
One Architecture. Multiple Devices. No Compromise.
One Architecture, SDK and P4 Forwarding Code
19.2 Tbps
1.6T Port
Silicon One ™ P100 ©Cisco 2021
New
12.8 Tbps
Silicon One™ Q200 ©Cisco 2020
8 Tbps
Silicon One™ Q211 ©Cisco 2020
6.4 Tbps
Silicon One™ Q201 ©Cisco 2020
3.2 Tbps
Silicon One™ Q202 ©Cisco 2020
10.8 Tbps
Silicon One™ Q100 ©Cisco 2019
Routing
25.6 Tbps
1.6T Port
Silicon One™ G100 ©Cisco 2021
Updated
12.8 Tbps
Silicon One™ Q200L ©Cisco 2020
8 Tbps
Silicon One™ Q211L ©Cisco 2020
6.4 Tbps
Silicon One™ Q201L ©Cisco 2020
3.2 Tbps
Silicon One™ Q202L ©Cisco 2020
Web Scale Switching
Service Provider
Aggregation
Core
Peering
Web Scale Provider
Peering
Core
DCI
Spine
Leaf
TOR
112G SerDes
56G SerDes
56G SerDes
28G SerDes
28G SerDes
56G SerDes

Cisco 8100 Series

Power Optimized

On-Chip Buffer

	8101-32H	8102-64H	8101-32FH*
FCS	**Shipping**		
Port Speeds	10/25/40/50/100		10/25/40/50/ 100/200/400
Ports & Line Cards	32X100G	64X100G	32X400G
100G Native/BO	32	64	32/128
400G Native/BO	N/A	N/A	32/0
Throughput	3.2Tbps	6.4Tbps	12.8Tbps
Buffer On/Off Chip	64MB	64MB	108MB
Data Sheet Power 27C / 768B / 15%	150W	220W	288W
27C / 500B / 80%	165W	235W	450W
ASIC	**Q201L (7nm)**	**Q202L (7nm)**	**Q200L (7nm)**

Cisco 8200 Series

Power Optimized

High Buffer

MACsec

	8201	8202	8201-32FH	8202-32FH-M*
FCS	**Shipping**	**Shipping**	**Shipping**	**Q2 2022**
Port Speeds	10/40/100/200/400			
Ports & Line Cards	24X400G + 12X100G	60X100G + 12X400G	32X400G	32X400G
100G Native/BO	36/108	72/108	32/128	32/128
400G Native/BO	24	12	32	32
Throughput	10.8Tbps	10.8Tbps	12.8Tbps	12.8Tbps
Buffer On/Off Chip	36MB/8GB	36MB/8GB	108MB/8GB	108MB/8GB
Data Sheet Power 27C / 768B / 15%	415W	750W	288W	~750W
27C / 500B / 80%	510W	845W	450W	<TBD>
27 / 500B / 80% Low power mode			370W	<TBD>
ASIC	**Q100 (16nm)**	**Q100 (16nm)**	**Q200 (7nm)**	

Cisco 8800 Series

High Buffer
High Bandwidth

100G Optimized
FABRIC MODE

Power Optimized

N+1 Redundant
Fabric

	8804(10RU)	8808(16RU)	8812(21RU)	8818(33RU)
FCS	Q2 2021	Shipping	Shipping	Shipping
100G Count Native	192	384	576	864
400G Count Native	144	288	432	648
Throughput	57Tbps	115Tbps	172Tbps	260Tbps

Q100 Line Cards

48X100G MACsec Q100

36X400G Q100

Q200 (7nm) Line Cards

36X400G MACsec

36X400G

Q200 LCs – Q2 2021

ちょっとだけデモ
こんな感じで動きます

参考リンク:
https://devnetsandbox.cisco.com/RM/Diagram/Index/219b721f-4116-4e47-adfa-c41ab540ca22?diagramType=Topology
https://www.cisco.com/c/en/us/td/docs/iosxr/cisco8000-emulator/cisco8000-hardware-emulator-datasheet.html

