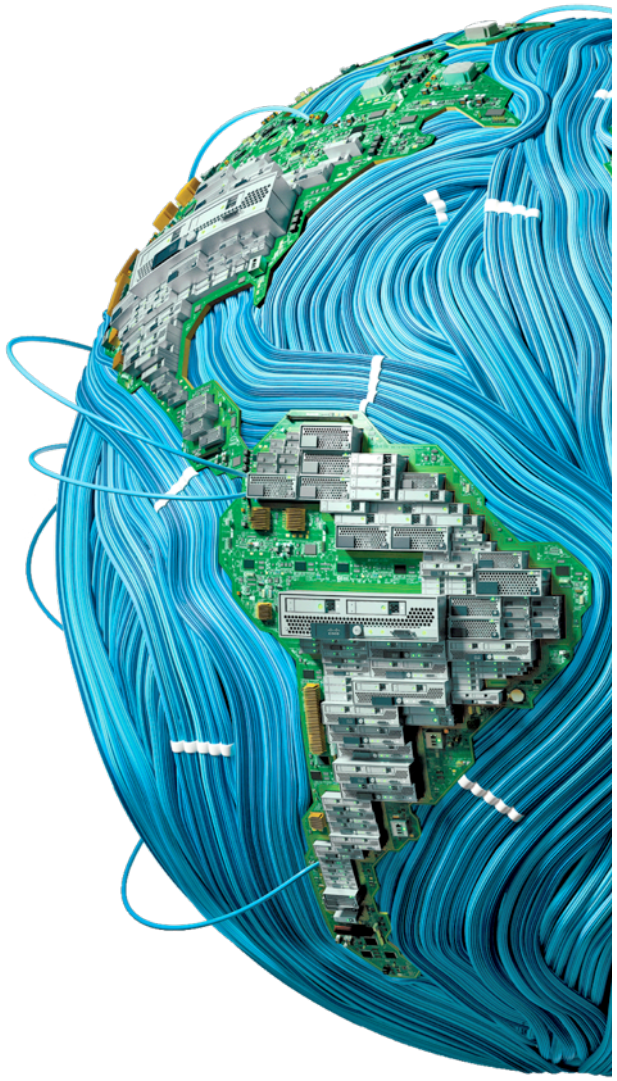
Model Driven APIs for the Network Infrastructure Layer

Akshat Sharma,

Technical Marketing Engineer, Cisco.



What brings us here?

Plenty of Standardization Efforts

- The **Network Processing Forum (NPF)** took the first stab at it back in 2003/2004 with its Service API definitions:
  - IPv4 Unicast Forwarding Service API Revision 2 (June 2004)
  - IPv6 Unicast Forwarding Service API Revision 2 (June 2004)
  - IPv6 Unicast Forwarding Service API (September 2003)
  - MPLS Forwarding Service API (September 2003)
  - IPv4 Unicast Forwarding Service API (April 2003)

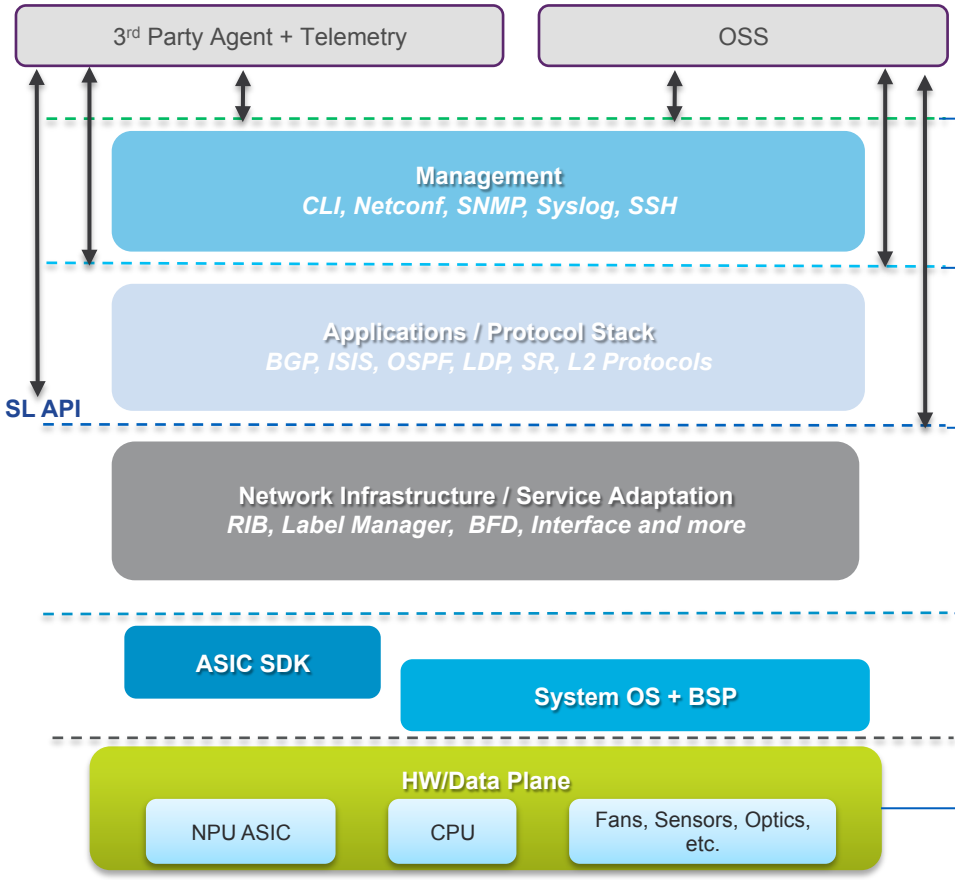
- **I2RS (interface to the Routing System) at IETF** is the latest ongoing iteration of the standardization exercise of this layer with primary focus on RIB and Label Switch Database APIs

- The Network operators are increasingly looking outside the capability of traditional Network protocols to influence the network through their own custom logic

- There is a need to have an interface to a lower layer of the stack!

Where do these APIs sit?

