

Nokia Service Router Linux

Release 24

Nokia Service Router Linux (SR Linux) is an open, extensible and resilient network operating system (NOS) designed to enable superior scalability, flexibility and efficiency for enterprise, service provider and webscale IP and data center networks.

Overview

Cloud has become a driver of change. Digital infrastructures are constantly transforming due to content located closer to end users, applications evolving to cloud-native approaches, the adoption of cloud-edge applications and the need to be ready for technology inflections. IP and data center networks need to keep pace and transform to become more “cloud like” with more agility, customizability, scalability and performance than previous generations of network infrastructures. Nokia SR Linux was designed to solve challenges in modern networks, where the primary challenges are lack of scalability, inflexibility and the need for operational simplification.

Nokia SR Linux is a truly open and modular NOS. It implements a ground-up, model-driven architecture combined with field-proven routing protocol stacks to create a unique foundation that delivers superior openness, extensibility and resiliency.

This innovative design foundation supports superior telemetry, unrivalled flexibility for implementing and customizing network tools and third-party applications, and plug-and-play integration—all critical features for modern networking.

Architecture building blocks and benefits

SR Linux was developed in close collaboration with some of the world’s largest network operators. Its design choices were motivated by the need to provide high levels of openness, programmability and flexibility to meet growing DevOps and agility requirements. Nokia SR Linux implements an architecture that delivers an open, extensible framework designed from the ground up for automation with advanced software features leveraging proven quality and resiliency.

Cloud-native design principles

Cloud-native application innovation and related technologies are making it easier to achieve the goals of business agility, continuity and innovation, while empowering staff and breaking down traditional silos.

Much like cloud-native applications, network and cloud infrastructures need to become much more dynamic and require more scalability and performance than previous generations of network infrastructure.

SR Linux embraces a cloud-native design approach that helps deliver superior programmability, unrivalled flexibility, and resilient IP routing capabilities to make network operations more agile and adaptable. Cloud-native innovations include the support for a Linux-based kernel, ground-up modular model-driven framework and a microservices-based, state-efficient design.

Linux-based NOS

Openness is best achieved with an underlying open operating system such as Linux®. Nokia SR Linux is a Linux-based NOS that builds on the Linux kernel to provide a set of loosely coupled services that come together to provide the functional blocks of an NOS, along with simple interfaces for their consumption.

SR Linux uses an unmodified Linux kernel as the foundation on which to build a suite of network applications (Nokia applications as well as customer applications) that are modular and isolated into their own failure domains. This provides many benefits, including reliability, portability and ease of application development. Using an unmodified kernel speeds the delivery of security patches

to critical system components, and leveraging a common Linux distribution provides access to equally field-proven management and infrastructure applications.

