citrix

White Paper

Fast-tracking Multi-cloud Applications that are Secure and Resilient

with Red Hat OpenShift, Red Hat Ansible and Citrix ADC

Introduction

Two trends that will dominate the coming years are application modernization and multi-cloud orchestration. This is because initial lift-and-shift approaches for applications were not cost-effective, according to Gartner. "Organizations are now re-rationalizing applications with containers, microservices and Kubernetes."

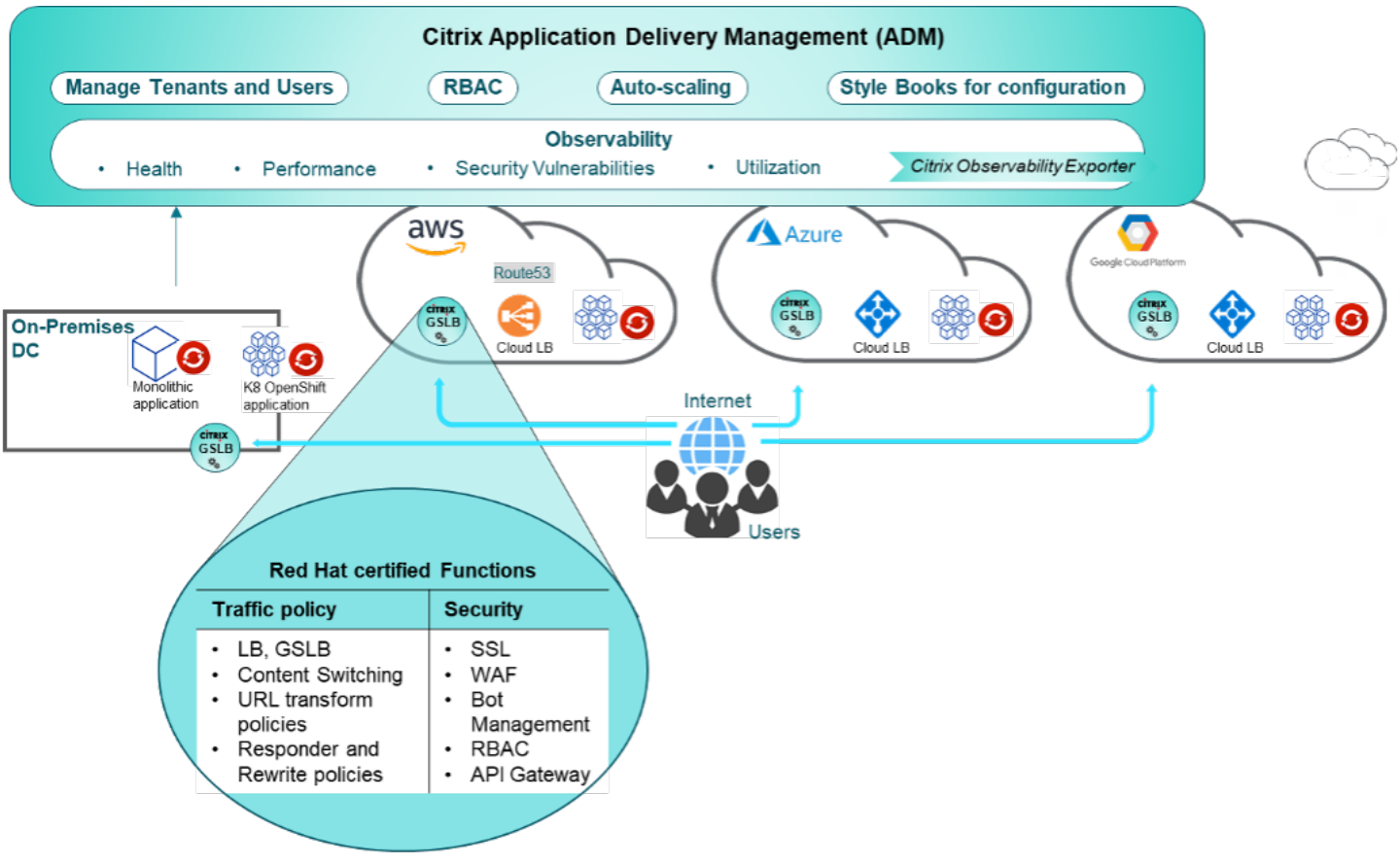
As cloud and platform teams take this journey, they are encountering prolonged complexity of managing multiple sites, cloud provider environments and application types. For a multi-cloud platform to be truly frictionless, it must provide:

- **Adaptive multi-cloud operations** that deliver a consistent experience across multiple public cloud providers and traditional infrastructure. Includes continuous delivery (CD) for evolving applications. See figure 1.
- **Fast-cycle development, configuration and deployment** using continuous integration and continuous delivery (CI/CD). See figure 2.