ついにContainer化
XRd vRouter

XRdとは？

DockerコンテナとしてIOS XRルーター

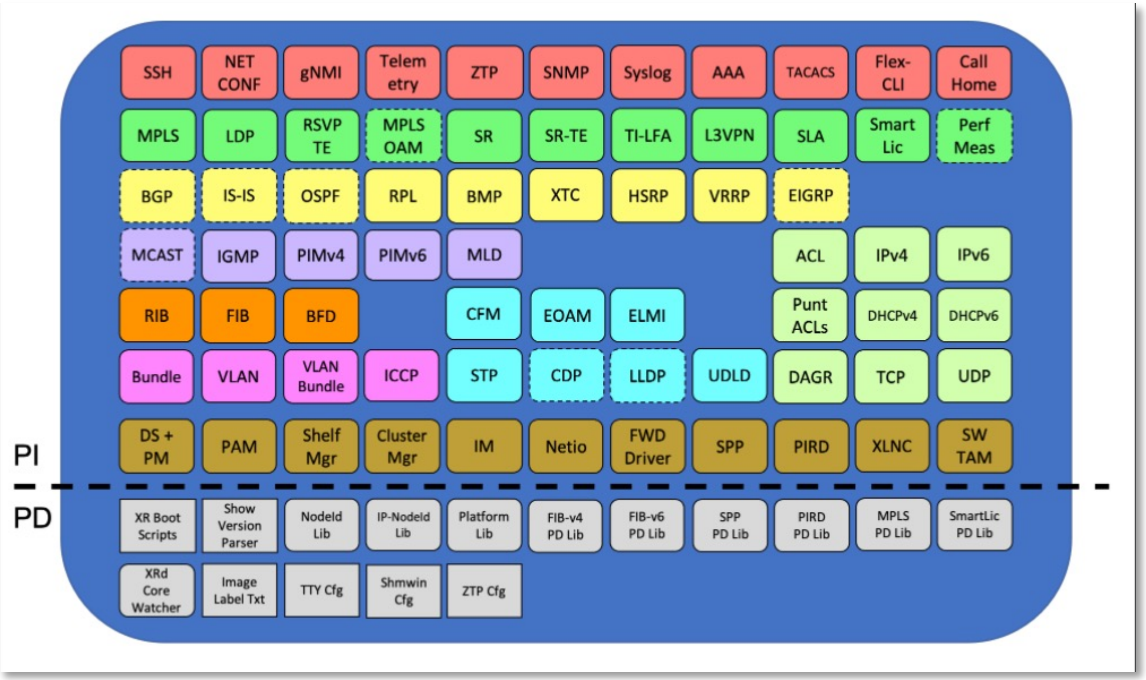
docker-composeを使いXRd Topologyを構築

XRv9K(VM)より圧倒的に軽量で立ち上がりも高速

Control planeだけのDockerイメージとData plane付きのvRouter

ユースケース

- BGP vRR及びSR-PCEなどのCP function
- 軽量でパフォーマンスの出るCloud Router
- ラボ利用での動作確認やトレーニング



	SSH	NET CONF	gNMI	Telem etry	ZTP	SNMP	Syslog	AAA	TACACS	Flex-CLI	Call Home
	MPLS	LDP	RSVP TE	MPLS OAM	SR	SR-TE	TI-LFA	L3VPN	SLA	Smart Lic	Perf Meas
	BGP	IS-IS	OSPF	RPL	BMP	XTC	HSRP	VRRP	EIGRP		
	MCAST	IGMP	PIMv4	PIMv6	MLD				ACL	IPv4	IPv6
	RIB	FIB	BFD		CFM	EOAM	ELMI		Punt ACLs	DHCPv4	DHCPv6
	Bundle	VLAN	VLAN Bundle	ICCP	STP	CDP	LLDP	UDLD	DAGR	TCP	UDP
PI	DS + PM	PAM	Shelf Mgr	Cluster Mgr	IM	Netio	FWD Driver	SPP	PIRD	XLNC	SW TAM
PD	XR Boot Scripts	Show Version Parser	NodeId Lib	IP-NodeId Lib	Platform Lib	FIB-v4 PD Lib	FIB-v6 PD Lib	SPP PD Lib	PIRD PD Lib	MPLS PD Lib	SmartLic PD Lib
	XRd Core Watcher	Image Label Txt	TTY Cfg	Shmwin Cfg	ZTP Cfg						

Cisco Cloud Native vRouter a.k.a XRd

Software based router to run on x86

- Cisco IOS-XR and Management
- DPDK and VPP based forwarding
- Kubernetes compliant
- Light footprint on x86 compute

Solution for Cloud native deployments

- Suitable for Cloud native environments
- Routing function at low-bandwidth cell site
- Physical CSR Feature parity

CPU Cores	2 physical cores: 1 for control plane ; 1 for dataplane (*)
Memory	11 GiB: 8 GiB regular memory + 3 GiB huge pages (**)
Disk	7 Gb (***)
Boot time	~2 mins (to BGP convergence)
Latency	50us via vRouter CNF

*   CPU may require hyperthreading

**   11 GiB provides equivalent memory to Physical Router

- 8 GiB is minimum to boot
- Real configuration expected to be < 10 GiB

Control planeのみの場合は2GBで動作

***   Includes provision for logs and other operational data; in most cases usage <= 2Gb

Cisco XRd Deployment Models

FLEXIBLE

Containerized/ Virtual Deployment

*Standard K8s, Vmware Tanzu K8s, standalone Docker, gVisor**

*1:1 Replacement of XRv9k on KVM/ESXI**

CONVENIENT

Feature parity

Telemetry & Yang Models Integration with Cisco Network Controller & Network Service Orchestrator

CLOUD

AWS (EKS), GCE GKE*, Azure AKS*

Easy to try, Easy to Scale, Easy to Manage

COMMON NETWORK OPERATING SYSTEM: IOS-XR

* = Future Release

Cisco XRd Use Cases

VIRTUAL ROUTE REFLECTOR

Industry Leading Scale

Upto 70 Mn Paths
20M Routes
100 RR Groups

VIRTUAL CELL SITE ROUTER

Lowest XR footprint

T’put: ~20 Gbps[#]
Routes: 20K VPNv4/V6
SRv6/SR-MPLS/SR-TE
HQoS

Light weight lab simulation

CI/CD deployment

Thorough XR Coverage
Fast boot time: ~2 Min

VIRTUAL PROVIDER EDGE*

Consistent architecture w/ ASR 9000

T’put: ~100Gbps
Business VPNs, M’cast, Peering

CLOUD ROUTER*

Public Cloud Gateway

Routes: Upto 100K[#]
GRE Tunnels: ~400[#]
BGP Sessions: 1000[#]
SRv6[$], SR-MPLS, GRE Overlay

Image, Scriptの取得

XRd Imageの取得

- https://software.cisco.com/download/home/286331236/type/280805694/release/7.7.1

XRdを操作するScriptの取得

- https://github.com/ios-xr/xrd-tools

xrd-tools 内容

/scripts：各種Scriptが格納

- host-check

  ホストが要件を満たしているか確認

- launch-xrd

  単一XRdコンテナを起動

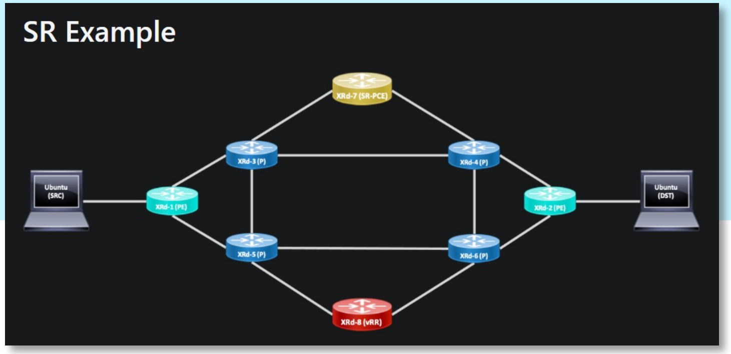
- xr-compose

  複数XRdによるトポロジを起動

/samples/xr_compose_topos: サンプルトポロジーが格納

- segment-routing

  複数のXRd間でSR-MPLSを構成



ちょっとだけデモ
こんな感じで動きます

参考リンク:
https://www.cisco.com/c/en/us/td/docs/routers/virtual-routers/xrd-77x/release/notes/b-release-notes-xrd-r771.html
https://github.com/ios-xr/xrd-tools

