EOS: the Modern Cloud Network Operating System

Introduction

The network is the enabler of new business models as well as the nervous system for all communications, entertainment, commerce and regulations. Network availability, performance, scale, automation, and security are now fundamental business requirements driving next-generation cloud, carrier, and enterprise networks. The underlying network operating system (NOS) provides the critical software foundation and is the core of this nervous system that enables businesses to operate in the modern world.

Over the past decade, Arista Networks has pioneered cloud networking solutions with a unique software-driven approach to building hyper-scale reliable networks. Arista EOS® (Extensible Operating System) developed from the ground-up is a robust, programmable and innovative operating system, with over fifty million ports deployed in the campus, hyperscalers, WAN, and in carrier-cloud environments.

As customers shift to adopting cloud first service delivery models with DevOps, the network software needs to evolve and meet the goals of the multi-cloud deployment at scale and provide a solid foundation for the Continuous Integration/ Continuous Delivery (CI/CD) model and quality of experience.

This whitepaper discusses how Arista's EOS architecture provides the foundation to build cloud networks and modern NetDevOps for the next generation of data driven network architectures, based on cloud principles.

Cloud Networking Requirements

The Internet is now like any utility in that any disruption or outage has a cascading impact for businesses and consumers alike. The next-generation networks, based on cloud architectures, provide a self-healing, scalable, automated platform supporting a variety of workloads from bare-metal to containerized and an IT infrastructure that is agile, servicing business applications across on-prem and multi-cloud. Ensuring network availability is a top priority for all businesses and this important metric is directly related to the architecture of the network and the operating system software capabilities.

The next generation networks are built on cloud principles, drive digital & technology transformations, and have the following foundational requirements:

Scale & Performance - Cloud based services have to scale and provide agility and performance - whether it is to meet bursts of demand or service global pods for faster delivery. The network operating system has to meet the demands of hyperscale environments with dynamic workloads, support for a large number of routes and peers and provide the performance with fast convergence, programmatic end-to-end traffic engineering with sub 50ms fault recovery, across data centers, routing edge, core as well as campus infrastructures.

Resiliency & Fault Containment - Network outages have cascading effects from loss of connectivity to application availability to loss of revenue and productivity. A fault event can originate from any aspect of the network - hardware, transceivers, cabling, software, etc. Fault detection, isolation, recovery, and resiliency are key functions, which, at its basic or foundational level, are part of the NOS. The network fabric, threaded together by the systems and software, must provide resiliency and workaround the faults. This directly relates to the quality of the software in production.

Simplification and Open Standards - Open standards drive efficiency. Proprietary technologies have shown accumulation of OpEx burden and rigidity of architectures over time, which eventually fail to meet the needs of newer technology shifts. Open standards further allow migration and interoperability, without needing CapEx for forklift upgrades and OpEx for managing disparate networks.

Programmability & Automation - NetDevOps is all about software - network elements as code objects to code and creating an operational workflow as part of Continuous Integration/Continuous Delivery (CI/CD) model for configuration, change management, segmentation & security policies, pre and post-deployment validation, etc. This modern approach requires deep programmability hooks into the NOS for tighter integration with applications utilizing open APIs, be it the control plane, management plane, or the data plane. Application monitoring, traffic engineering, and dynamic resource optimization, all require NOS to be programmable. Along with programmability, automation helps scale the management, provisioning & troubleshooting of hundreds, or possibly thousands, of networking devices at an increasing pace of change. Automation helps scale day-0 provisioning, implement on-demand security policies, and reduces the time to execute the repetitive tasks of sifting through telemetry and Syslog data for faster Mean Time to Repair (MTTR).

Modern Telemetry and Observability - While design and deployment are the initial steps, managing the modern networks need observability with real-time visibility, predictive analytics, and troubleshooting. This starts with software providing real-time state streaming and newer innovations in telemetry. Observability also needs to tackle additional challenges including quality of experience (QoE) management, and autonomous network detection and response (NDR) to help network operators find subtle 'grey failures'.

Digital transformation is driving large amounts of traffic, and generating telemetry data in the form of logs, security alerts, system state information, configuration changes, fault-related events, etc, from every network device. These 'network data lakes' create a foundation for the AI/ML operations to enhance observability and solve issues in real-time

Product Security - Networks traditionally are the first line of defense for cyber attacks. With programmable stacks, a new attack surface is added to the equation - the NOS itself. It's imperative to secure the management, control, and data planes of the network OS as they can be exploited as a backdoor to compromise the applications and the business.

Limitations of Legacy Operating Systems

Legacy Network Operating Systems - are still, unfortunately, in outdated production environments - and have been susceptible to a multitude of problems - software crashes, vulnerabilities, and scaling to name a few. The monolithic architecture severely limits the ability of the software to deliver the requirements of the technology shifts and business models such as network reliability, automation, deep programmability, real-time telemetry etc. A closer look, at the legacy OS architectures exposes the following limitations:

Lack of fault containment & isolation: Legacy NOS with their monolithic kernels lack a lot of modern software innovations - process restartability, CPU slicing, memory protection, isolation between processes & clear abstractions that avoid fate sharing, inefficient message processing etc. The lack of these capabilities causes higher incidents of network disruption - be it a defect leading to crash or processing messages of unrelated events by every agent. Over time, these issues manifest as poor application experience, slower convergence, and non-performance of the network

Lack of programmability & automation: Lack of SDK or extensibility in legacy network operating systems makes it harder and needs heavier lifting by engineering from the vendors side, to integrate with modern automation & tooling framework. Lack of APIs, SDKs and customized kernels prevent the users from implementing NetDevOps and software-defined controls to manage the network for agile service delivery. Finally, the internal state cannot be exposed and the only programmability that is possible in these stacks is via high-level wrappers, which may provide the same information as SNMP or the CLI in text or XML formats for monitoring purposes.

Legacy monitoring: For today's demanding networks, legacy NOS's methods of SNMP polling every few seconds creates a gap in capturing critical information. This information gap amplifies when operators would like to look at the network health in a holistic manner and leads to multiple blind spots in monitoring and results in prolonged outages. Production environments today mandate real-time streaming telemetry for accurate root cause analysis on first failure.

Versions and trains: Historically legacy vendors have had different NOS's by network location and often delivered different binaries per product line/chipset. The result of this is that customers end up with 3-4 different NOS with 8-12 different trains. This in turn creates a huge perpetual operational expense (OpEx) tax on customers as the network operations are burdened and trapped in the cycle of qualifying new bug fixes, maintaining many different versions of trains and customizing configurations. The deviation of software trains across segments adds to silos in the operations, adding unnecessary cost for NetOps.

Security - Security vulnerabilities in legacy NOS have been in the news constantly. The design of legacy NOS inherently lacks process & memory protection, and can be exploited very easily with the latest off-the-shelf and open-source hacking tools.