In traditional IT, these requirements align with "Day 2" operations and "Day 1" install-configure-deploy, although they overlap as CI/CD practices are adopted. As we'll see, Red Hat and Citrix fully address both sets of requirements through integration. Red Hat's OpenShift Container Platform offers platform consistency and compliance features to deploy Kubernetes apps across cloud providers. Red Hat Ansible radically simplifies IT automation of cloud provisioning and configuration management. This is ideal for three-tier applications and for migrating traditional applications to hybrid cloud.

A critical complement to Red Hat OpenShift cloud deployments is the Application Delivery Controller (ADC). Originally ADCs simply provided load balancing and traffic steering. As applications become connected across the internet, ADCs became the natural gatekeeper for all traffic to/from applications. This

Figure 1: Adaptive multi-cloud operations showing Red Hat and certified Citrix elements



enabled ADCs to evolve rapidly to assist in every aspect of application migration and cloud operations including security, performance, scaling, troubleshooting and availability. Today's ADCs can drive IT speed, costs, and compliance.

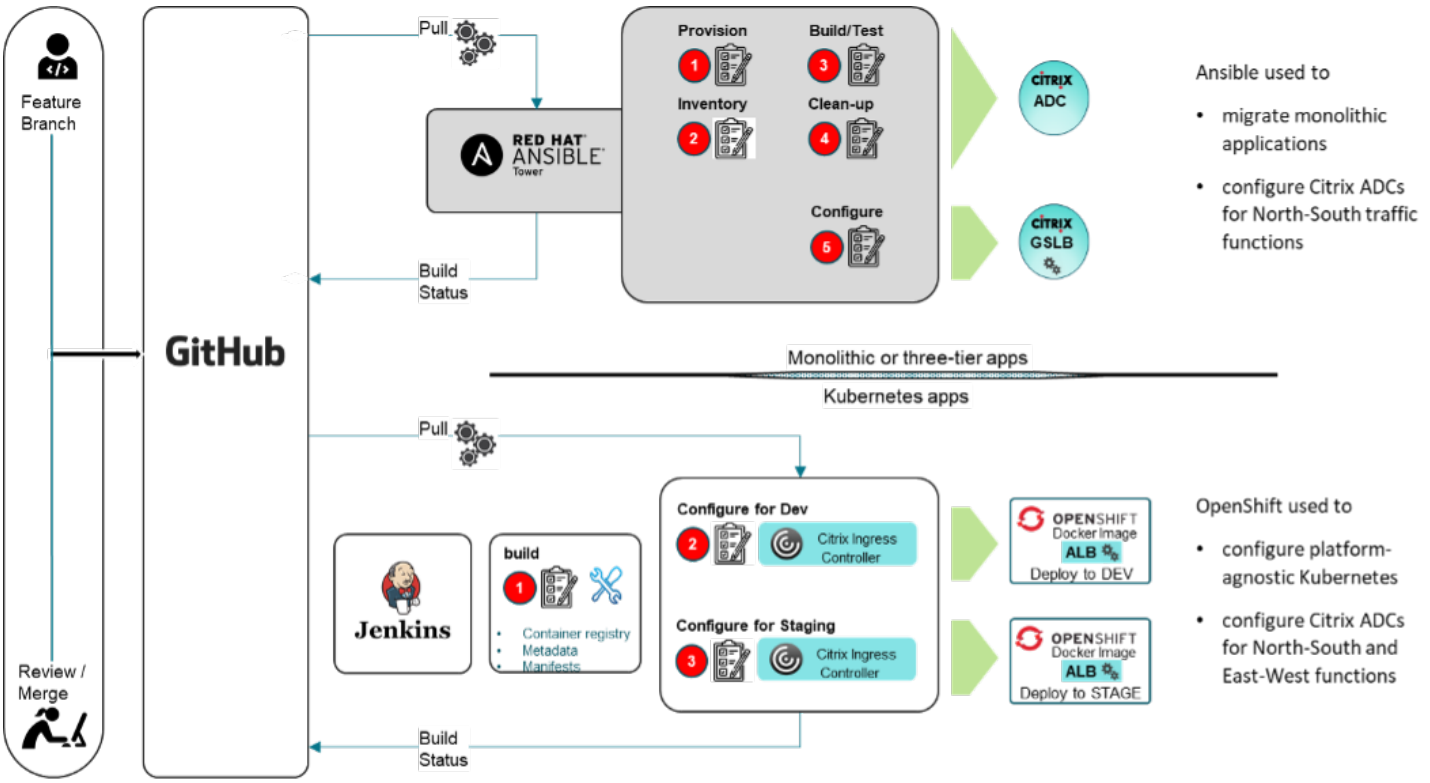
Citrix ADCs are full-featured and Red Hat certified with OpenShift and Ansible. Using Citrix Application Delivery Management, they optimize not only the user experience but also the operational metrics in delivering it, at dynamic scale and performance, securely from any device. The integrations and certifications give Red Hat customers the confidence to manage large multi-site operations with any application (monolithic or microservice-based) and the fastest cycles achievable for configuration and deployment.

