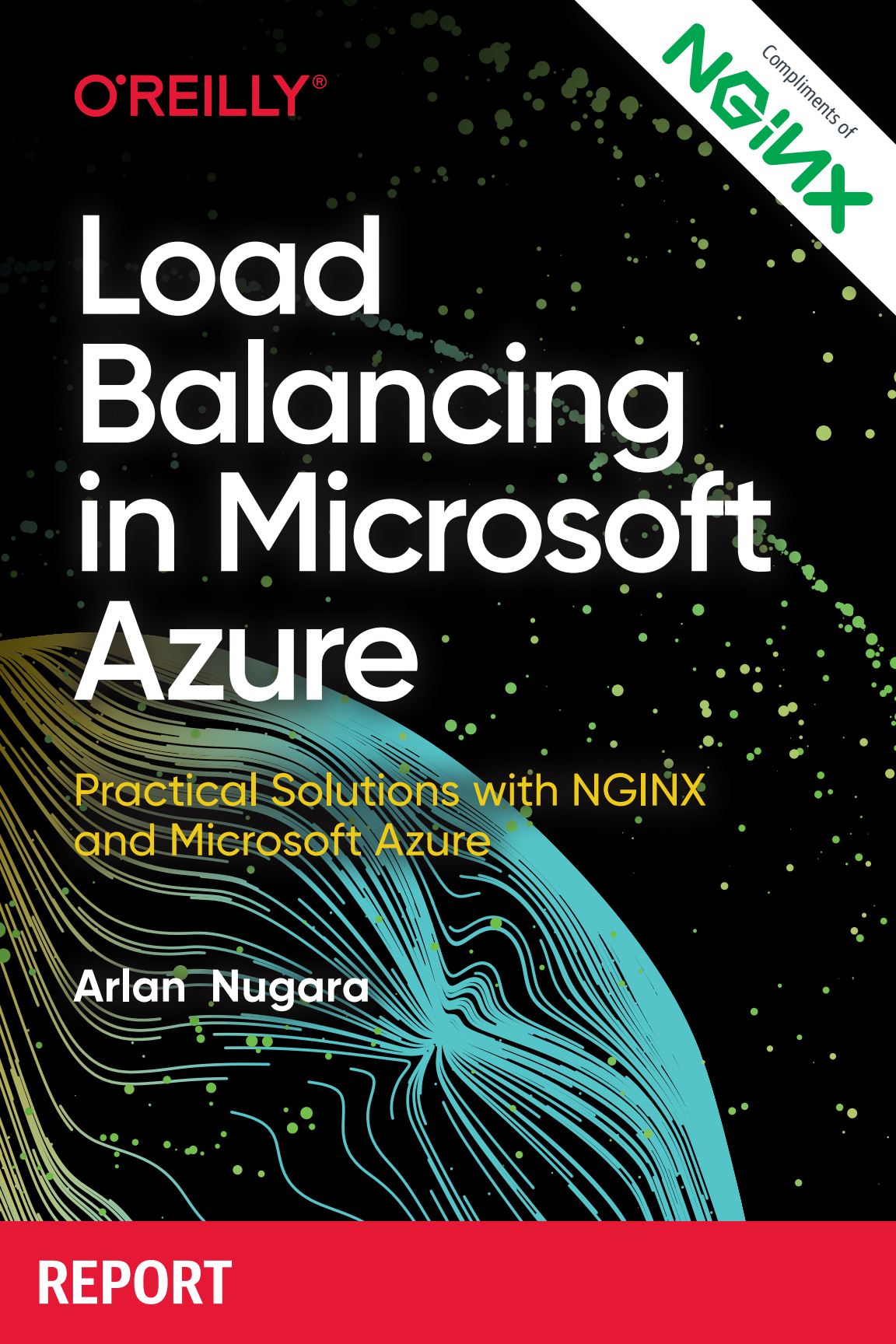
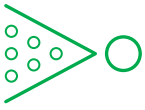

O'REILLY®
Compliments of NGINX
Load Balancing in Microsoft Azure
Practical Solutions with NGINX and Microsoft Azure
Arlan Nugara
REPORT

Try NGINX Plus and NGINX WAF free for 30 days

Get high-performance application delivery for microservices. NGINX Plus is a software load balancer, web server, and content cache. The NGINX Web Application Firewall (WAF) protects applications against sophisticated Layer 7 attacks.