De-Layering The Network Stack



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

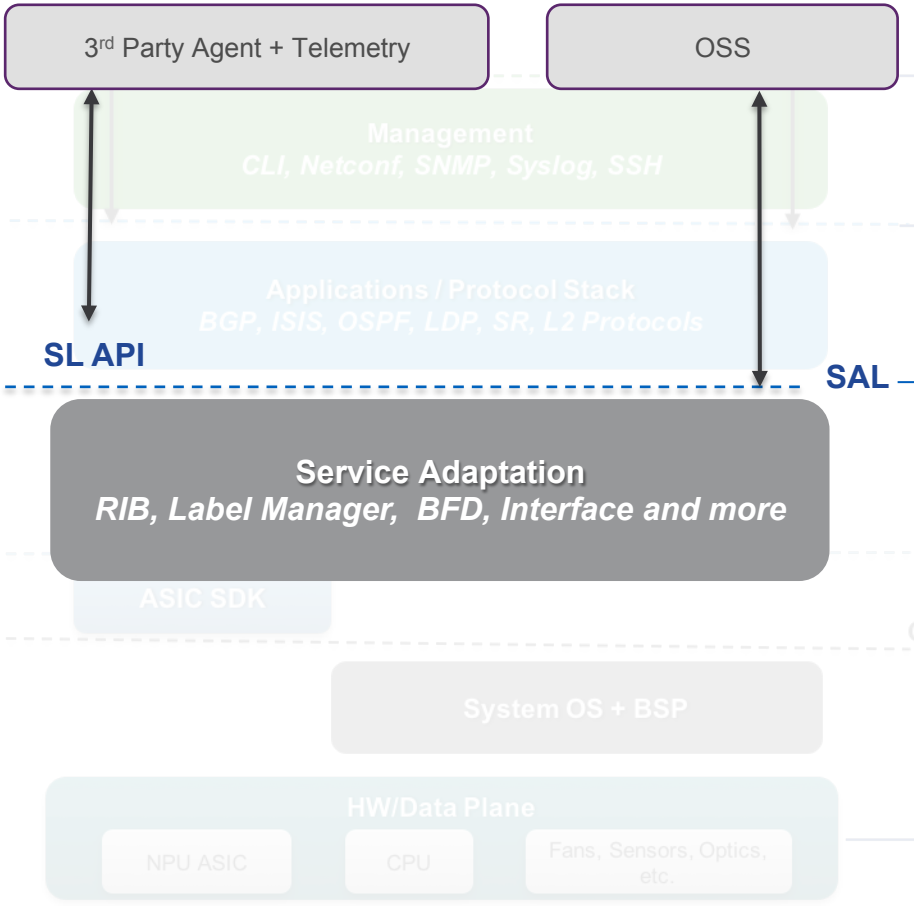
System OS - Linux Kernel
BSP(Board Support Package) - Boot Loader, Device Drivers, etc.

ASIC SDK and drivers for the SDK

Hardware

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

Zooming in .....

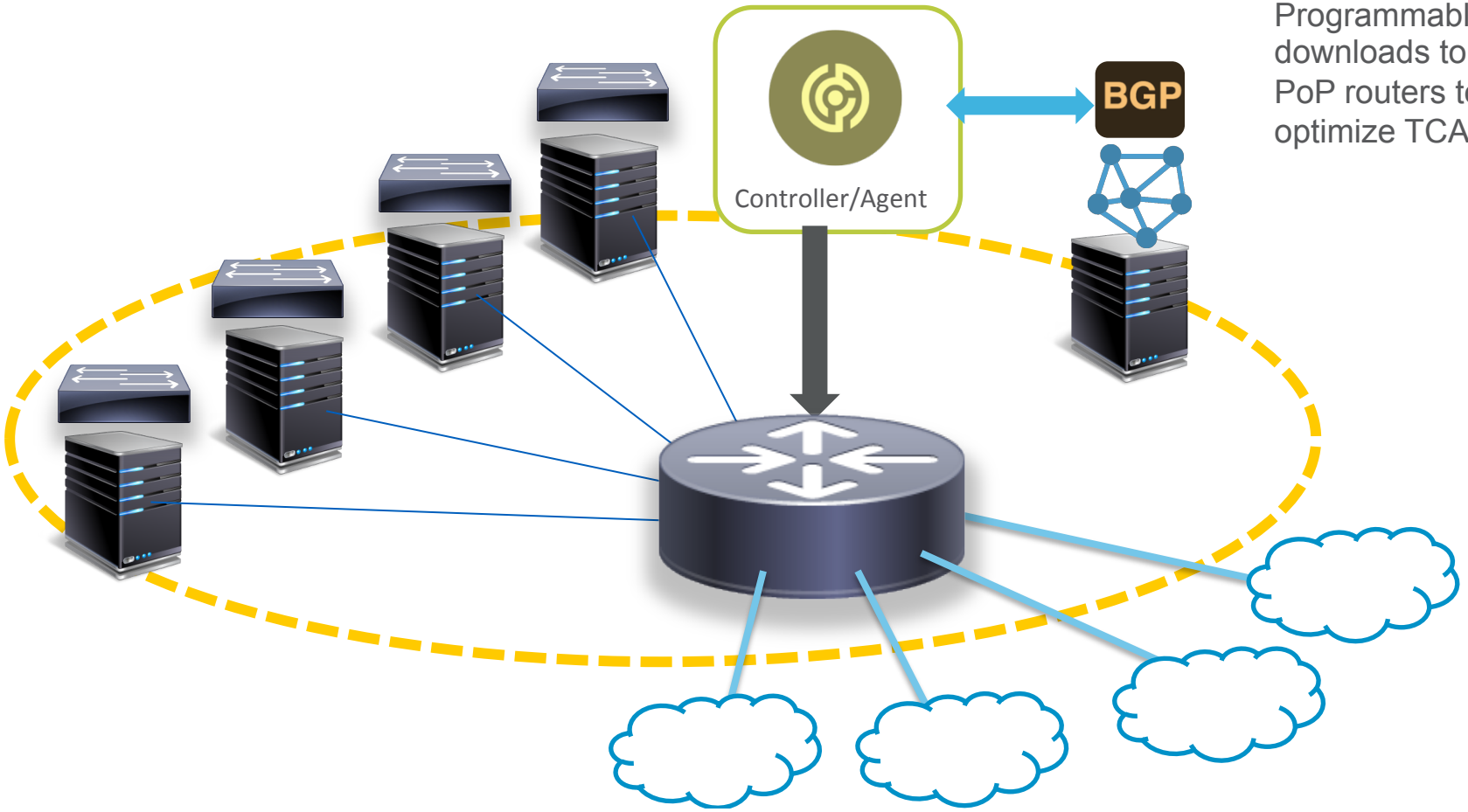


Network Infrastructure Layer / Service Adaptation Layer

- 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

“The Use Cases are Evolving ...”