当面の面白い使い方として

- Scrap and Buildが非常に簡単

  30秒ほどでXRdが起動する

- XRd 1ノードにつき2GB RAMと軽くてエコ
- XRd Topology, startup-configも自由自在

ラボ・トレーニング用途に非常に向いている

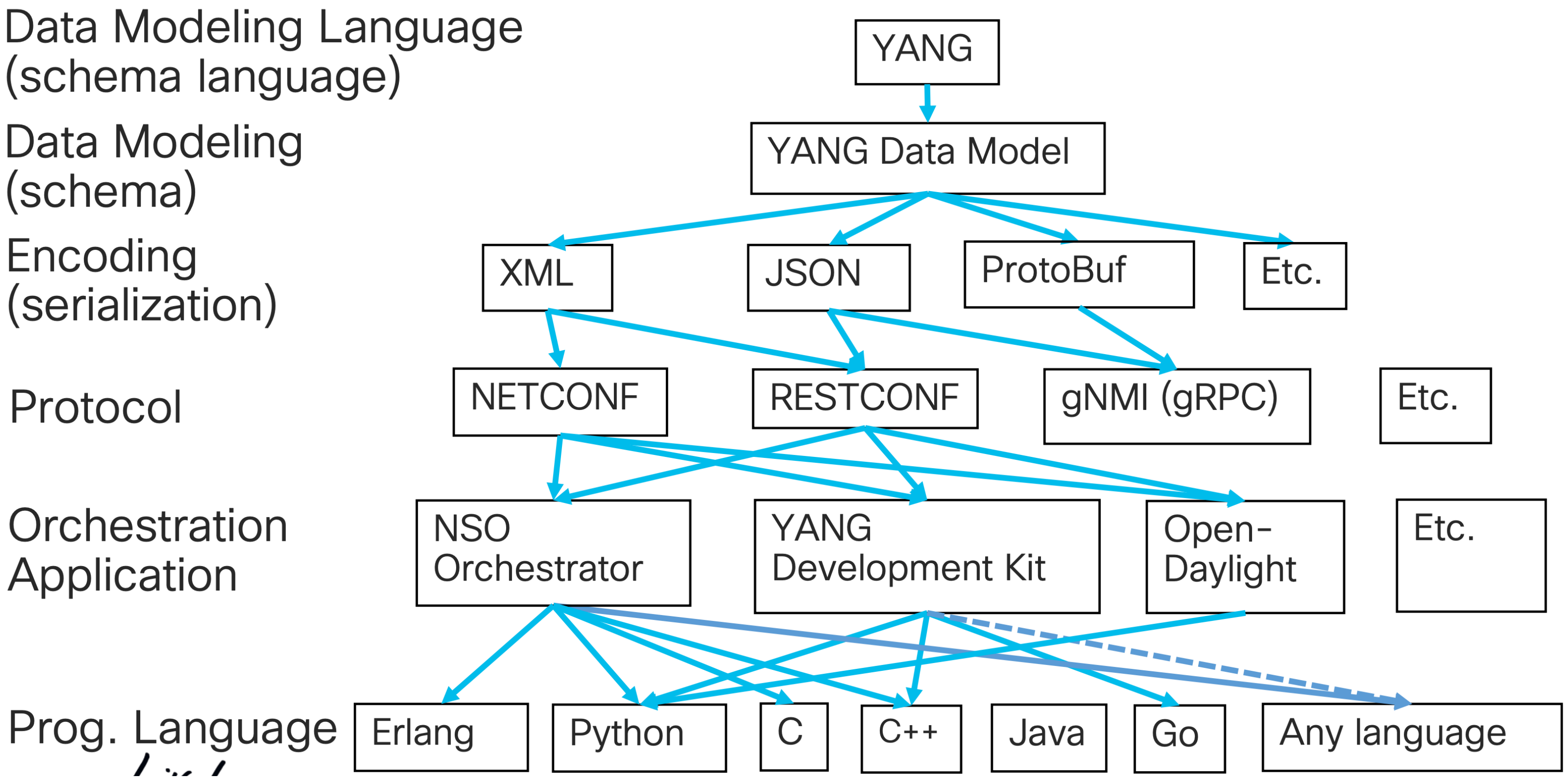
IOS-XRのあまり知られていない使い方

Service Layer API

IOS-XRのAPIの普及

- IOS-XRも多様なApplicationに向けて様々なAPIを用意しています

Data Model-driven Management



Data Modeling Language (schema language): YANG

Data Modeling (schema): YANG Data Model

Encoding (serialization): XML, JSON, ProtoBuf, Etc.

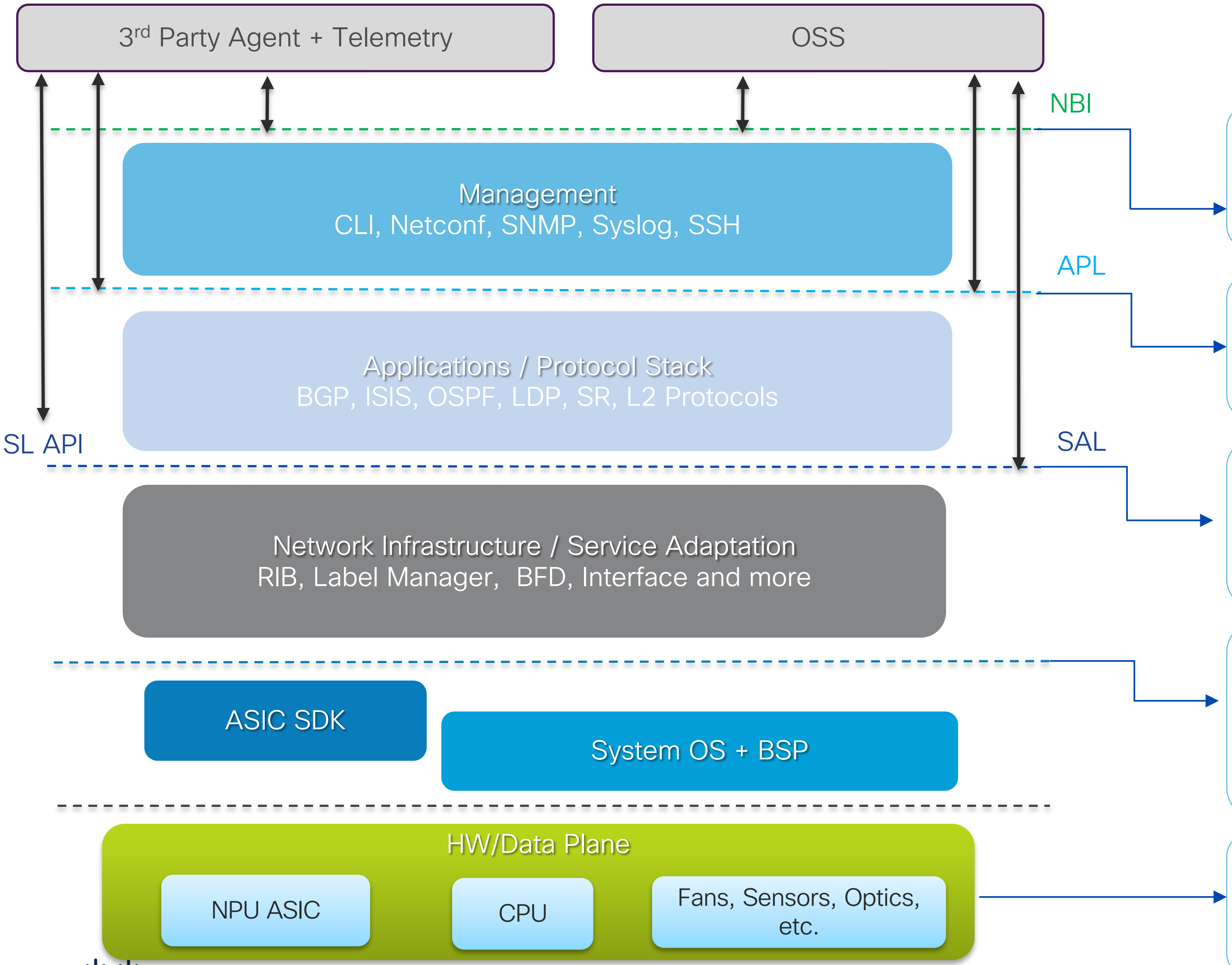
Protocol: NETCONF, RESTCONF, gNMI (gRPC), Etc.

