WHITE PAPER–JULY 2020

Deliver Elastic Kubernetes Ingress Controller and Services

A single platform for consolidated traffic management, security, and observability

vmware

Table of Contents

Introduction

Kubernetes/OpenShift offers an excellent automated application deployment platform for container-based workloads. However application services such as traffic management, load balancing within a cluster and across clusters/regions, service discovery, monitoring/analytics, and application security are critical for modern application infrastructure. Enterprises require a scalable, real-world-tested, and robust services fabric to deploy microservices applications in Kubernetes/OpenShift clusters ready for production environments. This whitepaper provides an overview of the requirements for such application services and explains how the VMware NSX® Advanced Load Balancer™ (by Avi Networks) provides a proven solution to deploy container-based workloads in production environments using Kubernetes/OpenShift clusters. You will learn about the following:

- Ingress Controller
- Multi-cluster, multi-site container support
- Dynamic service discovery
- Application performance monitoring and analytics
- Traffic management local and global load balancing
- Advanced network and application security
- Integrated DNS and IPAM
- Performance based elastic autoscaling

Challenges with Application Services for Kubernetes

Common application services, such as load balancing, network performance monitoring, and application security, that are available in traditional applications often need to be implemented or approached differently in container-based applications. Here are some of the challenges in deploying container-based applications.

Multiple discrete solutions

Modern application architectures based on microservices have made appliance-based load balancing solutions obsolete. Traditional hardware/virtual load balancers or open source tools are not equipped to support the north-south ingress services, do not support application autoscaling, and lack the native integration with peripheral services such as DNS, IPAM and web application firewall (WAF).

Complex operations

With disparate solutions, IT faces more complex operations in managing and troubleshooting multiple independent components from different vendors.

Lack of observability

End-to-end visibility is especially important with container-based applications. Application developers and operations teams alike need to be able to view the interactions between the peripheral services and the container services to identify erroneous interactions, security violations, and potential latencies.

Partial automation

Application and networking services need to be API-driven and programmable without the constraints of hardware appliances. Multi-vendor solutions can limit their flexibility and portability across environments. Multi-vendor solutions also necessitate in depth scripting knowledge for different products to provide only partial automation, if any at all, leading to compromising between feature, automation, and scale. Therefore, it is necessary to have consolidated Kubernetes services from a single platform.

Application Services Requirements for Container-based Environments

Traffic Management - Local Load Balancing: Local load balancers or application delivery controllers (ADCs) need to provide application networking services such as load balancing, health monitoring, TLS/SSL offload, session persistence, content/URL switching, and content modification.

Traffic Management - Global Load Balancing: Global load balancing directs clients to the appropriate site/region based on several criteria including availability, locality of the user to the site, site persistence, site load, etc.

Service Discovery: Maps service host/domain names to their Virtual IP Addresses where they can be accessed.

Monitoring/Analytics: Enterprise applications deployed in production require constant monitoring and alerting based on application and network performance, health, and security.

Security: Enterprise-class secure applications require TLS/SSL cert management, microservice-based network security policies that control application access, DDoS protection/mitigation, and web application firewall (WAF).

Avi Consolidates Services for Kubernetes

The VMware NSX® Advanced Load Balancer™ (by Avi Networks) for Kubernetes extends applications services such as container ingress seamlessly to cloud-native applications in Kubernetes and OpenShift environments through a single platform a with a modern, distributed architecture converging application services for Kubernetes/OpenShift.

- **Integrated solution**
  In addition to container ingress services Avi Kubernetes Ingress Services offers advanced L4-L7 services including global server load balancing (GSLB), DNS/IPAM, application security, WAF and analytics. It helps deliver applications consistently in multi-cloud environments with industry's only complete L2-L7 networking and security stack.
- **Operational simplicity**
  Centralized policies and full lifecycle automation eliminate manual tasks providing administrators with central control, self-service application delivery automation and operational consistency.
- **Rich observability**
  Real-time telemetry with application insights across all components through closed-loop analytics and deep machine learning it provides holistic end-to-end insights, across the network, end users, security, and real-time application performance monitoring.
- **Cloud-native automation with elasticity**
  Elastic autoscaling based on closed-loop analytics and decision automation across on-premises data centers and public clouds, including VMware, OpenStack, AWS, Azure, and Google Cloud Platform.

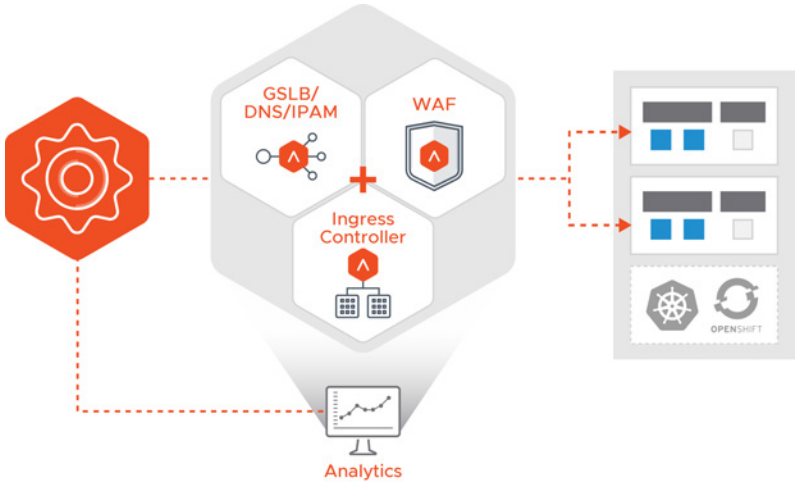


FIGURE 1: Avi Kubernetes Ingress Services

Consolidated Services Fabric for Container Ingress

VMware NSX Advanced Load Balancer integrates with container orchestration platforms such as Kubernetes/OpenShift on virtual machines and bare metal servers across on-prem, multi-cloud, multi-cluster, and multi-region environments. To deliver comprehensive container services for both traditional and cloud-native applications, Avi Kubernetes Services is optimized for North-South (ingress controller) traffic management including local and global server load balancing (GSLB), performance monitoring, dynamic service discovery, application security such as web application firewall (WAF), and DNS/IPAM management. Combining L4 through L7 load balancing, GSLB, DNS/IPAM management, and security functionalities in a single solution, Kubernetes Ingress Services provides operational consistency regardless of which on-prem, private-cloud or public-cloud environment the Kubernetes cluster is running on.

Kubernetes Ingress Services is based on a software-defined, distributed architecture with four major components:

Avi Controller: The Avi Controller is the central management component of the Avi architecture providing all control plane functionality of infrastructure orchestration, centralized management, and the analytics dashboard. In Kubernetes environments, the Avi Controller is in lock steps with Kubernetes primaries in a scalable manner. It can be deployed anywhere as long as connectivity and latency requirements are satisfied.

Avi Service Engines: In Kubernetes environments, the SEs are deployed external to the cluster and provide services such as load balancing, GSLB, analytics, DNS and WAF in the data plane.

Avi Kubernetes Operator: AKO is a pod running in Kubernetes clusters that provides communications with Kubernetes primaries to provide configuration. AKO remains in sync with the required Kubernetes objects and calls the Avi Controller APIs to deploy the Ingress Services via the Avi Service Engines. AKO is deployed via HELM.

Avi Multi-Kubernetes Operator: The AMKO facilitates multi-cluster application deployment extending application ingress controllers across multiple clusters. AMKO calls Avi APIs for Avi Controller to create GSLB services on the leader cluster which synchronizes with all follower clusters.

Kubernetes Ingress Services provides operational consistency regardless of which on-prem, private-cloud or public-cloud environment the Kubernetes cluster is running on (see Figure 2).

