NOKIA

Nokia Service Router Linux

NETOPS DEVELOPMENT KIT API REFERENCE

RELEASE 22.3

3HE 18309 AAAA TQZZA
Issue 01

March 2022

Nokia is committed to diversity and inclusion. We are continuously reviewing our customer documentation and consulting with standards bodies to ensure that terminology is inclusive and aligned with the industry. Our future customer documentation will be updated accordingly.

Table of contents

1 About this guide

The NetOps Development Kit (NDK) allows operators to program high-performance, integrated agents that run alongside the Nokia Service Router Linux (SR Linux). This document provides programming gRPC APIs used with the NDK.

This document is intended for users who plan to program high-performance, integrated agents for SR Linux.

Note:

This manual covers the current release and may also contain some content that will be released in later maintenance loads. See the *SR Linux Release Notes* for information on features supported in each load.

1.1 Precautionary and information messages

The following are information symbols used in the documentation.

DANGER: Danger warns that the described activity or situation may result in serious personal injury or death. An electric shock hazard could exist. Before you begin work on this equipment, be aware of hazards involving electrical circuitry, be familiar with networking environments, and implement accident prevention procedures.

WARNING: Warning indicates that the described activity or situation may, or will, cause equipment damage, serious performance problems, or loss of data.

Caution: Caution indicates that the described activity or situation may reduce your component or system performance.

Note: Note provides additional operational information.

Tip: Tip provides suggestions for use or best practices.

1.2 Conventions

Nokia SR Linux documentation uses the following command conventions.

- **Bold** indicates a command that the user must enter.
- Input and output examples are displayed in `Courier` text.
- An open right angle bracket indicates a progression of menu choices or simple command sequence (often selected from a user interface). Example: **start** > **connect to**
- A vertical bar (|) indicates a mutually exclusive argument.
- Square brackets ([ ]) indicate optional elements.

- Braces ({ }) indicate a required choice. When braces are contained within square brackets, they indicate a required choice within an optional element.
- *Italic* indicates a variable.

Generic IP addresses are used in examples. Replace these with the appropriate IP addresses used in the system.

2 What’s new

This chapter describes new features and enhancements documented in this guide since the previous release. See the *SR Linux Release Notes* for more information about features and enhancements in this release.

Table 1: What's new in this release

Update Description	Change
IfOperDownReason	Updated
KeepAliveRequest	New

3 Introduction

SR Linux provides a NetOps Development Kit (NDK), with a suite of libraries to assist operators with developing agents that run alongside SR Linux applications.

Agents built with the gRPC NDK function similar to other applications provided with SR Linux. SR Linux applications share state details with each other using a publish/ subscribe (pub/sub) architecture. Agents have their own table space within the IDB and can subscribe and receive a notification to events occurring on the device, or create their own table space and publish data to it. This data can be read by other applications within SR Linux, allowing route modifications by publishing routes to the IDB for selection by the FIB manager.

3.1 Datastore

The gRPC NDK allows you to add your own configuration to the system in a non-persistent manner. An NDK-added configuration is considered short-term, and its state is bound to the state of the agent. If an agent fails, the configuration added by the agent using the NDK is removed.

Agents can also add configurations through normal APIs (gNMI/JSON-RPC/CLI). This configuration persists across an agent failure, and the only way to remove it is to overwrite it with a commit command. The short-term datastore is used for agent route injection, while traditional methods for configuring the device are persistent.

3.2 gRPC

SR Linux uses gRPC for inter-process communication. gRPC is a client application that directly calls methods on a server application on a different machine as if it was a local object. The supported external APIs (CLI, gNMI, and JSON-RPC) communicate with the SR Linux and retrieve state information using gRPC.

On the server side, the server implements the interface and runs a gRPC server to handle client calls. On the client side, the client has a stub (or client) that provides the same methods as the server.

gRPC clients and servers can run and talk to each other in a number of environments and can be written in any supported gRPC language. Clients can be created in Go, Python, Ruby, or any other language with gRPC support.

3.3 Protocol buffers

SR Linux’s gRPC NDK uses protocol buffers. Protocol buffers are automated mechanisms for serializing structured data. You define how you want your data to be structured once, then you can use special generated source code to easily write and read your structured data to and from a variety of data streams using a variety of languages. You can also update a data structure without breaking deployed programs that are compiled against an old format.

When working with protocol buffers, structure is defined for serialized data in a proto file (a regular text file with a .proto extension). Each protocol buffer message is a small logical record of information containing a series of name-value pairs. Each message type has one or more uniquely numbered fields, and each field has a name and value type.

Messages also have optional arguments that specify if fields are optional, required, or repeated. New fields can be added to message formats without breaking backwards compatibility, and old binaries ignore any new fields when parsing the message. This allows the gRPC NDK to evolve over time without impacting current deployments.

Once the data structure is specified, a protocol buffer compiler (protoc) generates data access classes from the proto definition. These provide simple accessors for each field (like ConfigData() and set_ConfigData()) and methods to serialize and parse the complete structure to and from raw bytes.

4 Protocol Documentation

4.1 appid_service.proto

4.1.1 AppIdentData

Represents appid data.

Table 2: AppIdentData

Field	Type	Label	Description
name	string		Application name
author	string		Author name
is_connected	bool		Connected to IDB or not
version	string		Version string

4.1.2 AppIdentKey

Represents appid key.

Table 3: AppIdentKey

Field	Type	Label	Description
id	uint32		Application id

4.1.3 AppIdentNotification

Represents appid notification.