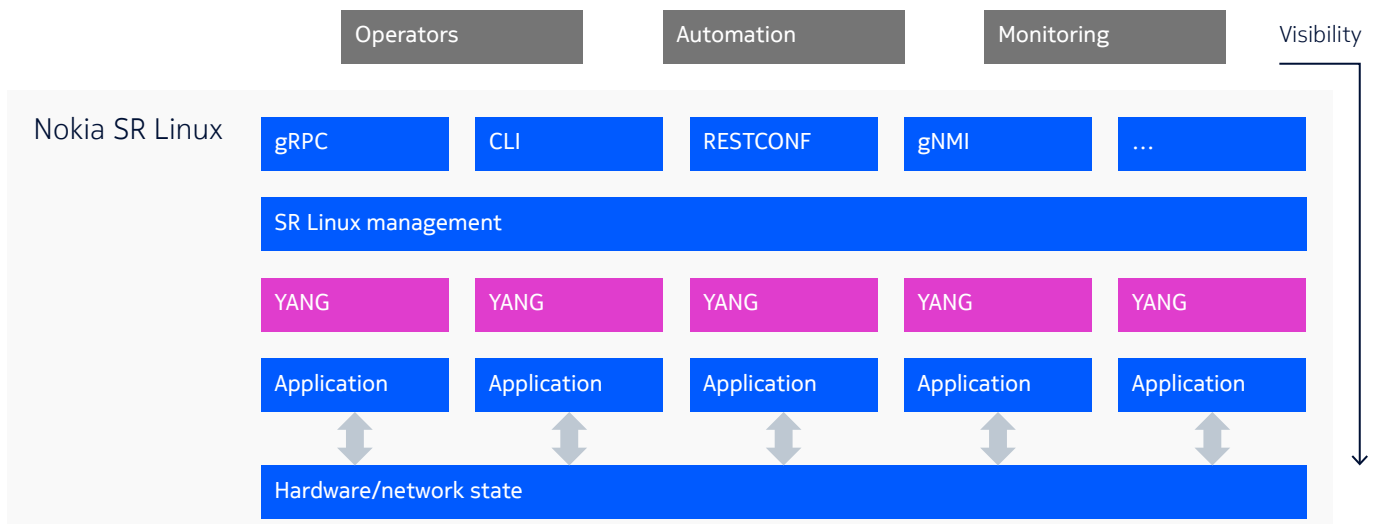
Ground-up, modular model-driven design

Openness cannot be an afterthought and must be designed from the foundation.

SR Linux implements a truly open architecture built around modular model-driven management and modern interfaces for its consumption (see Figure 1).

Our core design approach opens up the infrastructure by adopting emerging technologies to allow services to expose their functionality in the form of generalized Remote Procedure Call (gRPC) services and protocol buffers (protobufs). Our fully modular design, where each network application has its own YANG data structure, delivers complete openness. It gives network applications the ability to simplify their internal schemas based on data modelling languages, and then expose these schemas directly for consumption by northbound interfaces: any object, any interface.

Figure 1. Nokia SR Linux ground-up, model-driven design for true openness



State-sharing architecture

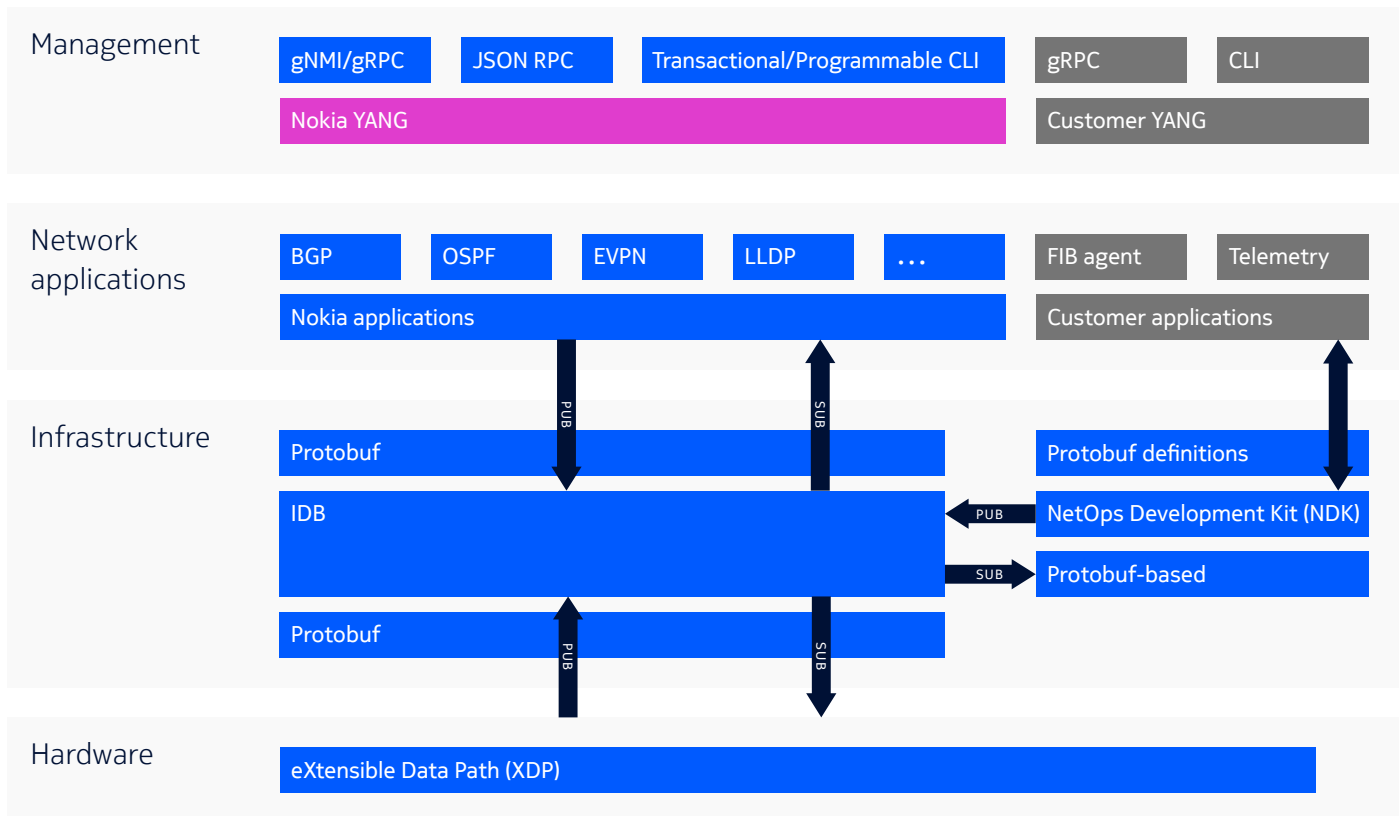
Nokia SR Linux applications share state with each other via a publish/subscribe (pub/sub) architecture (see Figure 2). The Nokia pub/sub architecture is implemented using protobufs, gRPC and the Nokia Impart Database (IDB).