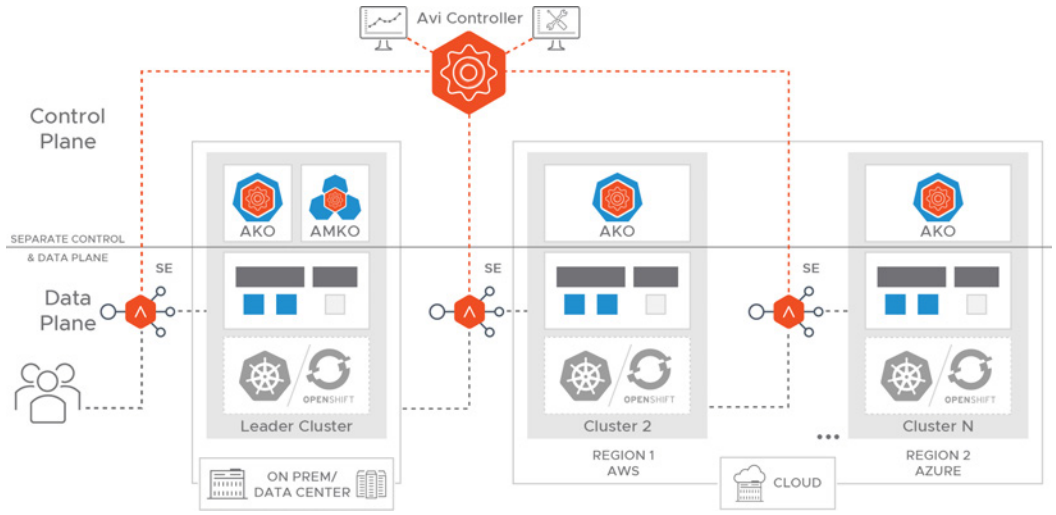


Figure 2: Application Services Architecture for Containers

Kubernetes/OpenShift Multi-Cluster Integration using Avi Kubernetes Operator

Avi Kubernetes Ingress Services can be used for integration with multiple Kubernetes clusters, with each cluster running its own instance of AKO. AKO is a pod running in Kubernetes clusters that provides communications with Kubernetes primaries to provide configuration. To extend applications across Multi Region and Multi Availability Zone deployments AMKO is required.

AKO synchronizes required Kubernetes objects and calls the Avi Controller APIs to deploy and configure the Ingress Services via the Avi Service Engines. Clusters are separated on SEs, which are deployed outside the cluster in the Data Plane by using VRF Contexts. Automated IPAM and DNS functionality is handled by the Avi Controller.

AKO synchronizes required Kubernetes objects and calls the Avi Controller APIs to deploy and configure the Ingress Services via the Avi Service Engines (SEE FIGURE 3).

Avi Kubernetes Operator

- Provides Ingress-Controller and Avi configuration functionality
- Translates and synchronizes Kubernetes objects to Avi Controller APIs
- Runs as a Pod in the Kubernetes/OpenShift Cluster

Avi Controller

- Automates the deployment and management of the Service Engines
- Provides centralized Analytics and Observability

Avi Service Engines (external)

- Hosts the Virtual-Services for Kubernetes/OpenShift Ingresses/Routes
- Handles Virtual-Service data plane traffic

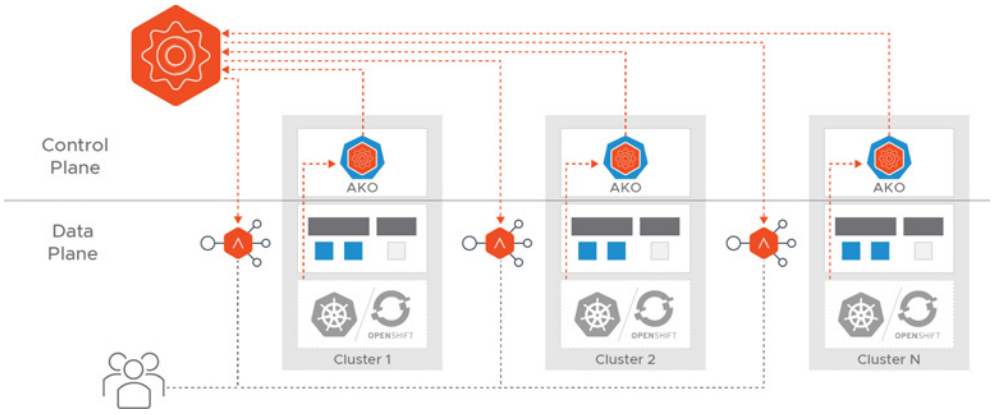


Figure 3: AKO Service Architecture

AKO Deployment Workflow

AMKO calls the Avi Controller APIs to deploy and configure GSLB services and DNS/IPAM settings which tie together the virtual services created by AKO on the leader and follower Kubernetes/OpenShift cluster sites (see Figure 4).

Infrastructure deployment workflow

1. The Avi Controller is deployed on the underlying Infrastructure, e.g. VMware vCenter
2. The underlying infrastructure parameters are configured via the Avi 'Cloud Object' on the Avi Controller
3. The Avi Controller communicates with the underlying infrastructure to orchestrates the lifecycle of the Service Engines

Per Kubernetes/OpenShift cluster workflow

4. Basic AKO parameters are configured via Helm Chart (Controller IP, Authentication, VRF etc.)
5. AKO is deployed in the cluster using Helm
6. AKO establishes a connection with Avi Controller

Application deployment workflow

7. Admin configures Ingresses/Routes and CRDs
8. AKO gets these objects from the Kubernetes/OpenShift primaries
9. AKO translates Ingresses/Routes and calls Avi APIs
10. Avi Controller allocates an IP address from IPAM and publishes FQDN to DNS
11. Avi Service Engines host Virtual Services for Ingresses/Routes

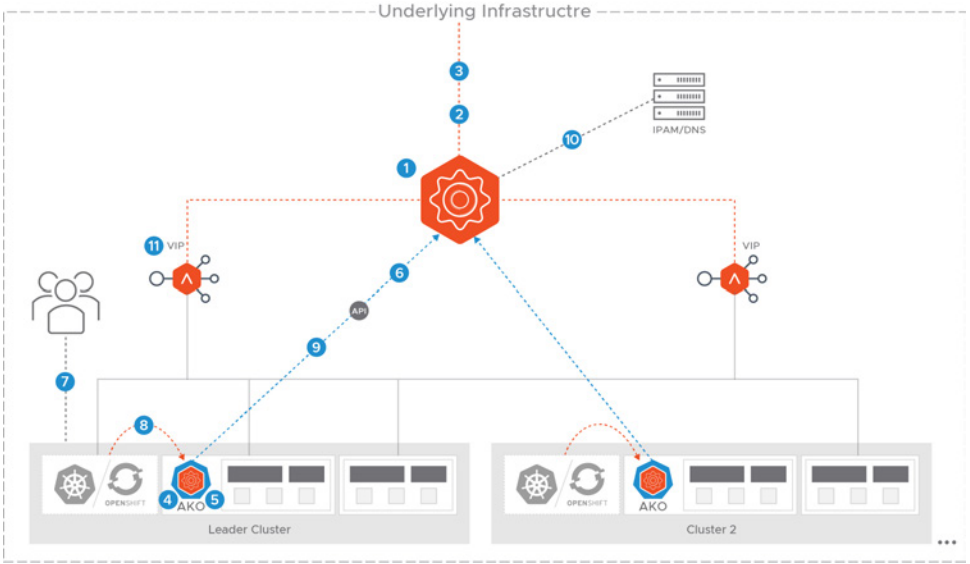


Figure 4: AKO Infrastructure Deployment Workflow

Comprehensive Multi-cluster Traffic Management

Container Ingress provides enterprise-class Kubernetes/OpenShift ingress traffic management across multi-cluster, multi-region, and multi-cloud environments, including local and global server load balancing (GSLB), web application firewall (WAF) and performance monitoring, Avi integrates seamlessly with Kubernetes for container and microservices orchestration and security.

Kubernetes/OpenShift GSLB Integration using Avi Multi-Cluster Kubernetes Operator

Avi Multi-Cluster Kubernetes Operator (AMKO)

- Facilitates Multi-Cluster Application deployment
- Runs as a Pod in the Kubernetes/OpenShift cluster
- Extends Application Ingresses across multiple clusters for application deployments

AMKO is an Avi pod running in the Kubernetes/OpenShift GSLB leader cluster and in conjunction with AKO, AMKO facilitates multi-cluster application deployment, mapping the same application deployed on multiple clusters to a single GSLB service, extending application ingresses across Multi Region and Multi Availability Zone deployments.