Table 4: AppIdentNotification

Field	Type	Label	Description
op	SdkMgrOperation		Operation such as create, delete, or update
key	AppIdentKey		AppIdent key

Field	Type	Label	Description
data	AppIdentData		AppIdent data

4.1.4 AppIdentSubscriptionRequest

Represents appid subscription request.

Table 5: AppIdentSubscriptionRequest

Field	Type	Label	Description
key	AppIdentKey		Optional, to filter on name

4.2 bfd_service.proto

4.2.1 BfdSessionNotification

Represents BFD session notification.

Table 6: BfdSessionNotification

Field	Type	Label	Description
op	SdkMgrOperation		Operation such as session create, delete, or update
key	BfdmgrGeneralSessionKeyPb		Session key
data	BfdmgrGeneralSessionDataPb		Session data

4.2.2 BfdSessionSubscriptionRequest

Represents BFD session subscription request.

Table 7: BfdSessionSubscriptionRequest

Field	Type	Label	Description
key	BfdmgrGeneralSessionKeyPb		Optional, to filter on name

4.2.3 BfdmgrGeneralSessionDataPb

Represents BFD session data.

Table 8: BfdmgrGeneralSessionDataPb

Field	Type	Label	Description
status	BfdmgrSessionStatus		Status of the session
sub_type	BfdmgrGeneralSessionDataPb.BfdmgrSessionSubType		Subtype of the session
src_if_id	uint32		src_if_id is only populated for P2P type Source interface ID

4.2.4 BfdmgrGeneralSessionKeyPb

Represents BFD session key.

Table 9: BfdmgrGeneralSessionKeyPb

Field	Type	Label	Description
type	BfdmgrSessionType		type is always present, other key field presence is determined by type Session type
src_ip_addr	IpAddressPb		key for type == SESSION_TYPE_P2P (if ipv6 link local also uses ipv6_ll_if_id below) Source IP address of the session
dst_ip_addr	IpAddressPb		Destination IP address of the session
instance_id	uint32		Network instance identifier
interface_name	string		type == SESSION_TYPE_MICROBFD
ipv6_ll_if_id	uint32		key for type == SESSION_TYPE_P2P (only ipv6 link local, in addition to other P2P fields above) Global if id for ipv6 link local session, otherwise 0

4.2.5 BfdmgrGeneralSessionDataPb.BfdmgrSessionSubType

Represents BFD session subtype.

Table 10: BfdmgrGeneralSessionDataPb.BfdmgrSessionSubType

Name	Number	Description
SESSION_SUB_TYPE_ UNKNOWN	0	Session subtype unknown
SESSION_SUB_TYPE_ SINGLE_HOP	1	Single-hop session
SESSION_SUB_TYPE_ MULTI_HOP	2	Multi-hop session
SESSION_SUB_TYPE_ MICROBFD	3	microbfd session
SESSION_SUB_TYPE_ SBFD_ECHO	4	microbfd session

4.2.6 BfdmgrSessionStatus

Represents BFD session status.

Table 11: BfdmgrSessionStatus

Name	Number	Description
INVALID	0	Session invalid
ADMIN_DOWN	1	Admin down
DOWN	2	Status down
INIT	3	Status initializing
UP	4	Status up and running

4.2.7 BfdmgrSessionType

Represents BFD session type.

Table 12: BfdmgrSessionType

Name	Number	Description
SESSION_TYPE_UNKNOWN	0	Unknown session type
SESSION_TYPE_P2P	1	Peer-to-peer session type

Name	Number	Description
SESSION_TYPE_MICROBFD	2	microbfd session type
SESSION_TYPE_SBFD_ECHO	3	seamless BFD session type, echo initiator

4.3 config_service.proto

4.3.1 ConfigData

Represents configuration data.

Table 13: ConfigData

Field	Type	Label	Description
json	string		Entire configuration fragment as JSON string

4.3.2 ConfigKey

Represents configuration key.

Table 14: ConfigKey

Field	Type	Label	Description
js_path	string		JSON path formatted string from YANG; for example, interface{.name==ethernet1/1}.my_field
keys	string	repeated	Value for keys

4.3.3 ConfigNotification

Represents configuration notification message to subscribe to configuration events

Table 15: ConfigNotification

Field	Type	Label	Description
op	SdkMgrOperation		Operation indicating create, delete, or update

Field	Type	Label	Description
key	ConfigKey		Configuration key
data	ConfigData		Configuration data

4.3.4 ConfigSubscriptionRequest

Represents configuration subscription request.

Table 16: ConfigSubscriptionRequest

Field	Type	Label	Description
key	ConfigKey		Optional, to filter on name

4.4 interface_service.proto

4.4.1 InterfaceData

Represents interface data.

Table 17: InterfaceData

Field	Type	Label	Description
admin_is_up	uint32		Admin state
mtu	uint32		Maximum transmission unit
if_type	IfMgrIfType		Interface type; for example, loopback, physical, or LAG
port_id	PortIdPb		Port identifier
description	string		Interface description
mac_addr	MacAddressPb		MAC address
aggregate_id	string		associated aggregate id
oper_is_up	uint32		Operational state

4.4.2 InterfaceKey

Represents interface key.

Table 18: InterfaceKey

Field	Type	Label	Description
if_name	string		Interface name; for example, ethernet 1/1

4.4.3 InterfaceNotification

Represents interface notification.

