Load Balancing

Standards and Shared Practices Document

Document Control Information

Document Information

Document Identification	
Document Name	Load Balancing Standard
Project Name	
Client	Yale University ITS
Document Author	E. Camden Fisher
Document Version	1.0.0
Document Status	Ratified
Date Released	May 17, 2019

Document Edit History

Version	Date	Additions/Modifications	Prepared/Revised By
0.0.1	April 7, 2019	Initial Draft	E Camden Fisher
0.0.2	April 15, 2019	Review Recommendations	E Camden Fisher
0.0.3	May 2, 2019	Review Recommendations	E Camden Fisher
0.0.4	May 3, 2019	Review Recommendations	E Camden Fisher
1.0.0	May 17, 2019	Ratified by TAC	E Camden Fisher

Document Review/Approval History

Date	Name	Organization/Title	Comments
April 08, 2019	John Coleman	Senior IT Forensic Security Analyst	
April 10, 2019	Lisa Sawin	Director of Web Technologies	
April 10, 2019	Louis Tiseo	Sr. Director, Foundational Technologies Services	
April 11, 2019	Louis King	Enterprise Architect	
April 11, 2019	Morrow Long	Principal Security Officer & Director of Engineering & Forensics	
April 11, 2019	Andrew Newman	Director, Design Services	
May 3, 2019	Ken Hoover	Sr. Solutions Architect	
May 3, 2019	Andrew Guirguis	Senior Cloud Automation Engineer	
May 3, 2019	Darrell Cook	Director, Application Services	
May 17, 2019	TAC		Ratified

Distribution of Final Document

The following people are designated recipients of the final version of this document.

Name	Organization/Title
Louis Tisseo	Sr. Director, Foundational Technologies Services
Andrew Newman	Director, Design Services
Kay Ratanasaka	Manager, Cloud Engineering
Michael Dunlap	Manager, Linux Systems Administration

Table of Contents

Introduction and Overview

This document is one of many technology standards developed as part of the *Operational Excellence* initiative. This standard's focus is to support consistent, repeatable, secure and reliable service delivery across Yale Information Technology Services by documenting a standard set of practices for designing load balancing configurations.

As referenced in this document, a *load balancer* is a reverse proxy that distributes network or application traffic across a set of servers or services. Load balancers are generally used to improve application performance and availability. It is also common practice to use load balancers to present services running on internal networks to lower security networks (such as campus or the Internet).

Intended Audience

The intended audience for this document is architects and technical leads within Yale ITS. The document should be referenced when designing solutions that require or might require load balancing. Possible indicators for load balancing:

- Requirement to scale a service or application horizontally
- Vendor recommendation
- Exposing servers or services in Yale's data center or Yale's cloud providers to the Internet or to campus
- Exposing services that terminate SSL and/or require SSL certificate management

Governance and Compliance

The awareness of these standards and shared practices along with the adherence and adoption of these standards will be governed and administered through a variety of channels.

- **TAC**: The Technology Architecture Committee (TAC) helps ensure that there is awareness and visibility for this and other ratified technology standards. It is responsible for ratifying and aligning technology standards with projects passing through the committee.
- **PMO**: The Project Management Office (PMO) plays an active role in understanding what projects need engagement with architects and subject matter experts as well as how projects should be engaging with governing bodies.
- **ISO**: The Information Security Office (ISO) engages as a mechanism to ensure projects adhere to predefined security and risk standards through processes such as the SDR.
- **TAST**: The Technology Architecture Standards Team (TAST) identifies areas for standards and shared practices across Yale's technology landscape and supports the development and adoption of technology architecture standards.

Scope

Although some conventions and procedures exist for load balancing configurations within ITS, in practice, many decisions are made at the engineering and operational level on an ad hoc basis. This results in solutions that differ over time and produces sub-optimal results with regard to security posture and/or application performance.

This document provides a set of recommendations based on security classification, application topology and application requirements. The load balancing technologies in scope for this document are F5 BigIP appliances in our Equinix and West Campus datacenter as well as cloud native load balancing technologies available in Amazon Web Services and Microsoft Azure.

Assumptions

When discussing load balancers within Yale ITS, we are referring to Layer 4 or Layer 7 reverse proxies. DNS based, or other forms of load balancing are out of scope for this document.