In summary, legacy network operating systems, being inherently monolithic with heavily modified code-paths, are susceptible to software crashes causing unplanned outages and exposure to security vulnerabilities, and cannot provide the foundation for cloud networking. Additionally, having multiple release trains multiplicatively expands the complexity of operations as it relates to vulnerability and defect management. All these factors along with manual error-prone configuration models inevitably compromise network availability leading to poor service delivery and user experience.

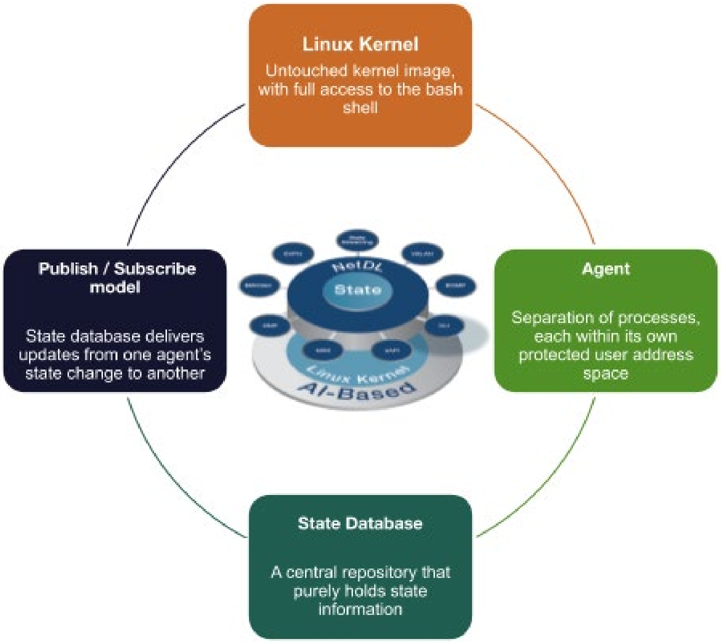

Linux Kernel
Untouched kernel image, with full access to the bash shell
Publish / Subscribe model
State database delivers updates from one agent's state change to another
NetDL
State
Linux Kernel
AI-Based
Agent
Separation of processes, each within its own protected user address space
State Database
A central repository that purely holds state information

Figure 1: Arista EOS - Robust Software Foundation

Arista's Extensible Operating System (EOS) has been designed from the ground up and is optimized for demanding environments such as hyper-scale data centers, large campuses, multi-cloud connectivity, and carrier networks. It combines modern-day software and operating system concepts, building on resiliency, programmability, transparently restartable processes, open platform development, an unmodified

Linux kernel, and a stateful publish/subscribe state database model. EOS simultaneously supports multiple chipset architectures, and virtualized and containerized deployment use cases with a single release train. This approach provides the same consistent operational experience and high quality across the entire Arista networking portfolio.

To fully understand the advantages and benefits of an open & programmable OS, we will examine the key components and attributes of EOS architecture as well as the rich array of network services built on top of EOS.

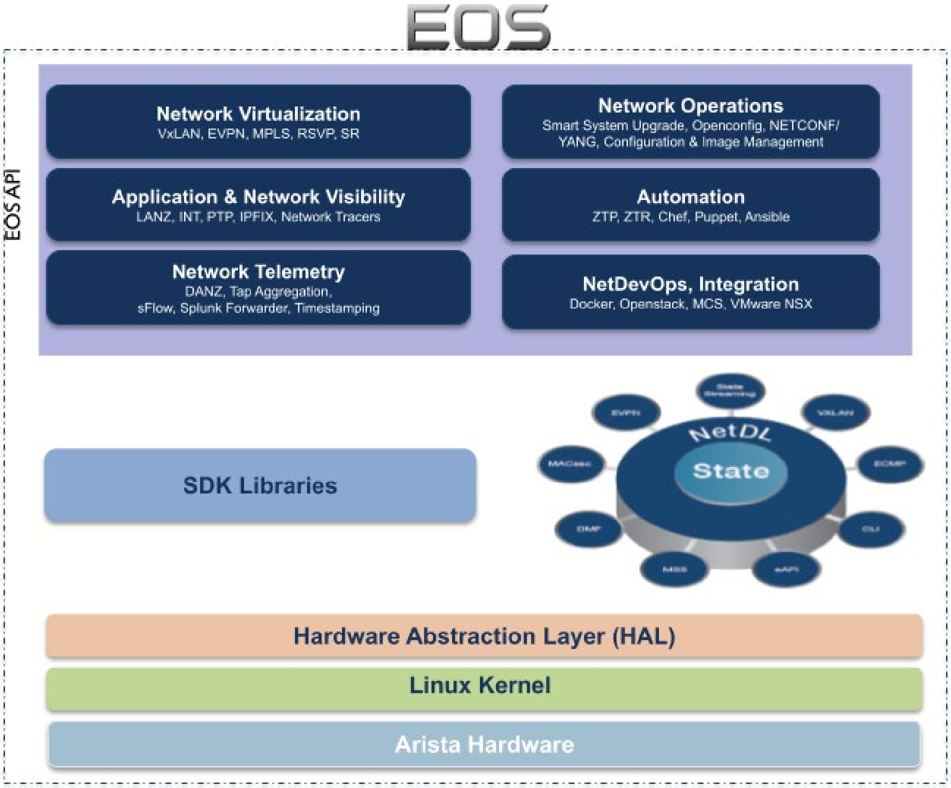

EOS
EOS API
Network Virtualization
VxLAN, EVPN, MPLS, RSVP, SR
Network Operations
Smart System Upgrade, Openconfig, NETCONF/
YANG, Configuration & Image Management
Application & Network Visibility
LANZ, INT, PTP, IPFIX, Network Tracers
Automation
ZTP, ZTR, Chef, Puppet, Ansible
Network Telemetry
DANZ, Tap Aggregation,
sFlow, Splunk Forwarder, Timestamping
NetDevOps, Integration
Docker, Openstack, MCS, VMware NSX
SDK Libraries
State Streaming
EVPN
VXLAN
MACsec
ECMP
DMF
CLI
MSS
eAPI
NetDL
State
Hardware Abstraction Layer (HAL)
Linux Kernel
Arista Hardware

Figure 2: Arista EOS Architecture

Modern OS Architecture

Fundamental to Arista EOS is a unique multi-process state sharing architecture that separates state information from the processes themselves. This well-defined abstraction between service layers is Arista's core software design philosophy to deliver network availability and superior quality. This enables fast convergence, fault recovery, and real-time software updates at a process level without affecting the running state of the system. Protocol processing, security functions, management services, and even device drivers run in user address space, not in the kernel itself. This separation greatly increases overall stability, and by maintaining the design discipline of keeping the Linux kernel environment pure, it provides the user the ability to leverage open-source Linux tools. The unique design supports a single image for different platforms, each with a separate silicon architecture, without having to build multiple software versions.