Orchestration Application: NSO Orchestrator, YANG Development Kit, Open-Daylight, Etc.

Prog. Language: Erlang, Python, C, C++, Java, Go, Any language

https://www.ciscolive.com/c/dam/r/ciscolive/emea/docs/2020/pdf/BRKNMS-2129.pdf

De-Layering The Network Stack



3rd Party Agent + Telemetry

OSS

NBI

Management
CLI, Netconf, SNMP, Syslog, SSH

APL

Applications / Protocol Stack
BGP, ISIS, OSPF, LDP, SR, L2 Protocols

SL API

SAL

Network Infrastructure / Service Adaptation
RIB, Label Manager, BFD, Interface and more

ASIC SDK

System OS + BSP

HW/Data Plane

NPU ASIC

CPU

Fans, Sensors, Optics, etc.

Management/Presentation Layer

Provides access to configure and manage the stack through Network config/oper DB: Yang Models, CLI.

Application/Protocol Layer

Provides APIs into the Routing Protocols (BGP, IGP, SR, etc.)

Network Infrastructure Layer / Service Adaptation Layer

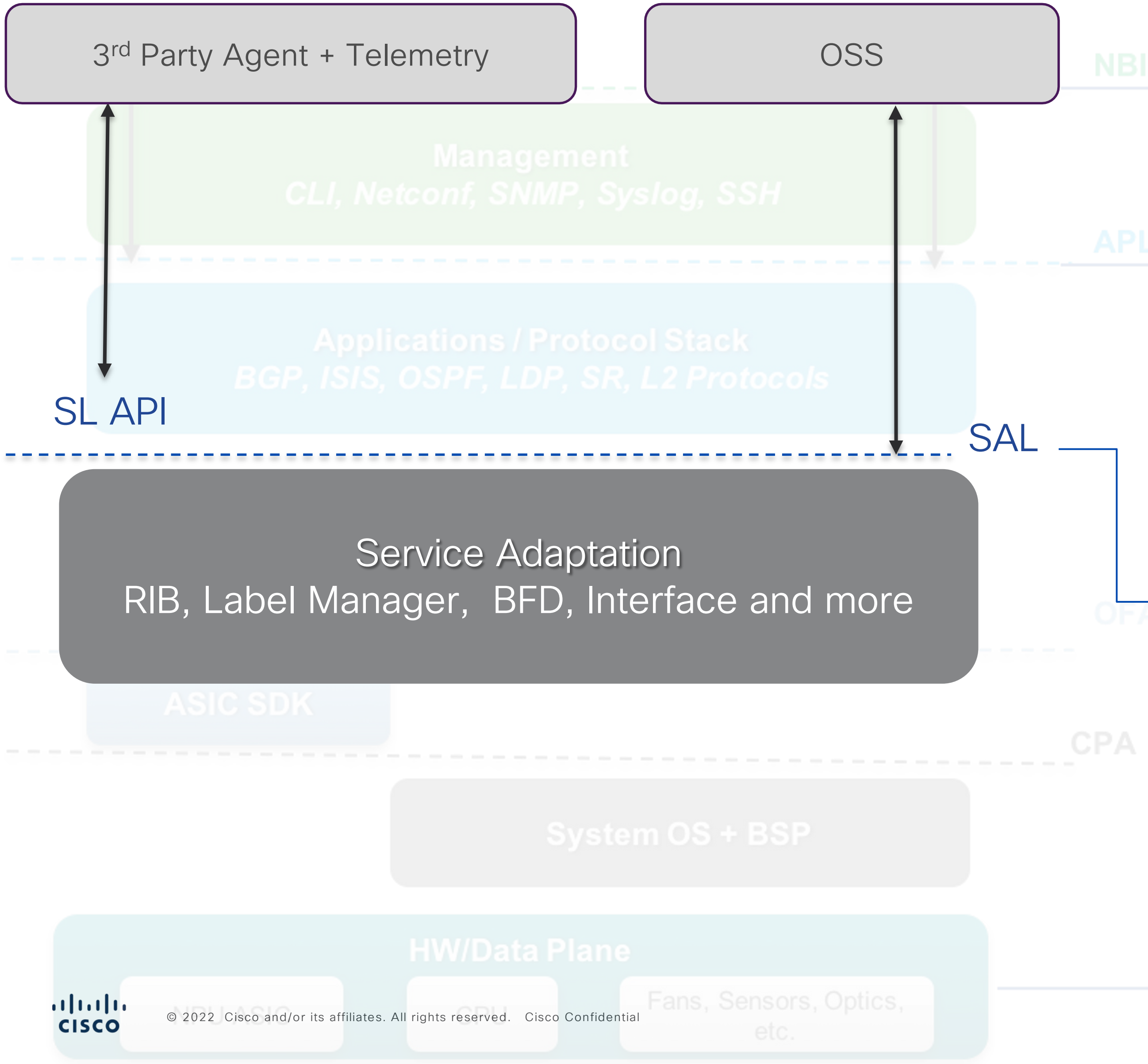
- Acts as the bridge between the Application Layer and the HW
- Presents abstractions to the Application/Protocol Layer

System OS - Linux Kernel
BSP(Board Support Package) - Boot Loader, Device Drivers, etc.