Constraints and Limitations

Only local forms of load balancing are in scope for this document. Yale does not currently engage in any cross-region/global load balancing. Technologies such as F5 or Azure Global Traffic Manager or AWS Global Load Balancing using Route 53 routing are out of scope for this document.

Dependencies

- Network connectivity between load balancing devices and application origin servers or services is required for “health checks” and proxied traffic.
- Network connectivity for the load balancer from the expected sources is required for clients to reach load balanced applications.
- DNS services may be required for clients to reach load balanced applications but may not be required for the load balancer to reach application origins.

Guiding Principles

Enterprise Specific Guiding Principles

Enterprise guiding principles should be applied and are referenced throughout many technology standards at Yale. These enterprise standards should also be applied to the load balancing standards.

Document	Document Location
Enterprise Guiding Principles	https://yaleits.atlassian.net/wiki/spaces/STAN/pages/781156666/ITS+Technology+Enterprise+Architecture+Principles

Service Specific Guiding Principles

Load Balancing is **adjacent to the origin**

Description: Load balancing should be done as close as possible to the application origin that is being load balanced.
Rationale: Load balancing technologies perform best when located as close to the application origin as possible. Most load balancing technologies perform "health checks" against origin servers, which can return false negatives when origin services are distant or logically separated from the load balancer. Adjacency to origin servers may also reduce the risk of traffic interception on compromised devices and problems caused by network segmentation.
Implications: Ideally, the load balancing device(s) should exist on the same Layer 2 network as the origin. When this is not possible, load balanced traffic should traverse as few network devices as possible.

Load Balancing is **symmetric**

Description: Traffic passing through the load balancer from a client will return back to the client via the same load balancing device.
Rationale: Load balancing solutions utilizing "direct routing" or "direct return" are overly complex and error prone and unsupported for cloud native load balancing. The performance capabilities of modern load balancers obviate the need for direct routing in all but the most demanding situations (such as high bandwidth video streaming).
Implications: Avoid the use of the direct routing or direct return load balancing type.

Load Balancing is **encrypted**

Description: Load balanced traffic must be encrypted in transit.
Rationale: Traffic between the client and the load balancer as well as traffic between the load balancer and the origin servers should be secure and unmolested in transit. Where possible, public certificates should be served and managed by the load balancer. Encrypted traffic should be terminated/decrypted at the load balancer and re-encrypted using certificates signed by an internal certificate authority (CA) between the load balancer and the origin server. This decryption may increase observability by security pipelines and may also enable application layer (ie. HTTP) management of load balanced traffic.
Implications: Load balanced services will need two certificates -- one hosted on the load balancer and one hosted at the origin. Certificates hosted at the origin may be self-signed or certificates signed by an internal certificate authority (CA).

Load Balancing is **consistent across environments**

Description: The same load balancing technology should be used across all environments for a given application or service. Ideally, load balancing is also done in the same location for all environments of a given application or service.
Rationale: Consistent behavior across all environments of an application is required for effective testing and validation. Even at Layer 4, there is no guarantee that a cloud based load balancer will behave the same way as an on-premises/hybrid load balancer.
Implications: When using on-premises or hybrid load balancing, all environments for a given application (dev, test, prod, etc) should use the same on-premises or hybrid load balancing technology. When using cloud native load balancing, all environments for a given application should use the same cloud native load balancing technology **in the same cloud.**

Load Balancing is **observable**

Description: At a minimum, network flow data should be available for all load balanced traffic. In some cases, access logging should also be made available.
Rationale: Observable traffic flows are easier to debug and also enable scanning for malicious intent. Traffic flows ingested into event processing pipelines can be used to detect (and eventually disable) attacks against Yale services and infrastructure as well as enabling application developers and administrators to trace and debug more effectively.
Implications: Hybrid and on-premises load balanced traffic should pass through a Yale managed firewall to allow for the collection of flow data. Flow log collection should be enabled for cloud native load balancing within the cloud provider.