With Citrix ADCs and Red Hat OpenShift working together, this idealized topology achieves operations across several sites, explicitly designed for high availability, security, performance, and scale. Moreover, it optimizes these characteristics continuously, using real-time observability and management across Red Hat OpenShift clusters and monolithic applications.

- Red Hat® OpenShift® Container Platform is used to deploy Kubernetes clusters both in public clouds and on-premises.
- Citrix ADCs, direct traffic toward to the nearest or most available application, across sites, clusters, nodes and Pods -- simply by configuring them as Global Server Load Balancers (GLSBs), Application Load Balancers (ALBs) or Service-proxies.

Best of all, it is very simple to add rich security functions and sophisticated traffic policies (above) to Citrix ADCs in any deployment scenario. These functions can be added within Red Hat's CI/CD workflows, for both Ansible and OpenShift, using the Citrix playbooks and Citrix Ingress Controller (CIC), respectively. (The CIC is an augmented version of the Kubernetes Ingress Controller for managing traffic in to Kubernetes cluster using Citrix ADCs). The platform team can use infrastructure-as-code to easily configure all ADC

Figure 2: Fast-cycle development, configuration, and deployment using CI/CID workflows with Ansible Tower or OpenShift



functions regardless of whether they are supporting monolithic applications (including three-tier apps) or Kubernetes microservices. (see Figure 2).

Two workflows are shown. While the pull mechanism from GitHub differs, the overall CI/CD workflow is similar between monolithic and Kubernetes application development. In the upper workflow, Red Hat Ansible Tower is a platform for Ansible that merges infrastructure operations and DevOps jobs into one workflow. It provisions and configures virtualized infrastructure that is not yet containerized, using declarative code. Here, Infrastructure Ops provisions the virtual infrastructure and then DevOps configures Citrix ADCs as a GSLB. Citrix provides templates for playbooks for all ADC functions that greatly simplify the effort.

In the lower workflow, Red Hat OpenShift provisions and configures Kubernetes clusters using declarative code. Here, DevOps fully configures the Application Load Balancer using the Citrix Ingress Controller (CIC). The CIC also uses declarative code based on Custom Resource Definitions (CRD), templates and policy mappings to simplify complex Kubernetes deployment architectures such as ServiceMesh.

The key point is that with these augmented Red Hat workflows, it is possible for ADCs and applications to be provisioned and configured together with a high level of CI/CD automation. This enables cloud teams to deploy robust HA and security functions as they transition to cloud based modernized applications.