At the core of EOS is the in-memory database (machine generated at run time), which runs in user space and contains the complete state of the system. It is designed for synchronizing state among processes, also called 'agents', by

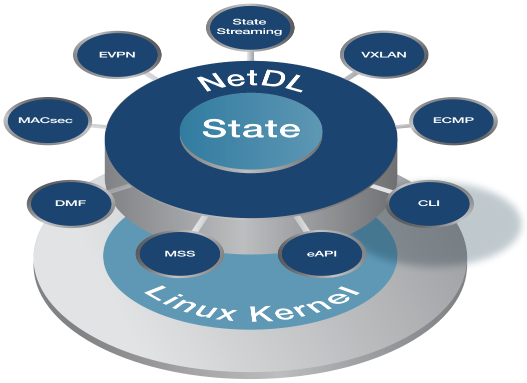


Figure 3: EOS State Database

notifying interested processes or agents when there is a change. The state database functions like an event-driven publish/subscribe model. Each EOS agent subscribes to this state database for notifications when the state of other agents change. The change notification is buffered and asynchronously sent to the state database, which then notifies all other agents who have subscribed to the changed state. This publish-subscribe architecture is a highly efficient model ensuring low overhead, scaling across multi-chip designs and provides resilience in case of faults.

Arista EOS has evolved from an in-system centralized state database that enables process level restarts with minimal system disruption, to a centralized network database (NetDB). This evolution combined multiple discrete network device's state into a centralized repository that enables easier troubleshooting, and better root cause analysis. This approach to passing state throughout the system, and the automated way the database code is generated, reduces risk and error, improves software feature velocity, and provides flexibility for customers who can use the same APIs to receive notifications from the database to customize and extend switch features.

Simplification

NetOps is challenged with multiple, siloed operational workflows and planning, even though the majority of operational procedures overlap across network roles. EOS brings consistency, flexibility and simplification in NetOps via:

1. Multiple silicon architecture support - EOS supports multiple silicon architectures - Broadcom DNX, Broadcom Strata, Intel Barefoot etc. This allows network architects to choose platforms for specific use cases such as Virtualization, HPC, Edge etc., and deploy both scale-out (leaf-spine) and scale-up (modular) designs.
2. Generational technology migrations - EOS provides the same software not only across silicon architectures, but also across silicon generations which helps customers migrate from 10G to 25G to 400G based IP fabrics. The ease of qualification of new generation of silicon cuts down qualification time and provides competitive advantage to customers
3. Consistent operational model - Having the same software image across multiple deployments means NetOps can use the same operational and integration model (integrate with external engines like Splunk or VMWare NSX) and drive a simplified end to end automation framework using the EOS Network APIs

Fault Containment & Resiliency

The EOS multi-process state-sharing architecture is also the key to providing high resiliency. Any fault is contained within the agent or driver where the fault originated. If the fault causes the agent to crash, then the EOS process manager restarts the agent immediately. If the fault causes the agent to hang or loop, the process manager detects the condition and restarts the agent. Thus, faults within EOS are self- healing.

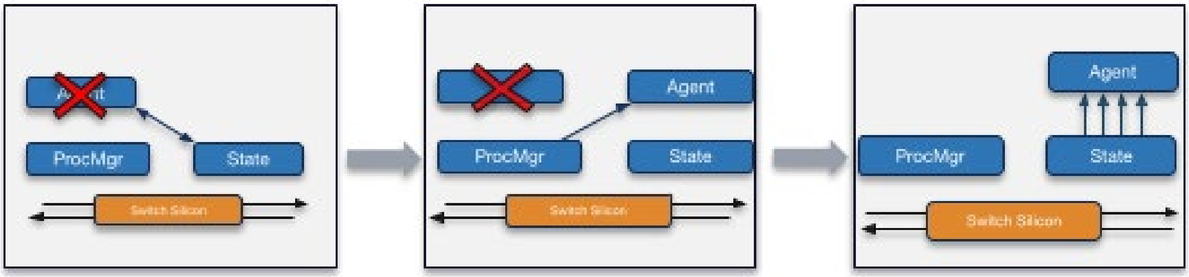


Figure 4: EOS Fault Containment and Self-Healing

Most EOS agents can be patched live and can be restarted without disrupting switch operation or the application data flow, so there is no user-perceptible downtime.

EOS provides a robust, protected environment for subsystems and agents to enable running third-party agents such as custom protocols or analytics agents within a specific customer environment. Partnering with Arista's EOS extensibility team, customers can deploy, with full confidence, these custom agents alongside their network operating system. Fault containment to a single module extends to security vulnerabilities. For example, if the SNMP subsystem has a vulnerability, then the exploit may read all SNMP-accessible state; however, the exploit will not be able to create additional user accounts, reconfigure interfaces, or run external software. Finally, third-party software may implement custom security policies or intrusion detection to further enhance security through the same extensibility mechanisms.

As a modern operating system EOS is developed using typesafe languages, signed images, signed reviews, and is inherently designed to be more resilient to persistent cyberattacks.

Open Standards

Proprietary solutions have locked the customers from innovating. Over time the inflexibility of the solution, at a control plane scale or performance level or interop level, fails to meet the requirements for a newer technology transition. This leads to a fork-lift upgrade tax on the customers.

Arista EOS supports a comprehensive suite of standards based layer-2 and layer-3 protocols such as EVPN, MPLS, Segment Routing etc., and open APIs for automation and integration. This allows customers the ability to migrate from brownfield/legacy to modern designs as well as adopt the latest in DevOps for management and monitoring.

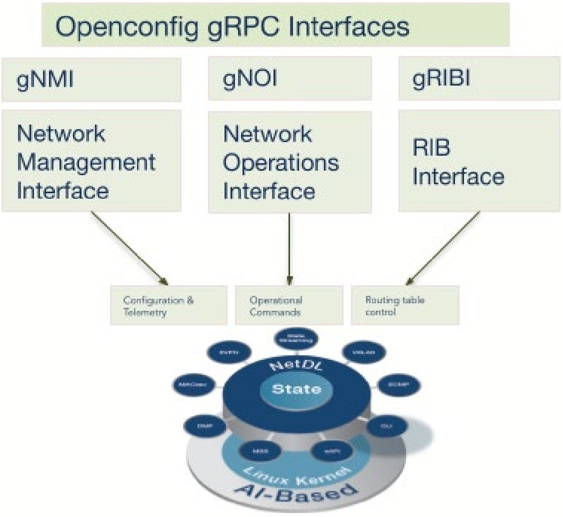


Figure 5: EOS Services - Simplified RPC Service Model

EOS supported Openconfig early on. Openconfig brings a common operational framework using declarative configuration and model-driven management for multivendor networks. It provides vendor neutral data models and streaming telemetry for network management. In addition to supporting NETCONF/YANG, EOS supports gNMI - gRPC Network Management Interface - an IETF draft for retrieval and manipulation of state from network elements. The gNMI service defines operations for configuration, management, operational state retrieval, and bulk data collection via streaming telemetry.