Load Balancing Architecture Decisions

Yale has made significant investments in F5 load balancing technology in purchasing hardware and building internal knowledge. Using this approach, Yale ITS has been very successful at terminating much of Yale's web access at F5 devices hosted on-campus. Although this has worked well, we are starting to see the limits of this approach as we attempt to move applications to the cloud. As a side effect, we have started to see pockets of "cloud native" load balancing being used across ITS. The two categories of load balancing architectures discussed in this document will be referred to as *on-premises/hybrid* load balancing and *cloud native* load balancing.

On-premises/hybrid load balancing in use today within Yale ITS includes F5 BigIP devices in on-campus data centers or colocation facilities. It may also include other hardware load balancing technologies as well as software based/virtual load balancing technologies that are centrally managed. For the sake of this document, we are specifically excluding "fit to purpose" load balancing software that is implemented for a single application or service (ie. Ingress controllers on a Kubernetes cluster or Docker Swarm).

Cloud native load balancing in use today within Yale ITS includes Amazon Web Services Elastic Load Balancers/Application Load Balancers/Network Load Balancers and Azure Load Balancers/Application Gateways. These load balancers are software defined and run within their respective cloud. They are generally applied to a single purpose/single application and are billed by the hour (or second in some cases). Cloud native load balancers support automation and service discovery for services running within their respective cloud and can offer internal or external load balancing. In general, these technologies are less feature-rich than a hardware appliance such as the F5 BigIP, but have the advantage of horizontal scalability and configurability via an API. Load balancer feature sets are converging as time goes on. As this happens, scalability needs and proximity to origin servers/services will become the deciding factor for load balancer technology selection.

Recommendation Matrix

The following is a recommendation matrix to align the location of application origins and application security classifications with load balancing solutions.

	Low Risk	**Moderate Risk**	**High Risk**
Cloud based origin with advanced rulesets or Multi-Cloud origin	Hybrid load balancing in the Cloud Hub	Hybrid load balancing in the Cloud Hub	Hybrid load balancing in the Cloud Hub
Cloud based origin with simple rulesets	Cloud Native load balancing	Cloud Native load balancing	Hybrid load balancing in the Cloud Hub
On Premises or mixed On Premises/Cloud based origin	On-Premises load balancing in the Yale Datacenter	On-Premises load balancing in the Yale Datacenter	On-Premises load balancing in the Yale Datacenter
Cloud Hub based origin	Hybrid load balancing in the Cloud Hub	Hybrid load balancing in the Cloud Hub	Hybrid load balancing in the Cloud Hub

Data Risk Classification

Data is classified according to its sensitivity and importance to the functioning of the University. More information on classifying data can be found in the *Data Classification Policy*.

Data Classification Policy	https://your.yale.edu/policies-procedures/policies/1604-data-classification-policy

Common Architecture Patterns

Load balancing architectures used within ITS will likely fit into one of the patterns referenced in the *Recommendation Matrix*.

On-premises origin	The servers and/or services being load balanced are in a Yale datacenter or colocation facility considered on-premises. Currently, these are West Campus and Cyrus One.
Cloud hub origin	The servers and/or services being load balanced are in the Equinix facility (the “cloud hub”).

Cloud based origin with advanced rulesets	The servers and/or services being load balanced are in one of Yale's cloud providers (currently AWS and Azure) **and** require advanced rulesets (examples can be found in the *On-Premises/Hybrid Load Balancing* section of this document).
Cloud based origin with simple rulesets	The servers and/or services being load balanced are in one of Yale's cloud providers (currently AWS and Azure) and require only the simple load balancing rulesets provided by the cloud native load balancers in the provider.
Mixed on-premises, hybrid and Cloud based origin	The servers and/or services being load balanced are in one or more of Yale's cloud providers (currently AWS and Azure) **and** are in the Equinix facility and/or on-premises (West Campus and Cyrus One).
Multi-cloud origin	The servers and/or services being load balanced are in more than one of Yale's cloud providers (currently AWS and Azure).

On-Premises/Hybrid Load Balancing

On-premises/Hybrid load balancers are physical or virtual load balancing technologies running in one of our on campus datacenters or in one of our colocation facilities. Traffic traverses at least one Yale managed firewall on its way to these devices. Today, on-premises/hybrid load balancing is accomplished with F5 BigIP technologies.

