WHITE PAPER: August 2024

Decision Automation with Load Balancers and WAF

Multi-cloud automation, self-service, security, and elastic scale

Table of contents

Introduction

Application-centric enterprises need to networking teams to automate and deliver self-service to lines of business. They simply cannot afford the operational inefficiencies and management challenges of fleets of legacy load balancing appliances. As enterprises extend out from data centers to "centers of data" across multiple public clouds using diverse computing infrastructure including bare-metal servers, VMs, and containers, they are looking to implement continuous delivery / continuous integration (CI/CD) practices. They need intelligent automation and consistent networking services across multi-cloud environments that eliminates silos caused by load balancing appliances in the data center, virtual ADCs in the cloud, or open source solutions for container networking. Appliance-based (physical or virtual) load balancers and most cloud load balancers do not offer real-time visibility into end-user experience or app performance, have no automated failure recovery capabilities, lack the ability and automation to scale across multiple clouds as well as scale automatically up or down based on traffic patterns and require extensive manual configuration and separate management for each instance. See Figure 1.

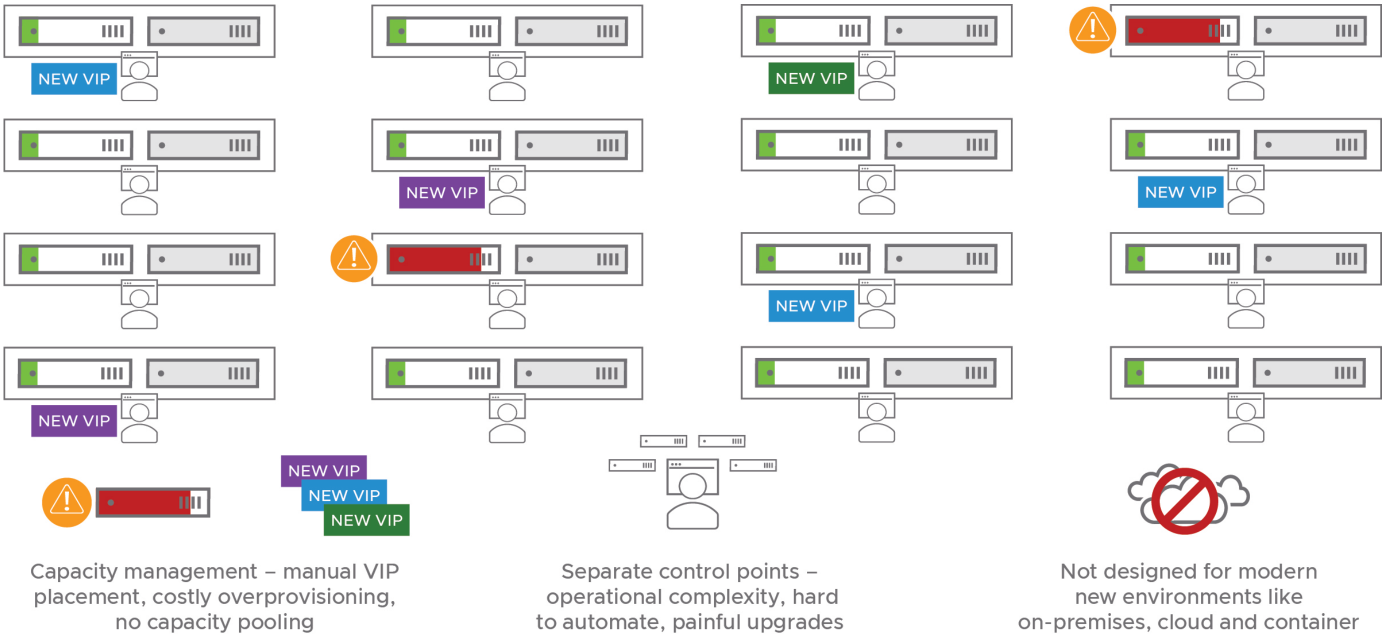
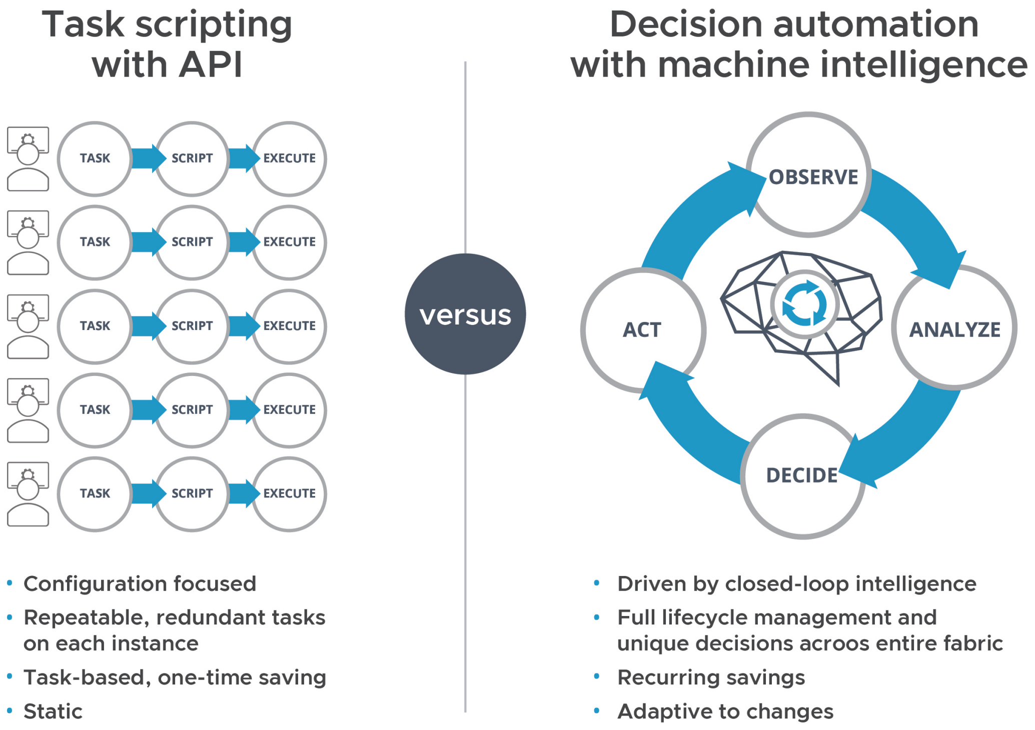

