Nokia Service Router Linux 7215 Interconnect System 7220 Interconnect Router 7250 Interconnect Router 7730 Service Interconnect Router

Release 25.3

NetOps Development Kit Reference

3HE 21393 AAAA TQZZA
Edition: 01
March 2025

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 Software Release Notes* for information on features supported in each load.

Configuration and command outputs shown in this guide are examples only; actual displays may differ depending on supported functionality and user configuration.

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

Configuration of the NDK server administrative state (default: disabled).	Configure the NDK server administrative state
Under the bfd_service.proto, all instances of Bfdmgr are updated to Bfd and have the Pb suffix removed.	bfd_service.proto
Under the lldp_service.proto, the Pb suffix is now removed from the following messages: • LldpNeighborData • LldpNeighborKey	LldpNeighborData LldpNeighborKey
Under the nexthop_group_service.proto, in the NextHopGroup message, the next_hop field is updated to nexthops	NextHopGroup
Under the nexthop_group_service.proto, in the NextHopGroupDeleteRequest message, the group_key field is updated to group_keys	NextHopGroupDeleteRequest
Under the route_service.proto, the RoutePb message is updated to Route, and within the Route message, the following fields are updated: • nhg_id field is updated to nexthop_group_id • nexthop field is updated to nexthops	Route
Under the route_service.proto, the RouteKeyPb message is updated to RouteKey, and within the RouteKey message, the net_inst_name field is updated to network_instance_name	RouteKey
Under the sdk_common.proto, in the IpAddressPb message, the addr field is updated to ip_address	IpAddressPb
Under the sdk_common.proto, in the SdkMgr Operation and SdkMgrStatus messages, all fields are updated to be capitalized with underscores	SdkMgrOperation SdkMgrStatus
Under the sdk_service.proto, in the Agent RegistrationRequest messge, the js_path field is updated to js_paths.	AgentRegistrationRequest
Under the sdk_service.proto, in the Notification message, the following fields are updated: • sub_id field is updated to subscription_id	Notification

• intf field is updated to interface • nw_inst field is updated to network_instance • appid field is updated to app_id • nhg field is updated to nexthop_group	
Under the sdk_service.proto, in the Notification QuerySubscription message, the sub_id field is updated to subscription_id	NotificationQuerySubscription
Under the sdk_service.proto, in the Notification RegisterRequest message, the following fields are updated: • intf field is updated to interface • nw_inst field is updated to network_instance • nhg field is updated to nexthop_group • appid field is updated to app_id	NotificationRegisterRequest
Under the sdk_service.proto, in the Notification RegisterResponse message, the sub_id field is updated to subscription_id	NotificationRegisterResponse
Under the sdk_service.proto, in the Notification StreamResponse message, the notification field is updated to notifications	NotificationStreamResponse
Under the sdk_service.proto, in the Notification RegisterRequest.Operation message, all fields are updated to be capitalized with underscores	NotificationRegisterRequest.Operation
Under the telemetry_service.proto, in the Telemetry DeleteRequest message, the key field is updated to keys	TelemetryDeleteRequest

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.

3.4 Configure the NDK server administrative state

Procedure

By default, the NDK server is disabled on all SR Linux platforms. To enable the NDK server globally, use the **system ndk-server admin-state enable** command.

Example: Enable NDK server

```
--{ candidate shared default }--[  ]--
# info system ndk-server admin-state
    system {
        ndk-server {
            admin-state enable
        }
    }
```

4 Protocol Documentation

4.1 appid_service.proto

4.1.1 AppIdentData

Represents appid data.

Table 1: 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 2: AppIdentKey

Field	Type	Label	Description
id	uint32		Application id

4.1.3 AppIdentNotification

Represents appid notification.

Table 3: 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 4: AppIdentSubscriptionRequest

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

4.2 bfd_service.proto

4.2.1 BfdGeneralSessionData.BfdSessionSubType

Represents BFD session subtype.