Table 19: InterfaceNotification

Field	Type	Label	Description
op	SdkMgrOperation		Operation such as create, delete, or update
key	InterfaceKey		Interface key
data	InterfaceData		Interface data

4.4.4 InterfaceSubscriptionRequest

Represents interface subscription request.

Table 20: InterfaceSubscriptionRequest

Field	Type	Label	Description
key	InterfaceKey		Optional, to filter on name

4.5 lldp_service.proto

4.5.1 LldpNeighborDataPb

Represents LLDP neighbor data.

Table 21: LldpNeighborDataPb

Field	Type	Label	Description
port_id	string		Port identifier
port_type	LldpNeighborDataPb.PortSubType		Port type

Field	Type	Label	Description
source_mac	MacAddressPb		Port MAC address
bgp_peer_address	IpAddressPb	repeated	LLDP BGP autodiscovered addresses
bgp_group_id	uint32		BGP group identifier
system_name	string		System name
system_description	string		System description

4.5.2 LldpNeighborKeyPb

Represents LLDP neighbor key.

Table 22: LldpNeighborKeyPb

Field	Type	Label	Description
interface_name	string		Local interface name
chassis_id	string		Chassis identifier
chassis_type	LldpNeighborKeyPb.ChassisIdType		Chassis type

4.5.3 LldpNeighborNotification

Represents LLDP neighbor notification.

Table 23: LldpNeighborNotification

Field	Type	Label	Description
op	SdkMgrOperation		Operation such as create, delete, or update
key	LldpNeighborKeyPb		LLDP neighbor key
data	LldpNeighborDataPb		LLDP neighbor data

4.5.4 LldpNeighborSubscriptionRequest

Represents LLDP neighbor subscription request.

Table 24: LldpNeighborSubscriptionRequest

Field	Type	Label	Description
key	LldpNeighborKeyPb		Optional, to filter on name

4.5.5 LldpNeighborDataPb.PortSubType

Represents port subtype.

Table 25: LldpNeighborDataPb.PortSubType

Name	Number	Description
RESERVED	0	Reserved for future use
INTERFACE_ALIAS	1	Alias of the interface
PORT_COMPONENT	2	Port identifier based on a locally defined port component
MAC_ADDRESS	3	MAC address
NETWORK_ADDRESS	4	Network address
INTERFACE_NAME	5	Name of the interface
AGENT_CIRCUIT_ID	6	Port identifier based on the circuit ID in the DHCP relay agent information option
LOCALLY_ASSIGNED	7	Port identifier based on a locally defined alphanumeric string

4.5.6 LldpNeighborKeyPb.ChassisIdType

Represents chassis type.

Table 26: LldpNeighborKeyPb.ChassisIdType

Name	Number	Description
RESERVED	0	Reserved for future use
CHASSIS_COMPONENT	1	Chassis identifier based on a locally defined chassis component
INTERFACE_ALIAS	2	Alias of the interface

Name	Number	Description
PORT_COMPONENT	3	Chassis identifier based on a locally defined port component
MAC_ADDRESS	4	MAC address
NETWORK_ADDRESS	5	Network address
INTERFACE_NAME	6	Name of the interface
LOCALLY_ASSIGNED	7	Chassis identifier based on a locally defined value

4.6 mpls_service.proto

4.6.1 MplsRouteAddRequest

Represents MPLS route add request, which can include one or more MPLS routes.

Table 27: MplsRouteAddRequest

Field	Type	Label	Description
routes	MplsRouteInfo	repeated	MPLS routes

4.6.2 MplsRouteAddResponse

Represents MPLS route add response.

Table 28: MplsRouteAddResponse

Field	Type	Label	Description
status	SdkMgrStatus		Status of MPLS route add request
error_str	string		Detailed error string

4.6.3 MplsRouteDeleteRequest

Represents MPLS route delete request, which can include one or more MPLS routes.

Table 29: MplsRouteDeleteRequest

Field	Type	Label	Description
routes	MplsRouteKeyPb	repeated	MPLS routes

4.6.4 MplsRouteDeleteResponse

Represents MPLS route delete response.

Table 30: MplsRouteDeleteResponse

Field	Type	Label	Description
status	SdkMgrStatus		Status of MPLS route delete request
error_str	string		Detailed error string

4.6.5 MplsRouteInfo

Represents MPLS route information; contains key and data.

Table 31: MplsRouteInfo

Field	Type	Label	Description
key	MplsRouteKeyPb		MPLS route key
data	MplsRoutePb		MPLS route data

4.6.6 MplsRouteKeyPb

Represents MPLS route key.

Table 32: MplsRouteKeyPb

Field	Type	Label	Description
top_label	MplsLabel		Top label
net_inst_name	string		Network instance name

4.6.7 MplsRoutePb

Represents MPLS route data.

Table 33: MplsRoutePb

Field	Type	Label	Description
nexthop_group_name	string		Next hop group name
operation	MplsRoutePb.Operation		Operation such as POP or SWAP
preference	uint32		Route preference

4.6.8 MplsRoutePb.Operation

Represents MPLS operation.

Table 34: MplsRoutePb.Operation

Name	Number	Description
INVALID_OP	0	Invalid operation
POP	1	Pop operation
SWAP	2	Swap operation

4.6.9 SdkMgrMplsRouteService

Represents service for MPLS programming.