Cost Savings

Over 80% cost savings compared to hardware application delivery controllers and WAFs, with all the performance and features you expect.



Reduced Complexity

The only all-in-one load balancer, content cache, web server, and web application firewall helps reduce infrastructure sprawl.

Exclusive Features

JWT authentication, high availability, the NGINX Plus API, and other advanced functionality are only available in NGINX Plus.

NGINX WAF

A trial of the NGINX WAF, based on ModSecurity, is included when you download a trial of NGINX Plus.

Download at nginx.com/freetrial

NGINX

Load Balancing in Microsoft Azure

Practical Solutions with NGINX and Microsoft Azure

Arlan Nugara

Beijing • Boston • Farnham • Sebastopol • Tokyo O'REILLY®

Load Balancing in Microsoft Azure
by Arlan Nugara

Copyright © 2019 O'Reilly Media, Inc. All rights reserved.

Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*http://oreilly.com*). For more information, contact our corporate/institutional sales department: 800-998-9938 or *corporate@oreilly.com*.

Editor: Kathleen Carr
Acquisitions Editor: Eleanor Bru
Production Editor: Katherine Tozer
Copyeditor: Octal Publishing, Inc.
Proofreader: Charles Roumeliotis
Interior Designer: David Futato
Cover Designer: Karen Montgomery
Illustrator: Rebecca Demarest

May 2019: First Edition

Revision History for the First Edition
2019-05-07: First Release

This work is part of a collaboration between O'Reilly and NGINX. See our *statement of editorial independence*.

The O'Reilly logo is a registered trademark of O'Reilly Media, Inc. *Load Balancing in Microsoft Azure*, the cover image, and related trade dress are trademarks of O'Reilly Media, Inc.

The views expressed in this work are those of the author, and do not represent the publisher's views. While the publisher and the author have used good faith efforts to ensure that the information and instructions contained in this work are accurate, the publisher and the author disclaim all responsibility for errors or omissions, including without limitation responsibility for damages resulting from the use of or reliance on this work. Use of the information and instructions contained in this work is at your own risk. If any code samples or other technology this work contains or describes is subject to open source licenses or the intellectual property rights of others, it is your responsibility to ensure that your use thereof complies with such licenses and/or rights.

978-1-492-05390-3

[LSI]

Table of Contents

Preface

This book is suitable for cloud solution architects and software architects looking to integrate NGINX (pronounced en-juhn-eks) with Azure-managed solutions to improve load balancing, performance, security, and high availability for workloads. Software developers and technical managers will also understand how these technologies in the cloud have a direct impact on application development and application architecture for more cloud-native solutions.

Load balancing provides scalability and a higher level of availability by distributing incoming network traffic efficiently across a group of backend servers, also known as a *server pool* or *server cluster*. This report provides a meaningful description of load-balancing options available natively from Microsoft Azure and the role NGINX can play in a comprehensive solution.

Even though the examples used are specific to Azure, these load-balancing concepts and implementations using NGINX apply equally to other large public cloud providers such as Amazon Web Services (AWS), Google Cloud Platform, Digital Ocean, and IBM Cloud along with their respective cloud platform–native load balancers.

Each cloud application has different load-balancing needs. I hope the information in this book helps you to design a meaningful solution that fits your performance, security, and high-availability needs while being economically practical.