Table 5: BfdGeneralSessionData.BfdSessionSubType

Name	Number	Description
BFD_SESSION_SUB_TYPE_UNKNOWN	0	Session subtype unknown
BFD_SESSION_SUB_TYPE_SINGLE_HOP	1	Single-hop session
BFD_SESSION_SUB_TYPE_MULTI_HOP	2	Multi-hop session
BFD_SESSION_SUB_TYPE_MICROBFD	3	microbfd session
BFD_SESSION_SUB_TYPE_SBFD_ECHO	4	microbfd session

4.2.2 BfdGeneralSessionKey.SbfdechoKey.SrpolicyUserType

Table 6: BfdGeneralSessionKey.SbfdechoKey.SrpolicyUserType

Name	Number	Description
SRPOLICY_USER_TYPE_ LOCAL	0	
SRPOLICY_USER_TYPE_ TYPE_PCEP	1	

4.2.3 BfdSessionStatus

Represents BFD session status.

Table 7: BfdSessionStatus

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

4.2.4 BfdSessionType

Represents BFD session type.

Table 8: BfdSessionType

Name	Number	Description
BFD_SESSION_TYPE_ UNKNOWN	0	Unknown session type
BFD_SESSION_TYPE_P2P	1	Peer-to-peer session type
BFD_SESSION_TYPE_ MICROBFD	2	microbfd session type
BFD_SESSION_TYPE_ SBFD_ECHO	3	seamless BFD session type, echo initiator

4.2.5 BfdGeneralSessionData

Represents BFD session data.

Table 9: BfdGeneralSessionData

Field	Type	Label	Description
status	BfdSessionStatus		Status of the session
subscription_type	BfdSessionSubType		Subtype of the session
source_interface_id	uint32		source_interface_id is only populated for P2P type Source interface ID

4.2.6 BfdGeneralSessionKey

Represents BFD session key.

Table 10: BfdGeneralSessionKey

Field	Type	Label	Description
type	BfdSessionType		type is always present, other key field presence is determined by type Session type
p2p	P2pKey		
microbfd	MicrobfdKey		
sbfdecho	SbfdechoKey		

4.2.7 BfdGeneralSessionKey.MicrobfdKey

Table 11: BfdGeneralSessionKey.MicrobfdKey

Field	Type	Label	Description
interface_name	string		

4.2.8 BfdGeneralSessionKey.P2pKey

Table 12: BfdGeneralSessionKey.P2pKey

Field	Type	Label	Description
source_ip_address	IpAddressPb		Source IP address of the session
destination_ip_address	IpAddressPb		Destination IP address of the session
instance_id	uint32		Network instance identifier
ipv4_unnumbered_or_ipv6_ll_interface_id	uint32		Global if id for ipv4 unnumbered or ipv6 link local
specified_discriminator	bool		session, otherwise 0 True if discriminators are specified

4.2.9 BfdGeneralSessionKey.SbfdechoKey

Table 13: BfdGeneralSessionKey.SbfdechoKey

Field	Type	Label	Description
instance_id	uint32		Network instance identifier
sr_policy_segment_list_id	uint32		
policy_name	string		
user_type	SrpolicyUserType		
sr_policy_endpoint	IpAddressPb		

4.2.10 BfdSessionNotification

Represents BFD session notification.

Table 14: BfdSessionNotification

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

4.2.11 BfdSessionSubscriptionRequest

Represents BFD session subscription request.

Table 15: BfdSessionSubscriptionRequest

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

4.3 config_service.proto

4.3.1 AcknowledgeConfigRequest

Represents config acknowledgment request; each config notification requires its own acknowledgement.

Table 16: AcknowledgeConfigRequest

Field	Type	Label	Description
infos	AcknowledgeConfigRequest Info	repeated	

4.3.2 AcknowledgeConfigRequestInfo

Represents configuration acknowledgment request information.

Table 17: AcknowledgeConfigRequestInfo

Field	Type	Label	Description
js_path_with_keys	string		JSON path formatted string from YANG; for example, interface{.name==ethernet1/ 1}.my_field
error	string		
warning	string		
output	string		

4.3.3 AcknowledgeConfigResponse

Represents config complete response.

Table 18: AcknowledgeConfigResponse

Field	Type	Label	Description
status	SdkMgrStatus		Status of commit complete request operation
error_str	string		Detailed error string

4.3.4 ConfigData

Represents configuration data.

Table 19: ConfigData

Field	Type	Label	Description
json	string		Entire configuration fragment as JSON string
data	bytes		Entire configuration fragment as binary data

4.3.5 ConfigKey

Represents configuration key.