F5 Local Traffic Manager (LTM) has the ability to manage traffic in a far more sophisticated way than cloud native load balancers at the cost of higher management burden and less automation. It is currently the only choice when advanced rulesets are required. LTM enables the control of network traffic for many common protocols and is programmable with iRules to allow for custom event based logic.

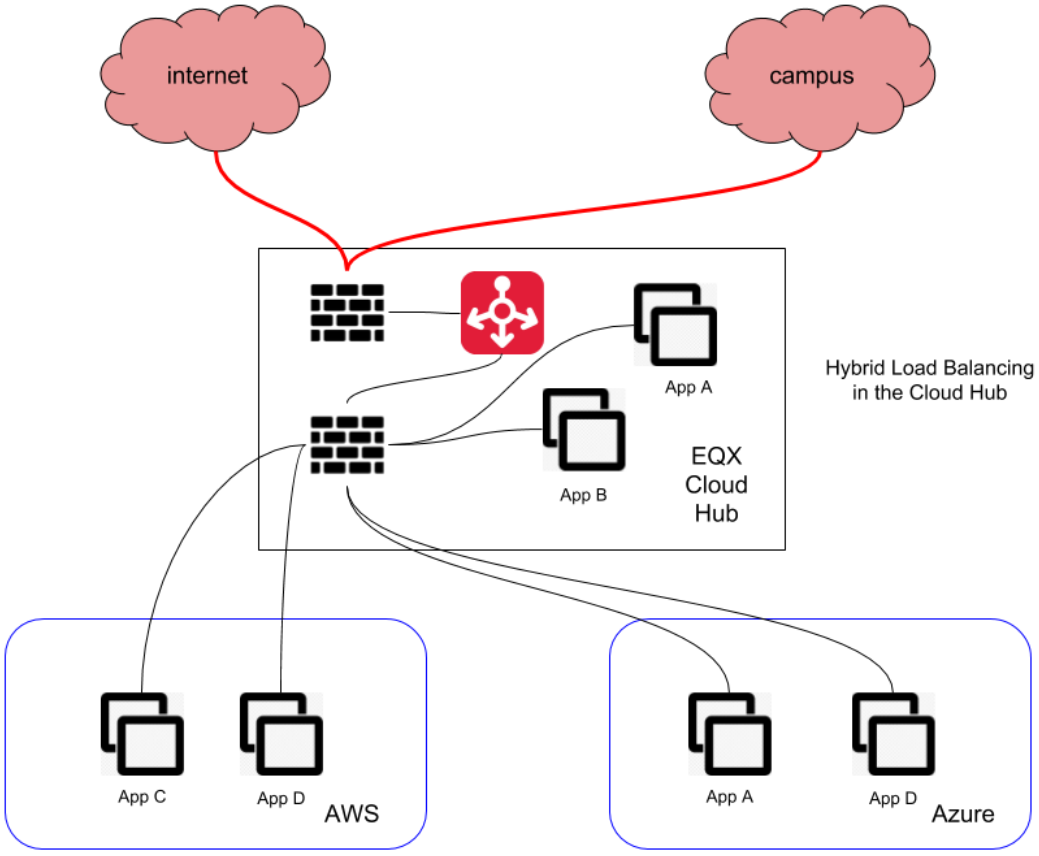
Referencing our principles, on-premises/hybrid load balancers should be selected when load balancing origins on campus/in our colocation facilities, across multiple clouds, or when advanced rulesets are required.