CHAPTER 1

What Load Balancing Is and Why It's Important

Load balancers have evolved considerably since they were introduced in the 1990s as hardware-based servers or appliances. Cloud load balancing, also referred to as Load Balancing as a Service (LBaaS), is an updated alternative to hardware load balancers. Regardless of the implementation of a load balancer, scalability is still the primary goal of load balancing, even though modern load balancers can do so much more.

Optimal load distribution reduces site inaccessibility caused by the failure of a single server while assuring consistent performance for all users. Different routing techniques and algorithms ensure optimal performance in varying load-balancing scenarios.

Modern websites must support concurrent connections from clients requesting text, images, video, or application data, all in a fast and reliable manner, while scaling from hundreds of users to millions of users during peak times. Load balancers are a critical part of this scalability.

Problems Load Balancers Solve

In cloud computing, load balancers solve three issues that fall under the following categories:

1. Cloud bursting
2. Local load balancing
3. Global load balancing

Cloud bursting is a configuration between a private cloud (i.e., on-premises compute environment) and a public cloud that uses a load balancer to redirect overflow traffic from a private cloud that has reached 100% of resource capacity to a public cloud to avoid decreases in performance or an interruption of service.

The critical advantage of cloud bursting is economic in the respect that companies do not need to provision or license excess capacity to meet limited-time peak loads or unexpected fluctuations in demand. This flexibility and the automated self-service model of the cloud means that only the resources consumed for a specific period are paid for until released again.

Organizations can use *local load balancing* within a private cloud and a public cloud; it is a fundamental infrastructure requirement for any web application that needs high availability and the ability to distribute traffic across several servers.

Global load balancing is much more complex and can involve several layers of load balancers that manage traffic across multiple private clouds, public clouds, and public cloud regions. The greatest challenge is not the distribution of the traffic, but the synchronization of the backend processes and data so that users get consistent and correct data regardless of where the responding server is located. Although state synchronization challenges are not unique to global load balancing, the widely distributed nature of a global-scale solution introduces latency and regional resource resiliency that requires various complex solutions to meet service-level agreements (SLAs).

The Solutions Load Balancers Provide

The choice of a load balancing method depends on the needs of your application to serve clients. Different load-balancing algorithms provide different solutions based on application and client needs:

Round robin
: Requests are queued and distributed across the group of servers sequentially.

Weighted round robin
: A round robin, but some servers are apportioned a larger share of the overall traffic based on computing capacity or other criteria.

Weighted least connections
: The load balancer monitors the number of open connections for each server and sends it to the least busy server. The relative computing capacity of each server is factored into determining which one has the least connections.

Hashing
: A set of header fields and other information is used to determine which server receives the request.

Session persistence, also referred to as a *sticky session*, refers to directing incoming client requests to the same backend server for the duration of a session by a client until the transaction being performed is completed.

The OSI Model and Load Balancing

The Open System Interconnection (OSI) model defines a networking framework to implement protocols in seven layers:

- Layer 7: Application layer
- Layer 6: Presentation layer
- Layer 5: Session layer
- Layer 4: Transport layer
- Layer 3: Network layer
- Layer 2: Data-link layer
- Layer 1: Physical layer

The OSI model doesn't perform any functions in the networking process. It is a conceptual framework to better understand complex interactions that are happening.

Network firewalls are security devices that operate from Layer 1 to Layer 3, whereas load balancing happens from Layer 4 to Layer 7. Load balancers have different capabilities, including the following:

Layer 4 (L4)
: Directs traffic based on data from network and transport layer protocols, such as IP address and TCP port.

Layer 7 (L7)
: Adds content switching to load balancing. This allows routing decisions based on attributes like HTTP header, URL, Secure Sockets Layer (SSL) session ID, and HTML form data.

Global Server Load Balancing (GSLB)
: GSLB extends L4 and L7 capabilities to servers in different geographic locations. The Domain Name System (DNS) is also used in certain solutions and this topic is addressed when Azure Traffic Manager is used as an example of such an implementation.

As more enterprises seek to deploy cloud-native applications in public clouds, it is resulting in significant changes in the capability of load balancers.