AMKO calls the Avi Controller APIs to deploy and configure GSLB services and DNS/IPAM settings which tie together the virtual services created by AKO on the leader and follower Kubernetes/OpenShift cluster sites.

Global Service Load Balancing (GSLB)

When enterprise applications are deployed across multiple data centers and/or private/public cloud regions, Avi's provides GSLB services by returning the applicable Virtual IP address for an application based on its DNS query. For best performance and optimal user experience, Avi's GSLB services use a combination of geo-location, site persistence, and availability to direct users to the appropriate site/region's VIP. (SEE FIGURE 5).

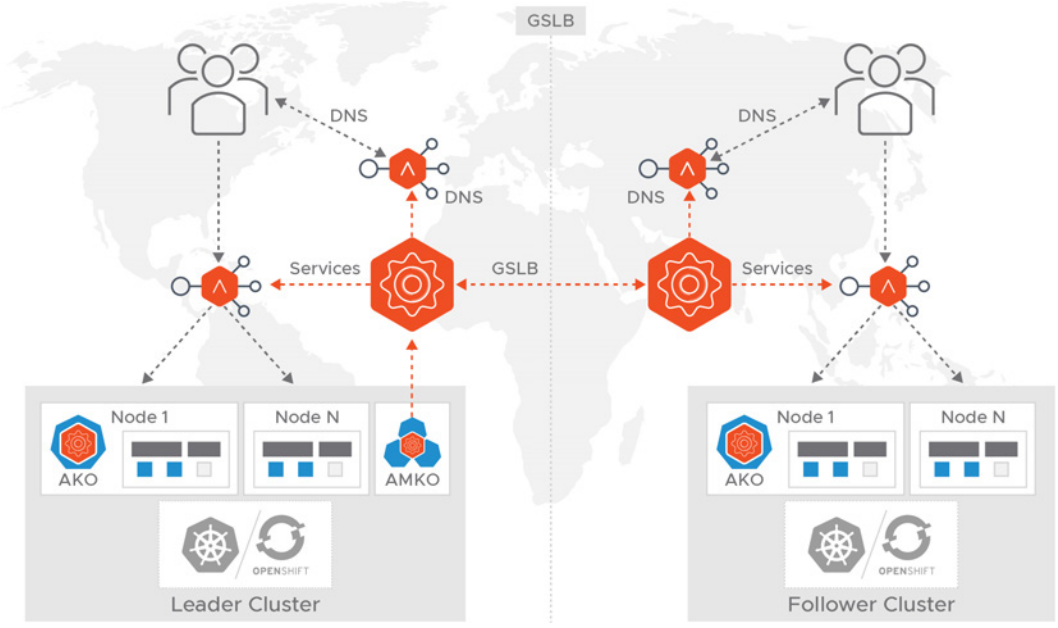


Figure 5: AMKO GSLB Service Architecture

Multi Availability Zones

Multi- Availability Zones provides load balancing of an application (virtual service) spread across multiple availability zones with application awareness across multiple clusters. To span an application across multiple availability zones Avi uses a tiered load balancing deployment methodology. A single virtual service is scaled out where the first tier of SE's load balances L4 TCP connections between the second tier of SE's, which provide L7 load balancing among the nodes within the chosen cluster. For best performance and optimal user experience, Avi dynamically routes traffic based on per-application availability across different AZs. The Avi Controller centrally manages the lifecycle for SE's, combines all analytics and logs across the VIPs of the same virtual service into a single consolidated view for the applications. (SEE FIGURE 6).