Table 35: SdkMgrMplsRouteService

Method Name	Request Type	Response Type	Description
MplsRouteAdd OrUpdate	MplsRouteAddRequest	MplsRouteAdd Response	MPLS route add or update; can add or update more than one MPLS route in one request
MplsRouteDelete	MplsRouteDeleteRequest	MplsRouteDelete Response	MPLS route delete; can delete more than one MPLS route in one request
SyncStart	SyncRequest	SyncResponse	Synchronization start to begin synchronization operation
SyncEnd	SyncRequest	SyncResponse	Synchronization end to close synchronization operation

4.6.10 Methods with deprecated option

Represents methods with deprecated option.

Table 36: Methods with deprecated option

Method Name	Option
MplsRouteAddOrUpdate	true
MplsRouteDelete	true

4.7 networkinstance_service.proto

4.7.1 NetworkInstanceData

Represents network instance data.

Table 37: NetworkInstanceData

Field	Type	Label	Description
net_inst_id	uint32		Network instance identifier
base_name	string		Base name
loopback_addr	IpAddrPrefLenPb		Loopback address of network instance
oper_is_up	bool		Operation status
router_id	string		Router identifier
inst_type	NetworkInstanceData.NetInstType		Network instance type

4.7.2 NetworkInstanceKey

Represents network instance key.

Table 38: NetworkInstanceKey

Field	Type	Label	Description
inst_name	string		Network instance name

4.7.3 NetworkInstanceNotification

Represents network instance notification.

Table 39: NetworkInstanceNotification

Field	Type	Label	Description
op	SdkMgrOperation		Operation such as create, delete, or update
key	NetworkInstanceKey		Network key
data	NetworkInstanceData		Network data

4.7.4 NetworkInstanceSubscriptionRequest

Represents network instance subscription request.

4.7.5 NetworkInstanceData.NetInstType

Represents network instance type.

Table 40: NetworkInstanceData.NetInstType

Name	Number	Description
DEFAULT	0	Default network instance type
L3VRF	1	L3VRF network instance type

4.8 nexthop_group_service.proto

4.8.1 MplsNextHop

Represents MPLS next hop.

Table 41: MplsNextHop

Field	Type	Label	Description
ip_nexthop	IpAddressPb		Next-hop IP address
label_stack	MplsLabel	repeated	MPLS label stack

4.8.2 NextHop

Represents next-hop.

Table 42: NextHop

Field	Type	Label	Description
resolve_to	NextHop.ResolveToType		Resolve-to type
type	NextHop.ResolutionType		Resolution type
ip_nexthop	IpAddressPb		IP next-hop address
mpls_nexthop	MplsNextHop		MPLS next-hop

4.8.3 NextHopGroup

Represents next-hop group.

Table 43: NextHopGroup

Field	Type	Label	Description
next_hop	NextHop	repeated	Next-hops

4.8.4 NextHopGroupDeleteRequest

Represents next-hop group delete request.

Table 44: NextHopGroupDeleteRequest

Field	Type	Label	Description
group_key	NextHopGroupKey	repeated	Next-hop group key details

4.8.5 NextHopGroupDeleteResponse

Represents next-hop group delete response.

Table 45: NextHopGroupDeleteResponse

Field	Type	Label	Description
status	SdkMgrStatus		Response for next-hop group request
error_str	string		Detailed error string

4.8.6 NextHopGroupInfo

Represents next-hop group information.

Table 46: NextHopGroupInfo

Field	Type	Label	Description
key	NextHopGroupKey		Next-hop group key
data	NextHopGroup		Next-hop group data

4.8.7 NextHopGroupKey

Represents next-hop group key.

Table 47: NextHopGroupKey

Field	Type	Label	Description
name	string		Next-hop group name
network_instance_name	string		Next-hop group network instance name

4.8.8 NextHopGroupNotification

Represents next-hop group notification.

Table 48: NextHopGroupNotification

Field	Type	Label	Description
op	SdkMgrOperation		Operation such as create, delete, or update
key	uint64		Next-hop group key
data	NextHopGroup		Next-hop group data

4.8.9 NextHopGroupRequest

Represents next-hop group request.

Table 49: NextHopGroupRequest

Field	Type	Label	Description
group_info	NextHopGroupInfo	repeated	Next-hop group details

4.8.10 NextHopGroupResponse

Represents next-hop group response.

Table 50: NextHopGroupResponse

Field	Type	Label	Description
status	SdkMgrStatus		Response for next-hop group request
error_str	string		Detailed error string

4.8.11 NextHopGroupSubscriptionRequest

Represents next-hop group subscription request.

Table 51: NextHopGroupSubscriptionRequest

Field	Type	Label	Description
key	NextHopGroupKey		Optional, to filter on name

4.8.12 NextHop.ResolutionType

Represents resolution type.

Table 52: NextHop.ResolutionType

Name	Number	Description
INVALID	0	Invalid resolution
REGULAR	1	Regular resolution
MPLS	2	MPLS resolution

4.8.13 NextHop.ResolveToType

Represents resolve-to type.

Table 53: NextHop.ResolveToType

Name	Number	Description
LOCAL	0	Resolve to local routes
DIRECT	1	Resolve to direct routes
INDIRECT	2	Resolve to indirect routes

4.8.14 SdkMgrNextHopGroupService

Represents service for next-hop group operations.

Table 54: SdkMgrNextHopGroupService

Method Name	Request Type	Response Type	Description
NextHopGroupAddOrUpdate	NextHopGroupRequest	NextHopGroupResponse	Add or update one or more next-hop groups.
NextHopGroupDelete	NextHopGroupDeleteRequest	NextHopGroupDeleteResponse	Delete next-hop group.
SyncStart	SyncRequest	SyncResponse	Synchronization start to open synchronization operation.
SyncEnd	SyncRequest	SyncResponse	Synchronization end to close synchronization operation.