Examples of Advanced Rulesets

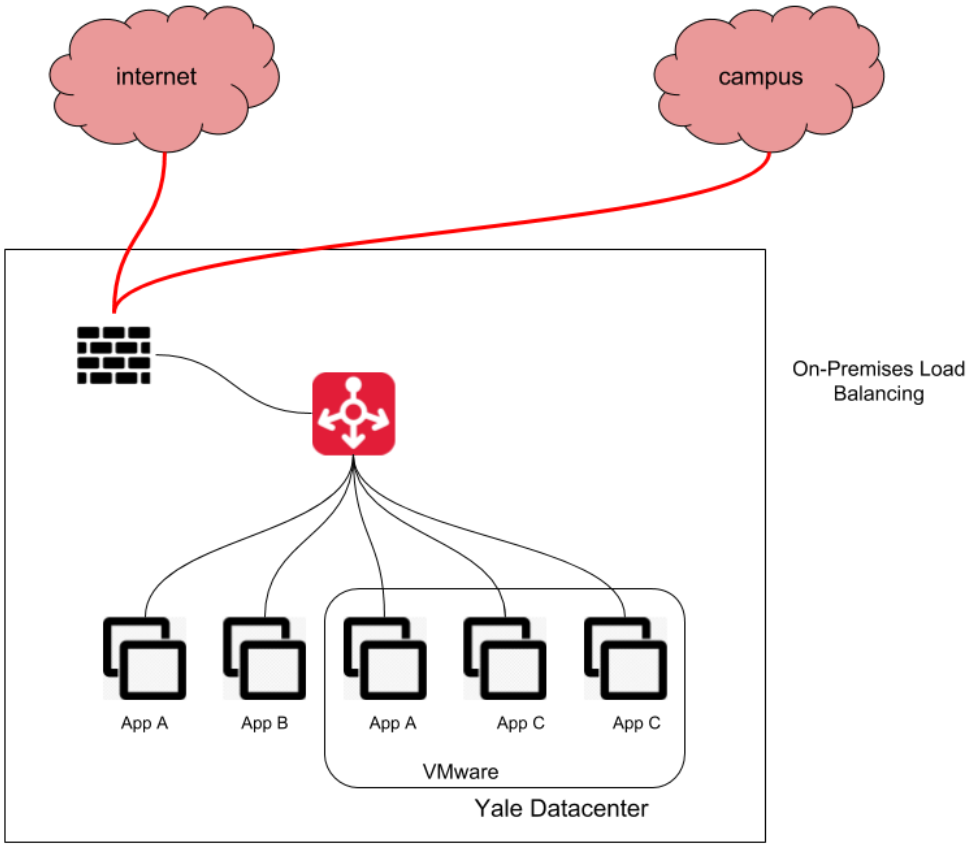
- Performance based
    - Profiles to enable caching and compression
    - Long lived/reused connections to the origin
    - Advanced load balancing algorithms (e.g., least connections or fastest response)

- Custom logic based
    - Complex rule based routing (e.g.., Source address, User-agent, etc.)
    - Redirection (i.e. 301/302)
    - Header injection/removal

