
# Avi Vantage Platform Reference Architecture for OpenStack

Avi Technical Reference (v18.1)

# Avi Vantage Platform Reference Architecture for OpenStack

view online

## Introduction

### About OpenStack

OpenStack has become the de facto standard for open source cloud orchestration because of its openly designed, collaboratively developed architecture by a vast community. OpenStack provides Infrastructure as a Service (IaaS) for deploying scalable cloud infrastructure. Furthermore, OpenStack allows the user to program and automate infrastructure through REST APIs.

### Purpose of This Document

As OpenStack deployments get more and more mature, cloud architects increasingly face real-world issues with deploying production applications due to the lack of an enterprise-class, cloud-native load-balancing solution. The Avi Vantage Platform from Avi Networks is an elastic application services fabric that delivers an enterprise-class cloud-native application services solution that include load balancing, application analytics, predictive auto scaling, micro-segmentation, and security. This reference architecture document discusses how to deploy an enterprise-class OpenStack IaaS, at scale with Avi Vantage.

### Intended Audience

To understand this document, we assume the reader is familiar with: * Basic OpenStack functionality. (Find more information: https://docs.openstack.org) * The basics of load balancing and ADC. (Find more information: https://avinetworks.com/docs)

### Avi Vantage Architecture

The Avi Vantage Platform is built on software-defined architectural principles, and separates the data and control planes. From the way it is defined:

The distributed data plane: * Is also known as the forwarding plane. * Exposes virtual services to clients. * Proxies and load-balances back-end servers. * Processes all virtual service traffic. * Forwards traffic to the next hop along the path to the selected destination network according to control plane logic.

The central control plane: * Sets and controls all ADC functions. * Collects, collates and analyzes event and metrics data pertaining to traffic flowing through the data plane. * Serves as an ADC troubleshooting focal point. * Performs functions that don?t have a strict speed constraint, are less time-critical, and do not act upon client traffic.

The product components include: * Avi Service Engines (distributed data plane): Distributed load balancers (SEs) are deployed closest to the applications across multiple cloud infrastructures. The Avi Service Engines collect and send real-time application telemetry to the Avi Controller. * Avi Controller (central control plane): Central policy and management engine that analyzes the real-time telemetry collected from Avi Service Engines and presents in visual, actionable dashboards for administrators via an intuitive user interface and or CLI built on Avi's RESTful API. In this document "Controller" in the singular is used for simplicity, but is implemented as a cluster of 3 Controllers in production environments. Controller redundancy ensures HA for the control function and applies additional computational power for analysis of telemetry.

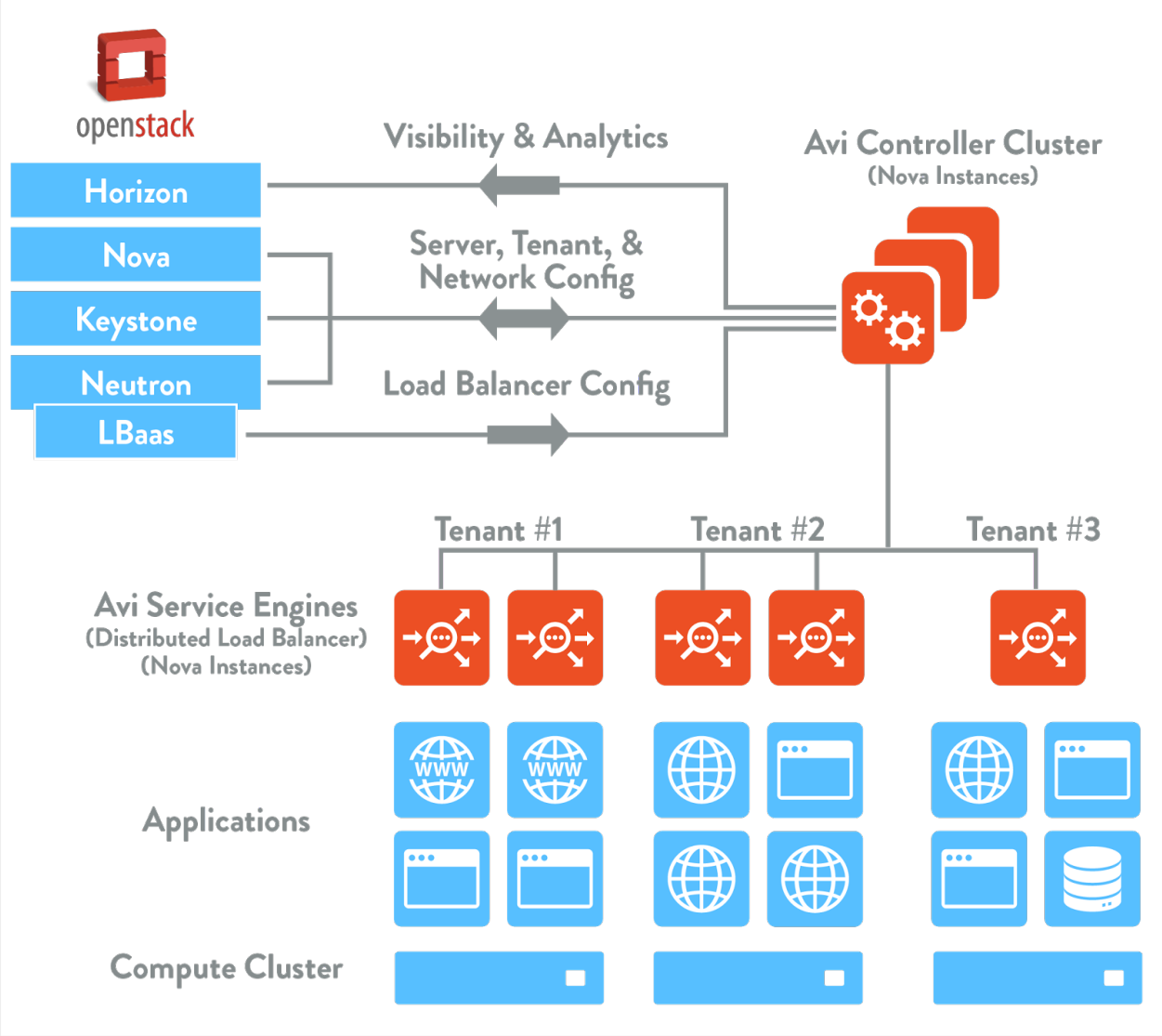



Figure 1. OpenStack deployment with Avi Vantage: Reference Architecture

The Avi Controller communicates with the OpenStack components listed below using standard OpenStack REST API calls to achieve the following: * Keystone: The Avi Controller communicates with Keystone to authenticate and authorize the users that invoke Avi REST API calls. * Glance: The Avi Controller communicates with Glance to upload and manage the image for distributed load balancers (aka Service Engines) * Nova: The Avi Controller communicates with Nova to create and manage the lifecycle of Avi Service Engines. * Neutron: The Avi Controller communicates with Neutron to create necessary ports and set up port properties for Service Engines to provide network services. * Heat: OpenStack administrators can optionally install the Avi Heat package on the Heat Engine servers to expose all Avi Controller API resource types for users to use in their heat templates. In contrast to LBaaS (v1 or v2) resource types, Avi Heat resource types expose significantly advanced features. More information on Heat is available here. * Horizon: OpenStack administrators can optionally install Avi Horizon Dashboard extensions to expose the full Avi UI directly embedded in the Horizon Dashboard. Users can then not only configure load balancer instances, but can also access the full analytics of their applications.