gRPC provides an efficient, secure communication channel for applications and allows native extensions through third-party applications.

IDB is a lightweight database that is optimized for handling high volumes of messages while protecting against any one application slowing down the whole system. IDB provides a reliable and scalable delivery mechanism, guaranteeing delivery of updates to subscribers.

SR Linux supports a variety of networking chipsets through the Nokia eXtensible Data Path (XDP). The Nokia XDP serves as a hardware abstraction layer that speeds time-to-market for multiple networking chipsets.

Figure 2. SR Linux modular, state-sharing and extensible architecture



Robust foundation

A robust NOS is characterized by operating system software stability and its ability to support field-proven, scalable and reliable software feature sets.

SR Linux was designed using field-proven IP protocol stacks and network functions from our Service Router Operating System (SR OS). The SR OS has a strong pedigree, with over 1.7 million

routers deployed in more than 2,500 IP networks, including the internet backbone and some of the largest enterprise, service provider, and webscale networks worldwide.

By using these field-proven protocol stacks, network design and operations teams can immediately benefit from a feature-rich, hardened, interoperable and secure NOS.



Resiliency and high availability

Hardware and software high availability is crucial for ensuring maximum system uptime in IP and data center networks. SR Linux-enabled hardware platforms support redundant fan and power configurations as well as hot-swappable, redundant control and fabric modules in modular chassis-based hardware platforms.

SR Linux microservices-based, state-efficient design paves the path for enabling per-application hitless upgrades and always on networking.

SR Linux supports warm reboot features that can be used to perform a soft reset or trigger an in-service software upgrade (ISSU). Nonstop forwarding (NSF) capabilities minimize data plane outages. NSF capabilities also enable hardware platforms to continue to forward packets. While leveraging control plane graceful restart, peers can continue to pass traffic.

Best-in-class streaming telemetry

SR Linux is designed to meet the demands of a model-driven world where visibility—and the scalability and granularity of that visibility—are paramount. SR Linux delivers an open, extensible and performant infrastructure that allows the retrieval of fine-grained system state, setting of configuration, and a scalable interface to support more granular data with push-based streaming.

SR Linux was built with an open, scalable telemetry framework at its core, internally using Google-defined Remote Procedure Calls (gRPC), gRPC Network Management Interface (gNMI) and protobufs. Because SR Linux is natively model-driven, it is immediately ready for streaming telemetry without requiring any translation layers.

NetOps Development Kit (NDK)

Nokia SR Linux allows third-party network applications to be fully integrated into the system with the same functionality as Nokia applications. This includes consistent configuration using YANG, telemetry support, life-cycle management and visibility of system resources.