Programmability and Extensibility

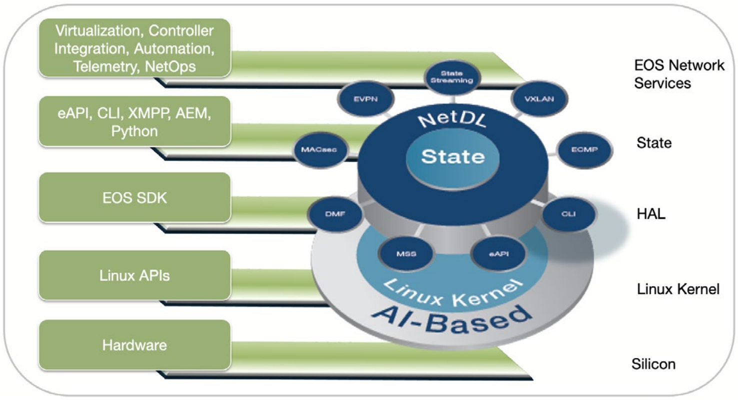


Figure 6: EOS Programmability Framework

With a software-first approach EOS was designed to be programmable across all layers – Linux kernel, hardware forwarding tables, switch configuration and CLI, switch control plane as well as management layer. Arista EOS's rich set of programmable interfaces including:

- Linux shell access and APIs
- State Database APIs
- Python, Perl scripting, Advanced Event Management
- EOS SDK
- JSON based eAPIs, CLI, SNMP , XMPP
- Containers, VMs and 3rd party agents

EOS provides extremely robust and reliable communication services while preserving the Linux heritage of security, stability, openness, modularity, and extensibility. This combination is unique in the industry. Arista EOS has full Linux shell access for root-level administrators and makes a broad suite of Linux-based tools available. DirectFlow allows customers to program the forwarding state of the switch in order to fine-tune packet forwarding based on application needs. State Database APIs provide access to all internal state, including low- level counters, temperature measurements, power supply status and all other parameters necessary to monitor and manage the system natively. This state data, part of NetDL, can be directly accessed by AI/ML engines as part of AIOps.

JSON-based EOS APIs (eAPI) provide easy web-based integration with tools commonly used to manage compute and storage resources as well as orchestration systems. Even the CLI written in Python is customizable. Scripts based on Python, go, etc., can also be developed as third party or native integration with applications, controllers & layer 4-7 services.

With EOS Software Development Kit (SDK), customers can develop their own customized EOS applications in C++ or Python. This development model allows third party applications to be first-class citizens of EOS along with other EOS agents. The SDK provides programming language bindings to software abstractions available in Arista EOS, so third party agents can access switch state and react to network events. These applications can, for example, manage interfaces, program IP and MPLS routes, Access Control Lists (ACLs), as well as use a range of APIs to communicate between the switch and network controllers. The SDK targets both long-running processes requiring event-driven notifications and scripts requiring high-performance interactions with other EOS agents. The state separation through system database and the inherent fault isolation enabled by the modular architecture allows customers to innovate/ develop and install their own applications without fear of disrupting the entire system.

EOS provides a platform for customers looking to deploy virtual machines or containers within the OS. This native support helps in custom application integration for dynamic environments managed by NetDevOps teams. For example, a customer wants to run a container to monitor CPU and memory utilization of a 3rd party application in EOS or run a custom process to measure latency & bandwidth between applications and end customers etc.

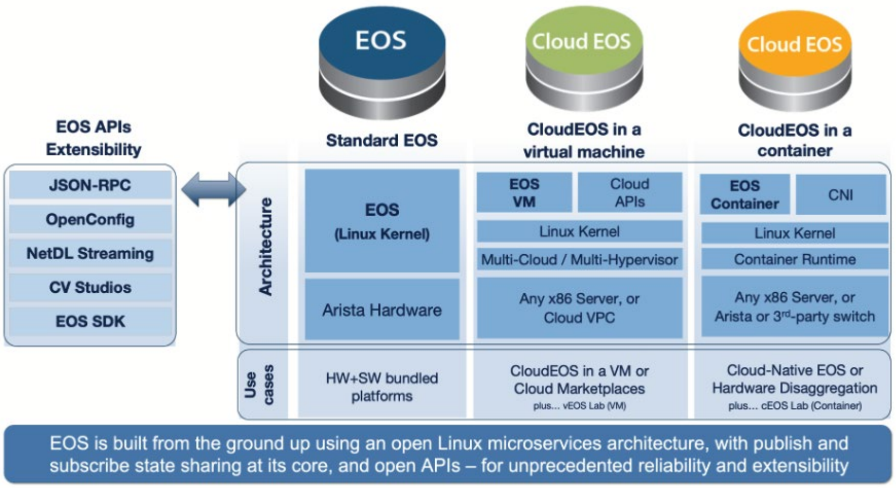
This rich programmability stack truly makes EOS the right network OS for next generation cloud networks.

Disaggregation/ Abstraction - Arista EOS's highly programmable architecture lends itself to another modern software paradigm - disaggregation. Customers are looking to disaggregate management, control and data planes at the OS levels, to provide them the flexibility of cost and control of the network.

EOS provides common APIs (eAPI), Opencofig and gRPC/ gNOI to achieve the management plane abstraction - a single dashboard for provisioning and monitoring the entire network via a common API and data model. For example, an operator defines a high level policy and this generates appropriate access lists and applying them to each physical device

Control plane abstraction is inherent in the architecture as EOS agents run in user address space, not in the kernel itself. This abstraction allows customers to write their own agents or even bring their own Linux version to run with EOS!

Data plane abstraction allows EOS to be run in embedded, virtualized and in containers. This allows customers the flexibility to extend EOS into the hypervisor or OS, for consistent operational control as well as deploy platforms with their choice of silicon for application needs.



EOS is built from the ground up using an open Linux microservices architecture, with publish and subscribe state sharing at its core, and open APIs – for unprecedented reliability and extensibility

Figure 7: EOS Platform Flexibility

Network Observability

Network observability is a next-generation, analytics-driven approach to pervasively monitor all application traffic by gaining complete visibility into physical, virtual, and cloud environments.