Furthermore, with Avi?s LBaaS driver installed on the Neutron API servers, the driver communicates the load balancer configurations to the Avi Controller using REST API calls.

To learn more about the functioning of these components, please refer to the OpenStack documentation. ### Interworking with other Ecosystems

Avi Vantage enables an OpenStack cloud to provide enterprise-class application services functionality with application analytics. Avi Vantage is fully supported by OpenStack distribution Havana release and above.

Built on software-defined architectural principles, Avi Vantage can integrate with software-defined networking (SDN) solutions to provide full-stack network and application services. Avi Controller communicates with SDN controllers such as Nuage VSP and Juniper Contrail to provide a complete TCP/IP stack.

## Advantages of Using Avi Vantage

A single Avi Controller cluster can manage 200 SEs. The Avi Controller has the intelligence to place virtual services on appropriate SEs. The distributed nature of Avi Vantage?s architecture offers unique high-availability features.

### Configuration and Analytics through OpenStack Horizon

Avi Vantage integrates with OpenStack Horizon and injects the Avi user interface within Horizon to deliver configuration and analytics capabilities to users. OpenStack networking and load balancing administrators can use the Avi dashboard from within Horizon to configure, scale and monitor OpenStack virtual services.

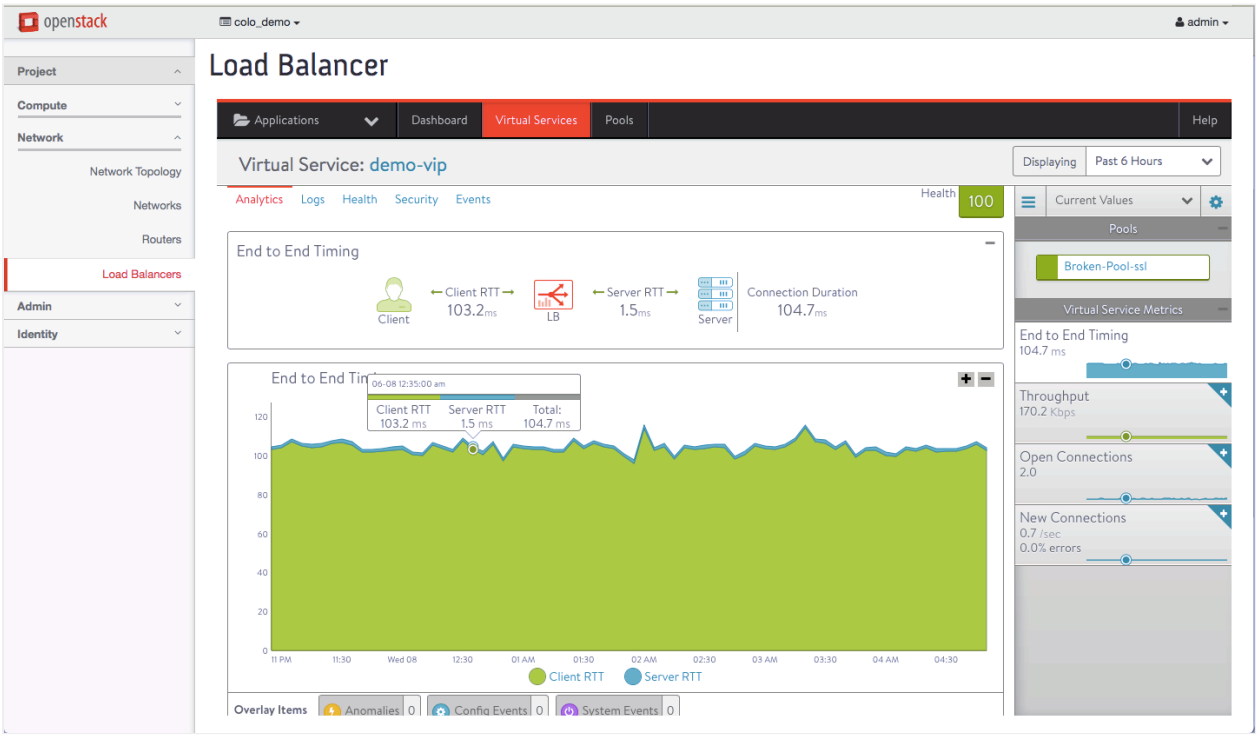
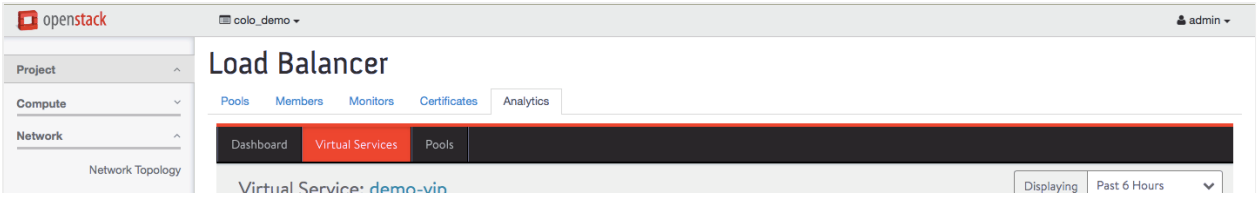


Figure 2. Graph of end-to-end timing of virtual service "demo-vip" transactions

Administrators can also configure Avi through Avi LBaaS to provide read-only analytics through Horizon.



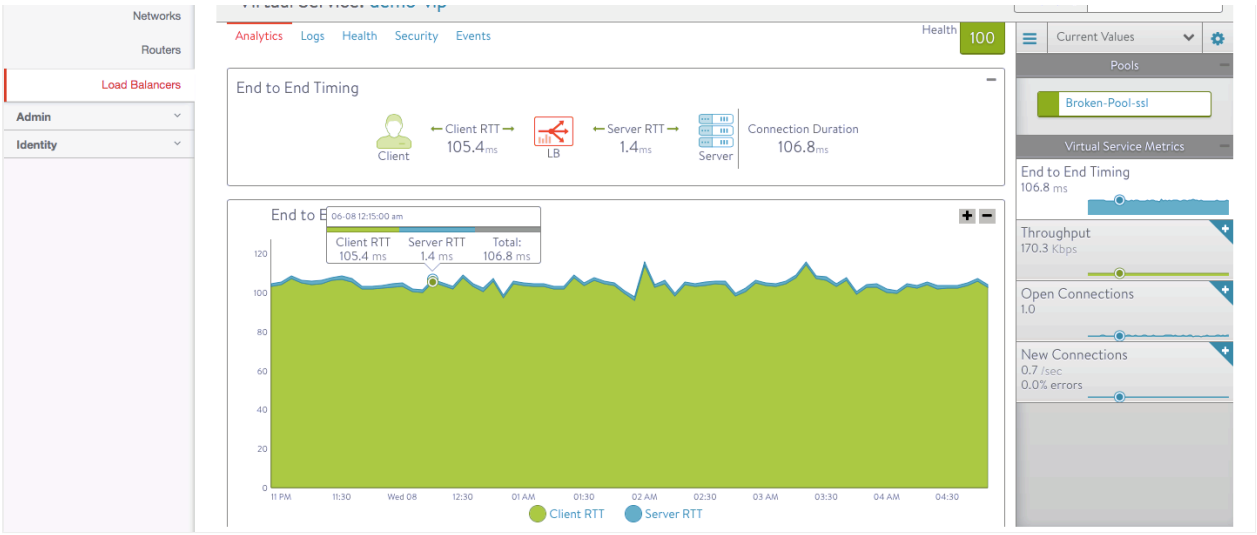



Figure 3. Metrics for VS "demo-vip" appear in the right frame of the window.

