ARISTA

EOS: An Extensible Operating System

Executive Summary

Performance and stability of the network is a business requirement for datacenter networking, where a single scalable fabric carries both network and storage traffic. Management processes must be scalable for successful deployment and operation, requiring a high degree of automation and integration into the customer's unique environment. The design and architecture of the network operating system provides the foundation for meeting these requirements.

The monolithic software architecture that runs on almost all of today's network switches presents a fundamental design limitation to high network reliability. The stark reality is that a single bug or defect anywhere in the OS exposes the entire system to disruption since there is no mechanism for software fault containment. At the same time, with no isolation between multiple tasks it is difficult to scale these systems from a performance perspective or to add new functionality. In fact, making any significant changes to a code base that is measured in millions of lines of code will reduce product stability and reliability. The fragile nature of the monolithic approach inherently prevents to extend the network operating system to implement new features such as integration with customer specific management processes.

Arista's Extensible Operating System (EOS) leapfrogs traditional network OS designs by providing the following capabilities and benefits:

In-Service Software Upgrade (ISSU)	Reduced maintenance windows due to ability to upgrade processes without system interruption
Software Fault Containment (SFC)	Faults are contained to a single module Provides superior system stability
Stateful Fault Repair (SFR)	Continuous health monitoring of all processes Enables invisible repair of faults
Security Exploit Containment (SEC)	Improves security by limiting any potential vulnerability to an individual module
Scalable Management Interface (SMI)	Enables automated maintenance, updates and integration with 3rd party NMS systems

Arista's EOS delivers these benefits with a unique multi-process state sharing architecture that separates networking state from the processing itself. This enables fault recovery and incremental software updates on a fine-grain process basis without affecting the state of the system, as well as security patches behind the scenes. In addition, protocol processing, management functions, and even device drivers run in user address spaces, not in the kernel itself. This greatly increases the stability of the kernel, which is a standard Linux kernel, making it safe to extend the operating system with additional functionality. Thus EOS provides extremely robust and reliable data center communication services while preserving the Linux heritage of security, stability, openness, modularity, and extensibility. This unique combination offers the opportunity to significantly improve the functionality and evolution of next

generation data center networks.

Introduction

Arista's EOS was designed from the ground up to provide the most robust foundation for the business needs of next-generation data center networks. EOS's advanced architecture delivers the following key benefits:

- **High Availability**

  EOS provides software fault containment (SFC), in which the impact of software problems is limited to a single module, preventing disruption of other switch functions. Further, EOS provides stateful fault repair (SFR), in which the faulty module is automatically restarted with its state intact and without affecting the rest of the system. EOS' superior fault isolation prevents problems such as console lock-up that plague less resilient software architectures. Finally, EOS may be extended via third-party software to provide custom monitoring, failover, and load balancing, all of which lead to higher availability.

- **Reduced Maintenance Windows**

  EOS supports in-service software upgrade (ISSU), allowing individual software modules to be upgraded without affecting the rest of switch operation including packet forwarding. In this way, patches or enhancements can be deployed outside of maintenance windows.

- **Scalable Management Processes**

  The familiar EOS command-line interface (CLI) avoids retraining costs, and EOS' single release train simplifies software upgrade deployment. Further, EOS may be extended through third-party software to enable tight integration with any in-house NMS system and the automation of routine maintenance tasks, including deployment, monitoring, maintenance, and upgrade.

- **Improved Security**

  If a vulnerability exists in an EOS module, the impact of any exploit is limited to the capabilities of that module. The vulnerability may be quickly and transparently patched with ISSU. With its support for third-party extensions, EOS may be extended with customized security policies and/or intrusion detection systems.

The next section explains the EOS architecture and how it yields the benefits above.

EOS Multi-Process State-Sharing Architecture

The key to EOS benefits is its unique multi-process state-sharing architecture that consists of multiple processes interacting with a central shared state repository called Sysdb (system database). This architecture is shown in Figure 1 below.

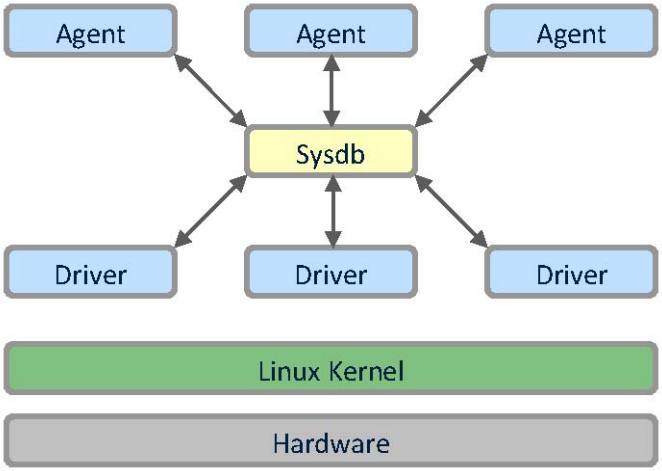

Figure 1: EOS Architecture

EOS derives its benefits from the essential characteristics of this architecture:

1. Each switch function is in a separate address space, including each CLI session, each hardware device driver, and each protocol daemon such as routing protocols, spanning tree, and LACP.