4.9 route_service.proto

4.9.1 IpRouteNotification

Represents IP route notification.

Table 55: IpRouteNotification

Field	Type	Label	Description
op	SdkMgrOperation		Operation such as create, delete, or update
key	RouteKeyPb		IP route key
data	RoutePb		IP route data

4.9.2 IpRouteSubscriptionRequest

Represents IP route subscription request.

Table 56: IpRouteSubscriptionRequest

Field	Type	Label	Description
key	RouteKeyPb		Optional, to filter on name

4.9.3 RouteAddRequest

Represents route add request; can contain more than one route.

Table 57: RouteAddRequest

Field	Type	Label	Description
routes	RouteInfo	repeated	IP routes

4.9.4 RouteAddResponse

Represents route add response.

Table 58: RouteAddResponse

Field	Type	Label	Description
status	SdkMgrStatus		Status of route add operation
error_str	string		Detailed error string

4.9.5 RouteDeleteRequest

Represents route delete request; can contain more than one route.

Table 59: RouteDeleteRequest

Field	Type	Label	Description
routes	RouteKeyPb	repeated	IP routes

4.9.6 RouteDeleteResponse

Represents route delete response.

Table 60: RouteDeleteResponse

Field	Type	Label	Description
status	SdkMgrStatus		Status of route delete operation
error_str	string		Detailed error string

4.9.7 RouteInfo

Represents route information.

Table 61: RouteInfo

Field	Type	Label	Description
key	RouteKeyPb		Route key
data	RoutePb		Route data

4.9.8 RouteKeyPb

Represents route key.

Table 62: RouteKeyPb

Field	Type	Label	Description
net_inst_name	string		Network instance name
ip_prefix	IpAddrPrefLenPb		IP prefix

4.9.9 RoutePb

Represents route data.

Table 63: RoutePb

Field	Type	Label	Description
nexthop_group_name	string		Next hop group name
preference	uint32		Preference
metric	uint32		Metric
nexthop	NextHop	repeated	List of next hops

Field	Type	Label	Description
owner_id	uint32		Next hop owner identifier returned only on notification.
nhg_id	uint64		Next-hop group identifier returned only on notification.

4.9.10 SdkMgrRouteService

Represents service for IP route operations.

Table 64: SdkMgrRouteService

Method Name	Request Type	Response Type	Description
RouteAddOrUpdate	RouteAddRequest	RouteAdd Response	Add or update IP routes.
RouteDelete	RouteDeleteRequest	RouteDelete Response	Delete IP routes.
SyncStart	SyncRequest	SyncResponse	Synchronization start for IP routes
SyncEnd	SyncRequest	SyncResponse	Synchronization end for IP routes

4.10 sdk_common.proto

4.10.1 AgentReply

Empty message from agent.

4.10.2 EvpnEthSegIdPb

Table 65: EvpnEthSegIdPb

Field	Type	Label	Description
es_id	bytes		Type 0 for now. hard-coded id

4.10.3 GlobalIfId

Represents global interface identifier.

Table 66: GlobalIfId

Field	Type	Label	Description
global_if_id	uint32		Global interface identifier

4.10.4 IpAddrPrefLenPb

Represents IP prefix.

Table 67: IpAddrPrefLenPb

Field	Type	Label	Description
ip_addr	IpAddressPb		IP address
prefix_length	uint32		IP address prefix length

4.10.5 IpAddressPb

Represents IP address.

Table 68: IpAddressPb

Field	Type	Label	Description
addr	bytes		IP address

4.10.6 IpInterfaceAddrPrefixPb

Represents IP prefix state.

Table 69: IpInterfaceAddrPrefixPb

Field	Type	Label	Description
prefix	IpAddrPrefLenPb		IP prefix
state	IpAddressState		IP prefix state

4.10.7 MacAddressPb

Represents MAC address.

Table 70: MacAddressPb

Field	Type	Label	Description
mac_address	bytes		MAC address

4.10.8 MplsLabel

Represents MPLS label.

Table 71: MplsLabel

Field	Type	Label	Description
mpls_label	uint32		MPLS label

4.10.9 NetInstanceId

Represents network instance identifier.

Table 72: NetInstanceId

Field	Type	Label	Description
instance_id	uint32		Network instance identifier

4.10.10 PortIdPb

Represents port identifier.

Table 73: PortIdPb

Field	Type	Label	Description
port_id	uint64		Port identifier

4.10.11 SyncRequest

Empty message for synchronization request.

4.10.12 SyncResponse

Empty message for synchronization end.

Table 74: SyncResponse

Field	Type	Label	Description
status	SdkMgrStatus		Error code
error_str	string		Detailed error string

4.10.13 IfEthernetDuplexModeType

Represents interface ethernet duplex mode. Corresponds to yang values

Table 75: IfEthernetDuplexModeType

Name	Number	Description
IF_ETH_DUPLEX_MODE_UNSET	0	duplex mode not supported
IF_ETH_DUPLEX_MODE_FULL	1	
IF_ETH_DUPLEX_MODE_HALF	2	

4.10.14 IfEthernetPortSpeedType

Represents interface ethernet port speed. Corresponds to yang values

Table 76: IfEthernetPortSpeedType

Name	Number	Description
IF_ETH_PORT_SPEED_UNSET	0	Speed unknown
IF_ETH_PORT_SPEED_10M	1	
IF_ETH_PORT_SPEED_100M	2	
IF_ETH_PORT_SPEED_1G	3	
IF_ETH_PORT_SPEED_10G	4	
IF_ETH_PORT_SPEED_25G	5	
IF_ETH_PORT_SPEED_40G	6	

Name	Number	Description
IF_ETH_PORT_SPEED_50G	7	
IF_ETH_PORT_SPEED_ 100G	8	
IF_ETH_PORT_SPEED_ 200G	9	
IF_ETH_PORT_SPEED_ 400G	10	
IF_ETH_PORT_SPEED_1T	11	

4.10.15 IfMgrIfType

Represents interface type.