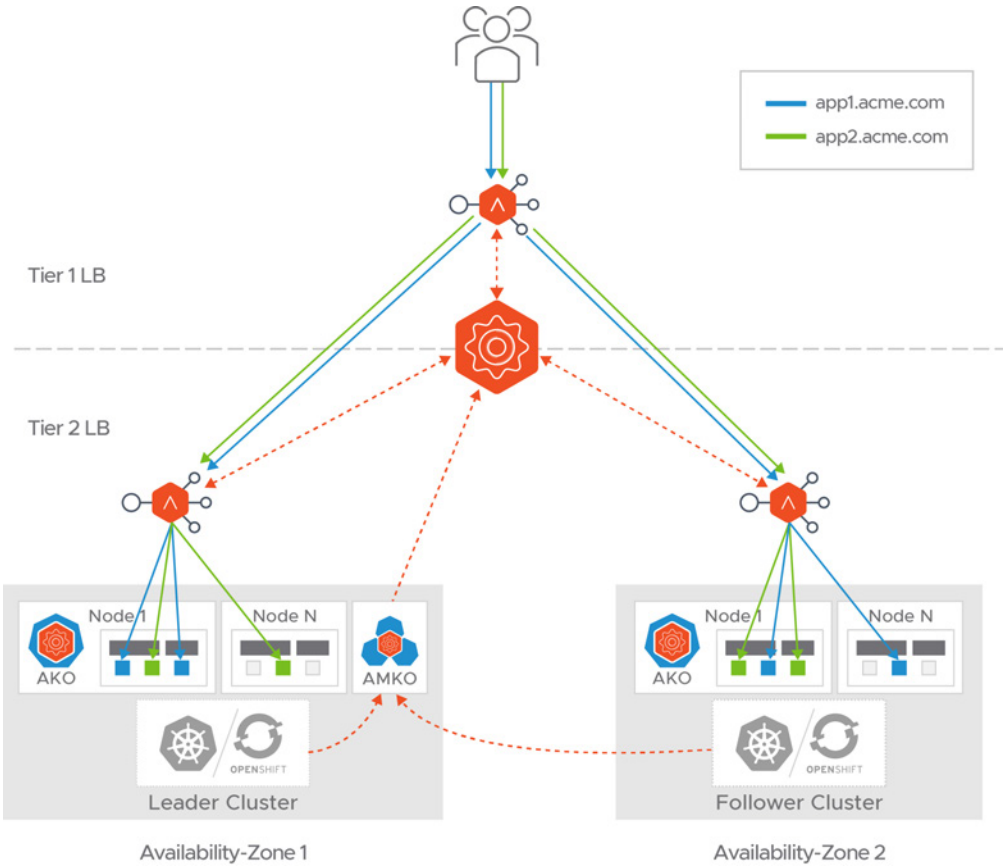


Figure 6: Multi Availability Zones Service Architecture

AMKO Infrastructure deployment workflow

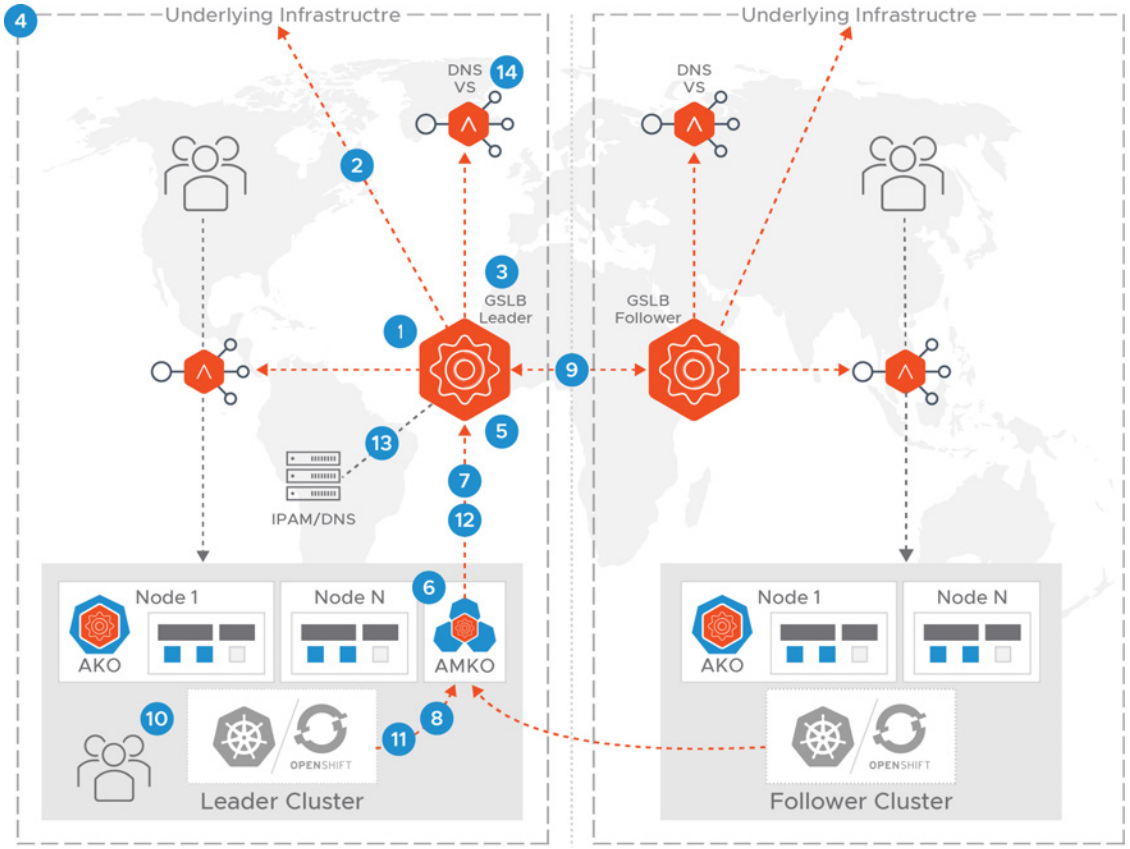


Figure 7: AMKO Infrastructure deployment workflow

Infrastructure deployment workflow

1. The Avi Controllers are deployed on the underlying Infrastructure, e.g. VMware vCenter
2. The Avi Controller communicates with the underlying infrastructure to orchestrates the lifecycle of the Service Engines
3. Kubernetes/OpenShift clusters are added to their Avi controller on their respective sites via AKO
4. One controller site is designated as the leader site
5. GSLB is enabled on the leader and all follower sites
6. AMKO is configured with Avi Controller GSLB leader and Kubernetes/OpenShift leader and follower cluster parameters via Helm Chart (Controller IP, Cluster API endpoint, Cluster and Controller Authentication, VRF etc.) and deployed on the Kubernetes/OpenShift cluster designated to the GSLB leader site
7. AMKO establishes a connection with Avi Controller for the leader cluster
8. AMKO establishes connection with Kubernetes/OpenShift leader and follower
9. GSLB leader Avi Controller synchronizes settings with Avi Controllers for all follower clusters

Application deployment workflow

10. Admin configures Ingresses/Routes and GSLB Custom Resource Definitions (CRD)
11. AMKO gets Ingresses/Routes and GSLB CRD's from the Kubernetes/OpenShift
12. AMKO translates GSLB CRD's and calls Avi APIs to configure GSLB service
13. Avi Controller publishes global FQDN to DNS
14. Avi Service Engines host DNS Virtual Services

Datapath

The Avi Controller automatically manages the available capacity of the Avi Service Engines based on traffic patterns.

North-South or External Services: These are external services accessed by users via Virtual IP address (VIP) per service. Avi SEs replace the default load balancers in front of the Router and the Routes themselves. Avi provides a single layer of proxying for ingress traffic into the Kubernetes/OpenShift clusters. Avi Controller and SEs natively integrate with the IaaS layer (AWS /GCP / Azure / vSphere / OpenStack) and automate network reachability for ingress VIPs.

These virtual services are proxied by one or several Avi Service Engines. The Avi Controller uses a combination of load and connectivity to find appropriate Service Engines and "places" these virtual services on those Service Engines.

Since Avi is a drop-in route provider, the Avi Controller has a built-in Route Controller that listens to create/modify/delete for Route objects and automatically creates/modifies/deletes corresponding Avi Virtual Service and Pool objects. Where required, additional features and functionality that are not covered by the standard Route functionality can be configured using Avi specific CRDs'. (SEE FIGURE 8).

1. Client DNS lookup for Service/Application via DNS
2. Service/Application FQDN resolves to Avi VIP
3. Client sends request to Avi VIP
4. Avi Service Engine chooses a backend pod for load-balancing.
5. Request is sent to the backend Pod IP
6. App responds back to the Client via Avi Service Engine

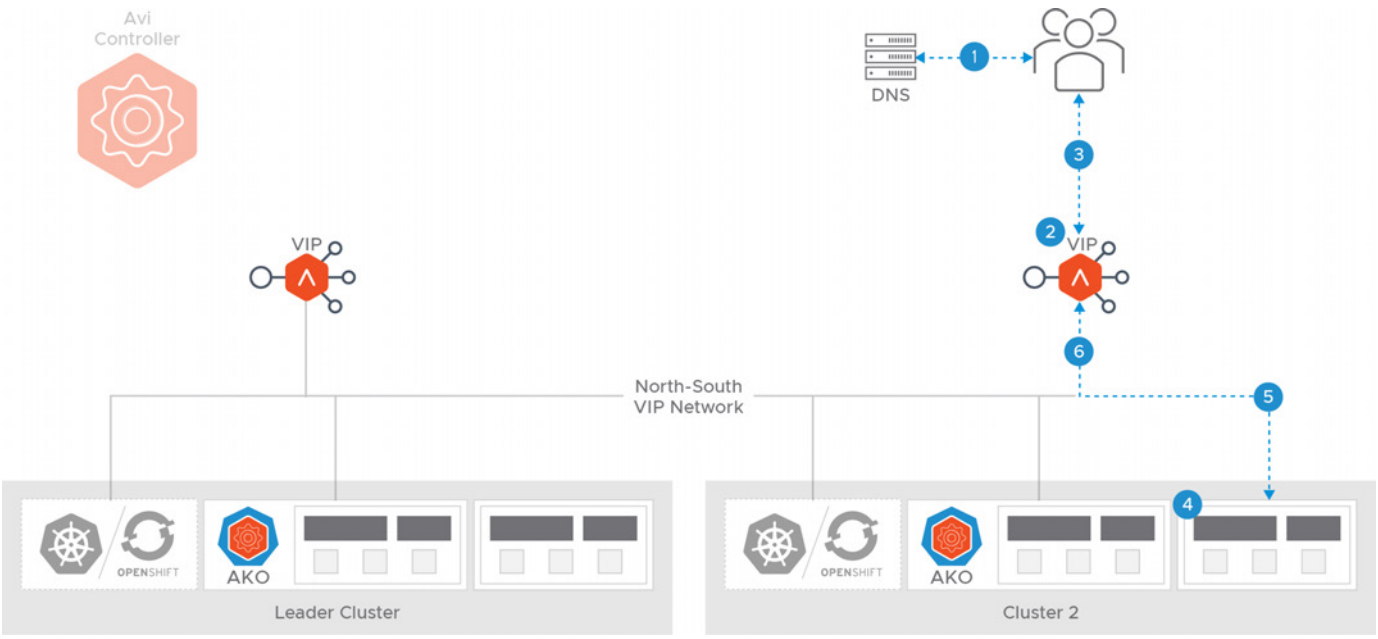


Figure 8: End-user datapath workflow

Enterprise-Grade Security for Kubernetes/OpenShift

Avi features an Intelligent Web Application Firewall (iWAF) with a distributed application security fabric to enforce security through closed-loop analytics and application learning mode. iWAF covers OWASP CRS protection, support for compliance regulations such as PCI DSS, HIPAA, and GDPR, positive security model, and signature-based detection. The built-in solution provides security and networking teams with a comprehensive security stack including DDoS, rate limiting, SSL/TLS offload and encryption, and ACL that simplifies policy customization and scales automatically on-demand across any environment. (SEE FIGURE 9).