NEW VIP
NEW VIP
NEW VIP
NEW VIP
NEW VIP
NEW VIP
NEW VIP
NEW VIP
NEW VIP
Capacity management – manual VIP placement, costly overprovisioning, no capacity pooling
Separate control points – operational complexity, hard to automate, painful upgrades
Not designed for modern new environments like on-premises, cloud and container

Figure 1: Legacy deployment creating new virtual service

API ≠ Decision Automation

Automation and self-service represent the evolution of modern, intelligent networking. All too often, vendors eager to jump on the automation bandwagon tell enterprises that their products are fully automated because they have REST API. However, there is a vast chasm between providing APIs and delivering automation for application services such as load balancing, WAF, GSLB, and container ingress networking. The issue comes down to writing scripts using the APIs versus letting the system automate mundane “decisions” using closed-loop intelligence about the environment and real time conditions. Correctly implemented, such declarative “Decision Automation” simplifies operations, improves agility, and improves security. Decision Automation lets administrators determine a desired outcome of a network implementation, then automates provisioning, configuration, on-demand availability of services, and manages the full lifecycle of the application services including automated failure recovery. The plans and policies specified in a declarative model are automatically implemented, delivered, and enforced with the help of closed-loop machine learning. This avoids laborious manual coding of scripts to cope with specific scenarios. Instead, the admin states the desired outcome (e.g. deploy new application), and the intelligence of the system then decides how best to meet it. Decision Automation also allows an IT generalist or even business owner to specify their intent, without having to rely on an IT specialist to manually provide input or code specialized scripts. See Figure 2.



- Configuration focused
- Repeatable, redundant tasks on each instance
- Task-based, one-time saving
- Static

- Driven by closed-loop intelligence
- Full lifecycle management and unique decisions acroos entire fabric
- Recurring savings
- Adaptive to changes

Figure 2: Task scripting vs Decision Automation

Decision Automation requires a fundamentally different architecture for applications services – one that is software-defined, agnostic to the underlying environment, and built on cloud-native principles. The architecture moves the focus from infrastructure to applications, proprietary hardware to software, and from manual configurations to centralized policies. It allows for programmability but more importantly leads to better automated, system delivered outcomes and reduced operational costs.

VMware Avi Load Balancer and WAF

VMware Avi Load Balancer uses a software-defined architecture that separates the central control plane (Avi Controller) from the distributed data plane (Avi Service Engines). Avi Load Balancer is 100% REST API based, making it fully automatable and seamless with the CI/CD pipeline for application delivery. The Avi Controller is the "brain" of the entire system and acts as a single point of intelligence, management, and control for the distributed data plane. The Avi SE represent full-featured, enterprise-grade load balancers, web application firewall (WAF), and analytics that provide traffic management and application security while collecting real-time analytics from the traffic flows. The Avi Controller provides comprehensive observability based on closed-loop telemetry and presents actionable insights to make decisions based on application monitoring, end-to-end timing, searchable traffic logs, security insights, log insights, client insights, and more.