Programmable Route Downloads



Programmable route downloads to CDN PoP routers to optimize TCAM space.

Traffic Engineering and Path Selection:

Engineering paths for applications through Route/label manipulation, all based on user specific logic.

Bring your own Protocol/Agent

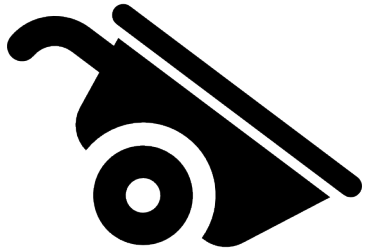
On-box agents and custom protocols that co-exist with standard protocols to influence routing.

“How do we build this API layer? ...”

The Primary Tenets

Performance

API for the
***"Do-it-yourself"* system**

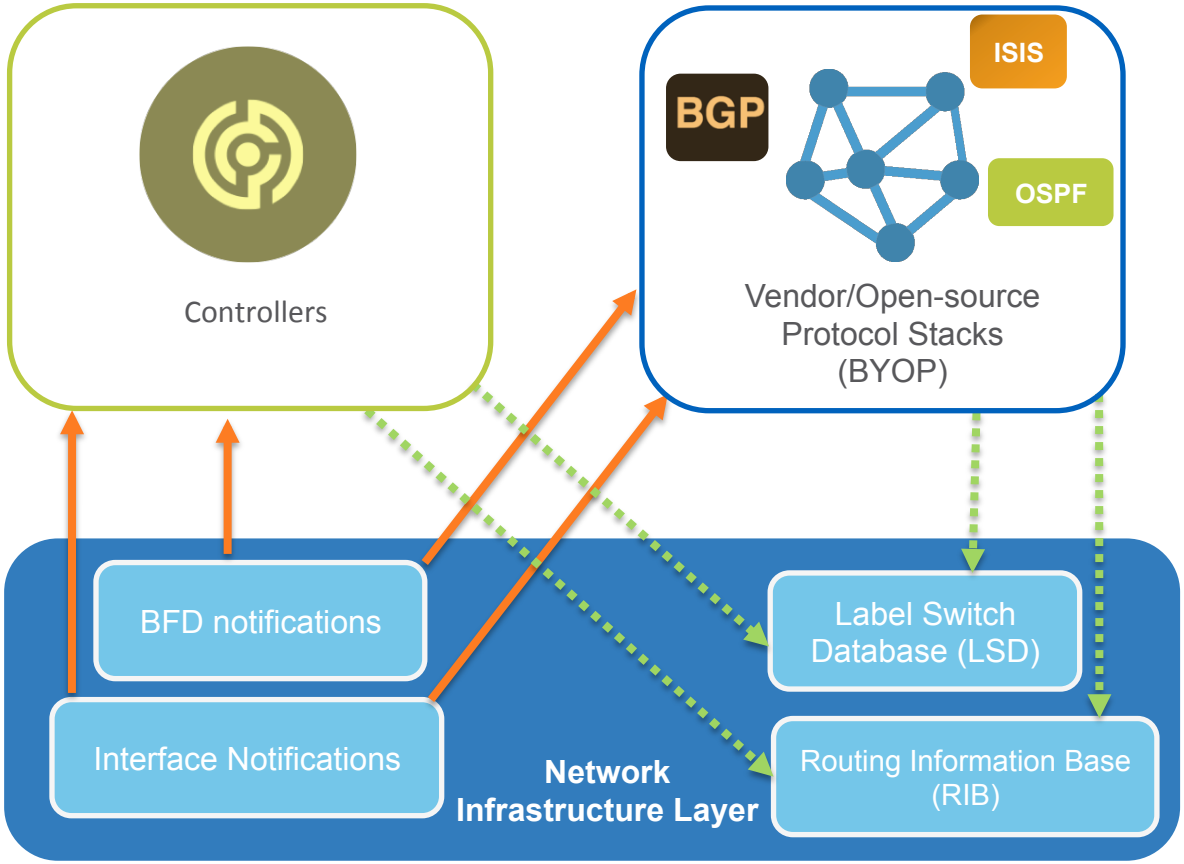


Offload Low-level tasks to Network OS

Highly Performant APIs

Performance

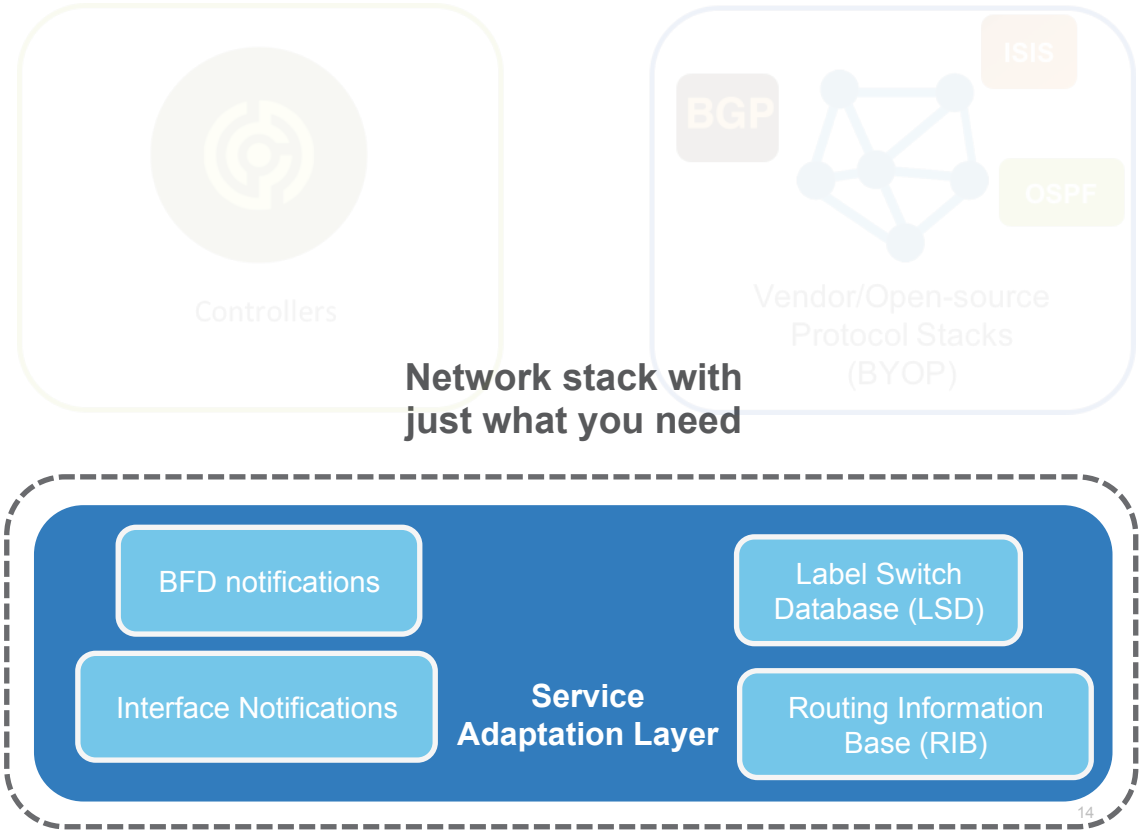
- Batch updates straight to RIB, LSD (and more in the future)
- Streaming Notifications (BFD events, Interface events…)



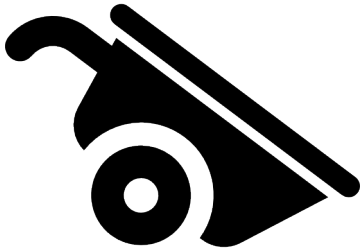
Minimal Network Stack With An Exhaustive API

API for the
***"Do-it-yourself"* system**

- Bring your own Protocol – Use the same APIs that Network OS protocol stacks use internally, over GRPC
- Enable a Network stack with just what you need, but with an exhaustive API

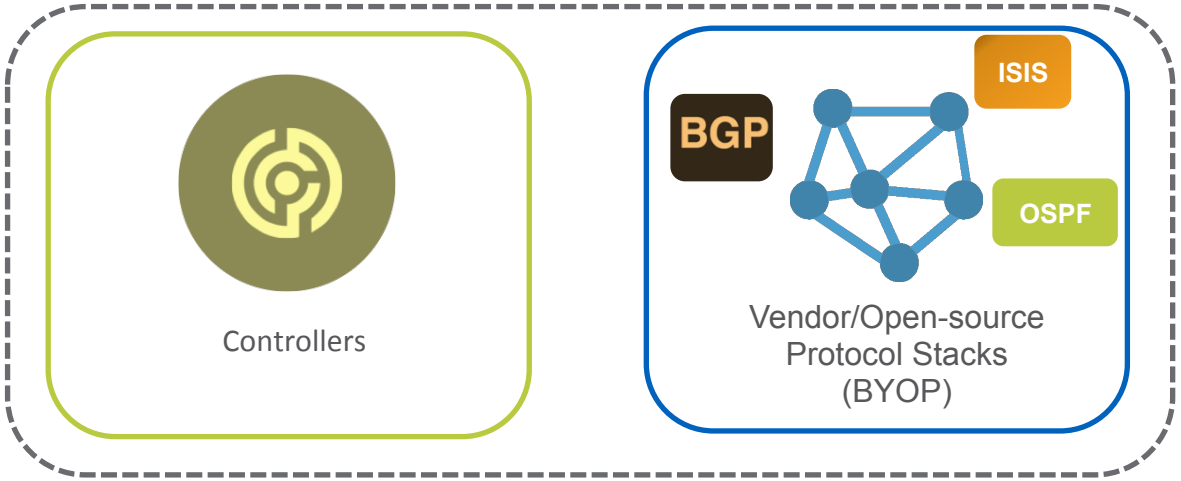


Strategic Offload To The Network Stack



Network OS handles the lower level functions