ADC Functions	North-South	North-South and East-West
	Ansible Tower	
	OpenShift	
Single cluster e.g. ALB	Local HA & Failover	Horizontal POD scaling
Multi-cluster e.g. GSLB	Global failover, autoscale	Global failover autoscale
Security	N-S policies/functions	+ E-W policies/functions

The following section will show four OpenShift architecture options, all using Red Hat certified components. They progressively bring additional visibility and control to a global multi-cloud network, with only minor increases in complexity. The remaining sections will show how Citrix optimizes operations with management, security, and scale at every point on the cloud native journey. Specifically, we'll discuss:

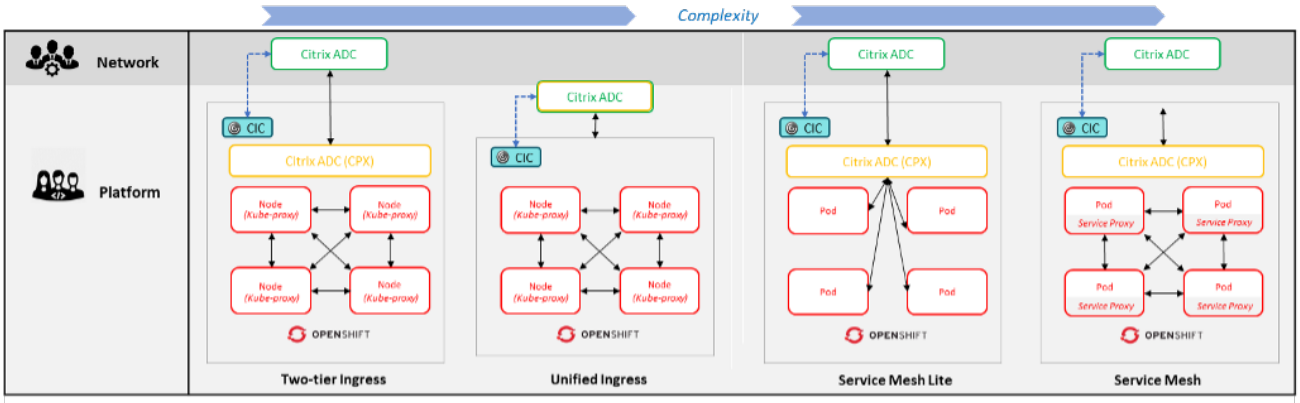
1. **OpenShift architectures:** Specifying the level of ADC mediation between Kubernetes PODs and Nodes, to maximize north-south and east-west visibility, availability, security and performance and scalability.
2. **OpenShift** failover scenarios: Using failover, policies and triggers for Control plane HA and application migration
3. **Multi-site observability and management:** Using telemetry to troubleshoot and adapt operations in real-time.
4. **Comprehensive Security Posture:** Easily configuring a layered security stack and deployment specific features such as Rate limiting, Authentication, Authorization, SSL offload and east-west communication policies to enable zero-trust.
5. **Performance and Scale:** Using multi-cloud observability to improve response time. Also includes many raw performance enhancements to Citrix ADCs

OpenShift Architectures

The most challenging part of building microservices has been choosing and configuring the right application deployment architecture. Organizations migrating to cloud native often sacrifice operational control and responsiveness to match their current team's skill set. But this tradeoff is greatly improved by the augmented OpenShift workflow with Citrix components.

Here we summarize the four common application delivery architectures in order of increasing complexity and benefit, dual-tier ingress, unified ingress, ServiceMesh Lite and ServiceMesh.

While visibility, control and complexity increase from left to right, Citrix ADCs export data to third-party tools like Prometheus, Grafana and Zipkin, natively, for all four architectures. The first two deployment architectures (Two-tier Ingress and Unified Ingress) provide visibility



Visibility	▪ North-South	+ East-West
Traffic	▪ Layer 4-7 Load balancing ▪ Observability export to popular opensource analytics ▪ CI/CID tools (e.g. Spinnaker)	 + Granular Observability/Troubleshooting + Control plane (e.g. Istio)
Security	▪ SSL Termination ▪ Web Application Firewall (WAF) ▪ Authentication	+ Inspect interpod traffic + Stop threat propagation with granular "Allow list" policies + Offload repetitive functions (e.g. encryption)

only to North-South traffic. Even with this constraint, scalability is excellent. Citrix ADCs can reach hundreds of Gbps—or even Tbps—throughput through active-active clustering of ADCs if required. With the right proxy, SSL termination can be done at the edge, and traffic can be inspected easily. This enables N-S traffic to be comprehensively secured across L3-7. ADC collects and exports telemetry on the N-S application traffic it sees, to a variety of open source and internal tools which means that these architectures can provide robust observability. ADC can also integrate with CI/CD tools like Spinnaker to provide excellent traffic management and continuous deployment capabilities.