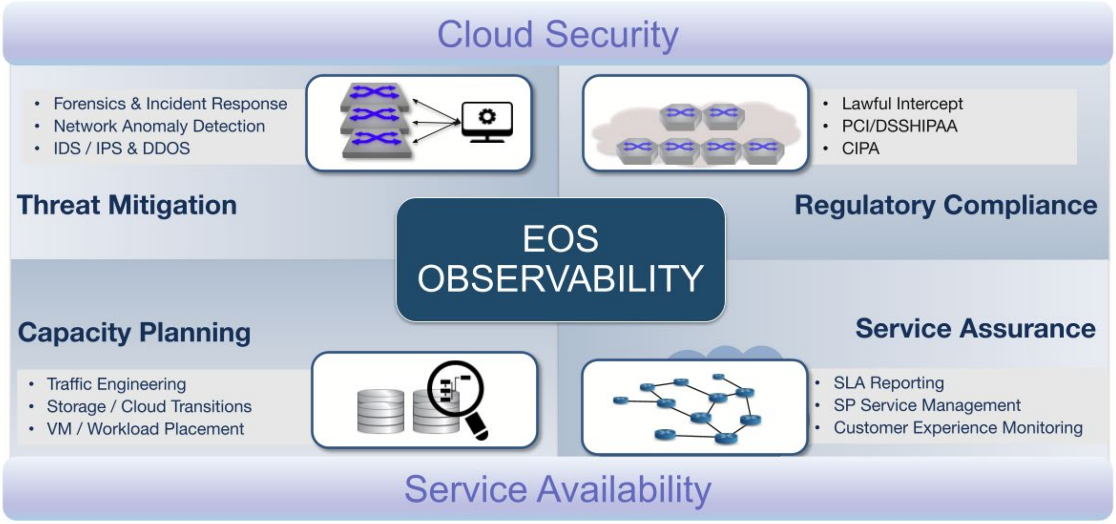


Figure 8: EOS Observability

EOS provides a rich set of observability tools to enable operations to monitor round the clock, network health, hotspots, faults, malicious traffic, regulatory compliance adherence and much more. EOS also provides for rapid integration with a wide range of third-party applications for telemetry, analytics, billing and OSS. These include:

- Real time state streaming/ NetDL
- LANZ
- DANZ
- sFlow
- IPFIX
- Mirroring
- Inband Network Telemetry
- Postcard based Telemetry

EOS state database can be streamed real time to provide information about every aspect of the system, including transceivers, interfaces, attached hosts MAC and IP address, routing peers, routing table, VLANs, ACLs, counters providing statistics of flow information and much more. Updated upon a state change, it allows operators to check real-time network health. Combined with automation, customers can mitigate issues within minutes, increasing OpEx and experience.

EOS NetDL - a Data Driven Approach to network operations

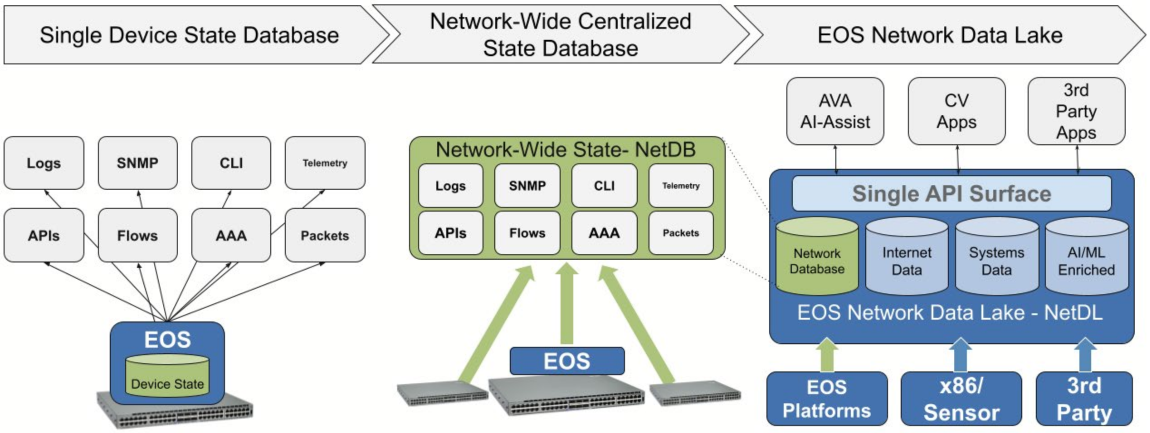


Figure 9: Evolution of EOS State to Network Data Lakes

From an observability perspective, the network is a repository of several data lakes - a collection of large amounts of state data from various sources (structured and unstructured). Arista NetDL builds onto the centralized NetDB architecture that aggregates all of the device state data from hundreds of network devices into a common, time-series, database using OpenConfig data models. Key characteristics being NetDL are:

- Multi-Tenant - NetDL is designed to support inherent multi-tenancy so it can be implemented on-premises or in an as-a-service offering.
- Multi-Modal - NetDL is designed to support multiple data modalities - from streaming telemetry, to full flow and packet-level capture, and external enrichment data sources
- AI/ML Ready - AI and ML technologies depend on accurate data for model generation and training data. By gathering high-fidelity 'perfect data' and storing it over time EOS NetDL is the network data foundation for AI/ML solutions
- Data Capable - EOS NetDL is designed to capture and aggregate data once, and make it available to multiple discrete applications concurrently, across the campus, WAN, data center, cloud, and branch networks - but then use this data for application performance monitoring, network performance monitoring, network detection and response, threat modeling, and AI Ops/NetOps workflows

The type of customer problems this solves are a look back and predict forward modeling, where customers can view historic data and then use that to create supervised learning models to predict what will happen on the network with specific changes.

EOS DANZ/ DANZ Monitoring Fabric (DMF)

Arista DANZ Monitoring Fabric (DMF) is a next-generation network packet broker (NPB) providing deep hop-by-hop visibility, predictive analytics and scale-out packet capture — integrated through a single dashboard . Complementing the DANZ Monitoring Fabric (DMF), DANZ EOS provides single-hop packet processing and data capture with highly accurate timestamps, fine-grained filtering, like MPLS header removal with traffic steering of mirrored packet, to GRE tunnels etc. It is both an in-band and out-of-band telemetry and packet capture architecture, transform opaque datacenter traffic into comprehensive visibility for security threat detection and mitigation, application and network performance management, service availability monitoring, traffic recording and troubleshooting.