ASIC SDK and drivers for the SDK

Hardware

Consists of ASIC/Chipset from HW vendors + CPU, Fans, Sensors

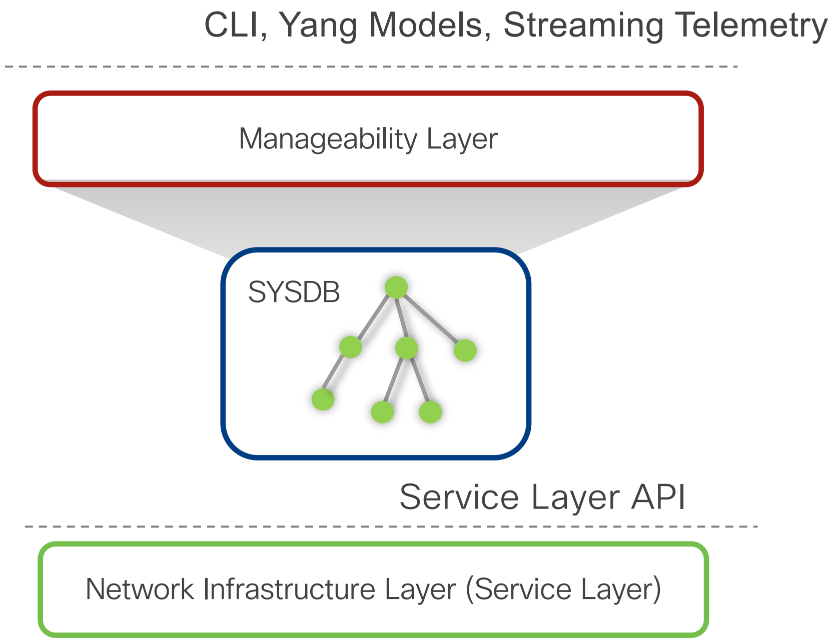

3rd Party Agent + Telemetry
OSS
NBI
Management
CLI, Netconf, SNMP, Syslog, SSH
APL
Applications / Protocol Stack
BGP, ISIS, OSPF, LDP, SR, L2 Protocols
SL API
SAL
Service Adaptation
RIB, Label Manager, BFD, Interface and more
OFA
ASIC SDK
CPA
System OS + BSP
HW/Data Plane
NPU ASIC
CPU
Fans, Sensors, Optics, etc.
ation Layer
Pro
age the stack through IOS-
dels, CLI.
ol Layer
Provides APIs into the Routing Protocols (BGP, IGP, SR, etc.)
Hardware
(Consists of ASIC/Chipset from Broadcom/Cisco custom Si + CPU, Fans, Sensors)

Network Infrastructure Layer / Service Adaptation Layer

- Acts as the bridge between the Application Layer and the HW
- Presents abstractions to the Application/Protocol Layer
- Highly Performant API outside the Central Config/Oper Database context

なんで他のAPIが必要なのか?

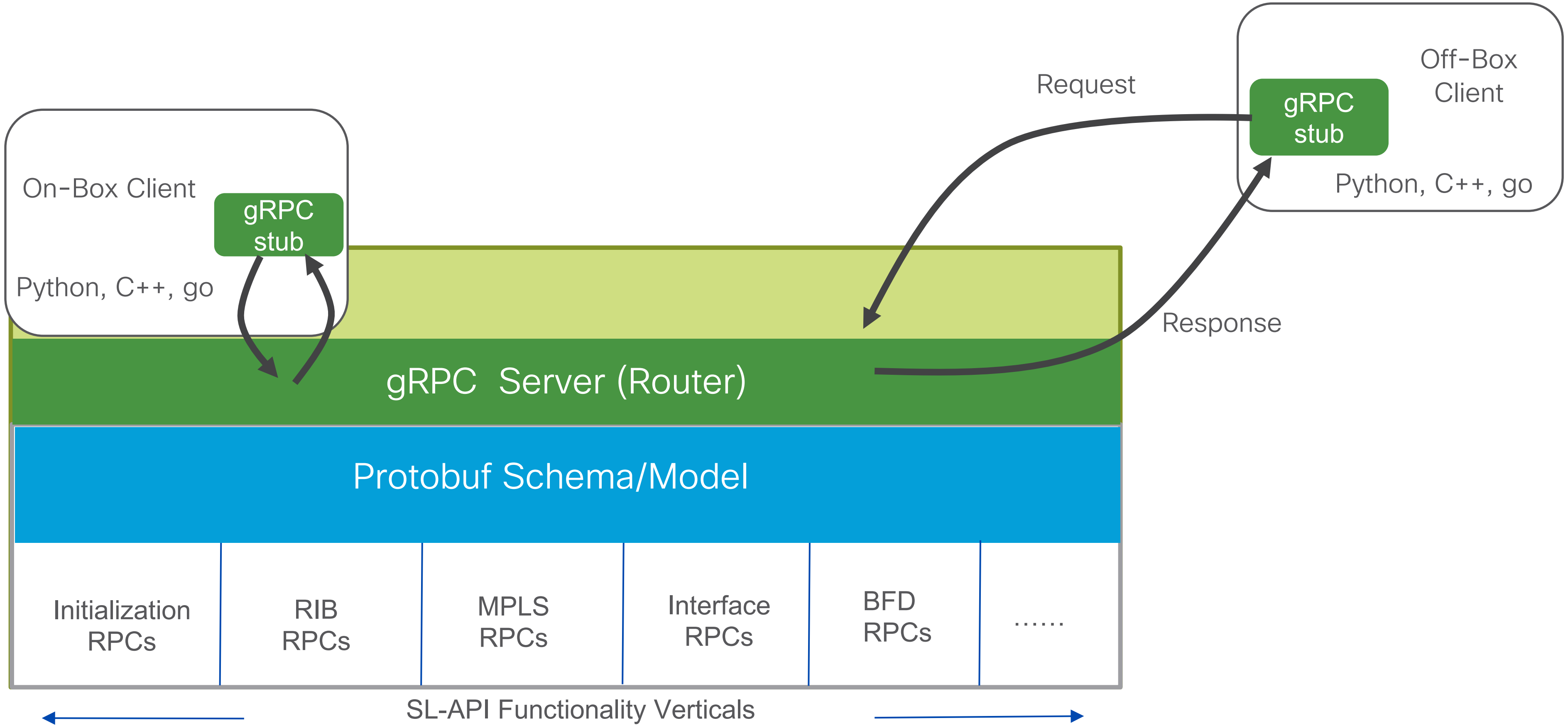
- 現在のAPIはほぼYANG Modelsにフォーカスしている
- YANGはもちろん重要なのですが、 基本的には “Feature” と紐付いています
  - SysDBがサポートしている (つまりFeature)をAPIで叩ける
- “The multiple layers in the stack get in the way – We need better performance!”
- “I have my own controller/protocol, just give me complete access to the infrastructure underneath”