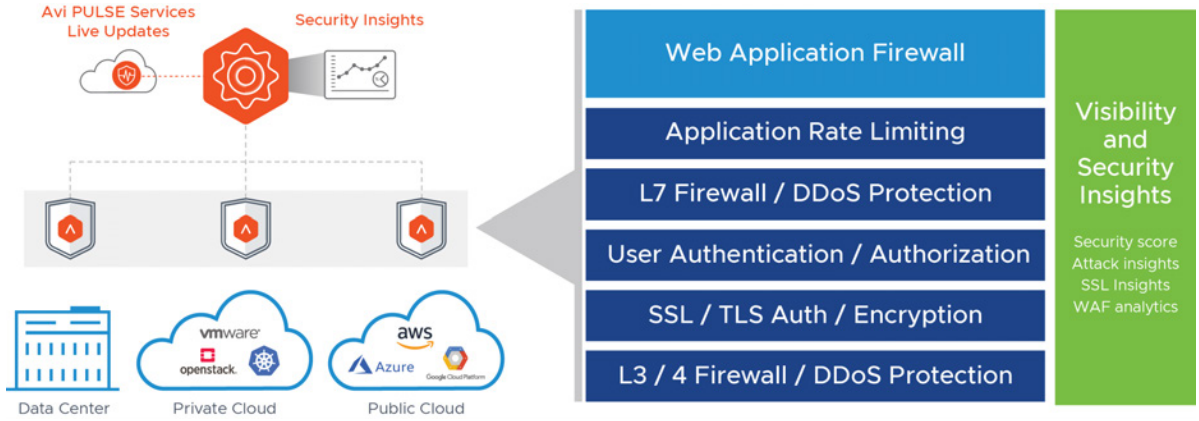


Figure 9: iWAF Comprehensive Security Stack and Insights

Avi's iWAF gives administrators end-to-end security insights and analytics on performance, end-user interactions and security events in a single dashboard (Avi App Insights) for actionable insights on security intelligence and enforcement. With Avi PULSE Services, live threat updates including IP reputation, signatures and more are sourced from industry leading threat analysis companies and curated through the Avi PULSE. It protects web applications from common vulnerabilities, such as SQL Injection (SQLi) and Cross-site Scripting (XSS), while providing the ability to customize the rule set for each application. iWAF analyzes the unvalidated traffic through the acceptlist engine, positive security model that validates known good behavior as applications and attack patterns are learned and last the signatures engine processes security rules that match a particular transaction – all these in real-time. The optimized security pipeline maximizes efficiency, sharply reduces false-positives, and blocks zero-day attacks. (SEE FIGURE 10).

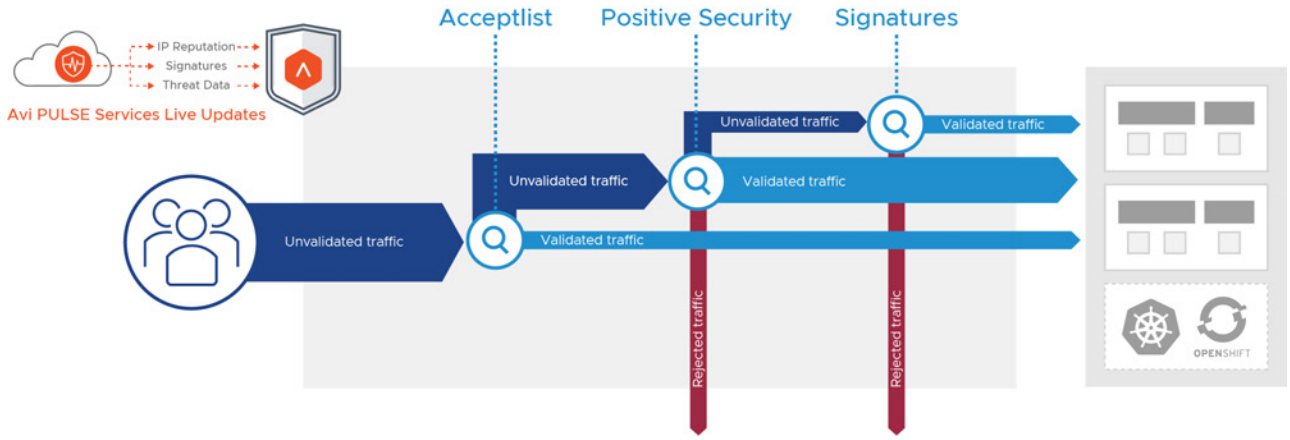


Figure 10: Avi iWAF Security Pipeline Optimization

Scale and Performance

While performing application delivery tasks, SEs could experience resource exhaustion. This may be due to CPU, memory, or traffic patterns. Monitoring several application and network telemetry real-time from the SEs, the Avi Controller can automatically migrate a virtual service to an unused SE or scale out the virtual service across multiple SEs to increase capacity. This allows multiple active SEs to concurrently share the workload of a single virtual service. In addition, Avi learns application access patterns and can perform intelligent, predictive autoscaling based on the learned traffic patterns and application usage, making services highly available before demand causes any service exhaustion or disruption.

Continuous Integration and Delivery (CI/CD)

Applications can be upgraded using a Blue-Green or canary deployment pattern. Avi offers an out-of-the-box non disruptive, graceful application upgrade capability. When a new application version is available the SEs can direct new users to the new application version while existing users continue to be serviced by the older version. Once all deployment criteria for the new application version is fulfilled all traffic is directed towards the new application version. After a sufficient period, when all existing users have disconnected, the older version is safely deleted. The entire process can be controlled by the administrator, or Avi can provide a policy-based Blue-Green orchestration that automates the entire process.

Monitoring and Analytics

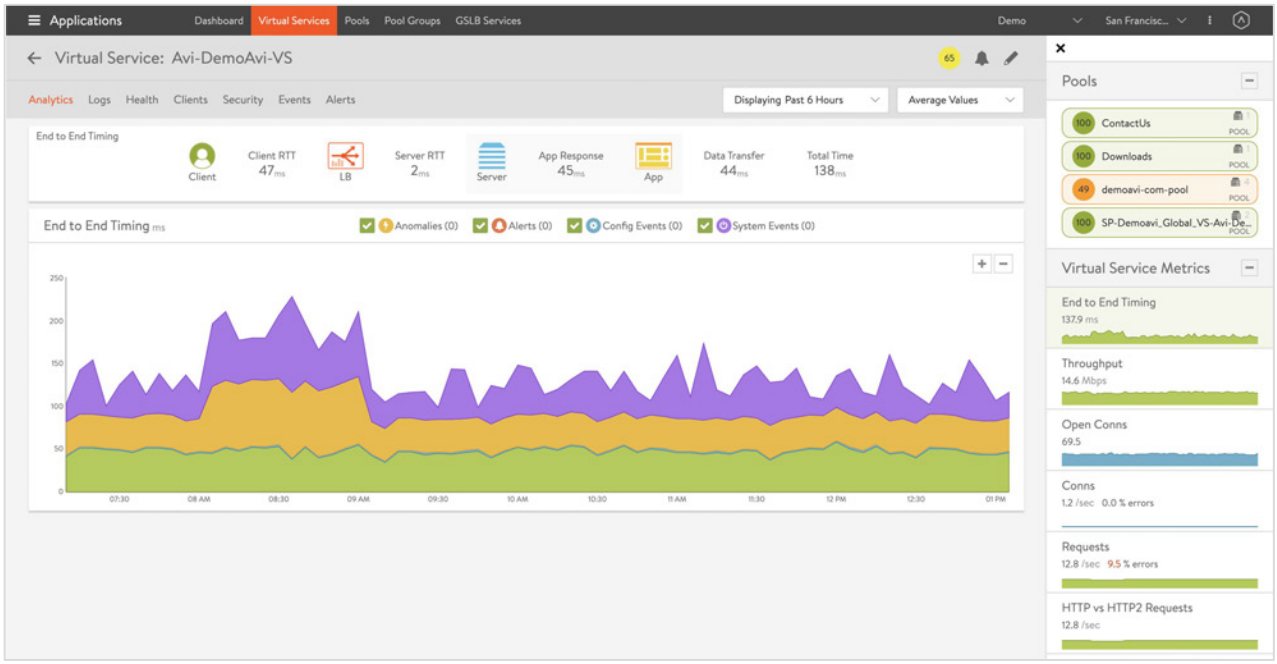
Application owners need to know how applications are performing in real-time and in the past, they need alerts when services degrade or go down, and they need to understand if an outage is caused by the infrastructure, the application, or a user error. A uniform, scalable, and robust monitoring/analytics system is required to collect, aggregate, accumulate, store, and rollup metrics and logs for all applications.