CHAPTER 2

Load-Balancing Options in Azure

Azure provides several options for managed load-balancing services:

- Azure Load Balancer
- Azure Application Gateway
- Azure Traffic Manager

We review each of these services to understand when to use them effectively.

Azure Load Balancer

A load balancer resource is either a public load balancer or an internal load balancer within the context of the virtual network.[1] Azure load balancer has an inbound and an outbound feature set. The Load Balancer resource's inbound load-balancing functions are expressed as a frontend, a rule, a health probe, and a backend pool definition. Azure load balancer maps new flows to healthy backend instances.

Azure load balancer is available in two different versions (SKUs). The Standard load balancer enables you to scale your applications and create high availability for small-scale deployments to large and complex multizone architectures. The Basic load balancer does not

1 Further reading: What is Azure Load Balancer?

support HTTPS and other basic functionality and is not suitable for production workloads.

A public load balancer maps the frontend IP address and port number of incoming traffic to the private IP address and port number of the virtual machine (VM), and vice versa for the response traffic from the VM. By applying load-balancing rules, you can distribute specific types of traffic across multiple VMs or services. For example, you can spread the load of web request traffic across multiple web servers.

Resources within the virtual network are not directly reachable from the outside unless a customer takes specific steps to expose them through public endpoints or connects them to on-premises networks through a virtual private network (VPN) or Azure ExpressRoute. Azure internal load balancer uses a private IP address of the subnet of a virtual network as its frontend. It directs traffic from within the virtual network or from on-premises networks to VMs within the virtual network.

An internal load balancer enables the following types of load balancing:

Within a virtual network
: Load balancing from VMs in the virtual network to a set of VMs that reside within the same virtual network.

For a cross-premises virtual network
: Load balancing from on-premises computers to a set of VMs that reside within the same virtual network.

For multitier applications
: Load balancing for internet-facing multitier applications where the backend tiers are not internet-facing. The backend tiers require traffic load balancing from the internet-facing tier.

For line-of-business (LoB) applications
: Load balancing for LoB applications that are hosted in Azure without additional load balancer hardware or software. This scenario includes on-premises servers that are in the set of computers whose traffic is load-balanced.

Azure Application Gateway for Load Balancing

An application gateway serves as the single point of contact for clients.[2] It distributes incoming application traffic across multiple backend pools, such as Azure VMs, VM scale sets, App Services, or on-premises/external servers. It is an application delivery controller (ADC) as a service and provides per-HTTP-request load balancing.

Azure Application Gateway is a Layer 7 (L7) web traffic load balancer that enables you to manage traffic to your web applications. Traditional load balancers operate at the transport layer (OSI Layer 4 [L4]—TCP and UDP) and route traffic based on source IP address and port to a destination IP address and port.

Web Application Firewall (WAF) is a feature of Application Gateway that provides centralized protection of your web applications from common exploits and vulnerabilities. WAF is based on rules from the Open Web Application Security Project (OWASP) core rule sets.

Azure Traffic Manager for Cloud-Based DNS Load Balancing

Azure Traffic Manager is a DNS-based traffic load balancer that enables you to distribute traffic optimally to services across global Azure regions while providing high availability and responsiveness.[3]

Traffic Manager uses DNS to direct client requests to the most appropriate service endpoint based on a traffic-routing method and the health of the endpoints. An endpoint is any internet-facing service hosted within or outside of Azure. Traffic Manager provides a range of traffic-routing methods and endpoint monitoring options to suit different application needs and automatic failover models. It is resilient to failure, including the failure of an entire Azure region.

2 Further reading: Azure Application Gateway Components

3 Further reading: Azure Traffic Manager

CHAPTER 3
NGINX Plus on Azure

NGINX Open Source Software (OSS) is free, whereas NGINX Plus is a commercial product that offers advanced features and enterprise-level support as licensed software by NGINX, Inc.[1]

NGINX Plus combines the functionality of a high-performance web server, a powerful frontend load balancer, and a highly scalable accelerating cache to create the ideal end-to-end platform for your web applications. NGINX Plus is built on top of NGINX OSS.