- Users can focus on higher layer protocols and Controller logic
- Lower Level Functionality – for eg. Route Conflict resolution (RIB), Label Mgmt etc. offloaded to the network OS



Users Focus on The Higher layer Protocols and Controller Logic

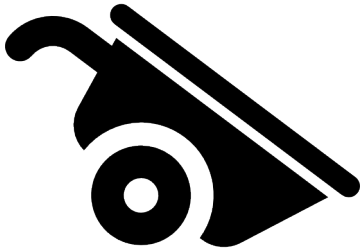
The case for Service Layer APIs

Performance

- Batch updates straight to RIB, LSD (and more in the future), without going through Network state database.
- Streaming Notifications (e.g. BFD events, Interface events…)

API for the *"Do-it-yourself"* system

- Bring your own Protocol – Use the same APIs that Network OS protocol stacks use internally, but over GRPC/thrift.



Offload Low-level tasks to Network OS

- Users can focus on higher layer protocols and Controller logic.
- Leverage Network OS infrastructure layer for Lower Level Functionality that includes scalability and data plane abstraction.

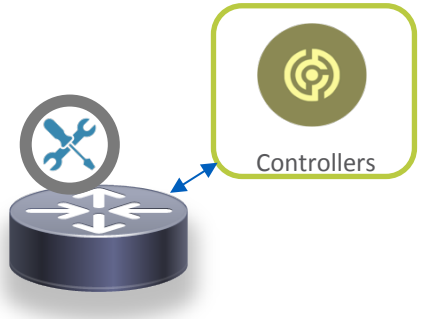
Building APIs for the Network Infrastructure layer

Building Network Infrastructure APIs for Today's Use Cases

Model Driven

Models act as versioned Contracts – easier to understand, document and version.

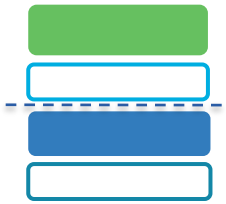
Protobuf IDLs, or YANG are examples of IDLs that may be used to used to model this API layer.