Avi Load Balancer delivers intelligent Decision Automation, autoscaling and full lifecycle management, enabled through in-depth ecosystem integration, 100% REST API and, analytics feed to the Controller. With that integration Avi achieves the automation and orchestration required to achieve outcomes based on the administrator's intent. The Avi Controller can natively talk to the APIs of private and public clouds such as vCenter, Microsoft Azure, Amazon AWS, Google Cloud Platform to deploy load balancers and WAF in an any of those environments. See Figure 3.

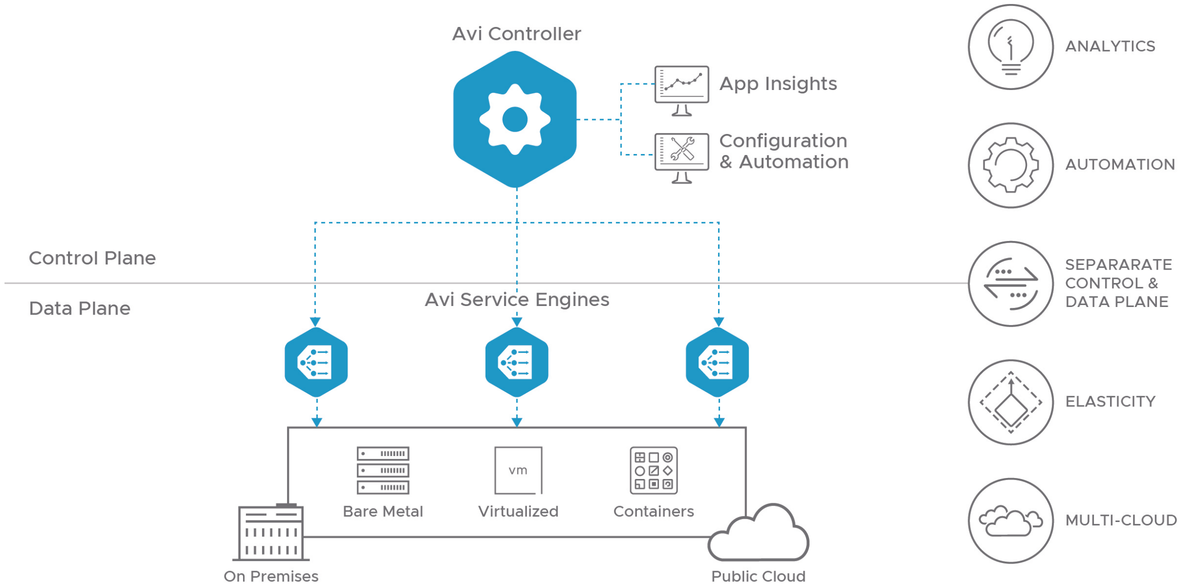


Figure 3: Avi Load Balancer High-level Architecture

Automated Application Delivery

Deploying applications and ensuring consistent end user experience requires decision-based automation across the entire lifecycle of an application. Starting with the initial deployment of the application to increasing capacity dynamically based on demand or automated failure recovery across on prem, multi cloud and multi zone deployments. See Figure 4.