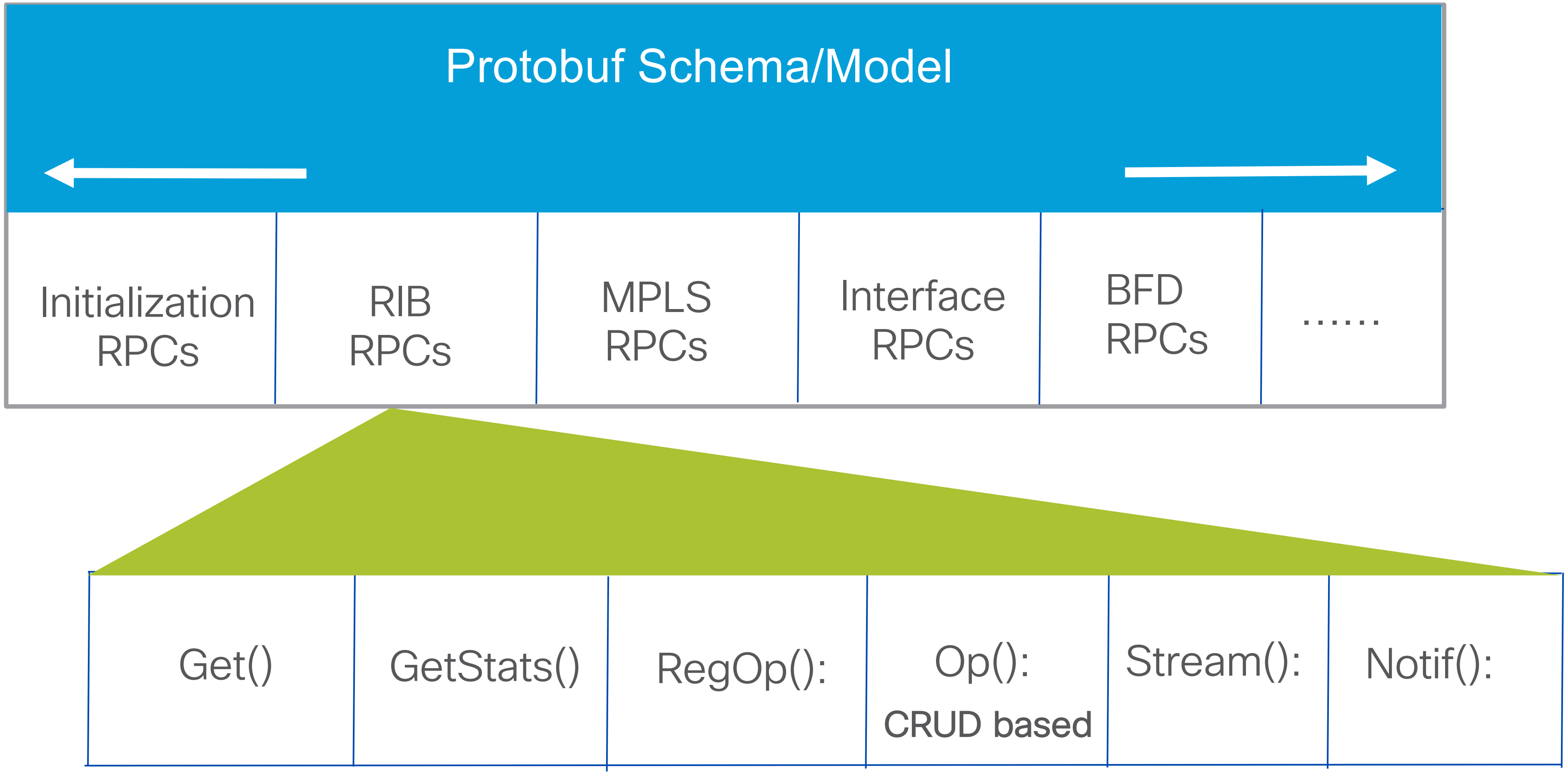
CLI, Yang Models, Streaming Telemetry
Manageability Layer
SYSDB
Service Layer API
Network Infrastructure Layer (Service Layer)

Service Layerに対するAPIを用意することで様々な要望に対応
特に直接RIBをAPIから制御する要望はある?

Service Layer API Architecture



Service Layer API Architecture



Protobuf Schema/Model

SL-API Functionality Domains

Initialization RPCs

RIB RPCs

MPLS RPCs

Interface RPCs

BFD RPCs

……

RPCs per Domain

Get()

GetStats()

RegOp():

Op(): CRUD based

Stream():

Notif():

ちょっとだけデモ…をやる時間はなさそうなので

- **Github:** Check out the Obj-model repository on Github at

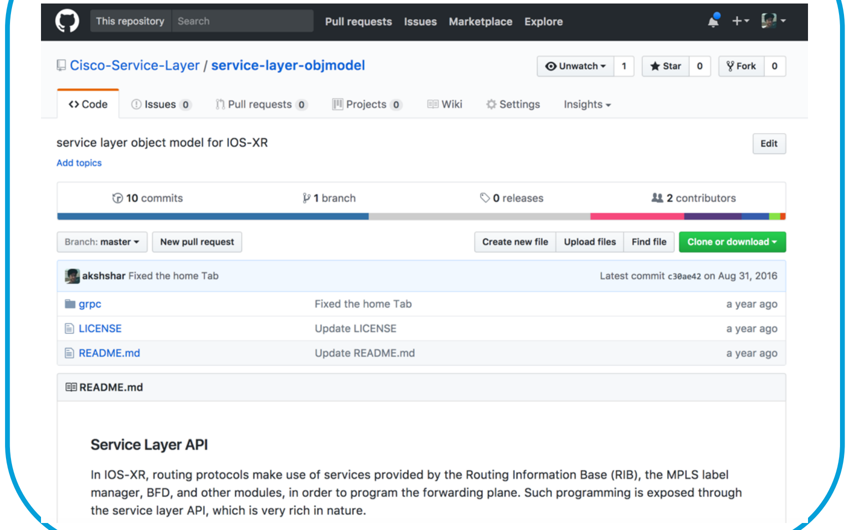
  https://github.com/Cisco-Service-Layer/service-layer-objmodel