Table 77: IfMgrIfType

Name	Number	Description
ETHERNET	0	Ethernet interface
LOOPBACK	1	Loopback interface
MANAGEMENT	2	Management interface
AGGREGATE	3	Aggregate(LAG) interface
IRB	4	Integrated Routing and Bridging (IRB) interface
SYSTEM	5	System interface
LIF	6	linux interface
NIC	7	linux nic interface (bus/dev/fn)
VHOST	8	vhost-net interface, vhn-<name> name for sock-path
KKLIF	9	temp name for new style of lif interface
KKVHOST	10	temp name for new style of vhost interface
OTC	11	SAS otc interface
MAX	12	

4.10.16 IfOperDownReason

Table 78: IfOperDownReason

Name	Number	Description
IF_OPER_DOWN_NONE	0	
IF_OPER_DOWN_PORT_ ADMIN_DISABLED	1	
IF_OPER_DOWN_MDA_ ADMIN_DISABLED	2	
IF_OPER_DOWN_TRANS_ LASER_DISABLED	3	
IF_OPER_DOWN_MDA_ NOT_PRESENT	4	
IF_OPER_DOWN_TRANS_ NOT_PRESENT	5	
IF_OPER_DOWN_PHY_INIT	6	
IF_OPER_DOWN_ LOOPBACK	7	
IF_OPER_DOWN_LOWER_ LAYER_DOWN	8	
IF_OPER_DOWN_MTU_ RESOURCES	9	
IF_OPER_DOWN_ UNSUPPORTED_SPEED	10	
IF_OPER_DOWN_ UNSUPPORTED_TRANS_ FEC	11	
IF_OPER_DOWN_OTHER	12	
IF_OPER_DOWN_PORT_ NOT_PRESENT	13	used internally by chassis mgr only - xdp never publish to IDB!
IF_OPER_DOWN_FABRIC_ AVAILABILITY	14	used internally by chassis mgr only - xdp never publish to IDB!
IF_OPER_DOWN_NO_ ACTIVE_LINKS	15	lag interface only

Name	Number	Description
IF_OPER_DOWN_MIN_ LINK_THRESHOLD	16	lag interface only
IF_OPER_DOWN_9_12_ SPEED_MISMATCH	17	Vodka port 9-12 must all be same speed as port 9
IF_OPER_DOWN_LAG_ RESOURCES	18	lag interface only
IF_OPER_DOWN_LAG_ MEMBER_RESOURCES	19	lag member interface only
IF_OPER_DOWN_ STANDBY_SIGNALING	20	ESM multihoming
IF_OPER_DOWN_HOLD_ TIME_UP_ACTIVE	21	interface hold-time up is actively holding the interface down
IF_OPER_DOWN_RELOAD_ TIME_ACTIVE	22	interface reload time is actively holding the interface down
IF_OPER_DOWN_ CONNECTOR_DOWN	23	parent connector oper down forces breakout port oper down
IF_OPER_DOWN_AUTO_ NEG_MISMATCH	24	
IF_OPER_DOWN_EVENT_ HANDLER	25	used internally by chassis mgr only - xdp never publish to IDB!

4.10.17 IfOperStateType

Represents interface operational state.

Table 79: IfOperStateType

Name	Number	Description
IF_OPER_STATE_UP	0	Interface operational state up
IF_OPER_STATE_DOWN	1	Interface operational state down
IF_OPER_STATE_TESTING	2	Interface operational state testing
IF_OPER_STATE_ UNKNOWN	3	Interface operational state unknown
IF_OPER_STATE_DORMANT	4	Interface operational state dormant

Name	Number	Description
IF_OPER_STATE_NOT_PRESENT	5	Interface operational state not present
IF_OPER_STATE_LOWER_LAYER_DOWN	6	Interface operational state lower layer down

4.10.18 IfTransceiverFecType

Represents interface transceiver fec. Corresponds to yang values

Table 80: IfTransceiverFecType

Name	Number	Description
IF_TRANS_FEC_UNSET	0	Fec unknown
IF_TRANS_FEC_DISABLED	1	
IF_TRANS_FEC_RS528	2	
IF_TRANS_FEC_RS544	3	
IF_TRANS_FEC_BASER	4	
IF_TRANS_FEC_RS108	5	

4.10.19 IpAddressState

Represents IP address state.

Table 81: IpAddressState

Name	Number	Description
IPADDR_STATE_UNKNOWN	0	IP address state unknown
IPADDR_STATE_TENTATIVE	1	IP address state tentative
IPADDR_STATE_DUPLICATED	2	IP address state duplicated
IPADDR_STATE_INACCESSIBLE	3	IP address state inaccessible
IPADDR_STATE_DEPRECATED	4	IP address state deprecated

Name	Number	Description
IPADDR_STATE_ PREFERRED	5	IP address state preferred

4.10.20 SdkMgrOperation

Represents enumeration value for operation in subscription.