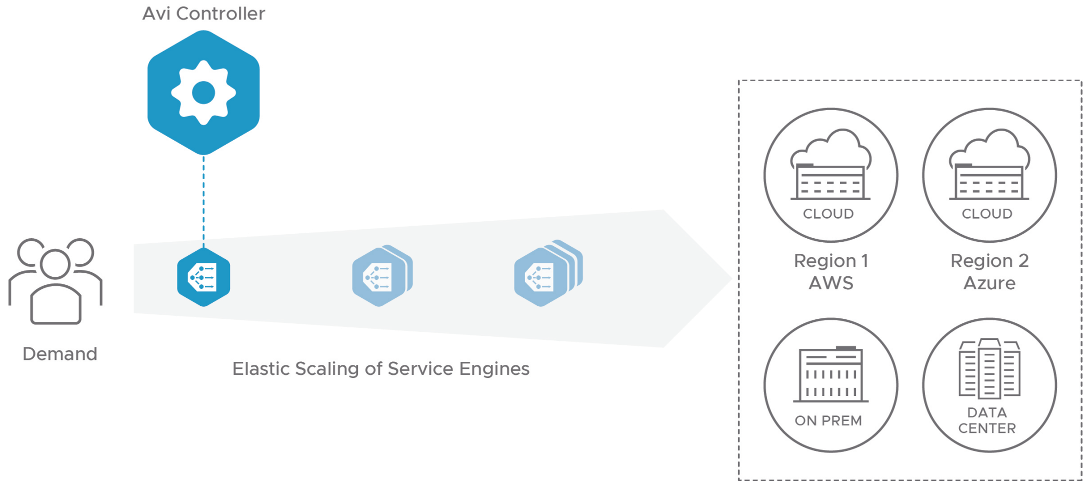


Figure 4: Avi elastic capacity

Monitoring several application and network telemetry real-time from the Service Engine, the Avi Controller can automatically create a new virtual service for a new application, automatically recover from Service Engine failure or migrate a virtual service to an unused Service Engine or scale out the virtual service across multiple Service Engines across Multi Region and/or Multi Availability Zone deployments.

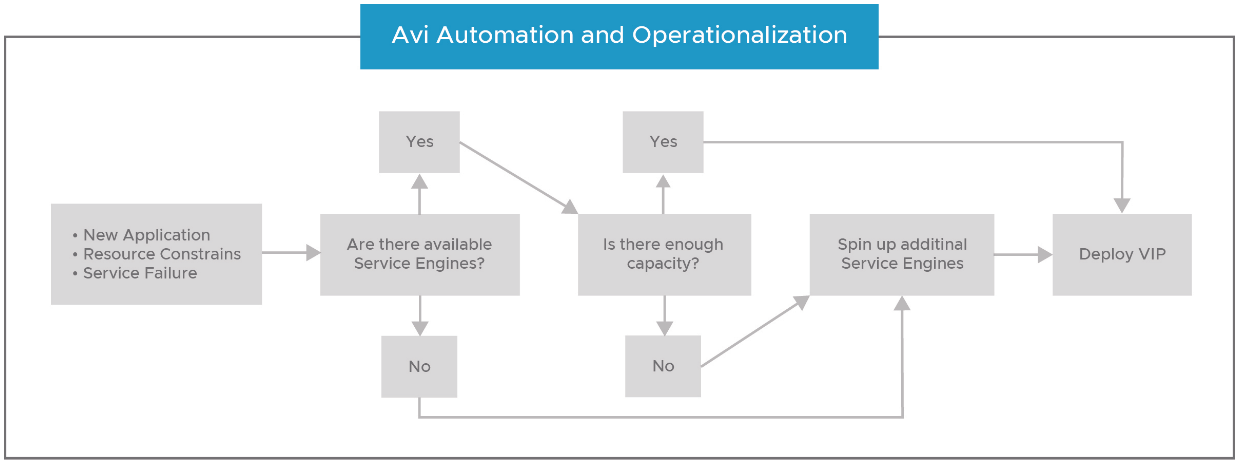

Avi Automation and Operationalization
• New Application
• Resource Constrains
• Service Failure
Are there available Service Engines?
Yes
No
Is there enough capacity?
Yes
No
Spin up additinal Service Engines
Deploy VIP

Figure 5: Avi Automation and Operationalization

Decision Automation for VIP creation with Avi Load Balancer

For example, an administrator states the intent of deploying a new application…

- With hardware based legacy systems the administrator has to identify if a load balancer exists, is there available capacity, if the environment variables are correct, are there any dependencies and once all these questions have been answered or resolved, each load balancer, firewall policies, authentication, IP and DNS settings have to be configured manually. See Figure 6.

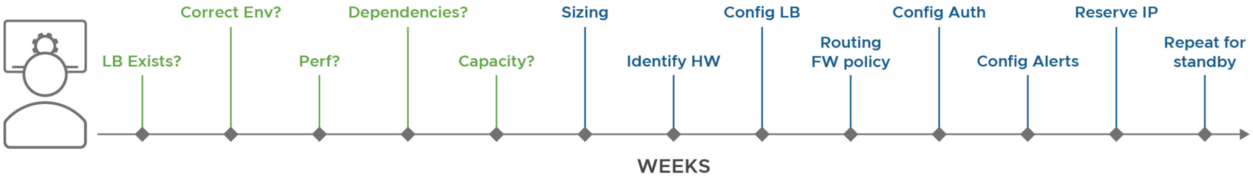


Figure 6: Legacy deployment creating new virtual service

- Avi will find the best location(s), the best host(s) to automatically deploy a new Service Engine, configure the Nics, virtual services, WAF, security policies, authentication and authorization, register DNS, automatically configure and pick up SSL certificates, etc. See Figure 7.