Two-tier ingress has one more hop to get to the backend servers than unified ingress but it requires a lower skill set. Two-tier ingress facilitates the current separation of roles that may exist between the networking and platform/application teams. The network team can own and manage the green ADC, and the platform/application team can work inside the Kubernetes environment. As more apps get onboarded, the delegation of roles with two-tier architecture can help with effective manageability and easier scaling. Conversely, with unified ingress, both the ADCs for N-S traffic and kube-proxy for the E-W traffic are managed by the platform team, who must be more network savvy to implement and manage this architecture.

A unified ingress proxy architecture can participate in the Kubernetes cluster's overlay network. This allows the external network to communicate with the microservices pods. Therefore, the platform team must be knowledgeable about layers 3-7 of the network stack to take full advantage of this architecture. However, as we will see, the CIC overcomes much of the complexity of Ingress while providing extensibility for protocols beyond HTTP/HTTPS.

Learn more about two-tier ingress and unified ingress.

The second two deployment architectures, Service Mesh Lite and Service Mesh, also provide visibility to East-West traffic between Kubernetes Pods. A service mesh is a dedicated infrastructure layer to control how different parts of an application communicate with one another. The service mesh landscape is seeing gradual adoption as customers become comfortable with the technology. It does offer the best observability, security, and fine-grained management for traffic among microservices—that is, for E-W traffic.

With Service Mesh, the microservice-to-microservice communication is enabled by a sidecar to each microservice pod (see Service Proxy in the Service Mesh architecture) or by a Citrix CPX (in the Service Mesh Lite architecture). In the latter case, all east-west traffic goes through Citrix CPX. In the former case, the sidecar

can also be a Citrix CPX, with very lightweight CPU and memory needs.

The presence of a ServiceProxy or cluster CPX enables the platform team to set security policies and control communication between microservices and logical groups of microservices. The team can operate in a zero-trust mode, i.e., block all communications except those explicitly allowed, thus dramatically reducing internal threat propagation. It can also mandate things like authentication, encryption, and rate limiting for APIs among microservices. Finally, it can provide sophisticated load balancing policies across Pods. Compared to two-tier and unified ingress, the limitations of kube-proxy are removed – namely lack of E-W observability, the lack of control of inter-Pod traffic and lack of granular load balancing.

Both Service Mesh and Service Mesh Lite are managed by the platform team and create a highly scalable, distributed architecture, but they also add complexity (and overhead) because of potentially thousands of sidecars in a large cloud native environment. As before, Citrix Ingress Controller can overcome much of the configuration complexity with highly repeatable configurations. Further, Citrix ADM can overcome the management complexity, as we'll see later.

Learn more about Service Mesh and Service Mesh Lite.

Using these deployment architectures, it is easy to complete the function table from the introduction.

For example, a Single cluster with east-west topology with Global Load Balancing and sophisticated failover policies can be configured simply by using declarative code that specifies these parameters for a Service-Mesh Lite architecture.

In each case, CIC can help build increasingly complex operational topologies without much additional knowledge. This helps the network team but is particularly helpful to the platform team to create sophisticated hybrid multi-cloud network topologies. The platform team can specify unified Ingress – where layer 3-7 traffic policies must be configured or Service Mesh/Service Mesh Lite, where inter-process communication, authentication, encryption, rate-limiting and other capabilities can be configured.

OpenShift failover scenarios

Configuring failover with appropriate local or global policies can streamline common operations. Here we describe control plane high availability and application migration from OpenShift 3.x to 4.x

Load balancing the control plane

Another example is load balancing the control plane, which is mandated to deploy Enterprise OpenShift even before the cluster comes up. To meet this requirement, multiple master nodes are needed in Active-Active mode to achieve a high availability control plane. Using CIC, the Citrix ADC can load balance between the master nodes. The network team creates the IP address, DNS entries and provides information to URLs. The application team runs the OpenShift installer that installs the cluster.

Migrating from OpenShift 3.x to 4.x

Another use of failover includes OpenShift version 3.x to 4.x migrations. There are typically two approaches:

- **Primary to Secondary:** Assume the primary application is in production on OpenShift 3.x. The

ADC Functions	North-South only	Architecture	North-South and East-West	Architecture
Single cluster e.g. ALB	Local HA & failover	Two-tier Unified Ingress	Horizontal POD scaling	ServiceMesh Lite ServiceMesh
Multi-cluster e.g. GSLB	Global failover, autoscale		Global failover autoscale	
Security	N-S policies/functions		+E-W policies/functions	

platform team would run the same instances on 4.x in standby mode. Then the team would shut down the app on 3.x. The ADC will detect non-response from the app if it has been configured with global traffic policy triggered by response time. Then it will automatically failover to 4.x and make it active. This approach requires the least overhead from the platform team.