For organizations currently using NGINX OSS, NGINX Plus eliminates the complexity of managing a "do-it-yourself" chain of proxies, load balancers, and caching servers in a mission-critical application environment.

For organizations currently using hardware-based load balancers, NGINX Plus provides a full set of ADC features in a much more flexible software form factor, on a cost-effective subscription.

NGINX Plus provides enterprise-ready features such as application load balancing, monitoring, and advanced management to Azure applications and services.

Table 3-1 shows the NGINX Plus feature sets compared to NGINX OSS. You can get more information on the differences between NGINX products at *nginx.com*.

1 Further reading: NGINX FAQs

Table 3-1. Feature set comparison of NGINX OSS and NGINX Plus from nginx.com

Feature Type	Feature	OSS	NGINX Plus
Load balancer			
	HTTP/TCP/UDP support	✓	✓
	Layer 7 request routing	✓	✓
	Active health checks	—	✓
	Session persistence	—	✓
	DNS service-discovery integration	—	✓
Content cache			
	Static/dynamic content caching	✓	✓
	Cache-purging API	—	✓
Web server/Reverse proxy			
	Origin server for static content	✓	✓
	Reverse proxy: HTTP, FastCGI, memcached, SCGI, uwsgi	✓	✓
	HTTP/2 gateway	✓	✓
	gRPC proxy	✓	✓
	HTTP/2 server push	✓	✓
Security controls			
	HTTP Basic Authentication	✓	✓
	HTTP authentication subrequests	✓	✓
	IP address-based access control lists	✓	✓
	Rate limiting	✓	✓
	Dual-stack RSA/ECC SSL/TLS offload	✓	✓
	TLS 1.3 support	✓	✓
	JWT authentication	—	✓
	OpenID Connect SSO	—	✓
	NGINX Web Application Firewall (additional cost)	—	✓
Monitoring			
	AppDynamics, Datadog, Dynatrace plug-ins	✓	✓
	Extended status with 90 additional metrics	—	✓
High availability (HA)			
	Active-active and active-passive modes	—	✓
	Configuration synchronization	—	✓
	State sharing: Sticky-Learn session persistence, rate limiting, key-value stores	—	✓
Programmability			
	NGINX JavaScript module	✓	✓
	NGINX Plus API for dynamic reconfiguration	—	✓
	Key-value store	—	✓

Feature Type	Feature	OSS	NGINX Plus
	Dynamic reconfiguration without process reloads	—	✓
Streaming media			
	Live streaming: RTMP, HLS, DASH	✓	✓
	VOD: Flash (flv), MP4	✓	✓
	Adaptive bitrate VOD: HLS, HDS	—	✓
	MP4 bandwidth controls	—	✓
Third-party ecosystem			
	Kubernetes Ingress controller	✓	✓
	OpenShift Router	✓	✓
	Dynamic modules repository	—	✓

Installing via Azure Marketplace

Azure Marketplace is a software repository for prebuilt and configured Azure resources from independent software vendors (ISVs). You will find open source and enterprise applications that have been certified and optimized to run on Azure.

NGINX, Inc. provides the latest release of NGINX Plus in Azure Marketplace as a virtual machine (VM) image. NGINX OSS is not available from NGINX, Inc., but there are several options available from other ISVs in Azure Marketplace.

Searching for "NGINX" in Azure Marketplace will produce several results, as shown in Figure 3-1.