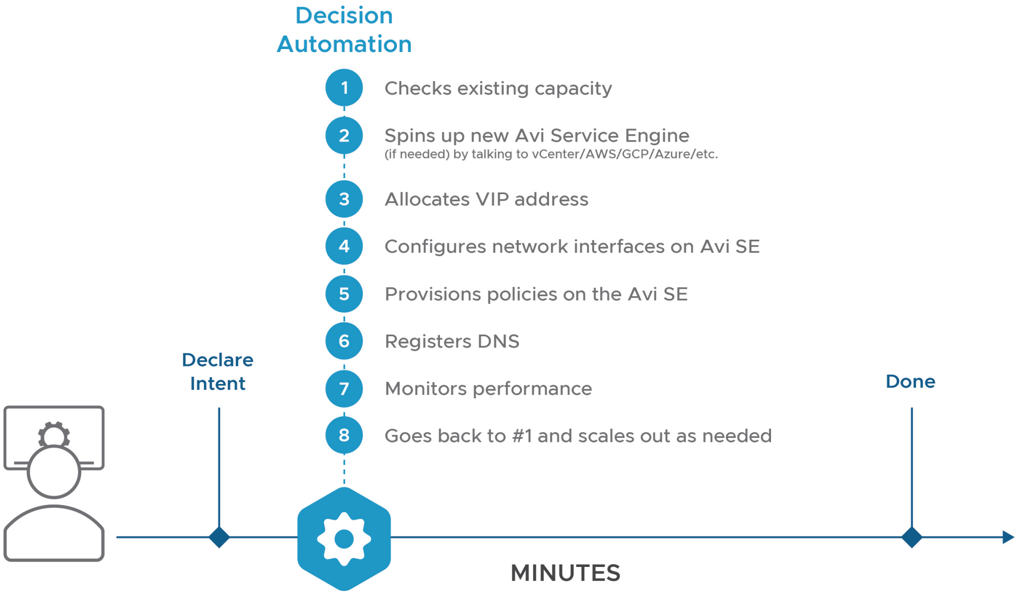

Decision Automation
1 Checks existing capacity
2 Spins up new Avi Service Engine
(if needed) by talking to vCenter/AWS/GCP/Azure/etc.
3 Allocates VIP address
4 Configures network interfaces on Avi SE
5 Provisions policies on the Avi SE
6 Registers DNS
7 Monitors performance
8 Goes back to #1 and scales out as needed
Declare Intent
Done
MINUTES

Figure 7: Avi automated deployment creating new virtual service

Upon initial application deployment the Controller will determine if SEs are available and if the SEs have sufficient capacity available to service the new application. Based on these criteria the Controller will automatically configure a new virtual service for the application on existing SEs or spin up new SEs as needed. See Figure 8.

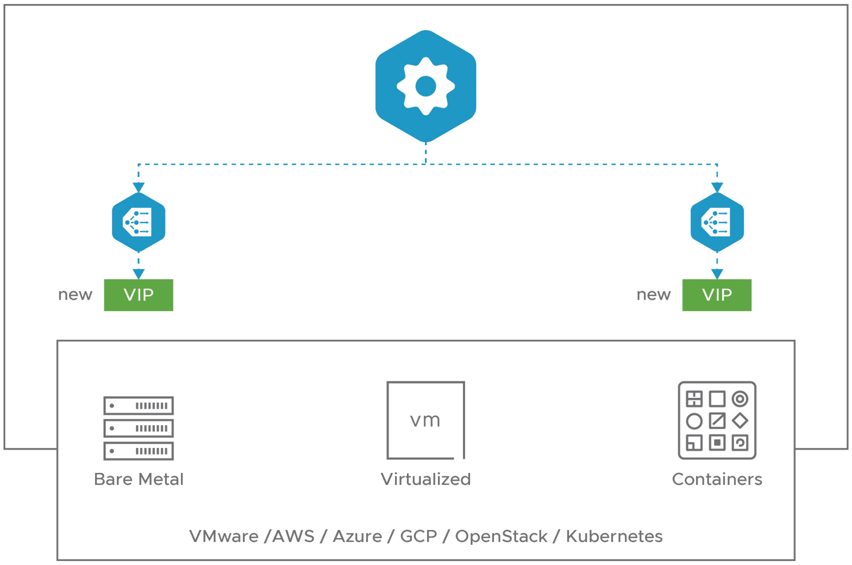


Figure 8: Avi New Virtual Service creation

Increasing capacity and rebalancing

While performing application delivery tasks, Service Engines could experience resource exhaustion. This may be due to the deployment of a new application, high CPU or memory utilization, or traffic patterns. Monitoring several application and network telemetry real-time from the SE, the Avi Controller can automatically migrate a virtual service to an unused SE or scale out the virtual service across multiple SEs across Multi Region and/or Multi Availability Zone deployments 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.
See Figure 9.