The instructions at https://github.com/avinetworks/avi-horizon-dashboard explain how to add the Horizon extension to an OpenStack installation.

**Per-application Load Balancer**

Avi?s ability to deploy hundreds of distributed load balancers makes it administratively simpler to dedicate a load balancer to each application. The resulting isolation reduces security concerns from neighboring apps and removes the noisy-neighbor problem. The feature can be configured by enabling the Per Application option within the Service Engine Group tab.

**Auto-healing Load Balancers**

When the Avi Controller detects an SE has gone down or faces some issues, it can communicate with OpenStack to dynamically spin up Service Engines to recover lost load balancing capacity. The Avi Controller can move virtual services from the problematic SE to spare capacity on the replacement SE or others having spare capacity. Auto-healing provides an easy to administer, highly available environment in which applications can be assured of the capacity they need with little to no human intervention.

**Elastic scaling of load balancers**

The Avi Controller can automatically scale load balancing capacity up or down, depending upon current virtual service requirements. The decision to autoscale is based on SE CPU usage. If a given Service Engine?s CPU utilization exceeds 80% (MAX_ALLOWED_CPU), the Avi Controller will spin up one more SE instance and perform a scale-out operation. On the other hand, if Service Engine CPU utilization drops below 30% (MIN_ALLOWED_CPU), the Avi Controller will spin down one of the instances of Service Engine and perform a scale-in operation.

# Design Considerations

## Requirements

The System Requirements: Ecosystem article lists requirements for these entities: Avi Controller, OpenStack, Neutron service, Neutron extension for allowed-address-pair, Avi LBaaS driver, Keystone, Avi SSL extension for OpenStack, and Horizon.

The Avi Vantage image is available as a Qcow2 or raw image of the Avi Controller at www.avinetworks.com/portal. SE software is embedded within the Avi Controller image and therefore requires no separate installation.

# Solution Overview

## Single-Arm v/s Double-Arm Deployments

### Single-Arm ADC

When virtual IP and back-end pool members are located on a single network, this topology is termed a single-arm ADC.

### Double-Arm ADC

When virtual-IP and back-end server pool members are located on a two separate networks, this topology is termed a double-arm ADC.

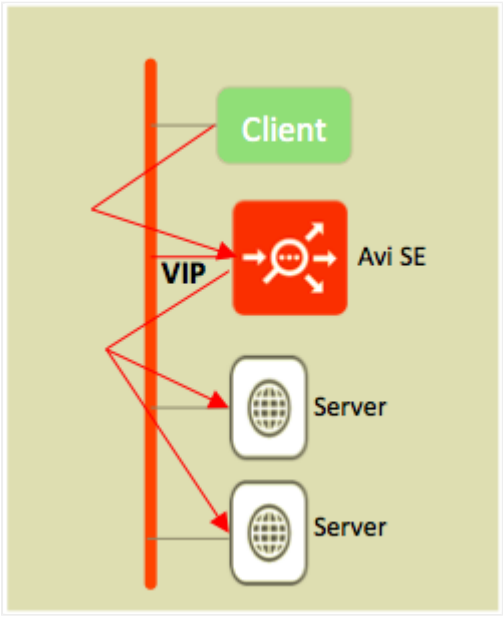



Figure 4a. Single-arm ADC

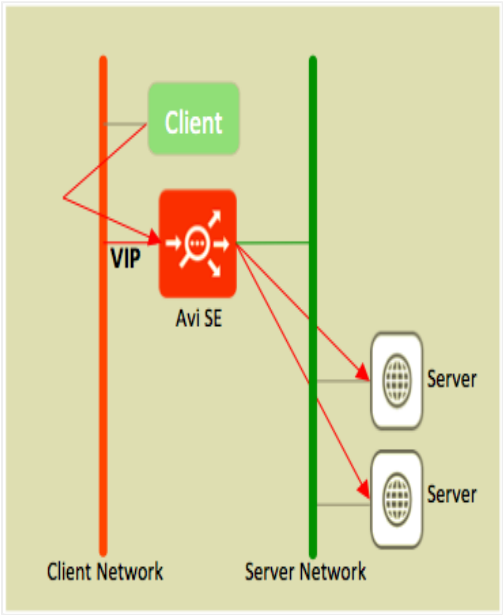



Figure 4b. Double-arm ADC

Note: Both single arm and double arm ADC are supported.

## North-South Traffic Flow

When clients outside the OpenStack cloud try to access a virtual IP address within an OpenStack cloud, it is termed north-south traffic (and hence, we refer to a north-south VIP). Since a north-south VIP is required for access from outside the OpenStack cloud, it requires a floating IP in addition to the internal virtual IP. Avi Vantage supports configuring a ?floating VIP? in the virtual service configuration. The Avi Controller then communicates with OpenStack Nova to assign an allocated floating IP to the virtual IP address.