The Nokia NDK enables networking teams to leverage SR Linux's underlying model-driven architecture, with a simple, clean, decoupled integration. This allows all applications in the system to support data modeling, transactional configuration and—most important—massively scalable streaming telemetry.

The NDK uses gRPC and protobufs to provide maximum flexibility for languages supported and backwards compatibility. This approach differs from others, which are restricted to certain languages, versions and/or libraries.

Unmatched extensibility

SR Linux enables network operations teams to leverage SR Linux's underlying open architecture to deliver unprecedented customizability.

Superior CLI programmability

Every SR Linux application (including third-party network applications) supports its own YANG model, which can be loaded into the system. With this design, the YANG data model is defined first, and from it, operators can derive the command line interface (CLI), the application programming interfaces (APIs), and the show-output formats related to software capabilities.

In addition to the gNMI interface, SR Linux includes an advanced CLI based on the Python programming language and a JSON-RPC API for management. The CLI provides a flexible framework for accessing the system's underlying data models and is based on quality of life (features that improve the operator experience) embraced by DevOps communities.

Operators can leverage CLI plugins to completely customize the way the CLI operates, plugging in Linux commands or pulling the state/configuration from various locations and combining these with system state/configuration to allow advanced logic. This capability streamlines the adoption of SR Linux because the interface can be customized.

Build custom apps with the SR Linux NDK

Nokia SR Linux enables network teams to create high-performance applications that run alongside native apps on SR Linux. These “on-box custom applications” can be deeply integrated with the rest of the SR Linux system and thus can perform tasks that are not possible with traditional management interfaces standard for the typical network operating systems.

The NDK allows network operators to teach the network their language and respond directly to business demands—all without worrying about the scalability or functionality of the routing stack or the underlying infrastructure.

The NDK provides the ability to be ready for technology inflections; for example, it's easy for operational teams to learn and experience how artificial intelligence/machine language (AI/ML) tools like ChatGPT can easily be integrated with their IP routers. With SR Linux, this is as simple as building an NDK application for that. To learn more about the SR Linux NDK, refer to <https://learn.srlinux.dev/ndk/>.

Support for open community ecosystems

- **Public SR Linux container image:** It's a well-established fact that learning by doing yields the best results. With that in mind, we offer the SR Linux container image at “no cost” and available without any registration or licensing requirements. The SR Linux image is available to everyone at a [publicly accessible GitHub container registry](#).
- **Containerlab:** [Containerlab](#) is a tool designed to deploy networking lab topologies. The public SR Linux container image, when powered by containerlab, allows network operations teams to deploy and configure their lab with network protocols and services they require and test the control and data plane functions. Containerlab supports native containerized NOS's as well as traditional virtual machine-based routers.

- **Integration with open ecosystems and tools:**

SR Linux supports integration with DevOps and automation tools like Ansible, NAPALM, NetBox, gNMIc and more. To learn more about SR Linux programmability and integration with open ecosystems and tools, refer to <https://learn.srlinux.dev/programmability/>

- **Learn SR Linux portal:** SR Linux packs many unique features that networking teams can leverage, several of which are truly new to the networking domain. Our [SR Linux portal](#) features tutorials, documentation, blogs, programmability and developer tools (YANG, NDK, event handler, plugins and more).

Automation at scale

SR Linux was designed and built to enable confident automation and operations at scale. Its ground-up, model-driven and modular architecture with independent data modeling protocols like YANG and OpenConfig and open scalable foundation are well suited for telemetry requirements in modern networks. Network management and automation platforms can immediately benefit from the granular and comprehensive multidimensional telemetry data from SR Linux to monitor and gain deep insights into network traffic. This helps in understanding the current state of the network and comparing with desired intent to identify and implement the required changes to the network.

The [Nokia Fabric Services System](#) complements and extends the capabilities provided by our SR Linux architecture foundation to enable a full turnkey data center solution for automation, intent-based approaches, telemetry collection and analytics. The Fabric Services System provides a modern, flexible toolkit that delivers automation at scale for all phases of data center fabric operations, including Day 0 design, Day 1 deployment and Day 2+ configuration, operation, measurement, and analysis of a data center fabric.