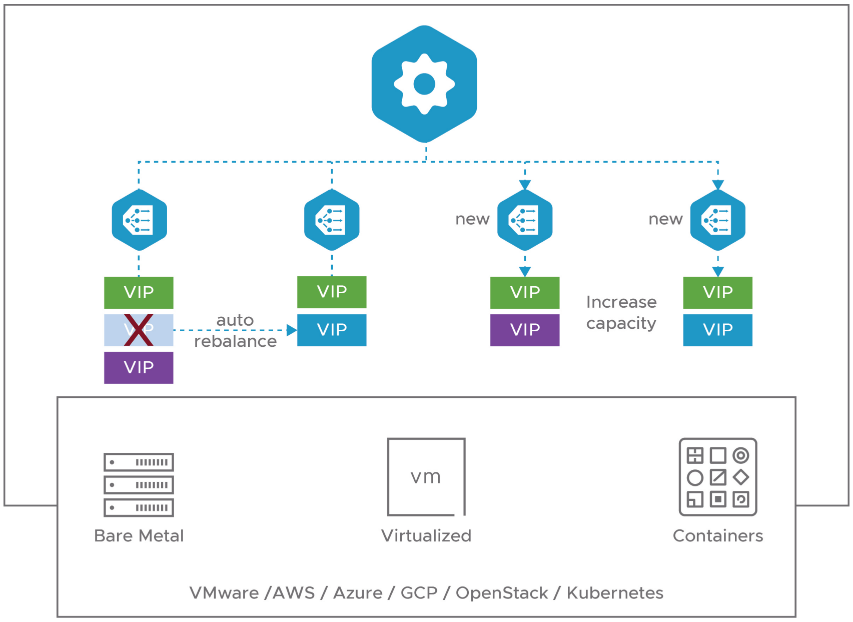


Figure 9: Avi Auto rebalance and capacity adjustment

Automated failure recovery

Automated failure remediation is essential in achieving consistent application availability. Avi relies on a variety of methods to detect Service Engine failures.

Controller-to- Service Engine Failure Detection:

In all deployment modes, the Avi Controller sends heartbeat messages to all SEs in all groups under its control. If there is no response from a specific SE for six consecutive heartbeat messages, the Controller concludes that the SE is down, and moves all virtual services to an available SE with sufficient capacity or spins up a new SE.

Service Engine -to- Service Engine Failure Detection Method:

In the above-mentioned Controller-to- Service Engine failure detection method, the Controller detects a SE failure by sending periodic heartbeat messages over the management interface. However, this method will not detect datapath failures for the data interfaces on SEs. To ensure holistic failure detection, SEs send periodic heartbeat messages over the data interfaces and if the Controller concludes that a SE is down it moves all virtual services to an available SE with sufficient capacity or spins up a new SE.

BGP-Router-to- Service Engine Failure Detection Method:

With BGP configured, the Service Engine -to- Service Engine failure detection is augmented by Bidirectional Forwarding Detection (BFD), which detects SE failures and prompts the router not to use the route to the failed SE for flow load balancing and by using BGP protocol timers, as well. See Figure 10.