  ➢ Proto definitions
  ➢ Exhaustive Unit Tests and tutorial apps

- **@xrdocs:** Blogs, Tutorials on Using Service Layer APIs and associated Libraries:
  https://xrdocs.github.io/cisco-service-layer/

- **APIdocs:** Doxygen based documentation, auto-generated from the proto files:
  https://xrdocs.github.io/cisco-service-layer/apidocs/

Github



Cisco-Service-Layer / service-layer-objmodel

service layer object model for IOS-XR

Add topics

10 commits · 1 branch · 0 releases · 2 contributors

akshshar Fixed the home Tab — Latest commit c30ae42 on Aug 31, 2016

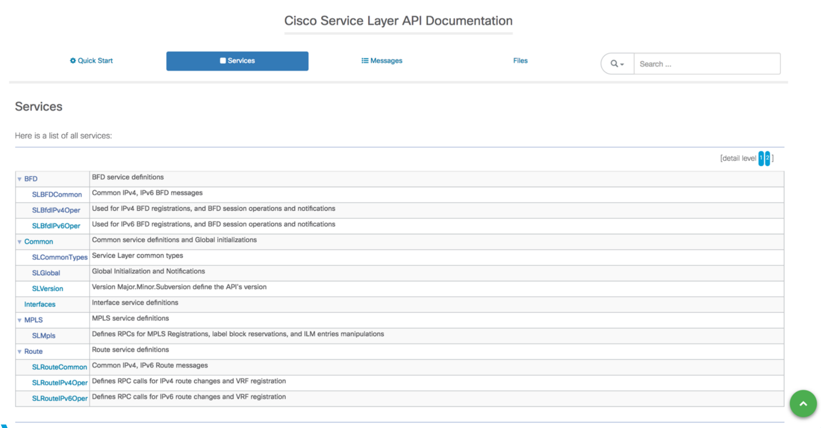
grpc	Fixed the home Tab	a year ago
LICENSE	Update LICENSE	a year ago
README.md	Update README.md	a year ago

README.md

Service Layer API

In IOS-XR, routing protocols make use of services provided by the Routing Information Base (RIB), the MPLS label manager, BFD, and other modules, in order to program the forwarding plane. Such programming is exposed through the service layer API, which is very rich in nature.