Hybrid Load Balancing in the Equinix Cloud Hub



On-Premises Load Balancing



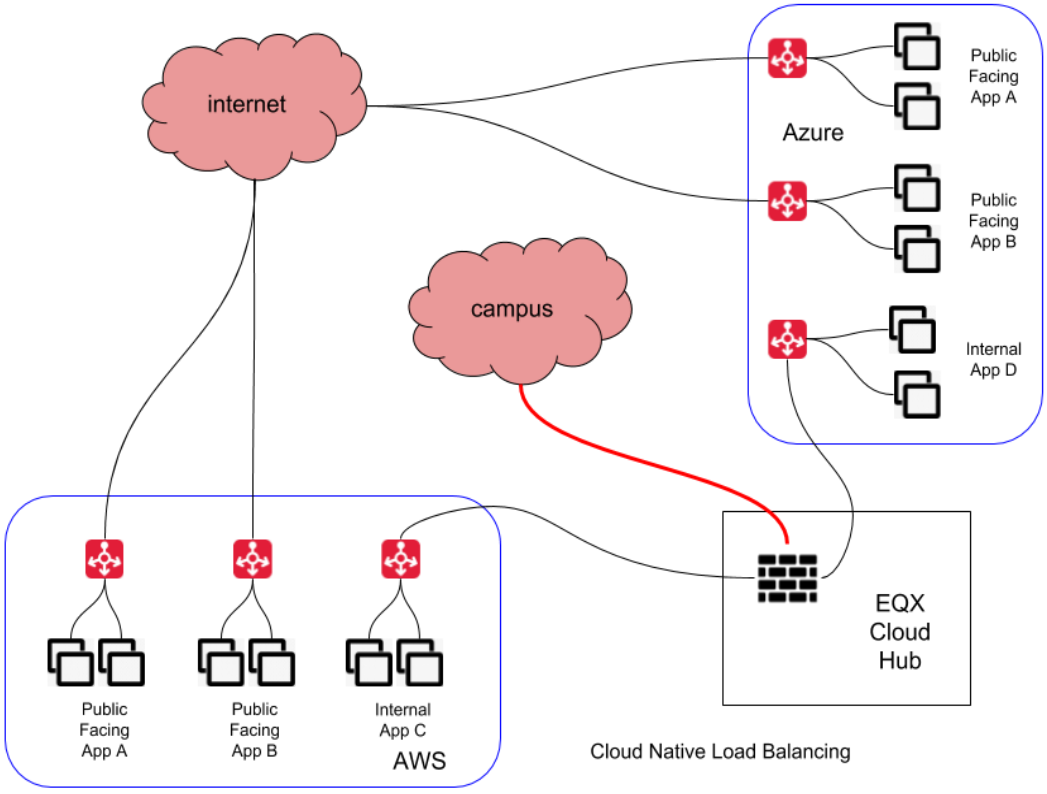
Access to the Service

Hybrid and On-Premises load balancing can be requested via the service catalog.

Load Balancing Service	https://yale.service-now.com/it?id=service_offering&sys_id=30688dcd6fbb31007ee2abcf9f3ee400

Cloud Native Load Balancing

Cloud native load balancers are software based load balancers provisioned in one of our supported clouds (currently AWS and Azure). Cloud native load balancers are horizontally scaled, transparent to the administrator and can handle traffic scale far beyond the needs of any application at Yale. In contrast to the on-premises/hybrid load balancing technologies discussed in this document, a single cloud native load balancer is generally associated with only one application. This has the advantages of mapping application lifecycle events to load balancer lifecycle events, isolation of changes, and independent scaling. It also presents better cost allocation opportunities and encourages the adoption of automation to manage the number of load balancers. Cloud native load balancers do not offer the same level of customization as hybrid or on-premises load balancing solutions in use today.



Cloud Native Load Balancing in AWS

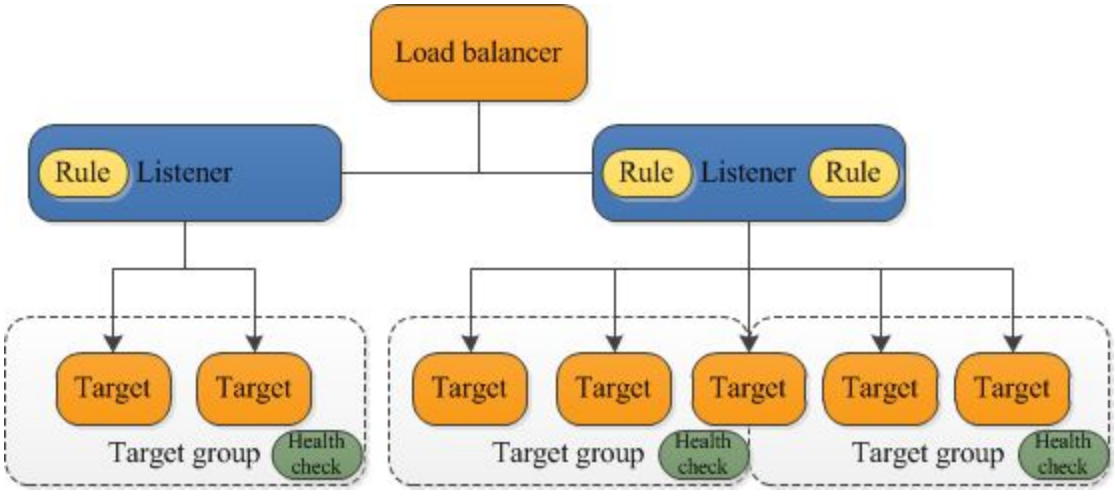
Amazon Web Services provides three flavors of load balancer to be used for workloads running within AWS. They are *Application Load Balancer (ALB)*, *Network Load Balancer (NLB)* and *Classic Load Balancer (ELB/CLB)*. Classic Load Balancers should not be used for services within Yale ITS and will not be discussed.

AWS native load balancers are able to scale automatically far beyond the needs of any service at Yale. Where possible, SSL certificates should be uploaded to AWS or acquired through AWS Certificate Manager. Those certificates should be applied to the AWS load balancer and traffic should be decrypted on the load balancer. Both ALB and NLB support SSL termination.