Table 82: SdkMgrOperation

Name	Number	Description
Create	0	Create operation
Change	1	Change operation
Delete	2	Delete operation

4.10.21 SdkMgrStatus

Represents status of network programming service calls.

Table 83: SdkMgrStatus

Name	Number	Description
kSdkMgrSuccess	0	Successful service call
kSdkMgrFailed	1	Failed service call

Network programming APIs and messages. This is the base layer for agent registration, router event notifications such as interface, LLDP, BFD, and so on; also provides keepalive functionality to detect agent liveliness

4.11 sdk_service.proto

4.11.1 AgentRegistrationRequest

Represents registration request message used in agent register and unregister.

Table 84: AgentRegistrationRequest

Field	Type	Label	Description
js_path	string	repeated	Optional, JSON path formatted strings, which are used in

Field	Type	Label	Description
			telemetry. Format of js_path follows hierarchical YANG. for example: .interface{.name==*}.my_app. .my_app.tunnel{.name==*} '*" needs to be replaced with a specific key.
agent_liveliness	uint32		Kill this agent unless a keepalive is received within this many seconds. Value of 0 means do not monitor this agent for liveliness.

4.11.2 AgentRegistrationResponse

Represents registration response in reply to registration request.

Table 85: AgentRegistrationResponse

Field	Type	Label	Description
status	SdkMgrStatus		Status of the register; for example: kOk, kFailed
error_str	string		Detailed error text
app_id	uint32		Application ID assigned by SDK manager.

4.11.3 AppIdRequest

Represents application identifier request from agent. All applications are assigned an identifier by IDB.

Table 86: AppIdRequest

Field	Type	Label	Description
name	string		Application name

4.11.4 AppIdResponse

Represents application identifier response to agent.

Table 87: AppIdResponse

Field	Type	Label	Description
status	SdkMgrStatus		Status of the call; for example, k Ok, kFailed
id	uint32		Identifier for the given application name

4.11.5 KeepAliveRequest

Represents keep alive request from agent to refresh liveliness of the agent.

4.11.6 KeepAliveResponse

Represents keepalive response.

Table 88: KeepAliveResponse

Field	Type	Label	Description
status	SdkMgrStatus		Status of keepalive; for example, k Ok or kFailed

4.11.7 Notification

Represents notification stream response.

Table 89: Notification

Field	Type	Label	Description
sub_id	uint64		Subscription identifier
intf	InterfaceNotification		Interface details
nw_inst	NetworkInstanceNotification		Network instance details
lldp_neighbor	LldpNeighborNotification		LLDP neighbor details
config	ConfigNotification		Configuration notification
bfd_session	BfdSessionNotification		BFD session details
route	IpRouteNotification		IP route details
appid	AppIdentNotification		App identification details

Field	Type	Label	Description
nhg	NextHopGroupNotification		Next-hop group details

4.11.8 NotificationQueryRequest

Represents notification query to return specific subscription details.

Table 90: NotificationQueryRequest

Field	Type	Label	Description
stream_id	uint64		Stream identifier, in Notification RegisterResponse

4.11.9 NotificationQueryResponse

Represents notification query response.

Table 91: NotificationQueryResponse

Field	Type	Label	Description
subscriptions	NotificationQuerySubscription	repeated	List of subscription details
status	SdkMgrStatus		Status of the query

4.11.10 NotificationQuerySubscription

Represents notification subscription.

Table 92: NotificationQuerySubscription

Field	Type	Label	Description
sub_id	uint64		Subscription identifier
description	string		Subscription description

4.11.11 NotificationRegisterRequest

Represents notification request from agent. Agent uses this message to subscribe to router events such as interface create, delete, or update, as well as LLDP neighbor create, delete, or update, and so on.

Table 93: NotificationRegisterRequest

Field	Type	Label	Description
stream_id	uint64		Unset on create, set otherwise
op	NotificationRegisterRequest.Operation		Specific operation in the notification register request
sub_id	uint64		Set for delete subscription, unset otherwise
intf	InterfaceSubscriptionRequest		Interface subscription request
nw_inst	NetworkInstanceSubscriptionRequest		Network instance subscription request
lldp_neighbor	LldpNeighborSubscriptionRequest		LLDP neighbor subscription request
config	ConfigSubscriptionRequest		Configuration subscription request
bfd_session	BfdSessionSubscriptionRequest		BFD session subscription request
route	IpRouteSubscriptionRequest		IP route subscription request
appid	AppIdentSubscriptionRequest		App identification subscription request
nhg	NextHopGroupSubscriptionRequest		Nexthop Group subscription request

4.11.12 NotificationRegisterResponse

Represents notification response.

Table 94: NotificationRegisterResponse

Field	Type	Label	Description
stream_id	uint64		Stream identifier. This needs to be passed to the SDK manager for further notification subscription changes specific to the current subscription
sub_id	uint64		Subscription identifier. Each subscription gets an identifier, which can be used to delete a subscription

Field	Type	Label	Description
status	SdkMgrStatus		Status of subscription

4.11.13 NotificationStreamRequest

Represents notification stream request.

Table 95: NotificationStreamRequest

Field	Type	Label	Description
stream_id	uint64		Stream identifier

4.11.14 NotificationStreamResponse

Represents notification stream response that contains one or more notification.

Table 96: NotificationStreamResponse

Field	Type	Label	Description
notification	Notification	repeated	Notification details

4.11.15 NotificationRegisterRequest.Operation

Represents notification stream subscription request operation.