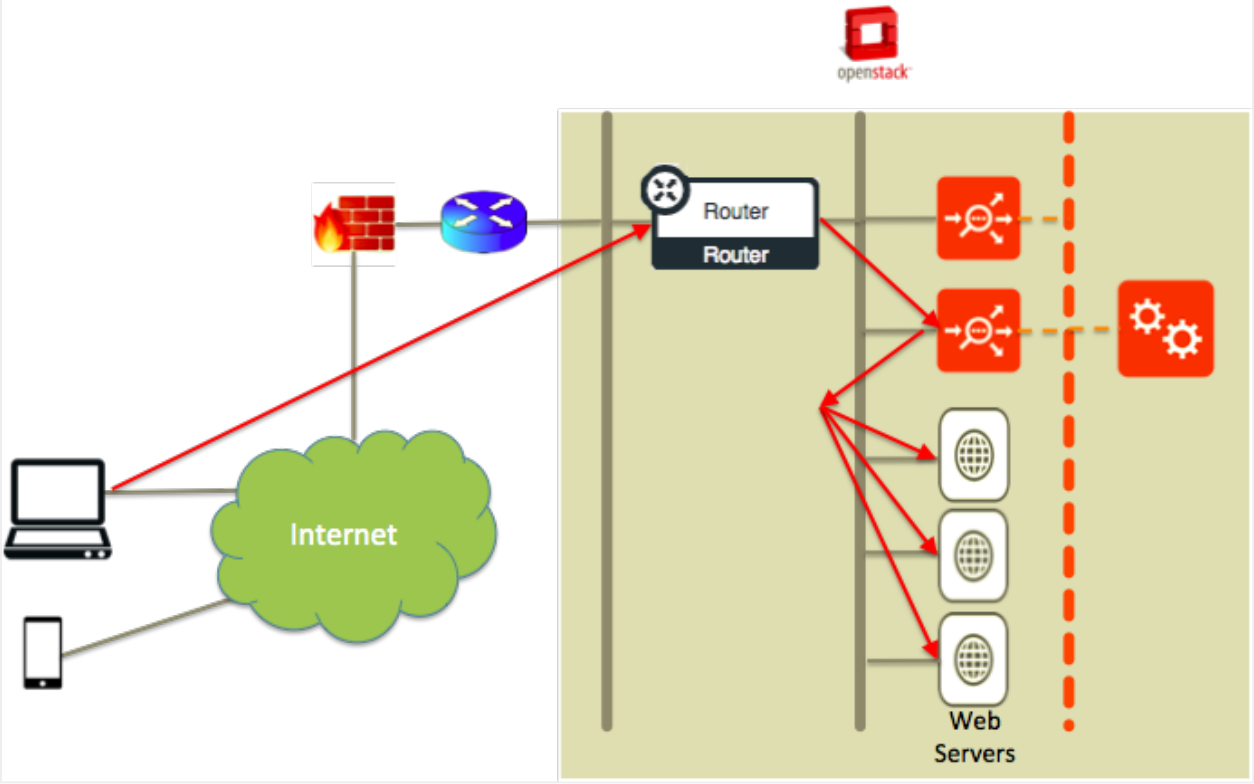
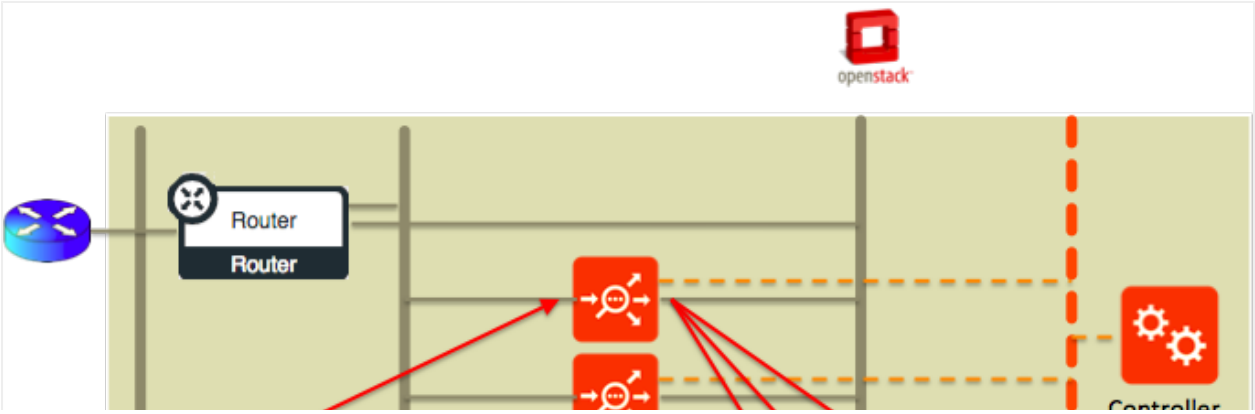



Figure 5. North-south traffic flow

## East-West Traffic Flow

When clients within the OpenStack cloud access virtual IP addresses within the OpenStack Cloud, it is termed east-west traffic (and hence, we refer to an east-west VIP). Since an east-west VIP is only accessed from within the OpenStack cloud, it does not require a floating IP; it can be accessed using an internal VIP.



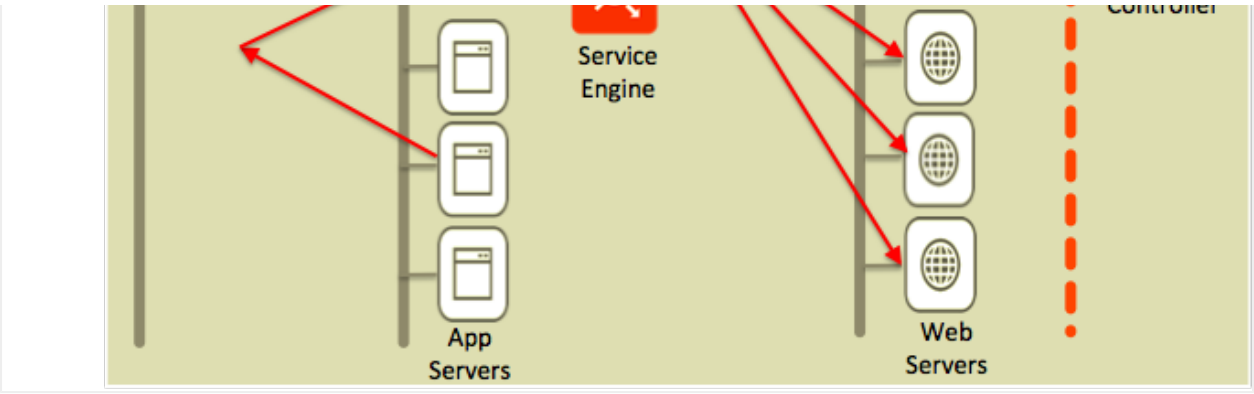



Figure 6. East-west traffic flow

## Scale-Out Virtual Services

Avi Vantage enables a single virtual service to be scaled out across multiple Service Engines. That means a single virtual IP can be served by multiple SE VMs.

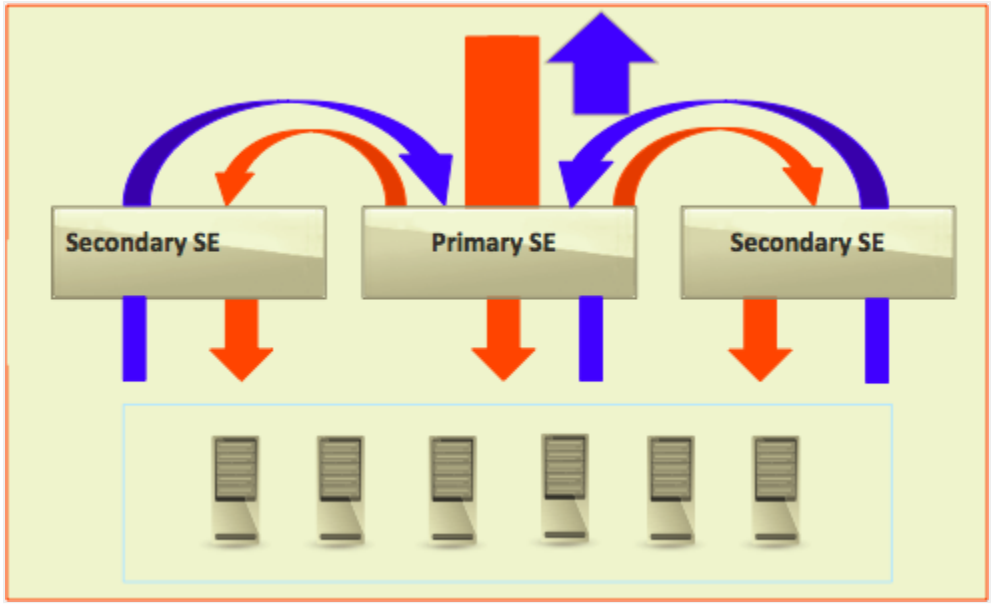



Figure 7. Scaling out virtual services The primary Service Engine directs some connections to each of the secondary Service Engine to share CPU-intensive SSL offload or memory-intensive connection setup. These packets are tunneled as MAC-in-MAC encapsulated packets with a custom EtherType. The secondary Service Engines have the option to send their response packets either directly to the client using OpenStack security group that requires tunnel mode.