Table 20: ConfigKey

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

4.3.6 ConfigNotification

Represents configuration notification message to subscribe to configuration events

Table 21: ConfigNotification

Field	Type	Label	Description
op	SdkMgrOperation		Operation indicating create, delete, or update
key	ConfigKey		Configuration key
data	ConfigData		Configuration data

4.3.7 ConfigSubscriptionRequest

Represents configuration subscription request.

Table 22: ConfigSubscriptionRequest

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

4.3.8 SdkMgrConfigService

Represents service for config operations.

Table 23: SdkMgrConfigService

Method Name	Request Type	Response Type	Description
AcknowledgeConfig	AcknowledgeConfigRequest	Acknowledge ConfigResponse	Acknowledge received configuration. When agent is registered with `wait_config_ack` flag set to true it needs to acknowledge received configuration.

4.4 interface_service.proto

4.4.1 InterfaceData

Represents interface data.

Table 24: InterfaceData

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

4.4.2 InterfaceKey

Represents interface key.

Table 25: InterfaceKey

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

4.4.3 InterfaceNotification

Represents interface notification.

Table 26: 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 27: InterfaceSubscriptionRequest

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

4.5 lldp_service.proto

4.5.1 LldpNeighborData.PortSubType

Represents port subtype.

Table 28: LldpNeighborData.PortSubType

Name	Number	Description
PORT_SUB_TYPE_RESERVED	0	Reserved for future use
PORT_SUB_TYPE_INTERFACE_ALIAS	1	Alias of the interface
PORT_SUB_TYPE_PORT_COMPONENT	2	Port identifier based on a locally defined port component
PORT_SUB_TYPE_MAC_ADDRESS	3	MAC address
PORT_SUB_TYPE_NETWORK_ADDRESS	4	Network address
PORT_SUB_TYPE_INTERFACE_NAME	5	Name of the interface
PORT_SUB_TYPE_AGENT_CIRCUIT_ID	6	Port identifier based on the circuit ID in the DHCP relay agent information option
PORT_SUB_TYPE_LOCALLY_ASSIGNED	7	Port identifier based on a locally defined alphanumeric string

4.5.2 LldpNeighborKey.ChassisIdType

Represents chassis type.

Table 29: LldpNeighborKey.ChassisIdType

Name	Number	Description
CHASSIS_ID_TYPE_RESERVED	0	Reserved for future use
CHASSIS_ID_TYPE_CHASSIS_COMPONENT	1	Chassis identifier based on a locally defined chassis component
CHASSIS_ID_TYPE_INTERFACE_ALIAS	2	Alias of the interface
CHASSIS_ID_TYPE_PORT_COMPONENT	3	Chassis identifier based on a locally defined port component
CHASSIS_ID_TYPE_MAC_ADDRESS	4	MAC address
CHASSIS_ID_TYPE_NETWORK_ADDRESS	5	Network address
CHASSIS_ID_TYPE_INTERFACE_NAME	6	Name of the interface
CHASSIS_ID_TYPE_LOCALLY_ASSIGNED	7	Chassis identifier based on a locally defined value

4.5.3 LldpNeighborData

Represents LLDP neighbor data.

Table 30: LldpNeighborData

Field	Type	Label	Description
port_id	string		Port identifier
port_type	PortSubType		Port type
source_mac	MacAddressPb		Port MAC address
bgp_peer_addresses	IpAddressPb	repeated	LLDP BGP autodiscovered addresses
bgp_group_id	uint32		BGP group identifier
system_name	string		System name

Field	Type	Label	Description
system_description	string		System description

4.5.4 LldpNeighborKey

Represents LLDP neighbor key.

Table 31: LldpNeighborKey

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

4.5.5 LldpNeighborNotification

Represents LLDP neighbor notification.

Table 32: LldpNeighborNotification

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

4.5.6 LldpNeighborSubscriptionRequest

Represents LLDP neighbor subscription request.

Table 33: LldpNeighborSubscriptionRequest

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

4.6 networkinstance_service.proto

4.6.1 NetworkInstanceData.NetworkInstanceType

Represents network instance type.

Table 34: NetworkInstanceData.NetworkInstanceType

Name	Number	Description
NETWORK_INSTANCE_ TYPE_DEFAULT	0	Default network instance type
NETWORK_INSTANCE_ TYPE_L3VRF	1	L3VRF network instance type

4.6.2 NetworkInstanceData

Represents network instance data.