Flow Monitoring

EOS supports standard flow monitoring tools such as sFlow and IPFIX, with hardware assist capabilities providing flow visibility into multi-terabits scalable platforms. In addition, mirroring capability of the platforms augmented with header stripping, user defined fields, PTP timestamping and steering of mirrored traffic provide a powerful toolset for operations to monitor at scale. In order for the collectors to correlate data collected from all sources, EOS further provides two key functionalities:

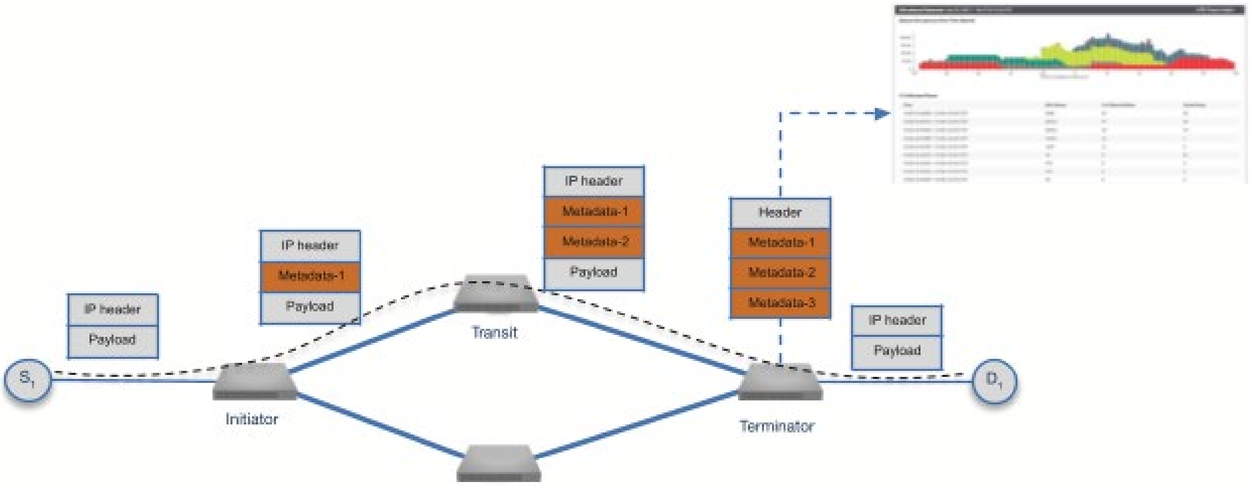


Figure 10: EOS Inband Network Telemetry

Inband Network Telemetry

Inband Network Telemetry is used to gather per flow telemetry information like path, per hop latency and congestion. This information is exported to the collector using protocols like IPFIX and may be stored in a time series database for providing per flow historical telemetry information and network analytics.

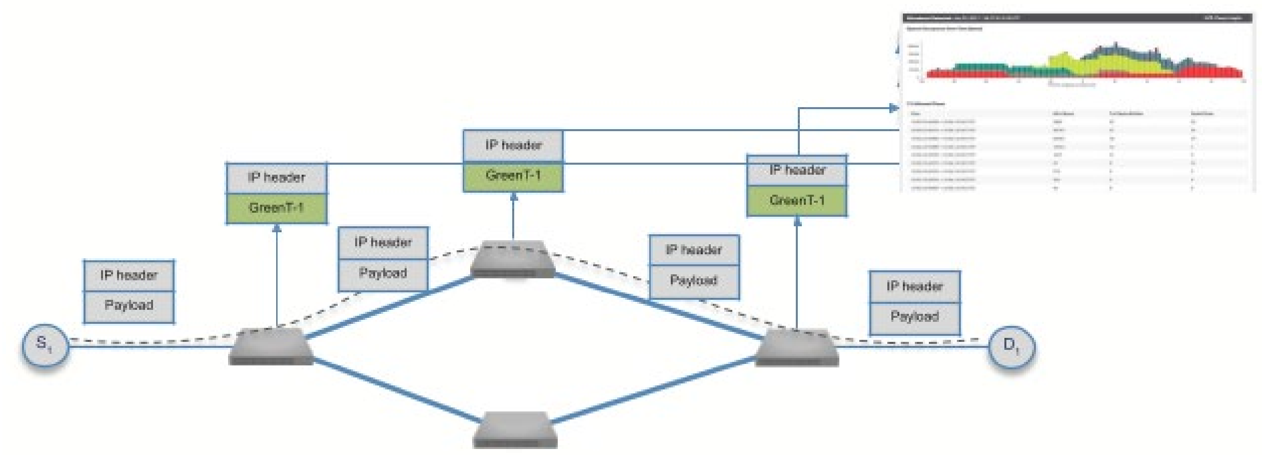


Figure 11: EOS Postcard Based Telemetry

Postcard based Telemetry is used to gather per flow telemetry information like path and per hop latency. Postcard telemetry samples flow at every switch, adding time stamps & aggregating them and sending the samples to a collector with path and latency information.

Arista Network Architectures - Leveraging EOS

Arista EOS: Universal Cloud Networking

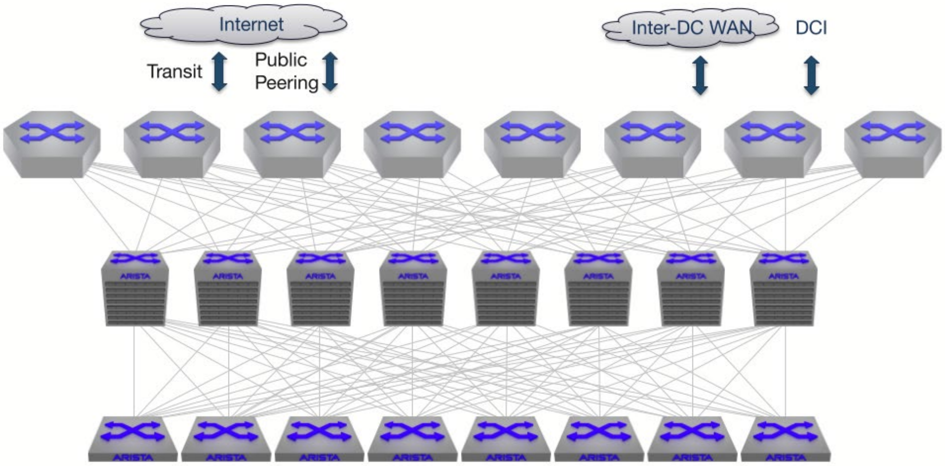


Figure 12: An example of a Universal Cloud Network

EOS provides the robust operating system, the building block, for Universal Cloud Networks (UCN). Cloud principles of open, scale-out, and programmability are the foundation for UCN designs and have been adopted across cloud titans with multi-layer Leaf-Spine topologies hosting thousands of customers, enterprise data centers with virtualized, containerized and bare metal workloads and Telco data centers hosting NFV applications. The UCN data centers are built to deliver a highly available self-healing architecture with link, path, device and network wide redundancy for guaranteed uninterrupted application performance.

EOS enables IP Fabrics based on open standards (BGP, EVPN) helping customers simplify designs, enabling migration to newer generations of silicon and interoperating with brownfield or legacy deployments. EOS's NetDL provides the foundation for modern real time system-wide telemetry coupled with Artificial Intelligence (AI) and Machine Learning (MI) enabled systems for building a powerful predictive analytics framework for IT Operations.

Being open and programmable with full API support, EOS allows for deep integration with northbound orchestrators, controllers, security engines and monitoring tools. This provides consistent performance that scales to support extreme 'East-West' traffic patterns.