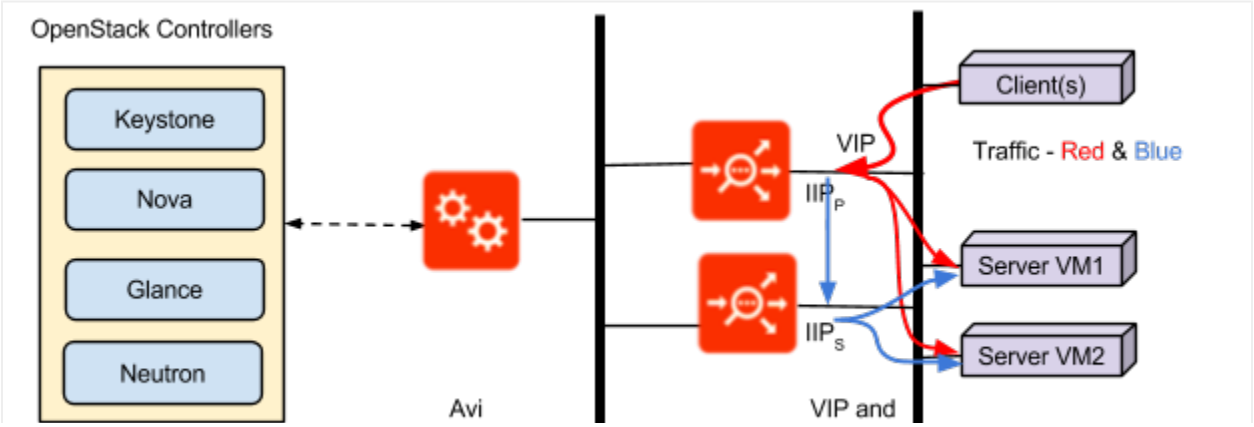



Figure 8. Scaling out virtual service within OpenStack

### Neutron ML2 Plugin

ARP is broadcast on the L2 network, so the ARP requests for the VIP address are received by the SE. However, outgoing packets with a VIP address are disallowed by default. Avi uses the allowed-address-pairs extension to let Neutron know it intends to use a VIP address on the SE even though the interface IP address is different. Once ARP is set on the correct Neutron port, the ML2 plugin allows traffic to go out with the VIP address.

> *A note about active/active SEs: Unfortunately, routing server responses through secondary SEs is not possible in the current implementation of Neutron ML2, even if ARP is set. This is because the ML2 agent uses `iptables` with the `nf_conntrack` module, which does not allow any out-of-sequence TCP packets. Note that the incoming packets are received as tunneled packets and hence, conntrack does not have the information about the connection on the incoming side. Avi tunnels the response packets back to the primary SE. If the Neutron port-security extension is supported, then responses via the secondary SEs is supported.*

## Planning Deployment

### Prerequisites

Refer to the following link for the same :

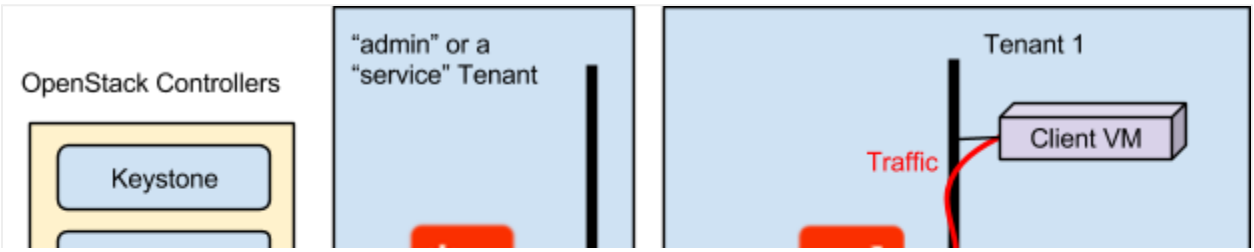
https://kb.avinetworks.com/installing-avi-vantage-for-openstack/#Deployment_Prerequisites ### Avi Deployment Models

Avi Vantage's basis on virtual-machine architecture results in high flexibility in terms of deployment. The Avi Controller VM can be instantiated in any tenant context. The Avi Service Engine (SE) VMs handling the traffic of a tenant?s applications can either be instantiated in the tenant context (and hence, be counted against the tenant?s resource quota) or can be instantiated in a separate ?LBaaS? tenant context, specifically created for tracking resources used for providing the load-balancing service. In the latter mode, it is further possible to share a single SE for handling traffic of applications across different tenants or always segregate the traffic of different tenants to different sets of SEs to ensure strong tenant isolation.

The following modes detail the most common deployment options on the Avi Vantage Platform.

**Mode 1: Avi with ?admin? privilege; SE in tenant?s context**

- Admin launches an Avi Controller VM
- The Avi Controller VM can reside in any tenant context. Recommended: Create it in admin context (or create a new LBaaS service tenant and create it under that tenant).
- This VM needs to have its first interface in a network set aside for management of Service Engines.
- Any OpenStack user can then log into the Avi Controller, but they are restricted to access their own tenant context objects (if any).
- A user can create virtual services using the REST API or UI. In either case, the Avi Controller creates SEs, either in that user?s project or in a specific configured LBaaS project.



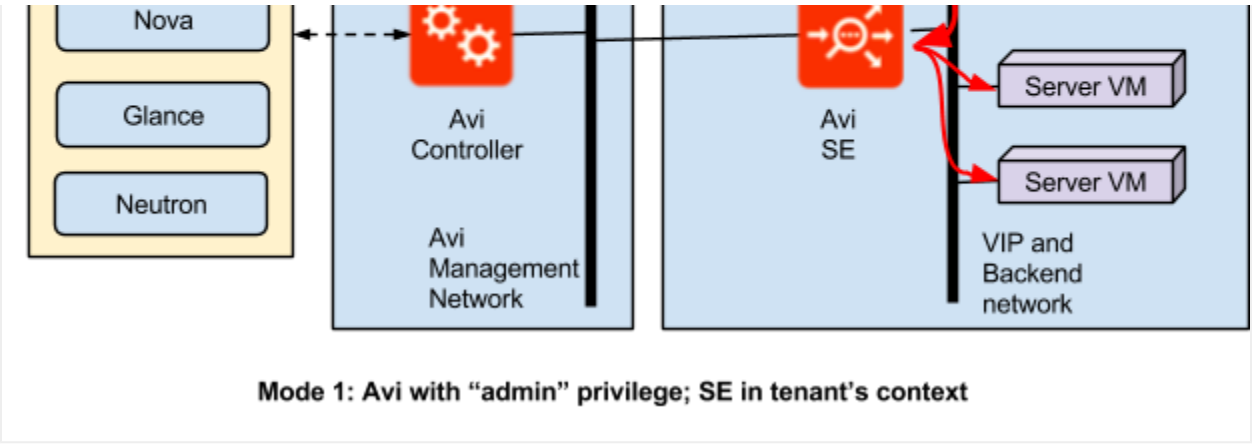



Figure 9. Avi with "admin" privilege; SE in tenant's context

**Mode 2: Avi with ?admin? privilege; SE in a ?service? tenant?s context**

In this method, there are two possible modes: 1. Exclusive SEs: Each tenant has his own set of Service Engines 2. Shared SEs: Service Engines shared across the tenants