Table 35: NetworkInstanceData

Field	Type	Label	Description
network_instance_id	uint32		Network instance identifier
base_name	string		Base name
oper_is_up	bool		Operation status
router_id	string		Router identifier
instance_type	NetworkInstanceType		Network instance type

4.6.3 NetworkInstanceKey

Represents network instance key.

Table 36: NetworkInstanceKey

Field	Type	Label	Description
instance_name	string		Network instance name

4.6.4 NetworkInstanceNotification

Represents network instance notification.

Table 37: NetworkInstanceNotification

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

4.6.5 NetworkInstanceSubscriptionRequest

Represents network instance subscription request.

4.7 nexthop_group_service.proto

4.7.1 NextHop.ResolutionType

Represents resolution type.

Table 38: NextHop.ResolutionType

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

4.7.2 NextHop.ResolveToType

Represents resolve-to type.

Table 39: NextHop.ResolveToType

Name	Number	Description
RESOLVE_TO_TYPE_LOCAL	0	Resolve to local routes

Name	Number	Description
RESOLVE_TO_TYPE_DIRECT	1	Resolve to direct routes
RESOLVE_TO_TYPE_INDIRECT	2	Resolve to indirect routes

4.7.3 MplsNextHop

Represents MPLS next hop.

Table 40: MplsNextHop

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

4.7.4 NextHop

Represents next-hop.

Table 41: NextHop

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

4.7.5 NextHopGroup

Represents next-hop group.

Table 42: NextHopGroup

Field	Type	Label	Description
nexthops	NextHop	repeated	Next-hops

4.7.6 NextHopGroupDeleteRequest

Represents next-hop group delete request.

Table 43: NextHopGroupDeleteRequest

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

4.7.7 NextHopGroupDeleteResponse

Represents next-hop group delete response.

Table 44: NextHopGroupDeleteResponse

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

4.7.8 NextHopGroupInfo

Represents next-hop group information.

Table 45: NextHopGroupInfo

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

4.7.9 NextHopGroupKey

Represents next-hop group key.

Table 46: NextHopGroupKey

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

4.7.10 NextHopGroupNotification

Represents next-hop group notification.

Table 47: 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

4.7.11 NextHopGroupRequest

Represents next-hop group request.

Table 48: NextHopGroupRequest

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

4.7.12 NextHopGroupResponse

Represents next-hop group response.

Table 49: NextHopGroupResponse

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

4.7.13 NextHopGroupSubscriptionRequest

Represents next-hop group subscription request.

Table 50: NextHopGroupSubscriptionRequest

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

4.7.14 SdkMgrNextHopGroupService

Represents service for next-hop group operations.

Table 51: SdkMgrNextHopGroupService

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

4.8 route_service.proto

4.8.1 IpRouteNotification

Represents IP route notification.

Table 52: IpRouteNotification

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

4.8.2 IpRouteSubscriptionRequest

Represents IP route subscription request.

Table 53: IpRouteSubscriptionRequest

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

4.8.3 Route

Represents route data.

Table 54: Route

Field	Type	Label	Description
nexthop_group_name	string		Next hop group name
preference	uint32		Preference
metric	uint32		Metric
nexthops	NextHop	repeated	List of next hops
owner_id	uint32		Next hop owner identifier returned only on notification.
nexthop_group_id	uint64		Next-hop group identifier returned only on notification.

4.8.4 RouteAddRequest

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

Table 55: RouteAddRequest

Field	Type	Label	Description
routes	RouteInfo	repeated	IP routes

4.8.5 RouteAddResponse

Represents route add response.

Table 56: RouteAddResponse

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

4.8.6 RouteDeleteRequest

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

Table 57: RouteDeleteRequest

Field	Type	Label	Description
routes	RouteKey	repeated	IP routes

4.8.7 RouteDeleteResponse

Represents route delete response.

Table 58: RouteDeleteResponse

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

4.8.8 RouteInfo

Represents route information.

Table 59: RouteInfo

Field	Type	Label	Description
key	RouteKey		Route key
data	Route		Route data

4.8.9 RouteKey

Represents route key.

Table 60: RouteKey

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

4.8.10 SdkMgrRouteService

Represents service for IP route operations.