Remote Procedure call (RPC) support

Enables consistency in Application Development.

gRPC, thrift are powerful RPC examples suitable for the performance requirements at this layer.



A Layered approach to APIs

A clean separation of concerns between the infrastructure layer and management/Protocol layer.

It is crucial to have a singular focus for this layer – enabling Vendors to focus on just the right amount of software, with a complete API.

Well, where is the code ?

Cisco Service Layer APIs

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

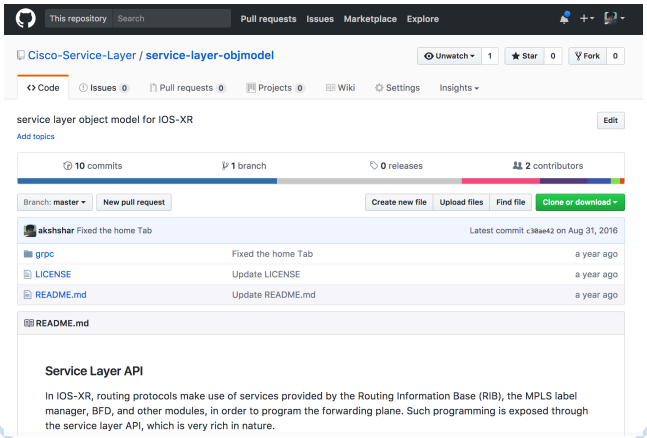
  https://github.com/Cisco-Service-Layer/service-layer-objmodel

  - ➢ Proto definitions of the latest RPC versions
  - ➢ Exhaustive python Unit Tests and tutorials to get started

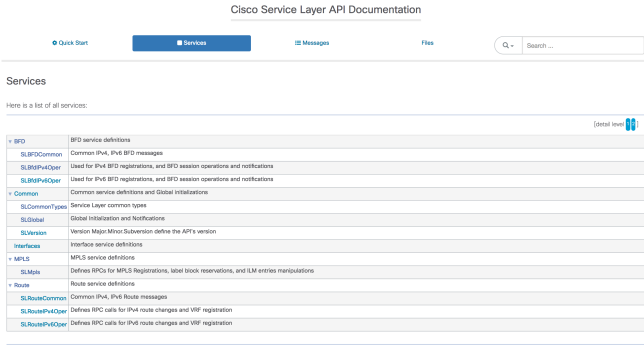
- **@xrdocs:** Blogs, Tutorials on Using Service Layer APIs and associated Libraries:
  https://xrdocs.github.io/cisco-service-layer/