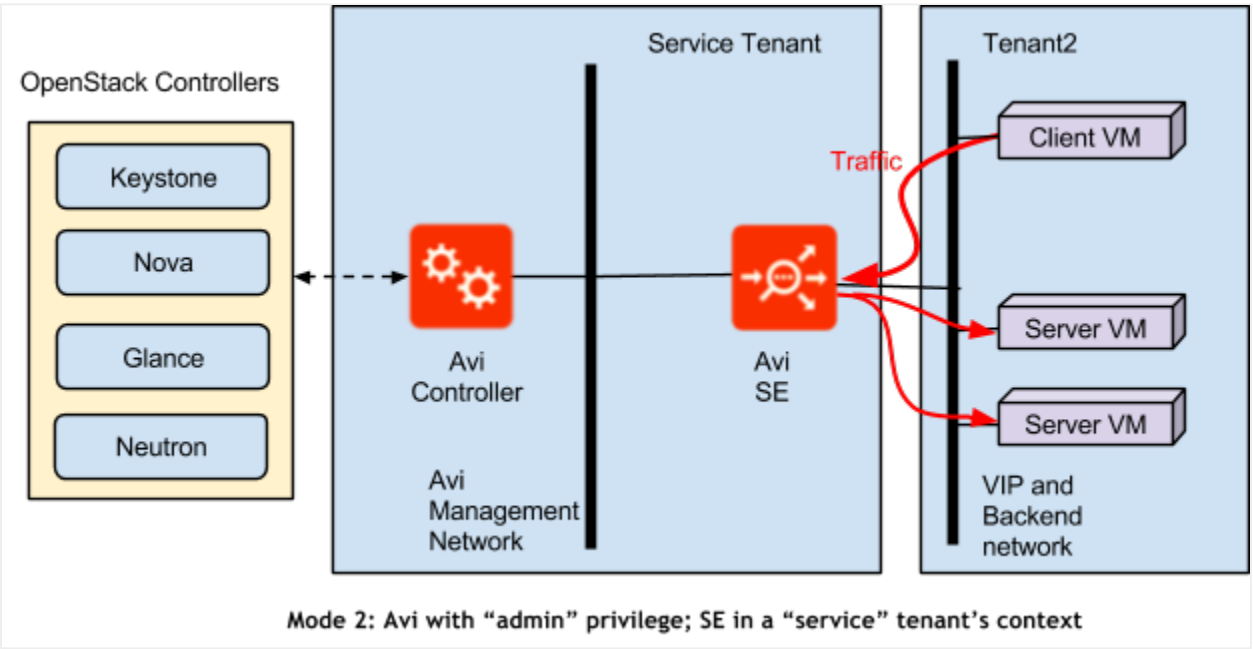



Figure 10. Avi with "admin" privilege; SE in tenant's context

**Mode 3: Avi with non-admin privilege; everything in a single tenant**

The Avi Controller and Service Engine VMs all are instantiated in a single tenant?s context. The Controller uses the OpenStack APIs to manage the lifecycle of Service Engines automatically. This requires that the Avi Controller VM be able to reach the OpenStack controller(s). Only users of that tenant can create virtual services. This mode is ideal for testing.

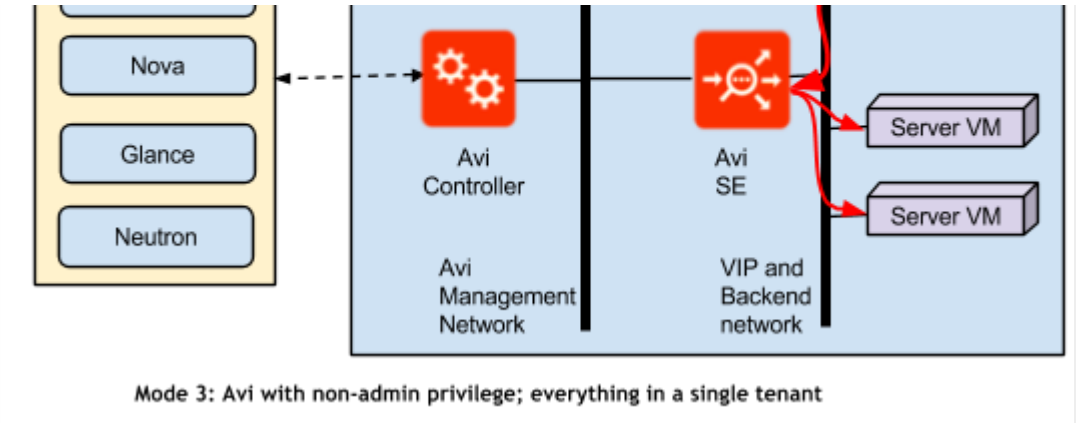



Figure 11. Avi with non-admin privilege; everything in a single tenant

## Avi Controller Design Considerations

The Avi Vantage Platform is a distributed system with the Controller serving as the master of multiple Service Engine slaves. The Avi Controller's control plane functions include defining the configuration of all SEs and the forwarding those configs to them. Additionally, the Avi Controller manages all analytics-related information, which includes virtual service metrics, log analytics, etc.

### Controller Sizing

During deployment of an Avi Controller, the system capacity of the Avi Controller can be specified based on allocations of the following system resources: * CPUs * Memory (RAM) * Disk

Refer to the Avi Controller Sizing article for more details.

### Control Plane High Availability

To ensure the highest level of uptime for a site, including through Avi Vantage software upgrades, careful consideration must be made to ensure the availability for both Avi Controllers and Avi Service Engines. * Avi Controller HA: Provides node-level redundancy for Avi Controllers. A single Avi Controller is deployed as the leader node, and then two additional Avi Controllers are added as follower nodes. [Note: Some older Avi literature may refer to the leader and followers as primary and secondaries, respectively.] An overview of the same is available here.

- Avi SE HA: Provides SE-level redundancy within an SE group. If an SE within the group fails, HA heals the failure and compensates for the reduced site capacity. Typically, this consists of spinning up a new SE to take the place of the one that failed. Avi Vantage supports two Avi Service Engine (SE) elastic HA modes, which combine scale-out performance as well as high availability:
- N+M mode (the default)
- Active/active
- Legacy mode enables a smooth migration from legacy appliance-based load balancer pairs.

HA for Avi SEs and HA for Avi Controllers are separate features and are configured separately. HA for Avi Controllers is a system administration setting. HA for Avi SEs is an SE group setting.

To ensure application availability in the presence of whole-site failures, Avi recommends use of Avi GSLB, explained below.

## GSLB

Global server loading balancing (GSLB) is the act of balancing an application's load across instances of the application that have been deployed to multiple locations (typically, multiple data centers and/or public clouds). Application load at any one of those locations is usually managed by a "local" load balancer, which could be Avi Vantage or a third-party ADC solution.