Table 61: 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

4.9 sdk_common.proto

4.9.1 IfEthernetDuplexModeType

Represents interface ethernet duplex mode. Corresponds to yang values

Table 62: 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	

4.9.2 IfEthernetPortSpeedType

Represents interface ethernet port speed. Corresponds to yang values

Table 63: IfEthernetPortSpeedType

Name	Number	Description
IF_ETH_PORT_SPEED_UNSET	0	Speed unknown
IF_ETH_PORT_SPEED_10M	1	
IF_ETH_PORT_SPEED_100M	2	

Name	Number	Description
IF_ETH_PORT_SPEED_1G	3	
IF_ETH_PORT_SPEED_10G	4	
IF_ETH_PORT_SPEED_25G	5	
IF_ETH_PORT_SPEED_40G	6	
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	
IF_ETH_PORT_SPEED_800G	12	

4.9.3 IfLoopbackModeType

Represents interface loopback mode. Corresponds to yang values

Table 64: IfLoopbackModeType

Name	Number	Description
IF_LOOPBACK_MODE_UNSET	0	loopback mode not supported
IF_LOOPBACK_MODE_NONE	1	
IF_LOOPBACK_MODE_FACILITY	2	
IF_LOOPBACK_MODE_TERMINAL	3	

4.9.4 IfMgrIfType

Represents interface type.

Table 65: 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
SYNC	11	1588 sync interface
IF_TYPE_MAX	12	

4.9.5 IfOperDownReason

Table 66: 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	

Name	Number	Description
IF_OPER_DOWN_MDA_NOT_PRESENT	4	
IF_OPER_DOWN_TRANS_NOT_PRESENT	5	
IF_OPER_DOWN_PHY_INIT	6	
IF_OPER_DOWN_LOWER_LAYER_DOWN	7	
IF_OPER_DOWN_MTU_RESOURCES	8	
IF_OPER_DOWN_UNSUPPORTED_SPEED	9	
IF_OPER_DOWN_UNSUPPORTED_TRANS_FEC	10	
IF_OPER_DOWN_OTHER	11	
IF_OPER_DOWN_PORT_NOT_PRESENT	12	used internally by chassis mgr only - xdp never publish to IDB!
IF_OPER_DOWN_FABRIC_AVAILABILITY	13	used internally by chassis mgr only - xdp never publish to IDB!
IF_OPER_DOWN_NO_ACTIVE_LINKS	14	lag interface only
IF_OPER_DOWN_MIN_LINK_THRESHOLD	15	lag interface only
IF_OPER_DOWN_9_12_SPEED_MISMATCH	16	Vodka port 9-12 must all be same speed as port 9
IF_OPER_DOWN_LAG_RESOURCES	17	lag interface only
IF_OPER_DOWN_LAG_MEMBER_RESOURCES	18	lag member interface only
IF_OPER_DOWN_STANDBY_SIGNALING	19	ESM multihoming
IF_OPER_DOWN_HOLD_TIME_UP_ACTIVE	20	interface hold-time up is actively holding the interface down
IF_OPER_DOWN_RELOAD_TIME_ACTIVE	21	interface reload time is actively holding the interface down

Name	Number	Description
IF_OPER_DOWN_CONNECTOR_DOWN	22	parent connector oper down forces breakout port oper down
IF_OPER_DOWN_AUTO_NEG_MISMATCH	23	
IF_OPER_DOWN_EVENT_HANDLER	24	used internally by chassis mgr only - xdp never publish to IDB!
IF_OPER_DOWN_UNSUPPORTED_BREAKOUT	25	interface doesn't support breakout config
IF_OPER_DOWN_CFM_CCM_DEFECT	26	
IF_OPER_DOWN_CRC_MON_FAIL_THRESH	27	crc-monitor signal failure threshold exceeded
IF_OPER_DOWN_SYMBOL_MON_FAIL_THRESH	28	symbol-monitor signal failure threshold exceeded
IF_OPER_DOWN_LINK_LOSS_FORWARDING	29	related to evpn-vpws mpls
IF_OPER_DOWN_TRANS_DOWN	30	
IF_OPER_DOWN_STORM_CONTROL_ACTION	31	
IF_OPER_DOWN_UNSUPPORTED_NUM_CHANNELS_FOR_SPEED	32	

4.9.6 IfOperStateType

Represents interface operational state.