2. Sysdb holds all state, while agents perform all processing. Sysdb is an address space that purely holds state and delivers state updates from one agent to another. For example, when a link goes down, a port driver updates the link state in Sysdb, which delivers the update to the SNMP service, which then sends a trap. Agents may contain copies of Sysdb state for efficiency, but all state is recoverable from Sysdb whenever needed. Like a traditional database engine, Sysdb itself is ultra-reliable because it contains no application code.

3. Processing is in user space. Protocol operation, management function, and device management, including hardware device drivers, run in user address spaces, not in the kernel. By keeping the vast majority of processing out of the kernel, system stability is enhanced, and extensibility is simplified, because adding a new user process to Linux is simpler and safer than adding kernel-level code.

How the Architecture Delivers the Benefits

This section explains how EOS' multi-process state-sharing architecture results in higher availability, reduced maintenance windows, improved manageability, and improved security.

The key to high availability is fault containment and software self-healing. On most switches, any software fault results in a reload, resulting in seconds or even minutes of downtime. Under EOS, 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 (ProcMgr) restarts it immediately. If the fault causes the agent to hang or loop, ProcMgr detects the condition and restarts the agent. Thus, faults within EOS are self-healing. This process is shown in Figure 2 below.

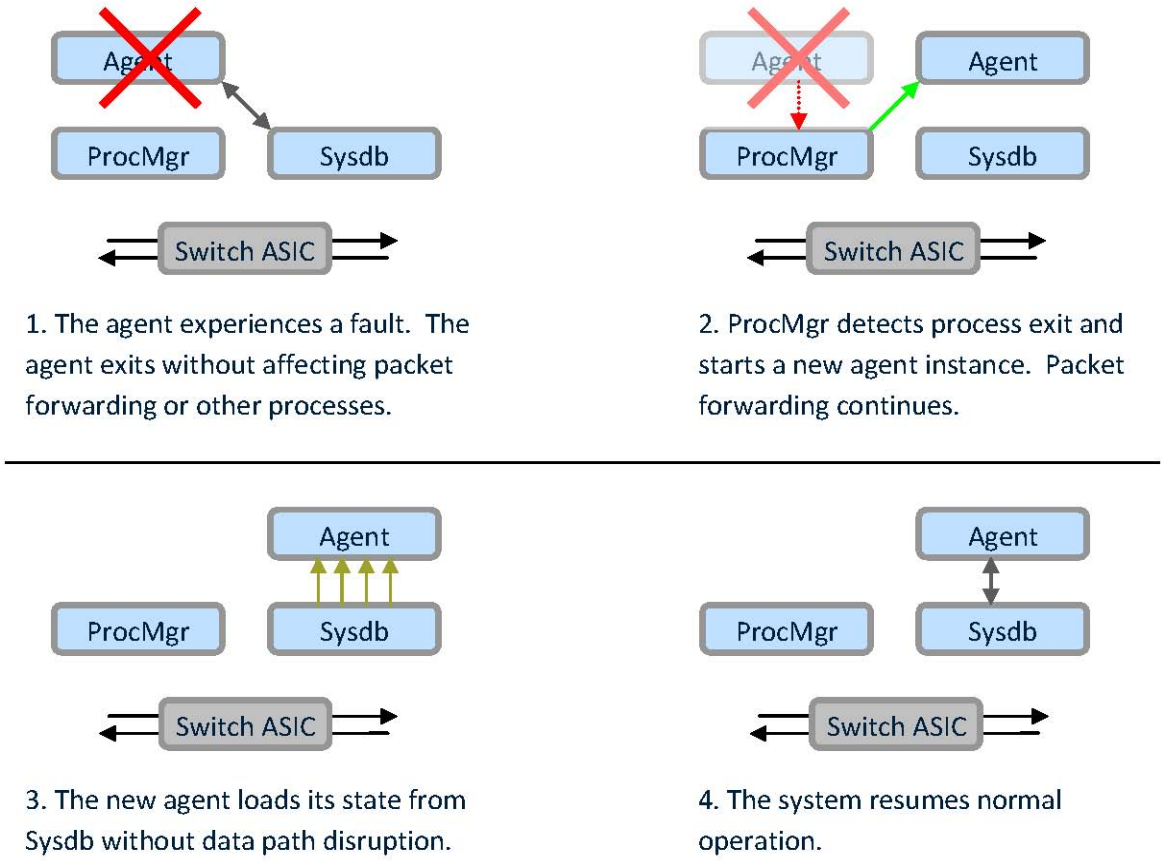

Figure 2: Fault Containment and Self-Healing