APIdocs @xrdocs



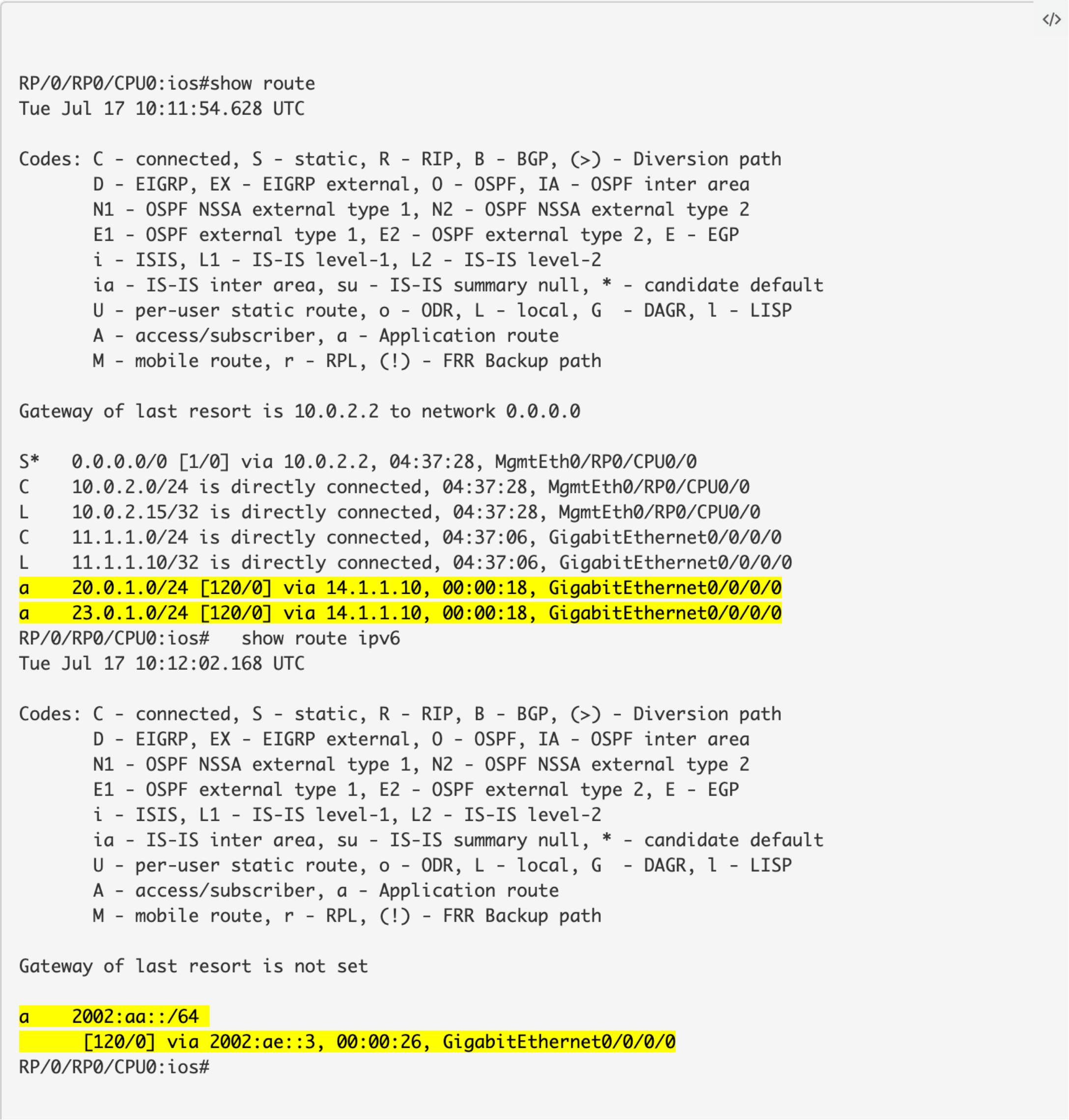
Cisco Service Layer API Documentation

Services

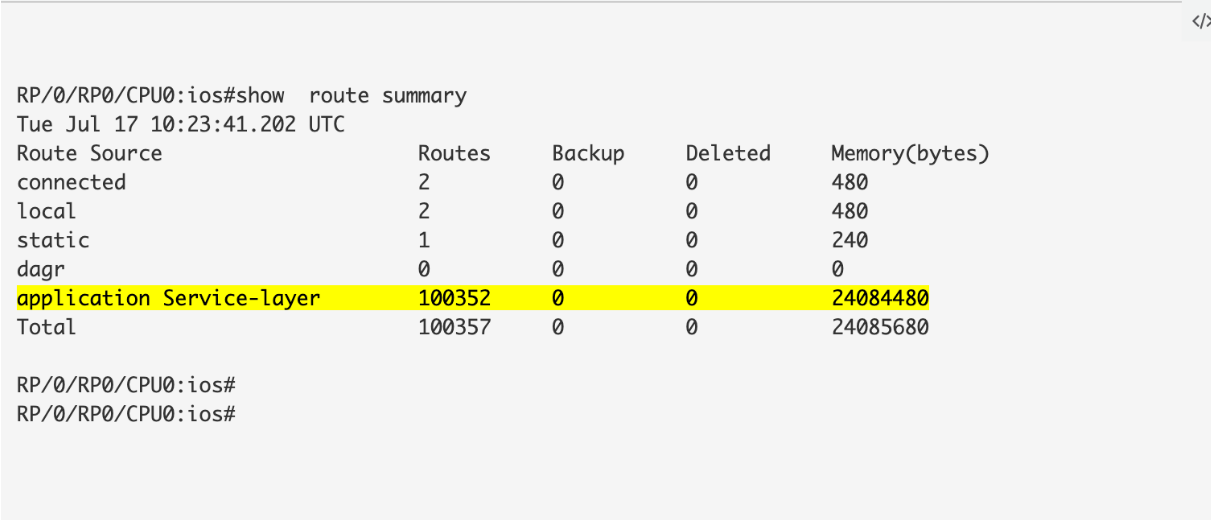
Here is a list of all services:

▼ BFD	BFD service definitions
SLBFDCommon	Common IPv4, IPv6 BFD messages
SLBfdIPv4Oper	Used for IPv4 BFD registrations, and BFD session operations and notifications
SLBfdIPv6Oper	Used for IPv6 BFD registrations, and BFD session operations and notifications
▼ Common	Common service definitions and Global initializations
SLCommonTypes	Service Layer common types
SLGlobal	Global Initialization and Notifications
SLVersion	Version Major.Minor.Subversion define the API's version
Interfaces	Interface service definitions
▼ MPLS	MPLS service definitions
SLMpls	Defines RPCs for MPLS Registrations, label block reservations, and ILM entries manipulations
▼ Route	Route service definitions
SLRouteCommon	Common IPv4, IPv6 Route messages
SLRouteIPv4Oper	Defines RPC calls for IPv4 route changes and VRF registration
SLRouteIPv6Oper	Defines RPC calls for IPv6 route changes and VRF registration

Application Routeの例



```
RP/0/RP0/CPU0:ios#show route
Tue Jul 17 10:11:54.628 UTC

Codes: C - connected, S - static, R - RIP, B - BGP, (>) - Diversion path
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
       U - per-user static route, o - ODR, L - local, G  - DAGR, l - LISP
       A - access/subscriber, a - Application route
       M - mobile route, r - RPL, (!) - FRR Backup path

Gateway of last resort is 10.0.2.2 to network 0.0.0.0

S*   0.0.0.0/0 [1/0] via 10.0.2.2, 04:37:28, MgmtEth0/RP0/CPU0/0
C    10.0.2.0/24 is directly connected, 04:37:28, MgmtEth0/RP0/CPU0/0
L    10.0.2.15/32 is directly connected, 04:37:28, MgmtEth0/RP0/CPU0/0
C    11.1.1.0/24 is directly connected, 04:37:06, GigabitEthernet0/0/0/0
L    11.1.1.10/32 is directly connected, 04:37:06, GigabitEthernet0/0/0/0
a    20.0.1.0/24 [120/0] via 14.1.1.10, 00:00:18, GigabitEthernet0/0/0/0
a    23.0.1.0/24 [120/0] via 14.1.1.10, 00:00:18, GigabitEthernet0/0/0/0
RP/0/RP0/CPU0:ios#   show route ipv6
Tue Jul 17 10:12:02.168 UTC

Codes: C - connected, S - static, R - RIP, B - BGP, (>) - Diversion path
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
       U - per-user static route, o - ODR, L - local, G  - DAGR, l - LISP
       A - access/subscriber, a - Application route
       M - mobile route, r - RPL, (!) - FRR Backup path

Gateway of last resort is not set

a    2002:aa::/64
      [120/0] via 2002:ae::3, 00:00:26, GigabitEthernet0/0/0/0
RP/0/RP0/CPU0:ios#
```



```
RP/0/RP0/CPU0:ios#show  route summary
Tue Jul 17 10:23:41.202 UTC
Route Source                 Routes    Backup    Deleted    Memory(bytes)
connected                    2         0         0          480
local                        2         0         0          480
static                       1         0         0          240
dagr                         0         0         0          0
application Service-layer    100352    0         0          24084480
Total                        100357    0         0          24085680

RP/0/RP0/CPU0:ios#
RP/0/RP0/CPU0:ios#
```

https://xrdocs.io/cisco-service-layer/tutorials/2017-09-25-using-service-layer-apis-with-vagrant-iosxr/

Summary

- 本セッションではCisco Routing platformのOpen化に対する取り組みを紹介しました
- 具体的な例として以下を説明しました
  - Cisco8000ルータでSONiCが動作
  - Container IOS-XR XRdで数十秒で大きなトポロジを起動
  - IOS-XRのAPI：Service Layer APIで経路注入
- 今後も続いていく営みとなると思いますので、面白い使い方など議論させていただければうれしいです

CISCO

The bridge to possible