Table 67: 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

Name	Number	Description
IF_OPER_STATE_ UNKNOWN	3	Interface operational state unknown
IF_OPER_STATE_DORMANT	4	Interface operational state dormant
IF_OPER_STATE_NOT_ PRESENT	5	Interface operational state not present
IF_OPER_STATE_LOWER_ LAYER_DOWN	6	Interface operational state lower layer down

4.9.7 IfTransceiverFecType

Represents interface transceiver fec. Corresponds to yang values

Table 68: IfTransceiverFecType

Name	Number	Description
IF_TRANS_FEC_UNSET	0	Fec unknown
IF_TRANS_FEC_DISABLED	1	
IF_TRANS_FEC_CL91_ RS528	2	
IF_TRANS_FEC_RS544	3	Deprecated as a state value, but still used by devmgr/txr code to represent any rs544 flavor
IF_TRANS_FEC_BASER	4	
IF_TRANS_FEC_CL108_ RS528	5	
IF_TRANS_FEC_CL91_ RS544	6	
IF_TRANS_FEC_CL119_ RS544	7	
IF_TRANS_FEC_CL134_ RS544	8	

4.9.8 IpAddressState

Represents IP address state.

Table 69: 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
IPADDR_STATE_ PREFERRED	5	IP address state preferred

4.9.9 PortOperDownReason

Various reasons for port being operationally down.

Table 70: PortOperDownReason

Name	Number	Description
PORT_OPER_DOWN_NONE	0	
PORT_OPER_DOWN_ PORT_ADMIN_DISABLED	1	Admin disabled (out of service). E.g lanes not created in physical layer devices
PORT_OPER_DOWN_ PORT_OPER_DISABLED	2	Admin enabled but operationally disabled. E.g lanes created but serdes are off
PORT_OPER_DOWN_MDA_ DISABLED	3	Parent MDA is disabled
PORT_OPER_DOWN_MDA_ FAILED	4	Parent MDA failed
PORT_OPER_DOWN_LINK_ DOWN	5	Operationally enabled but link/carrier down
PORT_OPER_DOWN_ TRANS_NOT_PRESENT	6	Transceiver is not plugged in or is dead (undetected)
PORT_OPER_DOWN_ TRANS_LASER_DISABLED	7	Transceiver laser is disabled

Name	Number	Description
PORT_OPER_DOWN_INCOMPATIBLE_CONNECTOR	8	Parent connector configuration does not match with this breakout port
PORT_OPER_DOWN_PHY_UNSUPPORTED_FEC_CFG	9	A physical layer device on the port does not support configured FEC
PORT_OPER_DOWN_PHY_UNSUPPORTED_TRANS_FEC	10	A physical layer device on the port does not support FEC configured on the transceiver
PORT_OPER_DOWN_PHY_INTERNAL_FEC_ISSUE	11	A physical layer device has internal/hardware problem on employing FEC
PORT_OPER_DOWN_UNSUPPORTED_BREAKOUT	12	Connector does not support the configured breakout mode.

4.9.10 PortWarningReason

Various warnings on port while it stays operational.

Table 71: PortWarningReason

Name	Number	Description
PORT_WARNING_NONE	0	
PORT_WARNING_TRANS_RX_LOS	1	Transceiver reported RxLos
PORT_WARNING_TRANS_UNSUPPORTED_SYNCE_CFG	2	Transceiver does not support synce configuration (e.g cannot do clock squelching on RxLos)
PORT_WARNING_PHY_UNSUPPORTED_SYNCE_CFG	3	A physical layer device does not support synce configuration
PORT_WARNING_PHY_UNSUPPORTED_PTP_CFG	4	A physical layer device does not support PTP configuration

4.9.11 SdkMgrOperation

Represents enumeration value for operation in subscription.

Table 72: SdkMgrOperation

Name	Number	Description
SDK_MGR_OPERATION_ CREATE	0	Create operation; returned if caching is enabled
SDK_MGR_OPERATION_ UPDATE	1	Update operation; returned if caching is enabled
SDK_MGR_OPERATION_ DELETE	2	Delete operation
SDK_MGR_OPERATION_ CREATE_OR_UPDATE	3	returned if caching is disabled; App can cache streaming data, if needed.

4.9.12 SdkMgrStatus

Represents status of network programming service calls.

Table 73: SdkMgrStatus

Name	Number	Description
SDK_MGR_STATUS_ SUCCESS	0	Successful service call
SDK_MGR_STATUS_FAILED	1	Failed service call

4.9.13 AgentReply

Empty message from agent.