Application Load Balancer

Application Load Balancers are best suited for the load balancing of HTTP and HTTPS traffic. ALBs operate at Layer 7 and provide advanced request routing. Advanced request routing allows the selection of origin servers or services based on HTTP specific rules such as request path or HTTP headers. ALBs are also capable of integrating with service discovery services within AWS to automatically scale and add application origins to the load balancer. ALBs route traffic to targets within Amazon Virtual Private Cloud (Amazon VPC) and are also capable of targeting Lambda functions as an origin.

More information about Application Load Balancer can be found in the documentation: https://docs.aws.amazon.com/elasticloadbalancing/latest/application



Network Load Balancer

Network Load Balancers are best suited for the load balancing of Transmission Control Protocol (TCP) and Transport Layer Security (TLS) traffic. NLBs operate at Layer 4 and are able to pass millions of requests per second with very low latency. NLBs route traffic to targets within Amazon Virtual Private Cloud (Amazon VPC) and are the only choice when not load balancing HTTP or HTTPS. NLBs can preserve the source IP of client traffic to the origin.

More information about Network Load Balancer can be found in the documentation: https://docs.aws.amazon.com/elasticloadbalancing/latest/network

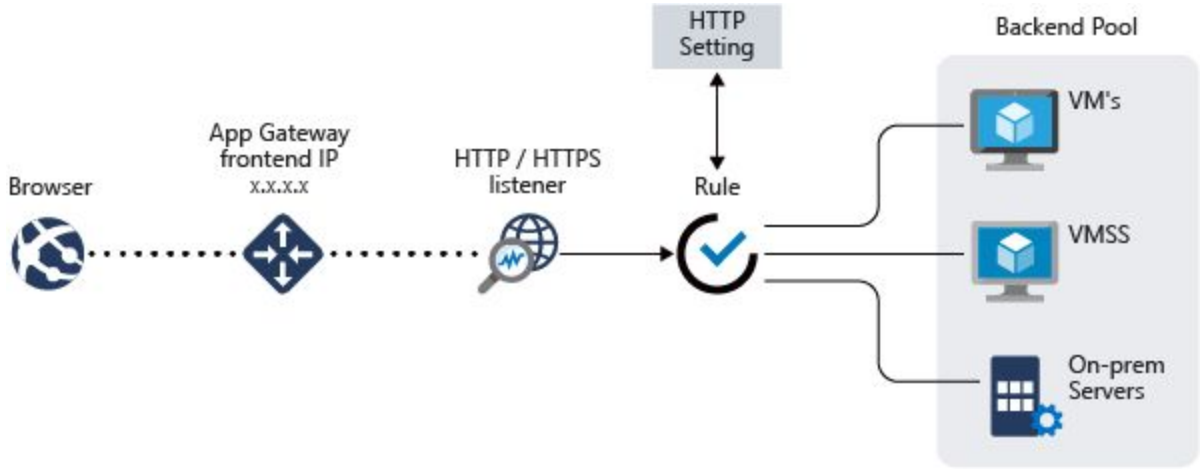
Cloud Native Load Balancing in Azure

Microsoft Azure provides two flavors of load balancers to be used for workloads running in Azure. They are *Azure Application Gateway* and *Azure Load Balancer*. Only Azure Application Gateway is able to terminate SSL traffic and is the best choice for load balancing HTTP or HTTPS services.

Azure Application Gateway

Azure Application Gateway is suited for use with HTTP or HTTPS traffic. They operate at Layer 7 and provides advanced request routing. Advanced request routing allows the selection of origin servers or services based on HTTP specific rules such as request path or HTTP headers. Some sizes (SKUs) of the Azure Application Gateway support autoscaling of the load balancer and Web Application Firewall (WAF) functionality.