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

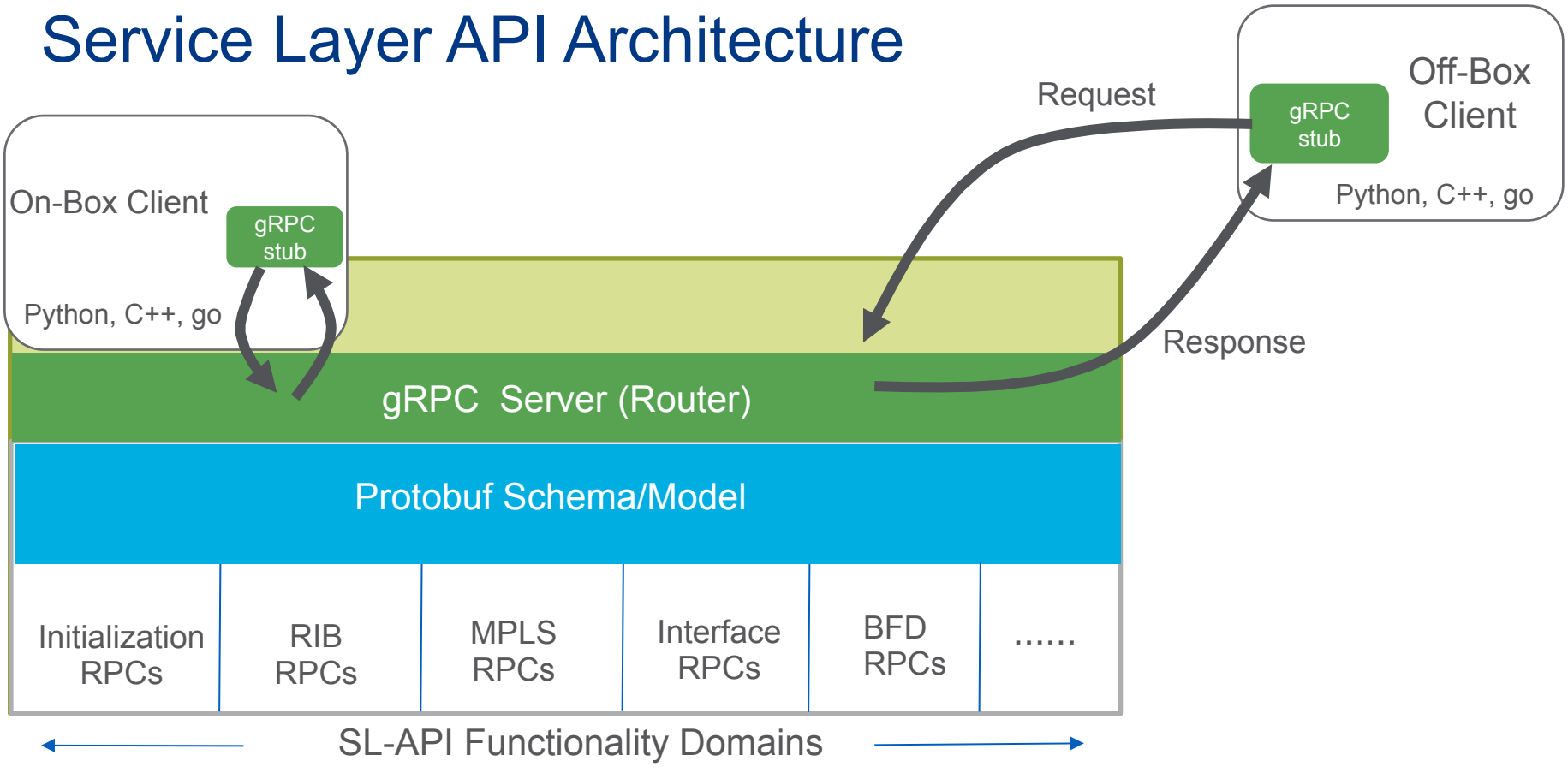
Github



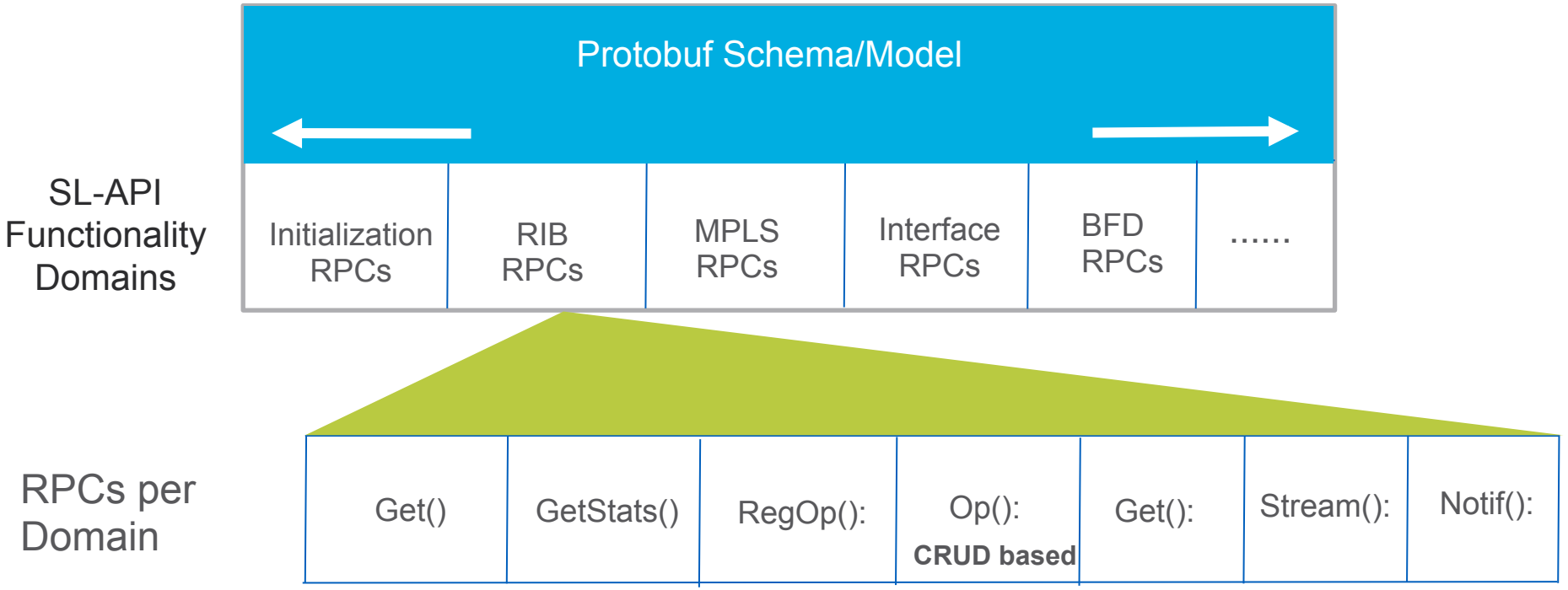
APIdocs @xrdocs



Service Layer API Architecture



Service Layer API Architecture



Other API examples that expose the infrastructure Layer

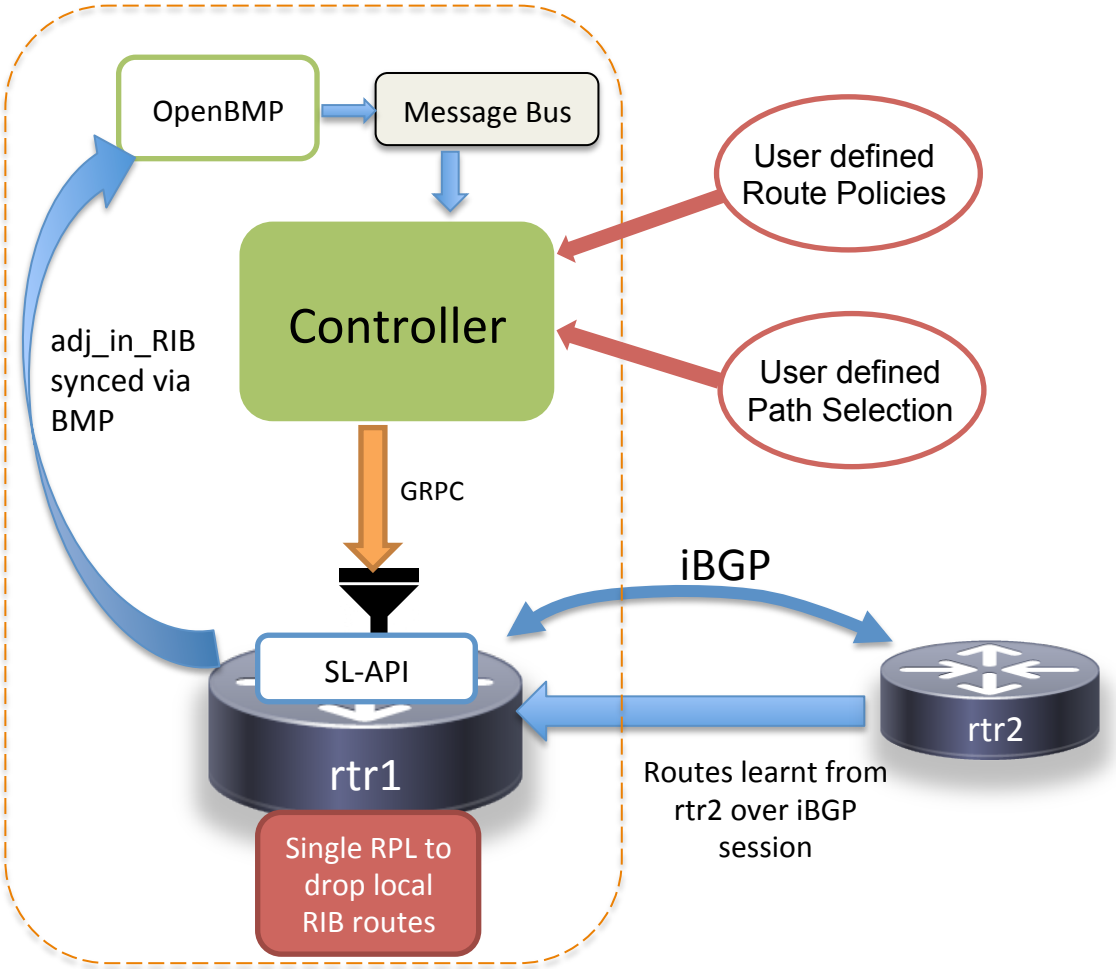
- **Juniper Extension Toolkit (JET):** (https://www.juniper.net/documentation/en_US/release-independent/jet/information-products/pathway-pages/index.html)
  - APIs exposed over gRPC
  - Addresses access to RIB, MPLS, Interface and more.