- **Gradual loading using Canary:** Here the ADC shifts traffic to the secondary cluster, perhaps 10% at a time until its fully loaded. After each incremental shift, ADM can be used to make sure there are no errors (e.g. 5xx errors) before loading the next increment. This approach is ideal for sites with time critical information or larger user communities because it limits customer dissatisfaction with any errors.

Multi-site Observability and Management

Observability features provide a uniform view of the production state across all sites, application types, and network dependencies. They actively debug and optimize the system by showing real-time patterns, not previously anticipated. A single-pane customizable dashboard monitors health, performance, utilization, and security vulnerabilities. It uses the metrics to create actionable service graphs and export to analytics tools such as Prometheus, Grafana, Elasticsearch, Kibana and Zipkin using Citrix Observability Exporter (CoE). Together with the Citrix GSLB controller, this enables rapid troubleshooting, capacity planning and optimization, such as redirecting traffic to an alternative geographic cluster.

Citrix Application Delivery Management (ADM) provides operational consistency across clusters and monolithic applications for managing tenants and users, implementing RoleBased Access Control (RBAC), auto-scaling, and rapidly configuring new repeatable instances of the cluster using simple composable stylebooks. With Citrix Application Delivery Management, it is easy to compare the performance of applications before and after migration to the cloud or after a migration to a new public cloud. Similarly, Citrix ADM can define, enforce, and monitor a common security posture for all applications across all public cloud and on-premises environments.

Further, Citrix ADM's intelligent analytics uses machine learning to understand application behavior characteristics across all deployments to automatically spot anomalies to reduce the turnaround time for troubleshooting issues and improving the overall application uptime.

Comprehensive Security Posture

Because security is a shared responsibility with cloud providers, it is difficult for application owners to see and apply common security policies across clouds that behave differently. For example, multi-cloud/hybrid operations can create a massive attack surface for malware that can use data paths intended for inter-application data. Clearly, there is a need to enforce and maintain a consistent security posture across all deployments.

Citrix Management and Observability drills down into specific applications to investigate communication polices and vulnerabilities as well as the client experience. Global Service Graphs offer a graphical end-to-end view of both monolithic and microservices-based applications.

Citrix ADCs can provide a layered security stack that integrates Bot management, DDOS mitigation, Web Application Firewall (WAF), API protection and best in class TLS encryption performance. Citrix offers these bundled as a service called Citrix Web App and API security (CWAAP) or as individual functions. With Red Hat Ansible Tower and OpenShift, developers can specify these functions declaratively:

- **Rate Limiting:** limits resource consumption as a policy e.g. "not more than 1000 requests/sec so that DDOS attacks cannot escalate
- **Authentication (Authn, Authz):** authentication and authorization to use internal APIs by all types of endpoints including web-based applications, mobile

apps, or IoT devices (Internet of Things)
- **SSL termination and re-encryption:** offload processing from app server; enable traffic inspection.
- **East-West security in service mesh:** Citrix is certified with the OpenShift service mesh, which uses Istio.

With OpenShift, WAF and rate limiting can be configured down to the pod level using proxies inside the cluster (CPX). Citrix ADM also provides holistic visibility to the type, volume, and origin of attacks against your applications wherever they reside. It even shows which apps are most at risk. Operations can prioritize remediation accordingly. Citrix ADC, in conjunction with Citrix Application Delivery Management, can define and enforce a consistent security posture across all applications and all hybrid cloud environments.

The analytics engine ensures policies are enforced. It tracks violations including config drift using the powerful SSL governance and enforcement tools included in Citrix Application Delivery Management.

Performance and Scale

When selecting the right ADC for public cloud deployments, scale and performance are important factors. There is always a need to scale applications in response to fluctuating demand. Under-provisioning may lead to lost customers, reduced employee productivity and lower revenue. Right-sizing infrastructure on demand is even more important in the public clouds where over provisioning is costly and typically goes against the design tenants of a scalable architecture.