GSLB is usually implemented to achieve one or more of the following goals for an application:

- Provide optimal application experience to users/clients who are in geographically distributed areas
- Offer resilience to loss of a data center or a network connection
- Perform non-disruptive migration to or addition of another data center

To achieve these goals, Avi GSLB performs the following functions: * Avi DNS for GSLB runs on designated Service Engines, as few as one in one "active" Avi GSLB site, but possibly several, one for each active site. When a client (typically a browser) performs a DNS query on fully qualified domain names (FQDNs), Avi DNS responds with the IP address (VIP) of the optimal application instance. The optimal address can and will change based on the GSLB load balancing algorithm, health of the application instances, and location of the clients. * The Avi DNS for GSLB monitors the health of the virtual services so that DNS responses point to the best location (i.e., ruling out unhealthy ones). * Avi GSLB synchronizes configuration and state across GSLB sites, so that GSLB functionality can continue despite certain failures. * Avi GSLB can integrate multiple OpenStack clouds and/or OpenStack and non-OpenStack environments (including public clouds) under a single solution. * Avi GSLB operates in a fashion that is totally opaque to end-users.

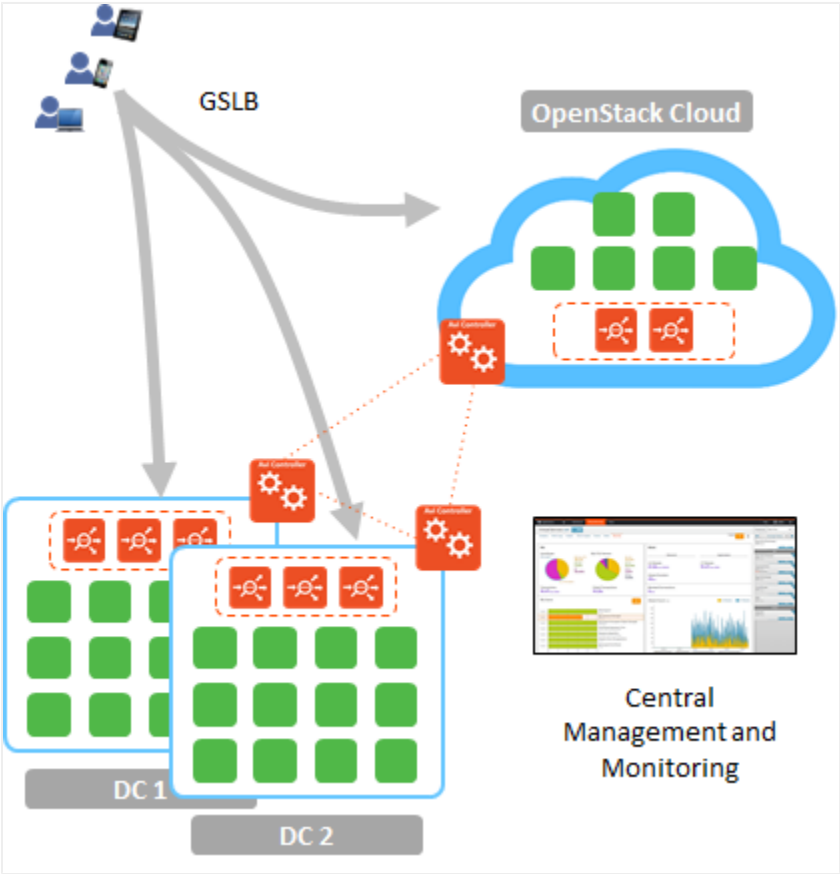



Figure 12. Avi GSLB supports multi-cloud, geographical distributed applications

## Avi Service Engine Design Considerations

### Service Engine Sizing

Avi publishes minimum and recommended resource requirements for new Avi Service Engines. These are somewhat abstract numbers though, as network and application traffic may vary dramatically. Refer to the Sizing Service Engines article for more details.

### Data Plane High Availability

#### Elastic HA Active/Active Mode

In this mode Avi places each new virtual service on a minimum of 2 Avi SEs within the SE group. This provides both better scale and faster failover times. If one of the Avi SEs fails, capacity is temporarily reduced while another Avi SE is configured or a new Avi SE is created to backfill the down Avi SE. This elastic mesh ensures maximum availability for virtual services while also ensuring that all Avi SEs are utilized.

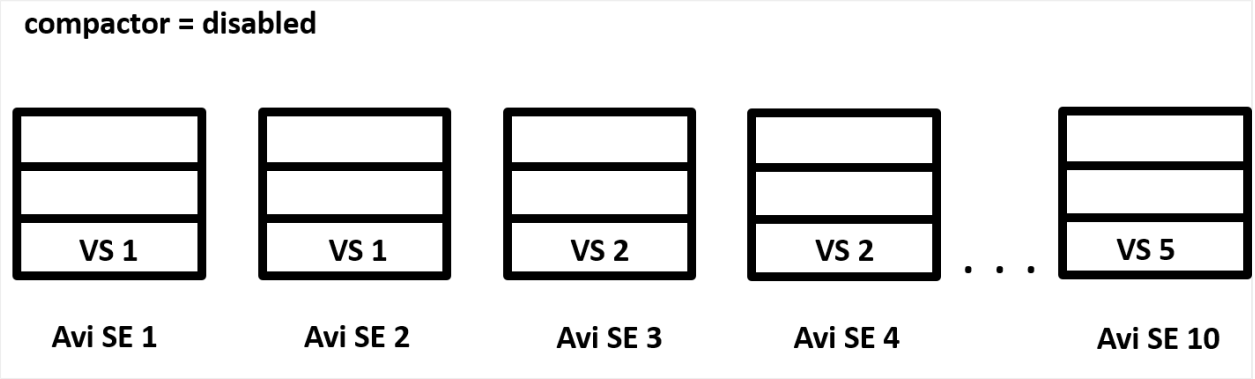



Figure 13. Elastic HA active/active mode

#### Elastic HA N+M Mode

In this mode, the Avi spins up as many SEs as needed to take the load of all virtual services. Additionally, Avi spins up a configured number of extra SEs to ensure spare capacity equivalent to M SEs.

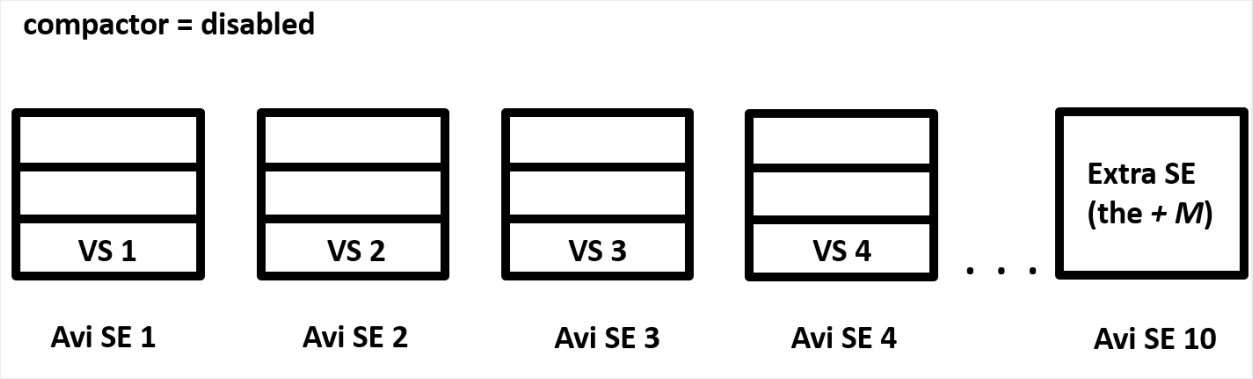



Figure 14. Elastic HA N+M mode

#### Compactor Option

Each elastic HA mode supports the compactor option. When the compactor option is disabled, virtual services are distributed evenly among the SEs within the group, except for the extra SEs in N + M mode (as shown above). When the compactor option is enabled, Avi Vantage fills up the SEs within the SE group one at a time.