4.9.14 EvpnEthSegIdPb

Table 74: EvpnEthSegIdPb

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

4.9.15 GlobalIfId

Represents global interface identifier.

Table 75: GlobalIfId

Field	Type	Label	Description
global_if_id	uint32		Global interface identifier

4.9.16 IpAddrPrefLenPb

Represents IP prefix.

Table 76: IpAddrPrefLenPb

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

4.9.17 IpAddressPb

Represents IP address.

Table 77: IpAddressPb

Field	Type	Label	Description
ip_address	bytes		IP address

4.9.18 IpInterfaceAddrPrefixPb

Represents IP prefix state.

Table 78: IpInterfaceAddrPrefixPb

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

4.9.19 MacAddressPb

Represents MAC address.

Table 79: MacAddressPb

Field	Type	Label	Description
mac_address	bytes		MAC address

4.9.20 MplsLabel

Represents MPLS label.

Table 80: MplsLabel

Field	Type	Label	Description
mpls_label	uint32		MPLS label

4.9.21 NetInstanceId

Represents network instance identifier.

Table 81: NetInstanceId

Field	Type	Label	Description
instance_id	uint32		Network instance identifier

4.9.22 PortIdPb

Represents port identifier.

Table 82: PortIdPb

Field	Type	Label	Description
port_id	uint64		Port identifier

4.9.23 SyncRequest

Empty message for synchronization request.

4.9.24 SyncResponse

Empty message for synchronization end.

Table 83: SyncResponse

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

4.10 sdk_service.proto

4.10.1 NotificationRegisterRequest.Operation

Represents notification stream subscription request operation.

Table 84: NotificationRegisterRequest.Operation

Name	Number	Description
OPERATION_CREATE	0	Create a subscription
OPERATION_DELETE	1	Delete all subscriptions
OPERATION_ADD_ SUBSCRIPTION	2	Add subscription to existing subscriptions
OPERATION_DELETE_ SUBSCRIPTION	3	Delete one subscription from existing subscriptions

4.10.2 AgentRegistrationRequest

Represents registration request message used in agent register and unregister.

Table 85: AgentRegistrationRequest

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

Field	Type	Label	Description
			Value of 0 means do not monitor this agent for liveliness.
wait_config_ack	bool		Indicate if SRLinux should wait for explicit ack from app after delivering configuration.
enable_cache	bool		Indicate if SRLinux should cache streaming notification response in ndk mgr. By default, caching is disabled.
auto_telemetry_state	bool		Indicate if SRLinux should automatically push configs as state. By default, auto_telemetry_state is disabled.

4.10.3 AgentRegistrationResponse

Represents registration response in reply to registration request.

Table 86: 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.

4.10.4 AppIdRequest

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

Table 87: AppIdRequest

Field	Type	Label	Description
name	string		Application name

4.10.5 AppIdResponse

Represents application identifier response to agent.

Table 88: AppIdResponse

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

4.10.6 KeepAliveRequest

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

4.10.7 KeepAliveResponse

Represents keepalive response.

Table 89: KeepAliveResponse

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

4.10.8 Notification

Represents notification stream response.

Table 90: Notification

Field	Type	Label	Description
subscription_id	uint64		Subscription identifier
interface	InterfaceNotification		Interface details
network_instance	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
app_id	AppIdentNotification		App identification details

Field	Type	Label	Description
nexthop_group	NextHopGroupNotification		Next-hop group details

4.10.9 NotificationQueryRequest

Represents notification query to return specific subscription details.

Table 91: NotificationQueryRequest

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

4.10.10 NotificationQueryResponse

Represents notification query response.

Table 92: NotificationQueryResponse

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

4.10.11 NotificationQuerySubscription

Represents notification subscription.

Table 93: NotificationQuerySubscription

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

4.10.12 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 94: NotificationRegisterRequest

Field	Type	Label	Description
stream_id	uint64		Unset on create, set otherwise
op	Operation		Specific operation in the notification register request
subscription_id	uint64		Set for delete subscription, unset otherwise
interface	InterfaceSubscriptionRequest		Interface subscription request
network_instance	NetworkInstanceSubscription Request		Network instance subscription request
lldp_neighbor	LldpNeighborSubscription Request		LLDP neighbor subscription request
config	ConfigSubscriptionRequest		Configuration subscription request
bfd_session	BfdSessionSubscription Request		BFD session subscription request
route	IpRouteSubscriptionRequest		IP route subscription request
app_id	AppIdentSubscriptionRequest		App identification subscription request
nexthop_group	NextHopGroupSubscription Request		Nexthop Group subscription request

4.10.13 NotificationRegisterResponse

Represents notification response.