Supported hardware platforms

Nokia Data Center Fabric solution

The Nokia portfolio of data center platforms addresses the needs of modern data centers. The portfolio offers a broad range of high-performance platforms for data center leaf-spine deployments. Both modular, chassis-based platforms and fixed-form-factor platforms are available. The portfolio includes,

- Nokia 7250 IXR-6e/10e, 7250 IXR-6/10 Interconnect Routers for data center fabrics
- Nokia 7220 IXR-H series Interconnect Routers for data center fabrics
- Nokia 7220 IXR-D series Interconnect Routers for data center fabrics

These hardware platforms implement SR Linux and support data center leaf, spine, super spine, management top of rack (ToR) features as well as data center interconnection and WAN use cases. For additional information on the Nokia Data Center Fabric solution, please refer to the [web page](#).

Software features

SR Linux supports, but is not limited to, the following software features. Certain features may be offered only on specific platforms. For platform-specific feature details and exceptions, refer to the hardware platform data sheets.

Open Linux support

- Support for unmodified Linux kernel
- Access to Linux tools, patching and packaging
- SR Linux container
- Linux control groups (cgroupsv2)

Platform features

- Dynamic Ternary Content Addressable Memory (TCAM) table allocation Layer 2 features

Layer 2 features

- Dot1q and untagged sub-interfaces
- Ethernet IEEE 802.1Q (VLAN) with support for jumbo frames
- Link aggregation: IEEE 802.3 ad Link Aggregation Group (LAG) and Link Aggregation Control Protocol (LACP)
- Link Layer Discovery Protocol (LLDP) on all interfaces
- Media access control (MAC) loop prevention
- MAC storm control
- Virtual routing and forwarding (VRF): MAC-VRF
- MAC access control lists (ACLs) with validation: accept, reject and log actions
- Multicast Listener Discovery (MLD) snooping in Layer 2 broadcast domains

Layer 3 features

- IPv4/v6 routing
- BGP with iBGP/eBGP: Support for IPv4/v6, including:
 - Core prefix independent convergence (PIC)
 - 4-byte autonomous system number
 - Route reflector
 - Dynamic BGP
 - BGP unnumbered
 - eBGP multi-hop
 - Add-paths for IPv4 and IPv6 routes
- Multiprotocol BGP (MP-BGP)
- Multi-topology, multi-instance IS-IS v4/v6
- Graceful restart client for IS-IS
- Multi-instance, multi-area OSPFv2 and OSPFv3
- Static routes for IPv4/v6
- Equal Cost Multi-Path (ECMP) with consistent and resilient hashing and configurable hash fields
- IPv6 flow label hashing
- VRF: Multiple VRF support
- Maintenance modes

- Bidirectional forwarding detection (BFD), micro BFD (mBFD)
- Interfaces: Loopback interfaces, Integrated Routing and Bridging (IRB)
- Proxy Address Resolution Protocol (ARP)/neighbor discover (ND)
- Routing policy:
 - Structured rules for accepting, rejecting and modifying routes that are learned and advertised to routing peers
 - Policy-based forwarding based on DiffServ Code Point (DSCP) and/or IP protocol
 - Routes can be matched based on prefix lists, autonomous system (AS) path regular expressions, BGP communities, Address Family Indicator/Subsequent Address Family Indicator (AFI/SAFI) protocol, etc.
 - Route leaking between network instances
- Layer 3/Layer 4 ACLs with validation; accept, reject and log actions

MPLS and segment routing (SR)

- Interface Link Distribution Protocol (LDP) over IPv4
- SR-ISIS over IPv4/v6
- BGP shortcuts over LDP
- BGP shortcuts over SR-ISIS
- MPLS QoS via EXP to forwarding class mapping
- MPLS ACL filters
- Internet Control Message Protocol (ICMP) tunnelling
- ICMP extensions for MPLS
- LSP ping and trace for LDP and SR-ISIS tunnels

Network virtualization

- EVPN with VXLANv4 encapsulation
- EVPN Layer 2 and Layer 3 connectivity
- EVPN all-active multi-homing; single active multi-homing for Layer 2 and Layer 3
- EVPN host route mobility