- **Arista EoS SDK:** (http://aristanetworks.github.io/EosSdk/docs/1.7.0/ref/)
  - On-box C API that exposes access to RIB, MPLS, BFD handlers and more.

Demo!

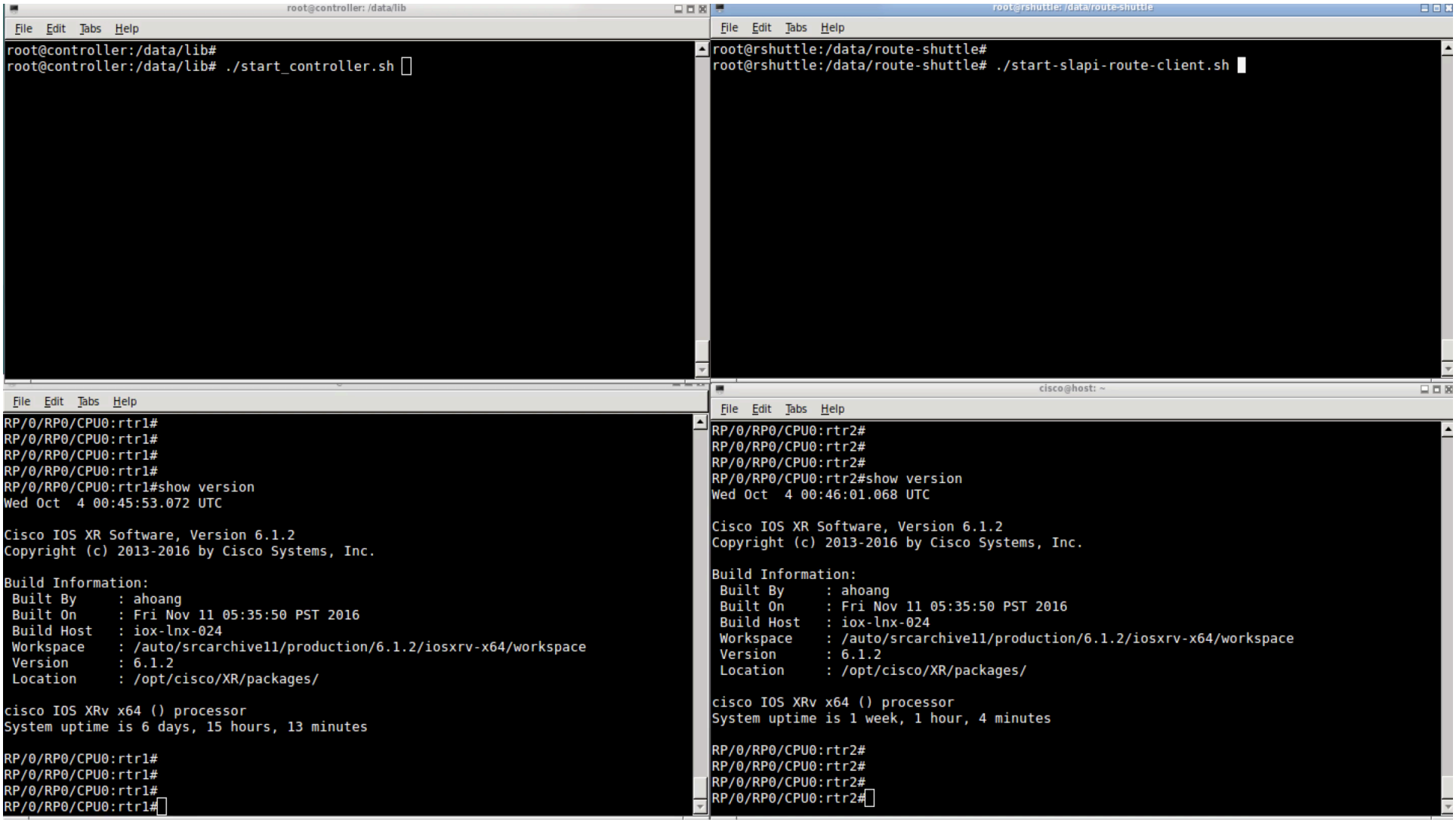
Programmable BGP Route Download

Programmable BGP Route Download

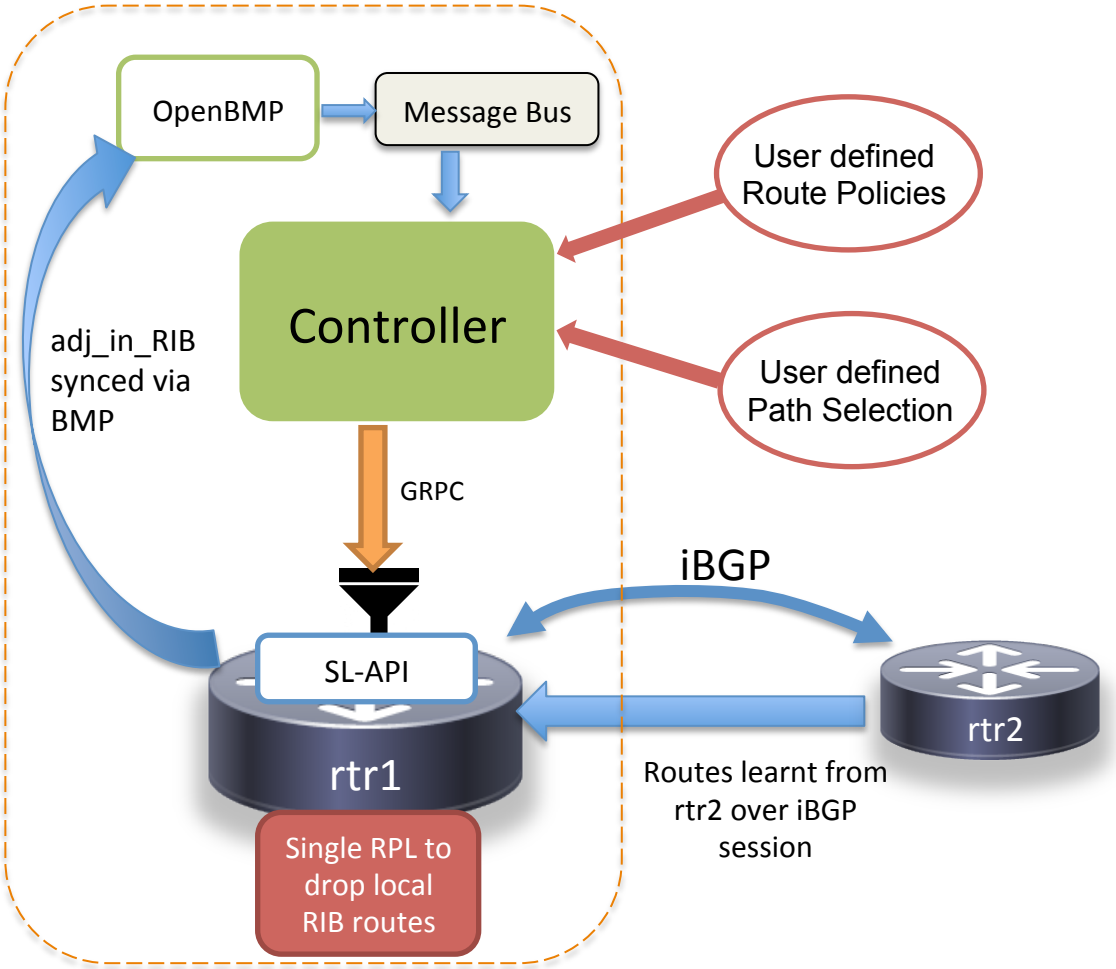


- **Performance:** The Controller and rtr1 form a single Logical Entity thanks to the **highly performant SL-API channel** and react seamlessly to route updates without drops.
- **Flexibility:** Users Can Define their own custom Route Policies and Path Selection algorithms, converting RIB handling into large data set operation.
- **Strategic Offload:** BGP on the router continues to form neighbors, get route updates and distribute routes. Only the RIB manipulation is handled in the controller, allowing user to focus on the core problem – thereby optimizing TCAM space.

https://github.com/Cisco-Service-Layer/openbmp-controller


root@controller: /data/lib
File Edit Tabs Help
root@controller:/data/lib#
root@controller:/data/lib# ./start_controller.sh
root@rshuttle: /data/route-shuttle
File Edit Tabs Help
root@rshuttle:/data/route-shuttle#
root@rshuttle:/data/route-shuttle# ./start-slapi-route-client.sh
File Edit Tabs Help
RP/0/RP0/CPU0:rtr1#
RP/0/RP0/CPU0:rtr1#
RP/0/RP0/CPU0:rtr1#
RP/0/RP0/CPU0:rtr1#
RP/0/RP0/CPU0:rtr1#show version
Wed Oct 4 00:45:53.072 UTC
Cisco IOS XR Software, Version 6.1.2
Copyright (c) 2013-2016 by Cisco Systems, Inc.
Build Information:
Built By : ahoang
Built On : Fri Nov 11 05:35:50 PST 2016
Build Host : iox-lnx-024