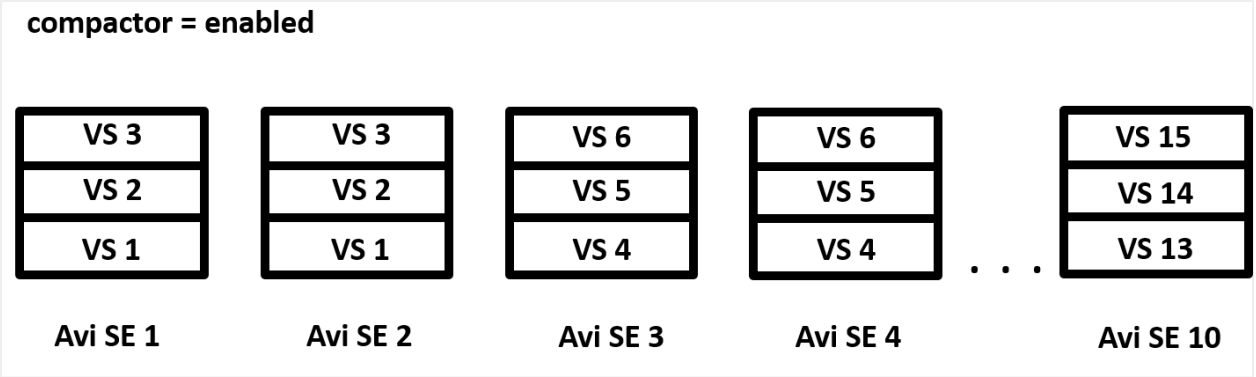



Figure 15. Elastic HA active/active mode with compactor enabled

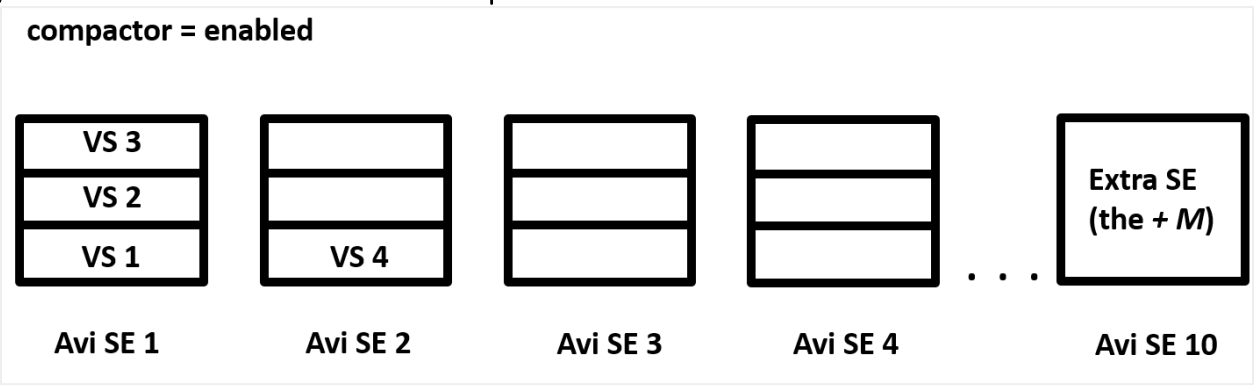



Figure 16. Elastic HA N+M mode with compactor enabled

#### Legacy HA Mode

In legacy HA mode, two Avi SEs are configured as an active-standby pair. The active SE load balances all the traffic for a particular VS, while the other SE in the pair is the standby, and receives no traffic for the VS. Upon a failover from the active SE, the former standby takes over the traffic. Upon failover, the standby also takes ownership of all other virtual services that had been active on the failed SE. This includes floating addresses such as VIP, SNAT-IP, and so on. Default failover time is 9 seconds, but may be decreased/increased by setting some parameters that govern fault detection.

## Configuration, Operations, Maintenance

#### Configuration

The Avi Vantage Platform with OpenStack integration supports ADC configuration using 2 methods.

1. Avi APIs, GUI or CLI

- The load balancing administrator has the option to configure VIP, pool, members and all associated objects directly through the Avi API, GUI or CLI.
- The Avi solution also integrates with OpenStack Horizon to support ADC configuration through ?Projects > Networks > Load Balancers?.

- The ADC administrator has the option to provide read-only access to application owners for analytics OR write-access to everyone to create/modify their own applications.

2. Avi LBaaS API

- The administrator can optionally install the Avi?s LBaaS plugin on their Neutron server node(s).
- Users can then use the LBaaS API to create pools and monitors, add members to the pools, associate monitors with pools, and associate VIPs to pools.
- Users can update the advanced properties of the objects created via the LBaaS API on the Avi Controller using Avi?s REST API or UI. However, certain basic changes to the objects (creation, deletion, or updating one of the fields that is accessible via LBaaS plugin) will not persist if done via Avi?s API, as they will be overwritten by the Avi LBaaS plugin, during a periodic synchronization cycle.
- The Avi LBaaS driver is configured with the Avi Controller?s admin credentials and uses those credentials to create LB instances on Avi on behalf of users of LBaaS API, in the correct tenant context.

**Operations**

Debugging or operationalizing Avi is straightforward. Avi Vantage supports * Events * Alerts * Services * Traffic Capture

**Events**

Events are used throughout Avi Vantage to provide a history of relevant changes that have occurred. Events are a permanent record, and can be used to generate alerts which can take action on the event. In the UI, events may be viewed within the context of specific objects, such as a virtual service, a pool, or a server. For more information, read the Events Overview article.

**Alerts**

In its most generic form, alerts are a construct that informs administrators of significant events within Avi Vantage. In addition to triggering based on system events, alerts may also be triggered based on one or more of the 200+ metrics Avi Vantage is tracking. For more information, read the Alerts article.

**Services**

Avi Vantage can define alert actions using various mechanisms. Users can configure a remote Syslog server, Splunk, email, SNMP traps, etc. to log appropriate events and logs into an external audit system.

**Traffic Capture**

Most troubleshooting of connection or traffic data may be done quickly via virtual services logs. However, some troubleshooting may require full visibility into the packet transmission. Avi Vantage provides a packet capture feature, which runs `tcpdump` against a designated virtual service. The packet capture is done on all Service Engines that may be hosting the virtual service, and followed by collation of the individual captures into a single capture of application traffic. For more information, read the Traffic Capture article.

**Maintenance**

1. Backup

You can export the Controller configuration from the Avi Controller. For more information, read the Backup and Restore of Avi Vantage Configuration article.

2. Improving Controller GUI Performance

Avi recommends dedicating CPU and memory resources to the Avi Controller. If it is stretched on resources, users face sluggishness while accessing the GUI and/or the CLI. You can verify Avi Controller resources by logging into bash prompt of Avi Controller and running the `htop` command. If you see excessive memory swapping, increase the Controller?s footprint.