Table 97: NotificationRegisterRequest.Operation

Name	Number	Description
Create	0	Create a subscription
Delete	1	Delete all subscriptions
AddSubscription	2	Add subscription to existing subscriptions
DeleteSubscription	3	Delete one subscription from existing subscriptions

4.11.16 SdkMgrService

Represents base service that defines agent registration, unregistration, notification subscriptions, and keepalive messages.

Table 98: SdkMgrService

Method Name	Request Type	Response Type	Description
AgentRegister	AgentRegistrationRequest	AgentRegistration Response	Register agent
AgentUnRegister	AgentRegistrationRequest	AgentRegistration Response	Unregister agent
NotificationRegister	NotificationRegisterRequest	Notification RegisterResponse	Register for event notifications
NotificationQuery	NotificationQueryRequest	NotificationQuery Response	Returns current or specific notification subscription details
KeepAlive	KeepAliveRequest	KeepAlive Response	Send periodic keepalive message
GetAppId	AppIdRequest	AppIdResponse	Get application name from application identifier

4.11.17 SdkNotificationService

Represents service for handling notifications.

Table 99: SdkNotificationService

Method Name	Request Type	Response Type	Description
NotificationStream	NotificationStreamRequest	NotificationStream Response stream	Send stream of event notifications based on the agent subscriptions

4.12 telemetry_service.proto

4.12.1 TelemetryData

Represents telemetry data.

Table 100: TelemetryData

Field	Type	Label	Description
json_content	string		Structured JSON telemetry data

4.12.2 TelemetryDeleteRequest

Represents telemetry delete request.

Table 101: TelemetryDeleteRequest

Field	Type	Label	Description
key	TelemetryKey	repeated	Telemetry key

4.12.3 TelemetryDeleteResponse

Represents telemetry delete response.

Table 102: TelemetryDeleteResponse

Field	Type	Label	Description
status	SdkMgrStatus		Status of delete request
error_str	string		Detailed error message

4.12.4 TelemetryInfo

Represents telemetry information.

Table 103: TelemetryInfo

Field	Type	Label	Description
key	TelemetryKey		Telemetry key
data	TelemetryData		Telemetry data

4.12.5 TelemetryKey

Represents telemetry key.

Table 104: TelemetryKey

Field	Type	Label	Description
js_path	string		JSON path referencing the key for telemetry data

4.12.6 TelemetryUpdateRequest

Represents telemetry update request.

Table 105: TelemetryUpdateRequest

Field	Type	Label	Description
state	TelemetryInfo	repeated	State of application

4.12.7 TelemetryUpdateResponse

Represents telemetry update response.

Table 106: TelemetryUpdateResponse

Field	Type	Label	Description
status	SdkMgrStatus		Status of telemetry update request
error_str	string		Detailed error message

4.12.8 SdkMgrTelemetryService

Represents service for telemetry service to store state data.

Table 107: SdkMgrTelemetryService

Method Name	Request Type	Response Type	Description
TelemetryAdd OrUpdate	TelemetryUpdateRequest	TelemetryUpdate Response	Add or update telemetry data
TelemetryDelete	TelemetryDeleteRequest	TelemetryDelete Response	Delete telemetry data

4.13 Scalar Value Types

4.13.1 Scalar

Scalar description

Table 108: Scalar

.proto Type	Notes	C++	Java	Python	Go	C#	PHP	Ruby
double		double	double	float	float64	double	float	Float
float		float	float	float	float32	float	float	Float
int32	Uses variable-length encoding. Inefficient for encoding negative numbers – if your field is likely to have negative values, use sint32 instead.	int32	int	int	int32	int	integer	Bignum or Fixnum (as required)
int64	Uses variable-length encoding. Inefficient for encoding negative numbers – if your field is likely to have negative values, use sint64 instead.	int64	long	int/long	int64	long	integer/ string	Bignum
uint32	Uses variable-length encoding.	uint32	int	int/long	uint32	uint	integer	Bignum or Fixnum (as required)
uint64	Uses variable-length encoding.	uint64	long	int/long	uint64	ulong	integer/ string	Bignum or Fixnum (as required)
sint32	Uses variable-length encoding. Signed int value. These more efficiently encode negative numbers than regular int32s.	int32	int	int	int32	int	integer	Bignum or Fixnum (as required)
sint64	Uses variable-length encoding. Signed int value. These more efficiently encode negative numbers than regular int64s.	int64	long	int/long	int64	long	integer/ string	Bignum

.proto Type	Notes	C++	Java	Python	Go	C#	PHP	Ruby
fixed32	Always four bytes. More efficient than uint32 if values are often greater than 2^28.	uint32	int	int	uint32	uint	integer	Bignum or Fixnum (as required)
fixed64	Always eight bytes. More efficient than uint64 if values are often greater than 2^56.	uint64	long	int/long	uint64	ulong	integer/ string	Bignum
sfixed32	Always four bytes.	int32	int	int	int32	int	integer	Bignum or Fixnum (as required)
sfixed64	Always eight bytes.	int64	long	int/long	int64	long	integer/ string	Bignum
bool		bool	boolean	boolean	bool	bool	boolean	TrueClass/False Class
string	A string must always contain UTF-8 encoded or 7-bit ASCII text.	string	String	str/ unicode	string	string	string	String (UTF-8)
bytes	May contain any arbitrary sequence of bytes.	string	Byte String	str	[]byte	Byte String	string	String (ASCII-8BIT)

Customer document and product support

Customer documentation

Customer documentation welcome page

Technical support

Product support portal

Documentation feedback

Customer documentation feedback