Avi Service Engines collect hundreds of individual metrics and log every HTTP or TCP/UDP transaction.

This real-time telemetry is processed by the Avi Controller continuously and made available via dashboards and REST APIs to provide unprecedented visibility and application insights for quick utilization by administrators. It enables network engineers and developers to get actionable insights into application performance, security, and end-user experience simplifying troubleshooting down to minutes.

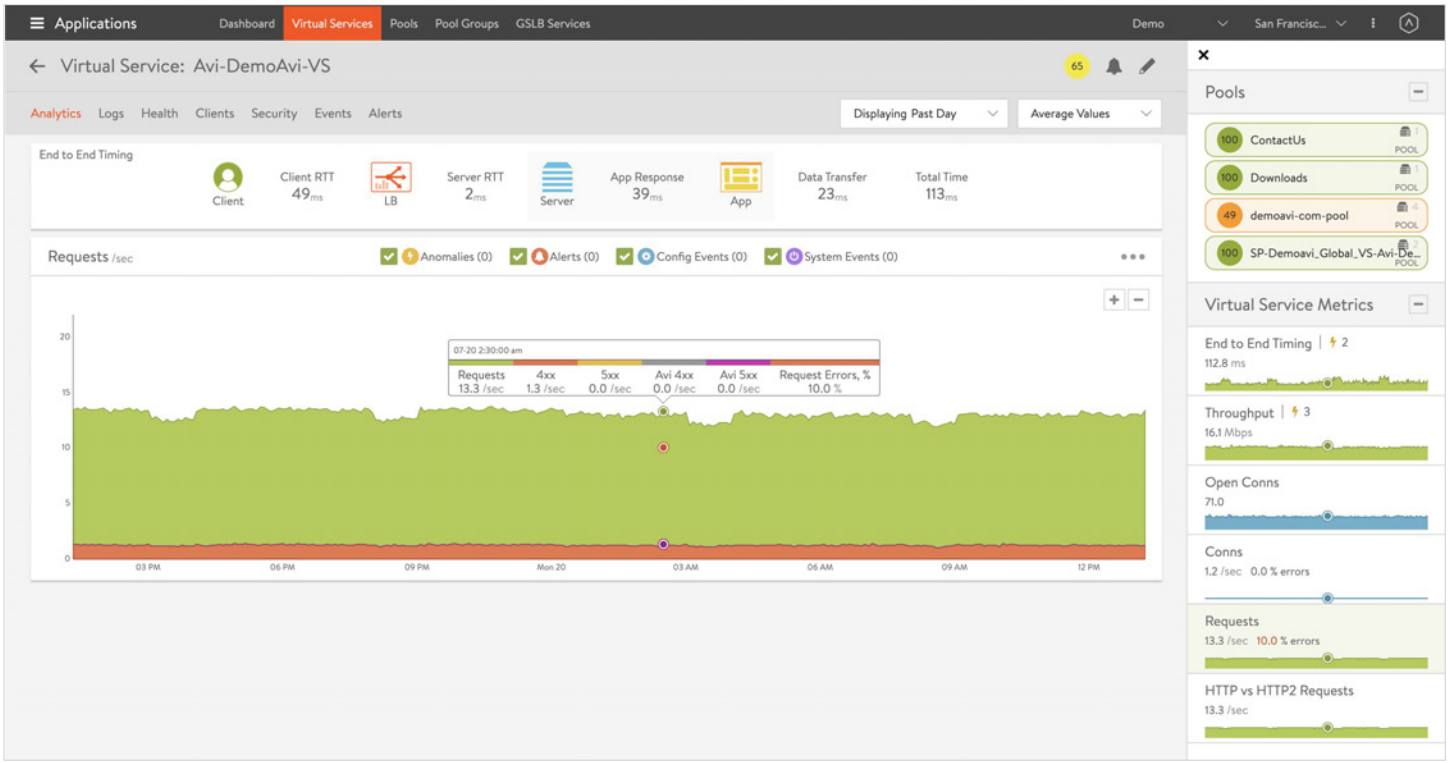
Analytics Dashboard

For every application, Avi insights provides an end-to-end view of all transactions, including current and historic views of critical metrics, such as requests/transaction/connection rate, throughput, etc.



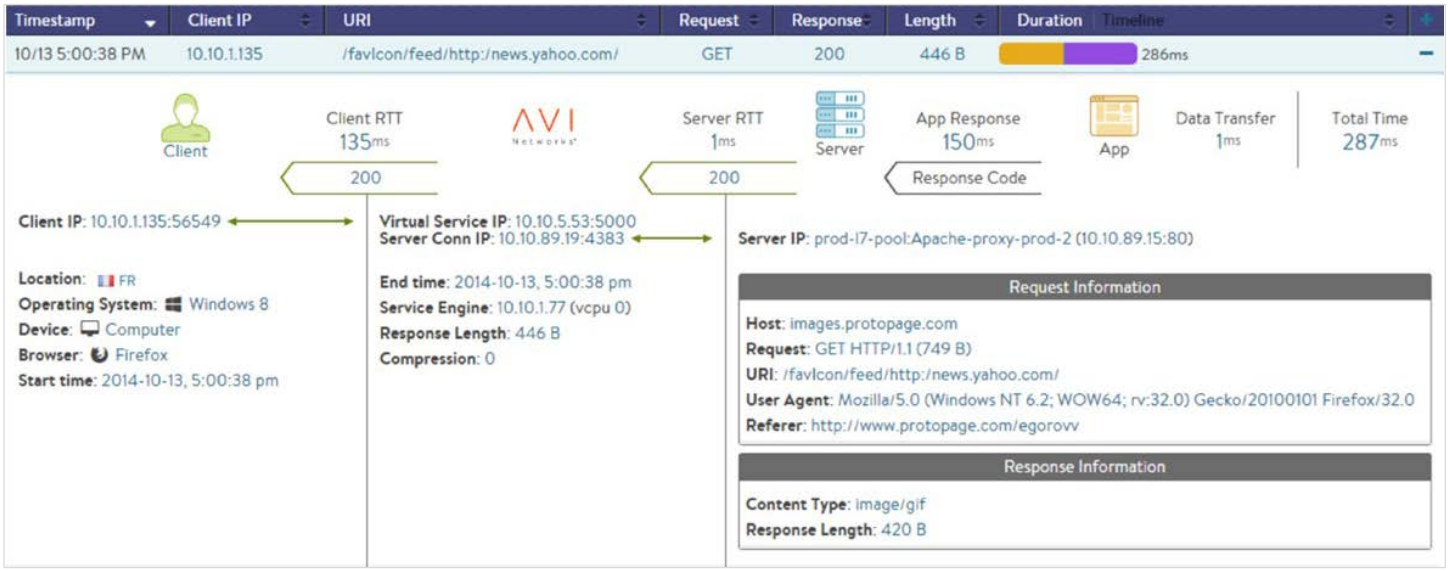
Application Performance Analytics

Avi Service Engines log every significant transaction, like end-to-end round-trip times with latencies between each network hop, and the Avi Controller indexes such logs and provides analytics based on several dimensions, such as pool member, response time, device type, etc. Log Analytics also provide a Google-like search with search filters transactions based on device, location, errors, etc. Application logs can also be directly forwarded by Avi Service Engines to an external log analytics platform like Splunk. And in combination with "Network DVR" like capability to record-and-replay specific transactions administrators can Troubleshoot application issues in less than a minute.