Table 95: 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
subscription_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

4.10.14 NotificationStreamRequest

Represents notification stream request.

Table 96: NotificationStreamRequest

Field	Type	Label	Description
stream_id	uint64		Stream identifier

4.10.15 NotificationStreamResponse

Represents notification stream response that contains one or more notification.

Table 97: NotificationStreamResponse

Field	Type	Label	Description
notifications	Notification	repeated	Notification details

4.10.16 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

Method Name	Request Type	Response Type	Description
GetAppId	AppIdRequest	AppIdResponse	Get application name from application identifier

4.10.17 SdkNotificationService

Represents service for handling notifications.

Table 99: SdkNotificationService

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

4.11 telemetry_service.proto

4.11.1 TelemetryData

Represents telemetry data.

Table 100: TelemetryData

Field	Type	Label	Description
json_content	string		Structured JSON telemetry data

4.11.2 TelemetryDeleteRequest

Represents telemetry delete request.

Table 101: TelemetryDeleteRequest

Field	Type	Label	Description
keys	TelemetryKey	repeated	Telemetry key

4.11.3 TelemetryDeleteResponse

Represents telemetry delete response.

Table 102: TelemetryDeleteResponse

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

4.11.4 TelemetryInfo

Represents telemetry information.

Table 103: TelemetryInfo

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

4.11.5 TelemetryKey

Represents telemetry key.

Table 104: TelemetryKey

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

4.11.6 TelemetryUpdateRequest

Represents telemetry update request.

Table 105: TelemetryUpdateRequest

Field	Type	Label	Description
states	TelemetryInfo	repeated	State of application

4.11.7 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

4.11.8 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

4.12 Scalar Value Types

4.12.1 Scalar

Scalar description

Table 108: Scalar

.proto Type	Notes	C++	C#	Go	Java	PHP	Python	Ruby
double		double	double	float64	double	float	float	Float
float		float	float	float32	float	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	int32	int	integer	int	Bignum or Fixnum (as required)

.proto Type	Notes	C++	C#	Go	Java	PHP	Python	Ruby
int64	Uses variable-length encoding. Inefficient for encoding negative numbers – if your field is likely to have negative values, use sint64 instead.	int64	long	int64	long	integer/ string	int/long	Bignum
uint32	Uses variable-length encoding.	uint32	uint	uint32	int	integer	int/long	Bignum or Fixnum (as required)
uint64	Uses variable-length encoding.	uint64	ulong	uint64	long	integer/ string	int/long	Bignum or Fixnum (as required)
sint32	Uses variable-length encoding. Signed int value. These more efficiently encode negative numbers than regular int32s.	int32	int	int32	int	integer	int	Bignum or Fixnum (as required)
sint64	Uses variable-length encoding. Signed int value. These more efficiently encode negative numbers than regular int64s.	int64	long	int64	long	integer/ string	int/long	Bignum
fixed32	Always four bytes. More efficient than uint32 if values are often greater than 2^28.	uint32	uint	uint32	int	integer	int	Bignum or Fixnum (as required)
fixed64	Always eight bytes. More efficient than uint64 if values are often greater than 2^56.	uint64	ulong	uint64	long	integer/ string	int/long	Bignum
sfixed32	Always four bytes.	int32	int	int32	int	integer	int	Bignum or Fixnum (as required)

.proto Type	Notes	C++	C#	Go	Java	PHP	Python	Ruby
sfixed64	Always eight bytes.	int64	long	int64	long	integer/ string	int/long	Bignum
bool		bool	bool	bool	boolean	boolean	boolean	TrueClass/False Class
string	A string must always contain UTF-8 encoded or 7-bit ASCII text.	string	string	string	String	string	str/ unicode	String (UTF-8)
bytes	May contain any arbitrary sequence of bytes.	string	Byte String	[]byte	Byte String	string	str	String (ASCII-8BIT)

Customer document and product support

Customer documentation

Customer documentation welcome page

Technical support

Product support portal

Documentation feedback

Customer documentation feedback