The EOS multi-process state-sharing architecture is also the key to reducing maintenance windows by allowing more maintenance tasks to be performed during normal switch operation. EOS' in-service software upgrade (ISSU) capability relies on the ability to restart the patched agent without disrupting switch operation. Packet forwarding is unaffected by the module upgrade process, so there is no user-perceptible downtime. ISSU is illustrated in Figure 3.

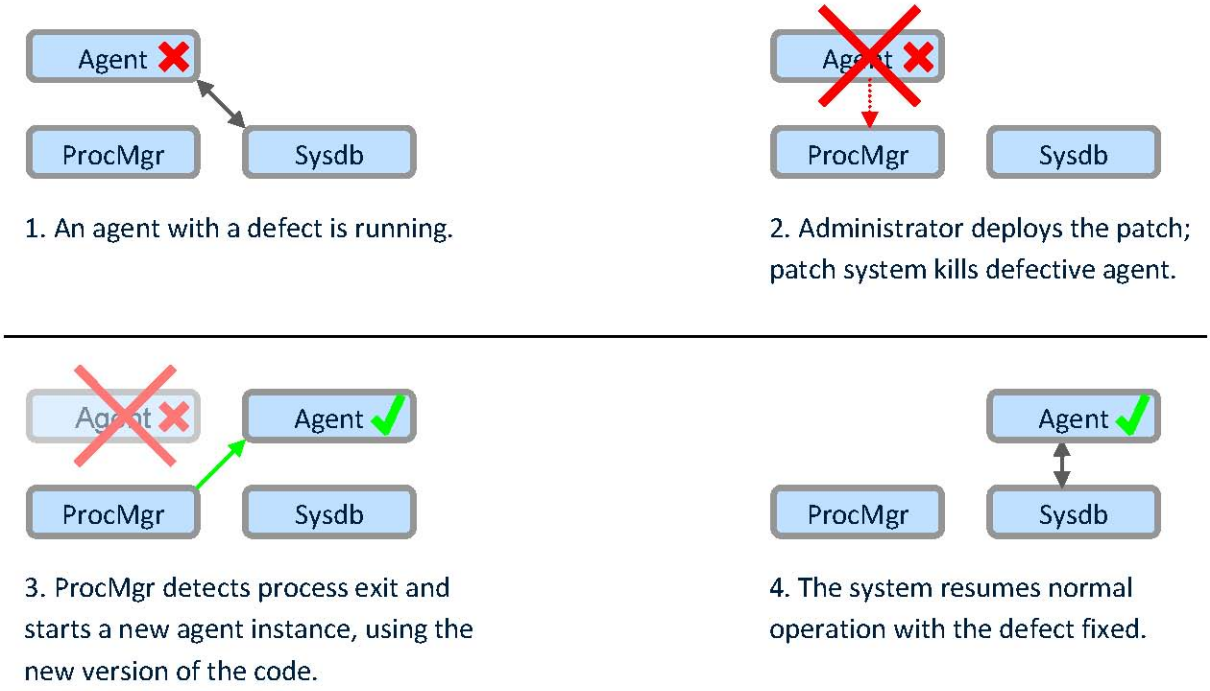

Figure 3: ISSU

The key to improved manageability is the capability to support third-party NMS integration and task automation software. Because EOS provides a robust, protected environment for agents, it is safe to run validated third-party agents as well, tailoring switch behavior to optimize manageability or automating common tasks within a specific customer environment. This extensibility is safe because switch state is protected within Sysdb, and any fault within the third-party extension is contained.

The key to improved security is containing the impact of a security vulnerability within the vulnerable agent. For example, if the SNMP agent 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. In other words, just as the EOS architecture contains faults to a single module, it also contains the impact of security vulnerabilities. Finally, through the same extensibility mechanisms that improve NMS integration, third-party software may implement custom security policies or intrusion detection to further enhance security.

Summary

The EOS multi-process state-sharing architecture provides superior network reliability through software fault containment (SFC), stateful fault repair (SFR), in-service software upgrade (ISSU), security exploit containment (SEC), and scalable management interfaces (SMI) for third-party tools for automated deployment, monitoring, maintenance, and upgrade.