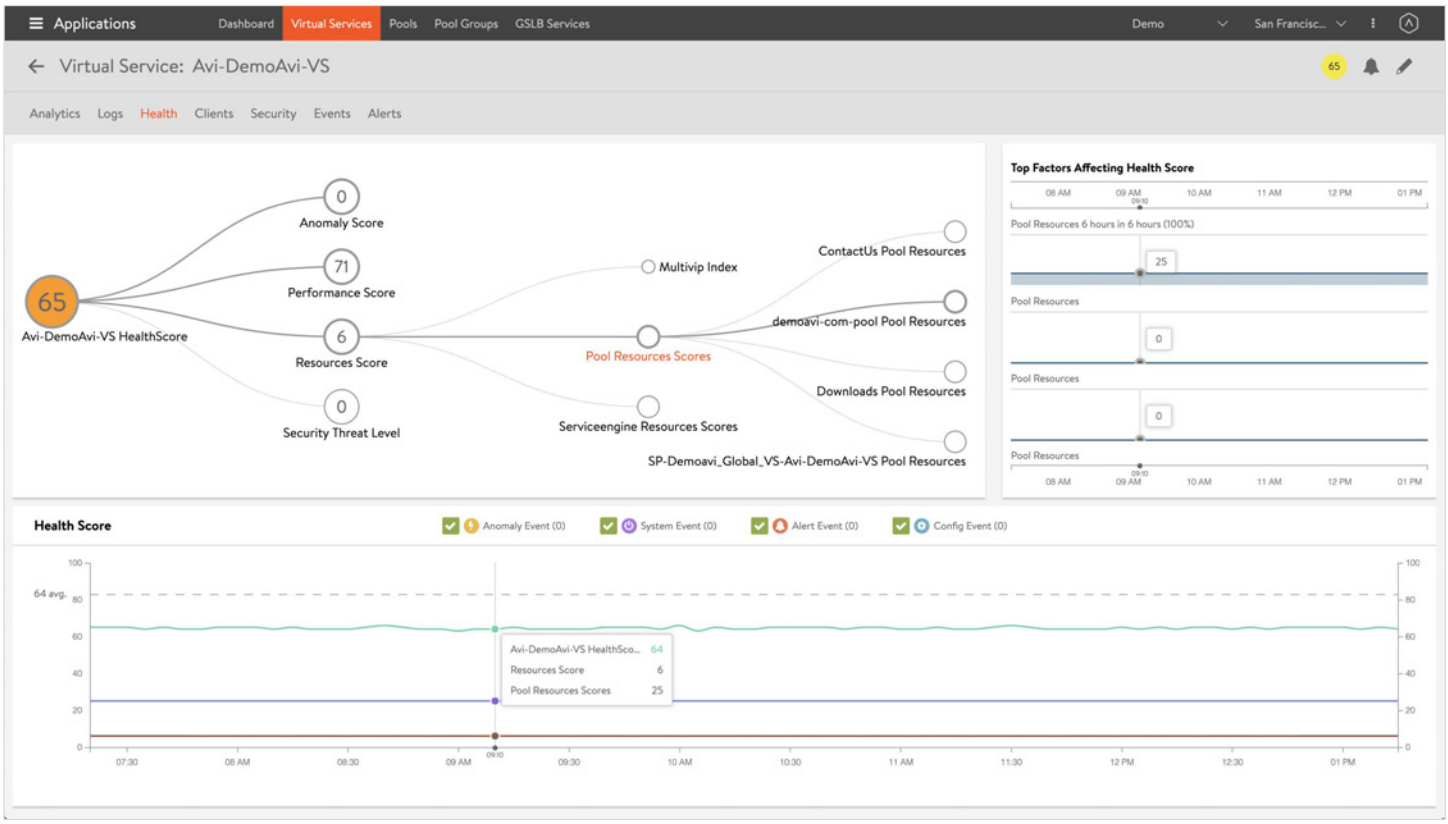
End User Experience Analytics

Client Analytics inserts a JavaScript resource into responses that reports back navigation, resource timing and server responses in real time information to Avi Controller. This provides aggregate page load times, dimensional analytics by device type, country, data center, server, application pool, etc., and details resource timing information for every page in the application giving granular visibility into end user experience and allows application owners to optimize and improve applications.