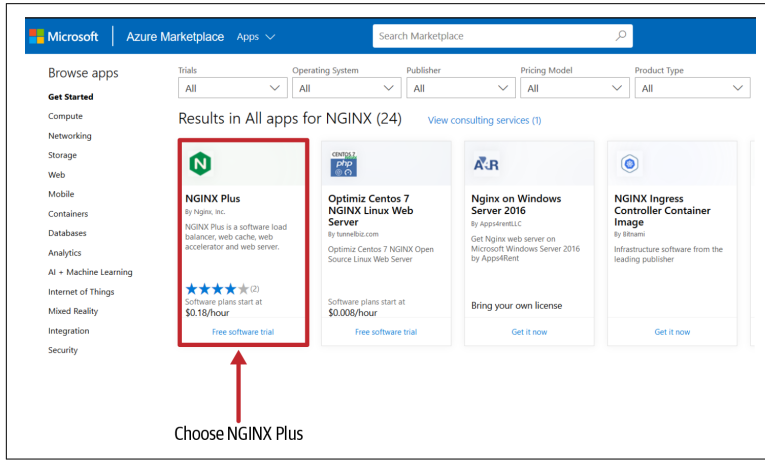


Figure 3-1. Searching for "NGINX" in Azure Marketplace

You will see several results besides the official NGINX Plus VM image from NGINX, Inc., such as the following examples from other ISVs for NGINX OSS:

- NGINX Web Server (Centos 7)
- NGINX Web Server on Windows Server 2016
- NGINX Ingress Controller Container Image

If you search for NGINX Plus in Azure Marketplace, there is only one option available from NGINX, Inc., as shown in Figure 3-2.

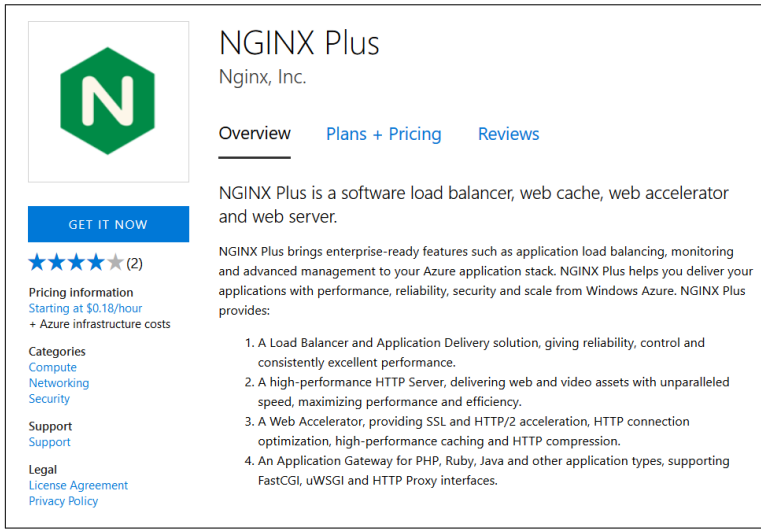


Figure 3-2. NGINX Plus available in Azure Marketplace

The initial page presented is the Overview page, which provides a summary of the NGINX Plus software functionality and pricing. For more details, click the "Plans + Pricing" link. You are presented with several important configuration options such as the Linux operating system (OS) and version as well as the recommended VM sizes and pricing available for the selected Azure Region, as shown in Figure 3-3.

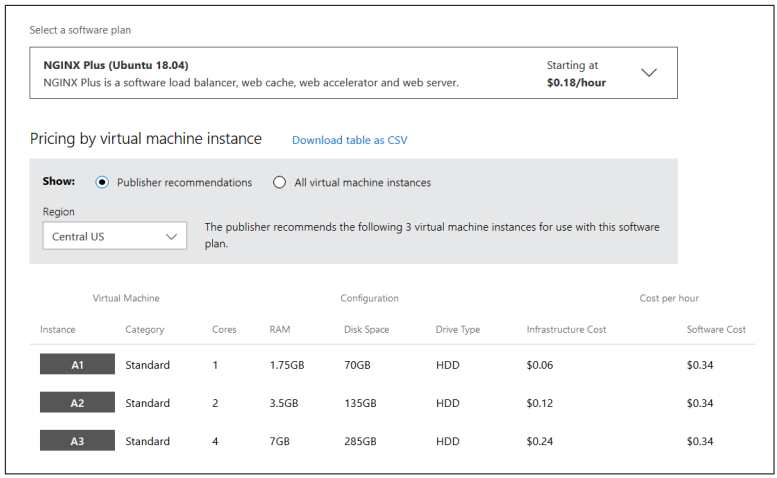


Figure 3-3. NGINX plans and pricing

The VM sizing or Azure Region can be changed later through Azure configuration options but a change to the Linux OS will require a rebuild of the NGINX Plus hosted VM.