A multi-cloud topology offers inherent performance benefits by taking advantage of proximity to the geographic areas with the highest demand. Citrix ADCs integrate with cloud provider load balancers and cluster capacity so that both will scale as needed in response to a flash spike and steer traffic to the closest cloud.

In addition, Citrix ADC stand-alone performance excels due to several features. This includes auto scaling up to 100 Gbps/region and because of its superior software architecture, it delivers a latency advantage of 100ms on a typical eCommerce webpage compared to other ADC vendors and cloud provider options. Citrix ADCs performance leadership has been benchmarked and independently tested by the Tolly Group. Citrix's content compression and caching saves bandwidth and server resources. This takes two forms. Auto-scaling inside the cluster or outside the cluster for non-Kubernetes apps with public cloud providers. Learn more about autoscaling with AWS. Learn more about Autoscaling with Microsoft Azure.

Other features enhance performance by optimizing the content delivery to the user

- Integrated Caching: Improves server response by offloading back-end resources from responding to every request. Caches responses to client requests in memory – which also saves hops to the origin server, reducing application response time and back-end overhead and costs.
- TCP optimization: Improves download speeds, especially in mobile networks, through advanced TCP tuning such as window scaling, selective acknowledgement, maximum segment size, buffer size. Supported TCP architectures include BIC, CUBIC, Westwood, and Nile.
- HTTP Compression: Improves download speed and reduces network demand by compressing HTTP responses received from the servers and send them to browsers.
- HTML Compression: Improves page download and render times and reduce network demand by compressing HTML content sent to the client. Includes combining CSS files, removing whitespace within CSS or JavaScript, and converting GIF images to PNG.
- Domain Sharding: Improves page render and download speed by concurrently downloading content split across multiple sub-domains.
- Image optimization: Improves page render times by tailoring image size to the device, automatically. For mobile devices, for example, Citrix ADCs can intelligently reduce image resolution without sacrificing apparent quality.
- The above is a subset of a Citrix ADC feature-set called Front-End Optimization which is designed to optimize the performance of web applications

(monoliths and microservices).

Why only Citrix and Red Hat

Using the rich capabilities of Citrix ADC and ADM together with Red Hat's CI/CD workflows, customers can easily design and implement highly performant and highly resilient modern applications across public clouds and on-premises. Customers can create simple, repeatable delivery of the most powerful, scalable multi-cloud deployment architectures, such as Service Mesh, without a lot of new skill set building for their platform teams. They can configure policies that makes the multi-cloud network perform optimally and adapt quickly to geographic demand peaks. During day-to-day operations, they can use Citrix Observability and Management to bring operational consistency across their sites. Finally, they can implement a complete layered security stack with policies that identify threats in both North-South and East-West traffic. In addition, for customers that have additional compliance requirements, Citrix can provide reference architectures with appropriate Federal Information Processing Standards (FIPS) ready solutions to accompany the North-South traffic requirements.

Conclusion

As businesses modernize applications and orchestrate multi-cloud deployments, they are encountering prolonged complexity from managing multiple sites, cloud provider environments and application types. Red Hat OpenShift provides a single platform-agnostic Kubernetes distribution to deliver modern applications on-premises and across multiple public cloud providers. Red Hat Ansible simplifies the IT automation and migration journey from monolithic applications, with IT automation of cloud provisioning and configuration management.

Due to their pervasive use as an intermediary for traffic to/from connected applications, and their typical location in the topology, ADCs are a natural control point for network security, availability, performance and scalability. To maximize these metrics, cloud teams must configure ADCs together with the applications they are serving. As a preferred certified partner for both OpenShift and Ansible, Citrix ADCs simplify and accelerate Red Hat's CI/CD configuration workflows with custom playbooks, ingress controllers, resource definitions and policy mappings. In addition, Citrix brings consistency and real-time adaptive control to global multi-cloud operations with observability and management. With Citrix and Red Hat integrations, cloud teams can measurably drive IT speed, costs and compliance.

Learn more

Citrix: The Basics of Cloud Native Applications

Red Hat: Citrix Ingress Controller for OpenShift

Red Hat: Certified Integration: Ansible and Citrix ADC

citrix

Enterprise Sales
North America | 800-424-8749
Worldwide | +1 408-790-8000

Locations
Corporate Headquarters | 851 Cypress Creek Road, Fort Lauderdale, FL 33309, United States
Silicon Valley | 4988 Great America Parkway, Santa Clara, CA 95054, United States