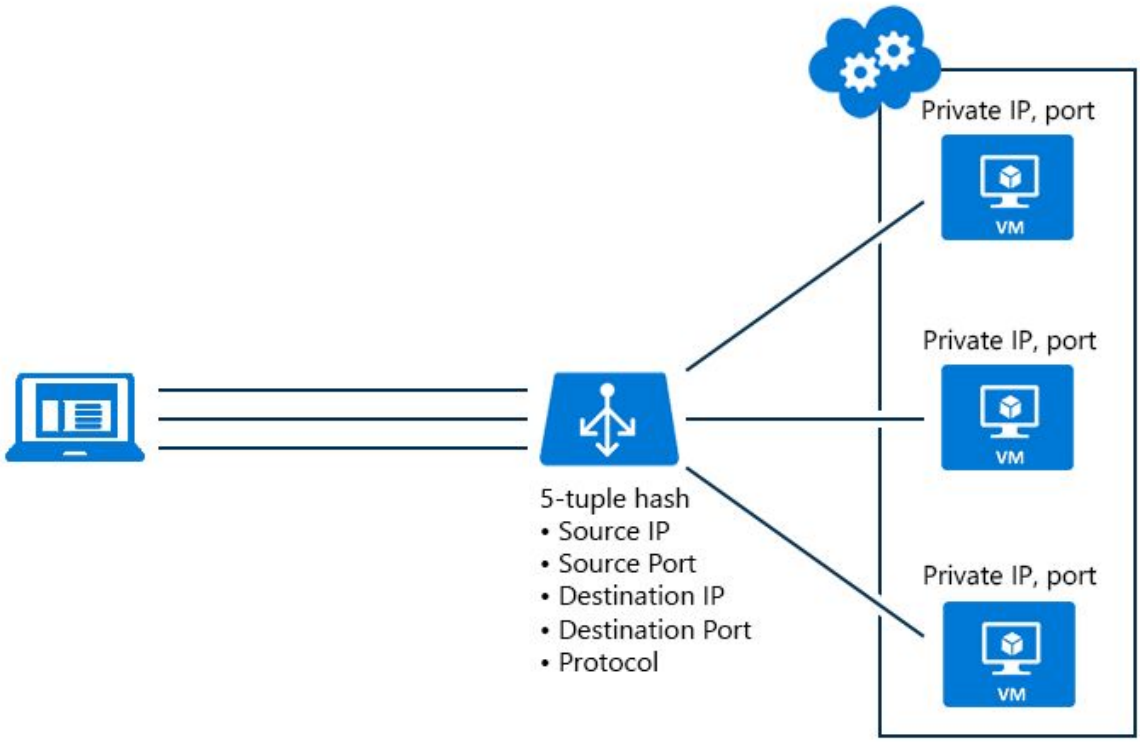
More information about Azure Application Gateway can be found in the documentation: https://docs.microsoft.com/en-us/azure/application-gateway/



Azure Load Balancer

Azure Load Balancers are best suited for the load balancing of Transmission Control Protocol (TCP) traffic. They operate at Layer 4 and offer basic load balancing. The Azure Load Balancer is offered in two sizes (SKUs) with differing capabilities. The *Basic* SKU is currently provided at no charge and is intended for simple use cases like a simple web server. The Standard SKU is more robust and can load balance traffic across any/all ports and protocols and is suitable for more complex workloads like clustered SQL servers.

More information about Azure Load Balancer can be found in the documentation: https://docs.microsoft.com/en-us/azure/load-balancer/


Private IP, port
VM
Private IP, port
VM
Private IP, port
VM
5-tuple hash
• Source IP
• Source Port
• Destination IP
• Destination Port
• Protocol

Access to Load Balancing

Service Description	Access Description
On-premises/Hybrid Load Balancing	Load Balancing Service in ServiceNow: https://yale.service-now.com/it?id=service_offering&sys_id=30688dcd6fbb31007ee2abcf9f3ee400
Cloud Native Load Balancing for Managed services	Through your managed service provider. **Linux:** https://yale.service-now.com/it?id=service_offering&sys_id=e8fa9965dbbe52403514f1c0ef9619ee **Windows:** https://yale.service-now.com/it?id=service_offering&sys_id=b4fa9965dbbe52403514f1c0ef9619f3
Cloud Native Load Balancing for Unmanaged services	Self-service through the cloud console or via the load balancing service in ServiceNow: https://yale.service-now.com/it?id=service_offering&sys_id=30688dcd6fbb31007ee2abcf9f3ee400