Workspace : /auto/srcarchive11/production/6.1.2/iosxrv-x64/workspace
Version : 6.1.2
Location : /opt/cisco/XR/packages/
cisco IOS XRv x64 () processor
System uptime is 6 days, 15 hours, 13 minutes
RP/0/RP0/CPU0:rtr1#
RP/0/RP0/CPU0:rtr1#
RP/0/RP0/CPU0:rtr1#
RP/0/RP0/CPU0:rtr1#
cisco@host: ~
File Edit Tabs Help
RP/0/RP0/CPU0:rtr2#
RP/0/RP0/CPU0:rtr2#
RP/0/RP0/CPU0:rtr2#
RP/0/RP0/CPU0:rtr2#show version
Wed Oct 4 00:46:01.068 UTC
Cisco IOS XR Software, Version 6.1.2
Copyright (c) 2013-2016 by Cisco Systems, Inc.
Build Information:
Built By : ahoang
Built On : Fri Nov 11 05:35:50 PST 2016
Build Host : iox-lnx-024
Workspace : /auto/srcarchive11/production/6.1.2/iosxrv-x64/workspace
Version : 6.1.2
Location : /opt/cisco/XR/packages/
cisco IOS XRv x64 () processor
System uptime is 1 week, 1 hour, 4 minutes
RP/0/RP0/CPU0:rtr2#
RP/0/RP0/CPU0:rtr2#
RP/0/RP0/CPU0:rtr2#
RP/0/RP0/CPU0:rtr2#

Programmable BGP Route Download



- **Performance:** The Controller and rtr1 form a single Logical Entity thanks to the **highly performant SL-API channel** and react seamlessly to route updates without drops.
- **Flexibility:** Users Can Define their own custom Route Policies and Path Selection algorithms, converting RIB handling into large data set operation.
- **Strategic Offload:** BGP on the router continues to form neighbors, get route updates and distribute routes. Only the RIB manipulation is handled in the controller, allowing user to focus on the core problem – thereby optimizing TCAM space.

https://github.com/Cisco-Service-Layer/openbmp-controller