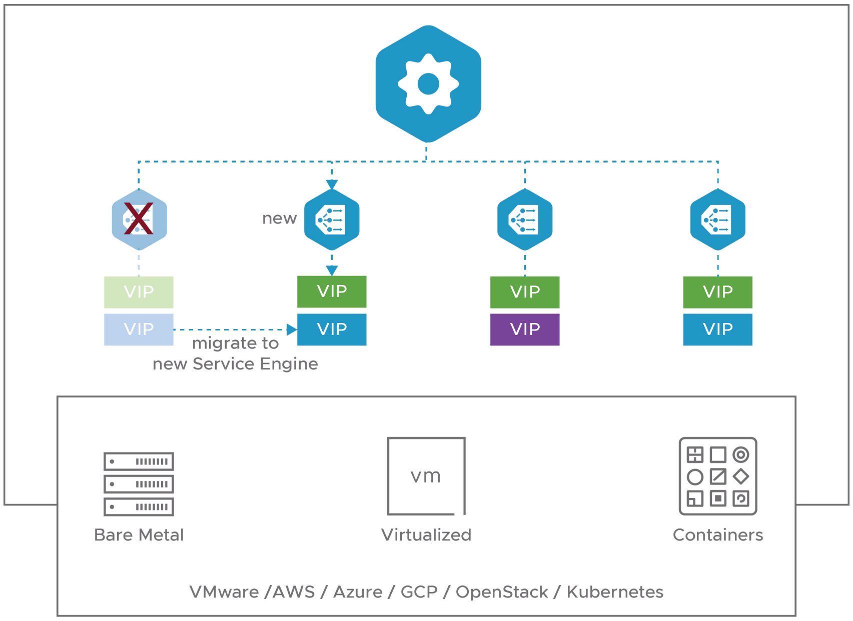


Figure 10: Avi automated failure recovery

Automated Load Distribution

With VMware Avi Load Balancer, enterprises can move workloads across multiple clouds effortlessly. Avi enables enterprises to use AWS/GCP/Azure as a natural extension to their data centers by automatically overflowing to the cloud during traffic peaks. Avi can automatically create app resources in public clouds to absorb traffic surges and scale them back down. For operational automation Avi is natively integrated with the surrounding ecosystem enabling firewalls to automatically initiate, IP addressing, and DNS automatically configured with Infoblox, Amazon RAW 53, etc., but can also provide some customization through REST APIs.
See Figure 11.

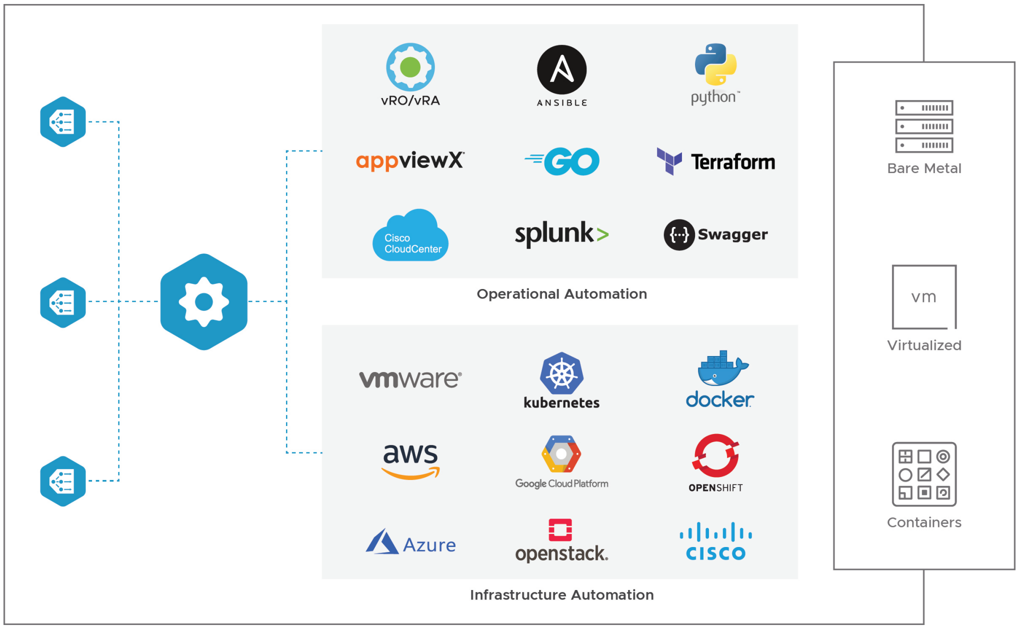


Figure 11: Avi ecosystem integration

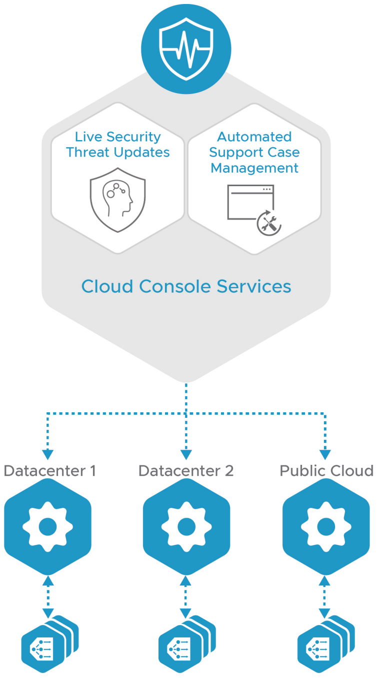

Live Security Threat Updates
Automated Support Case Management
Cloud Console Services
Datacenter 1
Datacenter 2
Public Cloud

Figure 12: Cloud Console Services

Automated Services to Manage Security, Support, and Anomalies

Cloud Console services which is part of the VMware Avi Load Balancer provide an automated and central facility to manage and operate security intelligence and case management for globally distributed Avi Controller clusters run by customers.

Any Avi Controller in any cluster can optionally (based on customer consent) connect to Cloud Console services to retrieve the latest threat intelligence updates including IP reputation, application signatures, WAF CRS rules and more, and automatically contact VMware support giving operations the choice to create a case manually directly from any Avi Controller or configure the Controller to automatically create a support case if a system failure is detected. See Figure 12.