Customers can leverage EOS to enable multi-cloud deployment models with their on-premise data center and off-premise public cloud for a unified, consistent service delivery

Arista EOS: Cloud Grade Routing

EOS routing, driven by software-first cloud principles of open, scale-out and software-defined, has disrupted the legacy routing architectures. This is a huge change from legacy hardware-first big routers.

The first step in the simplification is scale-out designs with ECMP, driving resiliency, CapEx and Opex efficiencies. For example - with 5G deployments accelerating, providers are disaggregating the closed & proprietary RAN (Radio Access Network) access architecture and bringing cloud computing with MEC (Multi-access Edge) pods, open interfaces and network automation. The mobile provider can deploy the MEC as a IP data center with scale-out architecture, providing low latency, high bandwidth services such as content delivery, location enabled analytics, security monitoring, etc.

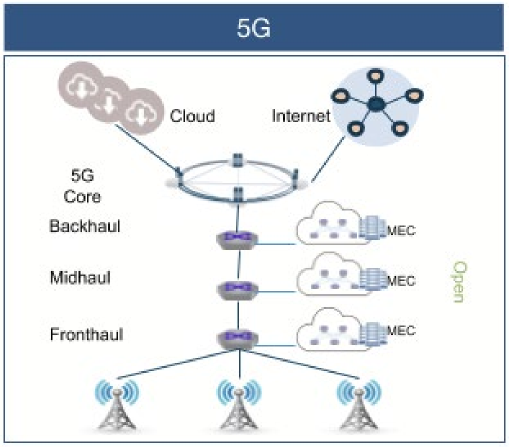


Figure 13: 5G deployment with MEC

The second step of simplification is reducing multiple protocols to one with EVPN, a standards-based approach, that simplifies provider edge use cases, shedding the legacy approach. EVPN addresses multiple use cases like layer-2 and layer-3 extension and edge services like VPN, and Pseudowires (PWEs). This reduces the overhead of using multiple protocols with a single one, across multiple deployments. Further, Segment Routing (SR), provides the perfect paradigm for intelligent software-driven source routing to traffic engineering that eliminates complexity and enables fine grained control. Segment Routing also offers better control plane and data plane scaling by removing the need for per flow state at every network hop and better ECMP characteristics compared to traditional TE solutions. The combination of EVPN for services and SR for backbone provides reduced qualification & troubleshooting time and allows a consistent operational framework across deployments.

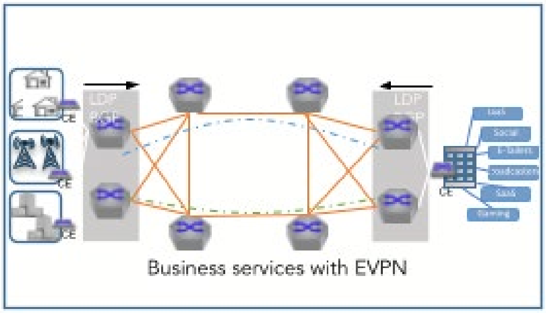


Figure 14: Deploying L2 & L3 Services with EVPN

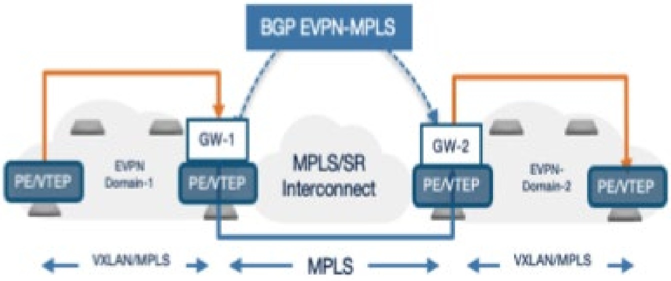


Figure 15: Simplification with EVPN for DCI

The third step being software-driven control - Arista EOS provides rich programmability tools (EOS SDK), open APIs (OpenConfig/YANG, NETCONF), allowing operators to build software driven network deployment for traffic engineering and consistent automation across routing edge and backbone.

For controller-based architectures, EOS SDK provides deep programmability to traffic engineer the network at the silicon level, providing access to label forwarding & route tables for MPLS traffic. In addition, export of routing state via BGP Link-State (BGP-LS) to an external controller for path computation and steering the traffic real time, providing the desired control for customers.

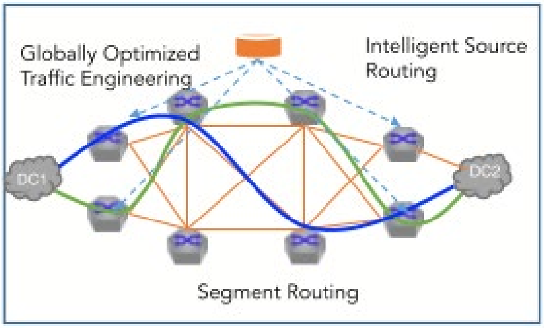

Globally Optimized Traffic Engineering
Intelligent Source Routing
DC1
DC2
Segment Routing

Figure 16: Programmatic Traffic Engineering with SR

Arista EOS: Cognitive Campus

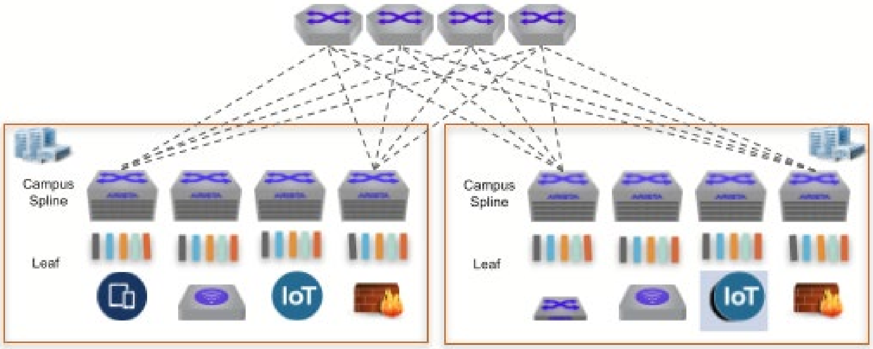


Figure 17: Cognitive Campus Deployments with EOS

Just as with data centers and routing, campus networks are undergoing a massive transition with the advent of hybrid workforce, IoT and new technologies for wired-wireless deployment. Arista's Cognitive Campus Architecture, with EOS and CloudVision, delivers a comprehensive set of capabilities based on the same cloud grade principles used for Cloud Data Centers and Cloud Grade Routing . These include:

Availability and Scale: Deploying efficient leaf-spine architectures - EOS brings cloud deployment maturity with active-active connectivity with dynamically load sharing paths. Open layer-2, layer-3 and virtual overlay (EVPN-VXLAN, transcend the limitations of 802.1q 4K VLANs) feature sets are scalable, interoperable, and dynamically reconfigurable to accommodate device and workload proliferation

Zero touch deployments: The new campus architecture now has to accommodate provisioning new IoT devices like badge readers, security cameras and environmental controllers as well as computers and smartphones. To address the challenge of providing consistent and secure deployment, as part of the cognitive management plane, EOS supports Zero Touch Provisioning (ZTP) to automate deployments and simplify infrastructure. Together with CloudVision for user and application monitoring, common provisioning and telemetry dashboards, customers can implement simple, repeatable and automated architecture ensuring error-free operations.