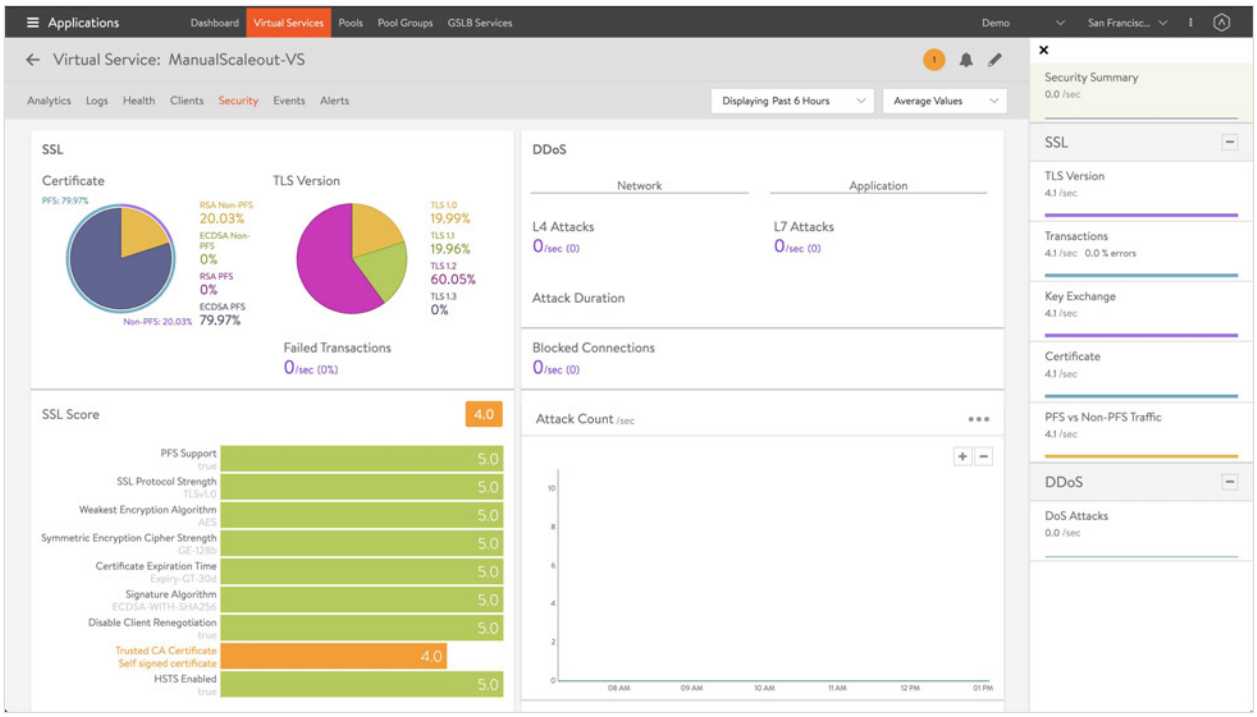
Application Health Score

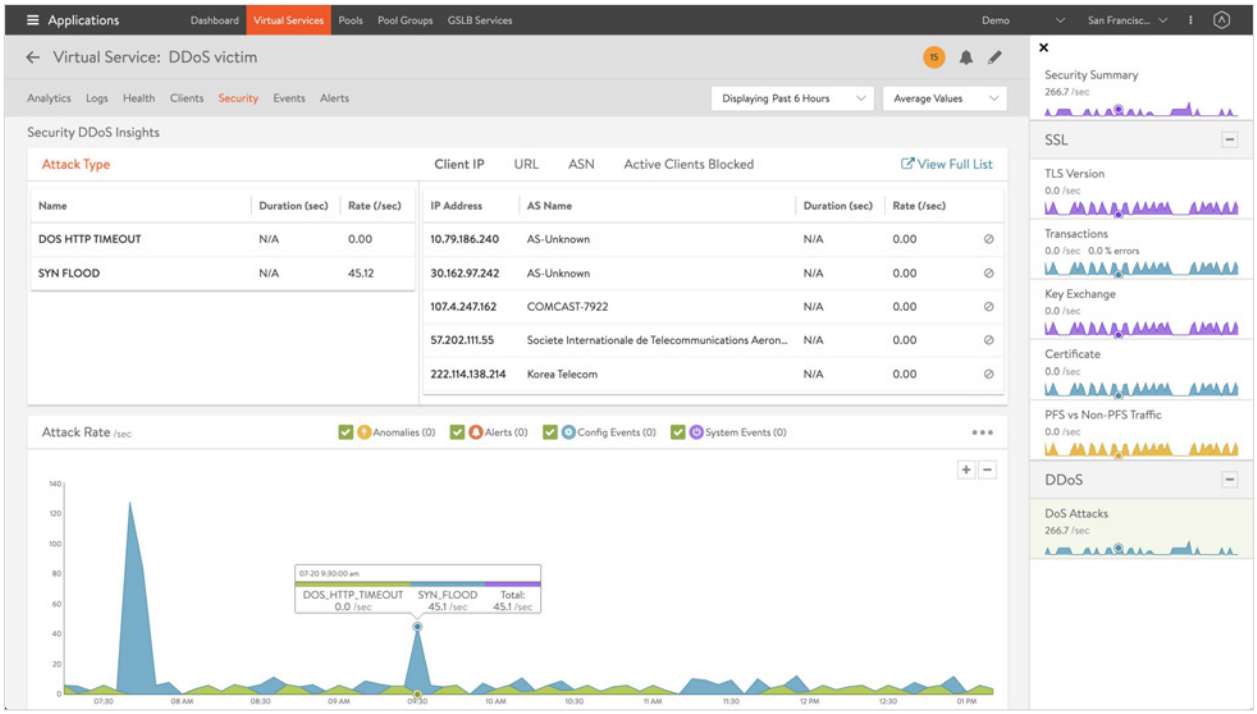
Avi summarizes analytics information into a health score that provides a quick snapshot of the overall applications. The Application health score is determined by Server Response Time, Resource Usage, Anomalous Traffic and Security Posture.



Security Analytics

Avi continually assesses the health of each virtual service providing insights into TLS/SSL versions and transaction rate. The SSL score is based on the SSL security profile which consists of PFS support, SSL/TLS version, encryption algorithm used, cipher strength, signature algorithm, trusted/untrusted certificates used, highlighting potential risks with SSL certificates or TLS versions used. DDoS attack analytics breaks down distributed denial of service data for the virtual service into the most relevant layer 4 and layer 7 attack data including information about the attack duration, blocked connections, attack count, the number of network attacks per second, such as IP fragmentation attacks or TCP SYN flood and application attacks per second, such as HTTP SlowLoris attacks or request floods.





Analytics Alerts and Notifications

Avi can trigger alerts based on system events, or on more of the 500+ metrics Avi is tracking. Once an alert is triggered, it may be used to notify an administrator through one or more of the notification actions. In addition, alerts can trigger ControlScripts for automated actions.

The following notification forms are available:

Local notifications:

- Alerts are logged and made visible within the Analytics Dashboard. Local notifications are marked by the alert action with an alert priority, which provides an informative mechanism for categorizing the alerts.

Email:

- Alert Actions may be configured to send alerts to administrators via email. These emails could be sent directly to administrators or to reporting systems that accept email.

Syslog:

- Syslog messages may be sent to one or more syslog servers. Communication is non-encrypted via UDP, using a customizable port. According to RFC 5426, syslog receivers must support accepting syslog datagrams on the well-known port 514, but may be configurable to listen on a different port.

SNMP traps:

- Alerts may be sent via SNMP traps using SNMP v2c. Multiple trap servers may be defined.
- Configuring SNMP traps is exclusively for sending alerts to an SNMP trap server, not for configuring how SNMP would poll Avi SNMP OIDs.
- Traps are sent from the Controller cluster leader, but the leadership role can move to either follower Controller after a failure. Consequently, the external SNMP server should be configured to allow traffic from any one of the three Controllers in the cluster.

Service Discovery

Service discovery is the process of automatically detecting devices and services on a network. Like Kubernetes service discovery it works by devices connecting through a common language on the network allowing devices and/or services to connect without any manual intervention.

There are two types of service discovery: Server-side and Client-side. Server-side service discovery allows clients applications to find services through a router or a load balancer. Client-side service discovery allows clients applications to find services by looking through or querying a service registry, in which service instances and endpoints are all within the service registry.

The Service Registry is a database that contains the network locations of service instances. The service registry needs to be highly available and up to date so clients can go through network locations obtained from the service registry. Microservices service discovery is a way for applications and microservices to locate each other on a network. Service discovery implementations within microservices architecture discovery includes a central server (or servers) that maintain a global view of addresses for clients that connect to the central server to update and retrieve addresses. The “global state” (available service IP addresses) of the application across sites and regions also resides in the service discovery database and is accessible by DNS. Users of all services (users using browsers or apps or other services) use well-known DNS mechanisms to obtain service IP addresses.

Avi service discovery automatically maps service host/domain names to their Virtual IP addresses across multiple clusters and availability zones and updates the service discovery database as services are created and disabled. Avi provides an authoritative DNS server for users’ devices and other services including a variety of DNS configuration options as well as integration with third-party DNS and IPAM services such as Infoblox.

For example, an application app1.acme.com is created and automatically associated with 2 VIPs belonging to 2 clusters in different zones/regions. Datacenter-1 has VIP-1 and Datacenter-2 has VIP-2. When a user does a DNS lookup for app1.acme.com, the user is returned an A record with either VIP-1 or VIP-2 based on the following criteria's:

- is the service active/active or active/passive?
- are both VIPs responsive/available?
- is the user geographically closer to Datacenter-1 or Datacenter-2?
- does the user have a site persistence cookie directing the user to a specific Data Center?
- If the user is returned VIP-1, the user will access the application in Datacenter-1. If the user is returned VIP-2, the user accesses the application in Datacenter-2. (SEE FIGURE 11)

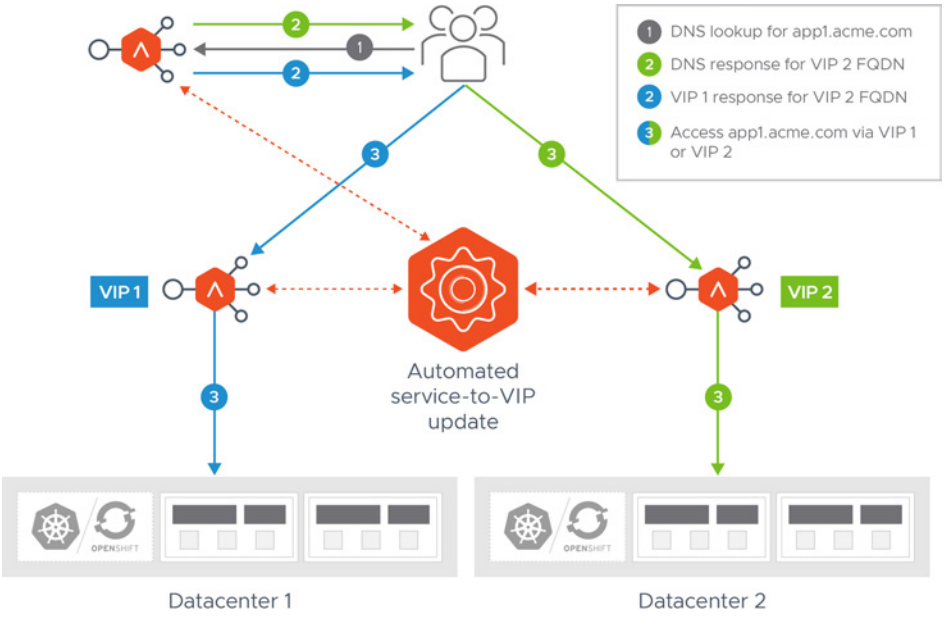


Figure 11: Avi automated service discovery

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 vmware.com.