- Provider edge customer edge (PE-CE) BGP path attribute propagation in EVPN
- EVPN IP aliasing

QoS and traffic management

- Intelligent packet classification, including IPv4, IPv6 match-criteria-based classification
- Ingress per forwarding class sub-interface policing
- Queuing/scheduling:
 - Strict priority
 - Weighted Round Robin (WRR)
 - Weighted Random Early Detection (WRED)
 - Explicit Congestion Notification (ECN)
- QoS classification and marking based on DSCP
- Ingress DSCP rewrite
- QoS classification and marking based on IEEE 802.1p
- Multi-field classification

System management and automation

- Native model-driven architecture, configuration candidates, exclusive mode, checkpoints, rollbacks:
 - Support for SR Linux and OpenConfig data models
- Model-driven management interfaces: gNMI, gRPC Routing Information Base Interface (gRIBI), JSON-RPC and CLI (transactional, Python CLI and CLI plugins)
- gRPC network operations interface (gNOI)
- gRPC Network Security Interface (gNSI)
- P4 runtime packet extraction and injection
- Per-user configurable options for CLI
- Local Authentication, Authorization and Accounting (AAA) with Role Based Access Control (RBAC)
- Terminal Access Controller Access Control System (TACACS+) AAA via privilege levels
- Remote Authentication Dial-In User Service (RADIUS) support for AAA



- Password complexity policies and lockout management
- Access to common Linux utilities: Bash, cron and Python
- Syslog RFC 5424
- Telemetry:
 - Subscription-based telemetry for modeled data structures, either on change or sampled
 - sFlow
 - Logging infrastructure
- Telemetry-driven event management
- Python-based Zero Touch Provisioning (ZTP)
- Address management: Dynamic Host Configuration Protocol (DHCP) v4/v6 relay
- DHCP v4/v6 server with static allocations
- Interactive mirroring
- Unified Forwarding Tables (UFT) profiles
- NetOps Development Kit (NDK)
 - gRPC and protobuf-based interface for tight integration
 - Leverages SR Linux model-driven architecture
 - Direct access to other application functionality, e.g., forwarding information base (FIB), Link LLDP and BFD
 - Native support for streaming telemetry

Resiliency

- Support for redundant fan and power configurations in hardware platforms
- Support for hot-swappable, redundant control and fabric modules
- Application warm restart to perform soft reset or triggered in-service software upgrade (ISSU):
 - Nonstop forwarding (NSF)
 - Graceful restart

Security

- Distributed and aggregated ACLs and policers for control and management plane
- Layer 2 through Layer 4 Control Plane Policing (CoPP)
- Mirroring from interface/sub-interface or ingress ACL
- Mirroring to Switch Port Analyzer (SPAN) and Encapsulated Remote SPAN (ERSPAN)
- IPv6 Router Advertisements (RA) guard
- MAC security (MACsec)

Timing and synchronization

- ITU-T Synchronous Ethernet (SyncE)
- IEEE 1588v2:
 - Boundary clock (BC)
 - Profiles: ITU-T G.8275.1
 - Ethernet encapsulation
- RFC 5905 Network Time Protocol (NTP)

About Nokia

At Nokia, we create technology that helps the world act together.

As a B2B technology innovation leader, we are pioneering networks that sense, think and act by leveraging our work across mobile, fixed and cloud networks. In addition, we create value with intellectual property and long-term research, led by the award-winning Nokia Bell Labs.

Service providers, enterprises and partners worldwide trust Nokia to deliver secure, reliable and sustainable networks today – and work with us to create the digital services and applications of the future.

Nokia operates a policy of ongoing development and has made all reasonable efforts to ensure that the content of this document is adequate and free of material errors and omissions. Nokia assumes no responsibility for any inaccuracies in this document and reserves the right to change, modify, transfer, or otherwise revise this publication without notice.

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis

© 2024 Nokia

Nokia Oyj
Karakaari 7
02610 Espoo
Finland
Tel. +358 (0) 10 44 88 000

Document code: 875208: (April) CID207598