Group segmentation for security: To ensure compliance and security, segmentation is needed based on functional roles across enterprise workspaces and independent of traditional network addressing. EOS supports Group-based segmentation whereby security policy enforcement is based on logical groups rather than traditional interfaces, subnets or physical ports. For example, to protect the organization from the well-publicized Mirai botnet, an administrator might want to define a group for security cameras and a different group for the networked digital video recorders (DVRs), and yet another one for the physical security administrators. A camera, per policy, will only be allowed to communicate with the DVR and security administrator. A camera will not be allowed to communicate with another camera even if it were on the same subnet. Group segmentation is built on an efficient data plane enforcement mechanism, avoiding the limitations of vendor lock-in solutions

Intelligent Monitoring: EOS NetDL provides the same rich telemetry information as seen in other use cases such as data centers. The telemetry data from campus deployments can now be used for monitoring the distributed campus workforce, allow pinpointing of hotspots with flow tracking, and provide improved security from audit to segmentation. This real time state telemetry allows identifying and inventorying campus devices, users and applications, monitoring key application and IoT SLAs, such as VoIP or security camera applications, and last but not least automatically capture device or user rogue behavior and quarantine them.

All the features are delivered via a single Image EOS that supports an ecosystem of solutions from industry-leading partners, for the latest use cases around dispersed workspaces, constant availability requirements.

Arista EOS: Multi-Cloud Networking

Enterprises are embracing multi-cloud workload deployment strategies as the next step in the evolution of their service delivery. The Arista CloudEOS Router delivers a multi-cloud gateway, with advanced routing and security features like Cloud Network Private Segments, BGP EVPN, IPsec, NAT, Dynamic Path Selection and In-band Network Telemetry. Arista CloudEOS provides consistent operational experience and automation, cloud grade monitoring and scalable solutions.

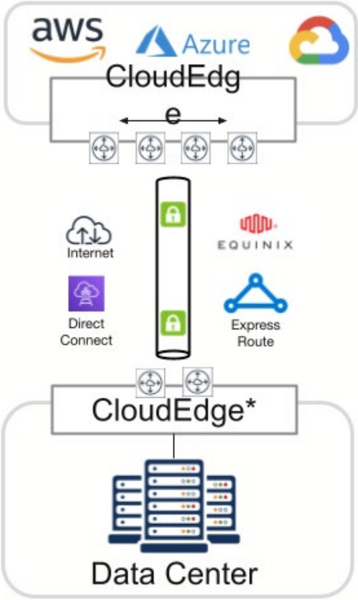


Figure 18: CloudEOS for Multi-Cloud Deployments

EOS Architecture Advantages Summary

Innovations driven by Cloud, Virtualization, IoT and 5G are moving at a breakneck speed. Network architects and operators are challenged to adopt the modern architectures, automation, cloud native build outs and simplifying operations across all network locations and environments, as business models are evolving.

Arista EOS is the modern operating system built for scaled environments using cloud principles. Here is the summary of the key advantages of EOS:

Open, programmable platform: Arista EOS shipping as a single software train supports multiple silicon chipsets across fixed and modular platforms. This provides the same consistent operational model across all locations/ profiles. Having a consistent operational model provides agility - a competitive advantage for rolling out new services.

Modern network observability: Real time state streaming in EOS enables modern telemetry for next generation cloud networks. NetDL - a multi-tenant and multi-modal data lake that stores all network state from EOS networking devices & additional data sources, provides the foundation for applying AI and ML technologies to the new observability frameworks.

Data Analysis (DANZ) provides rich traffic mirroring and monitoring capabilities. Integration with Splunk, sFlow-based collectors, and application monitoring tools such as Corvil provide traffic visibility. Tracer and monitoring capabilities for containers (Container Tracer), Virtualization (VM Tracer), Latency Analysis (LANZ), and Inband Network Telemetry provide a rich visibility and monitoring toolkit for monitoring network health .

Network automation for the cloud era: EOS natively supports Ansible, Puppet, and Chef which enables network configuration in the same manner as servers and storage within data center environments. In addition, Zero Touch Provisioning (ZTP) automates the provisioning of network infrastructure and speeds time to production for new services while eliminating the risk of human error and Zero Touch Replacement (ZTR) provides automated provisioning of replacement switches, significantly reducing mean-time-to-replacement of a failed switch.

Multi-Cloud deployments: Arista CloudEOS extends the EOS software platform running on the physical switches to a virtual machine/ container based offering. In addition to network design and validation, CloudEOS enables connectivity to multiple clouds as well as deploying virtual private clouds for enterprises extending services via the cloud.

Conclusion

Arista's EOS Software is the most advanced, resilient and programmable operating system and has continued to evolve the classic Software-Defined Networking (SDN) principles to software-driven networking control and an AI enabled holistic view with NetDL, while building on Arista's core pillars of reliability, open standards, and programmability. Arista's EOS provides industry leading network services, operational innovations and integration capabilities across data center, routing, campus and multi-cloud.

For more information, visit: https://www.arista.com/en/products/eos

References

Arista EOS Routing Feature Sheet

Arista Cloud Grade Routing

Cognitive Campus White Paper

Arista Multi Cloud Networking

Arista Zero Trust Security for Cloud Networking

Arista DANZ Monitoring Fabric

Arista NetDevOps on GitHub

Santa Clara—Corporate Headquarters
5453 Great America Parkway,
Santa Clara, CA 95054

Phone: +1-408-547-5500
Fax: +1-408-538-8920
Email: info@arista.com

Ireland—International Headquarters
3130 Atlantic Avenue
Westpark Business Campus
Shannon, Co. Clare
Ireland

Vancouver—R&D Office
9200 Glenlyon Pkwy, Unit 300
Burnaby, British Columbia
Canada V5J 5J8

San Francisco—R&D and Sales Office
1390 Market Street, Suite 800
San Francisco, CA 94102

India—R&D Office
Global Tech Park, Tower A, 11th Floor
Marathahalli Outer Ring Road
Devarabeesanahalli Village, Varthur Hobli
Bangalore, India 560103

Singapore—APAC Administrative Office
9 Temasek Boulevard
#29-01, Suntec Tower Two
Singapore 038989

Nashua—R&D Office
10 Tara Boulevard
Nashua, NH 03062