These new services provided by Cloud Console will reduce the complexity, management, and cost of your network operations. Reduce risk with automated compliance and remediation services, increase business agility, and accelerate the digital transformation of your business.

Automated 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. See Figure 13.

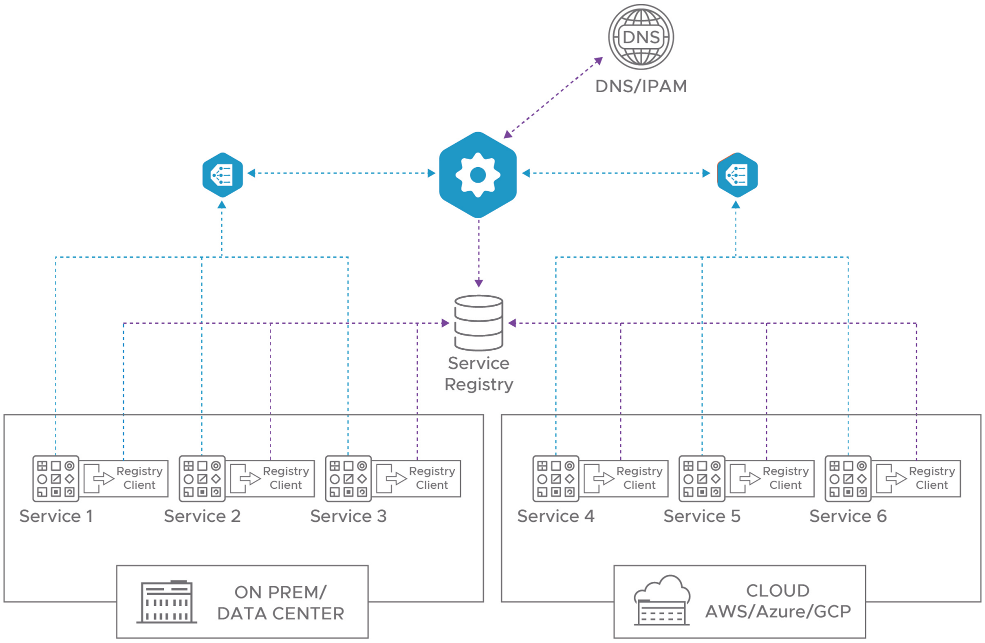
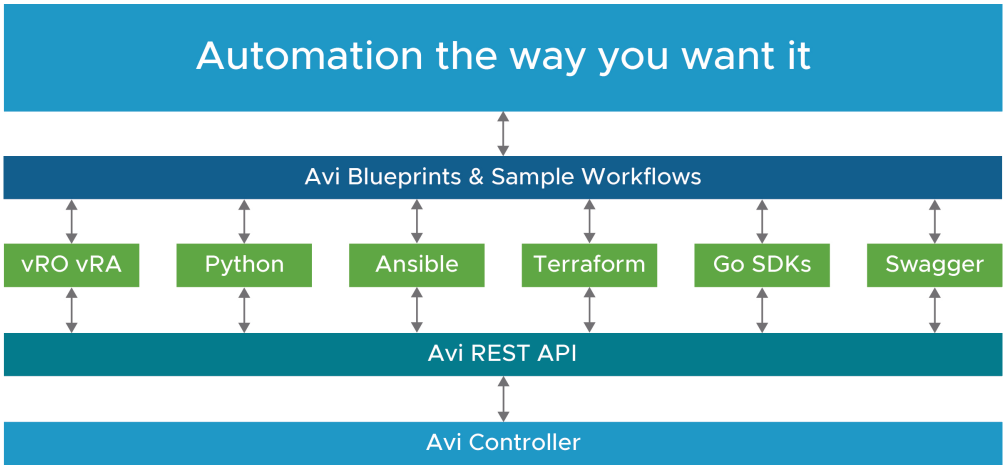


Figure 13: Avi third party services integration

Integration with Automation Tools

Many of these automation elements are something that Avi has natively baked in, but because Avi is built on top rest API, it allows for customization. That means that what is displayed in the UI and the CLI are just wrappers for the API. This is that it makes it easy build out some customization through Terraform, Ansible, vRO vRA, or just custom Python scripts to automate everything.



In Summary

VMware Avi Load Balancer delivers load balancing with integrated app monitoring and analytics, security, predictive autoscaling, and multi-cloud load balancing while offering operational simplicity through automation for enterprises that have their apps deployed in a mix of private data centers and multiple public clouds delivering uniform architecture and user experience, regardless of the environment.