First, create an Azure availability set of two or more NGINX Plus virtual machines, which adds redundancy to your NGINX Plus setup by ensuring that at least one VM remains available during a planned or unplanned maintenance event on the Azure platform. For more information, see "Manage the availability of Linux virtual machines" in the Azure documentation. The Azure VM deployment process involves configuration in the following areas: Basics, Disks, Networking, Management, Advanced (Settings), and Tags. Figure 3-4 depicts the start of this process.

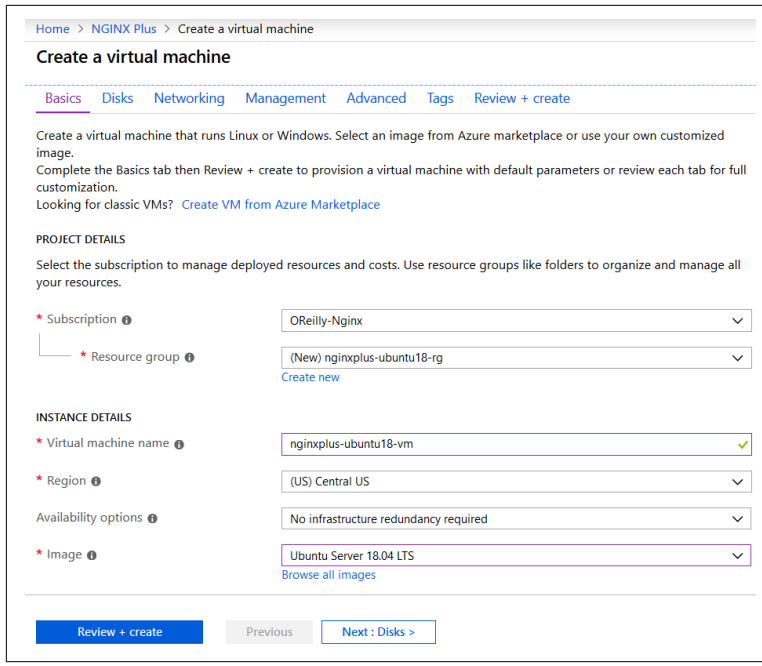


Figure 3-4. Azure VM deployment of NGINX (Azure Marketplace)

Next, create endpoints to enable clients outside the NGINX Plus VM's cloud or virtual network to access it. Sign in to the Azure Management Portal and add endpoints manually to handle the inbound network traffic on port 80 (HTTP) and port 443 (HTTPS). For more information, see "How to set up endpoints on a Linux classic virtual machine in Azure" in the Azure documentation.

As soon as the new VM launches, NGINX Plus starts automatically and serves a default *index.html* page. To verify that NGINX Plus is working properly, use a web browser to access the public DNS name of the new VM and view the page. You can also check the status of the NGINX Plus server by logging into the VM and running this command:

```
$ /etc/init.d/nginx status
```

Azure virtual machine scale sets (VMSSs) let you create and manage a group of identical, load-balanced VMs. VMSSs provide redundancy and improved performance by automatically scaling up or down based on workloads or a predefined schedule.

To scale NGINX Plus, create a public or internal Azure load balancer with a VMSS. You can deploy the NGINX Plus VM to the VMSS and then configure the Azure load balancer for the desired rules, ports, and protocols for allowed traffic to the backend pool.

The cost of running NGINX Plus is a combination of the selected software plan charges plus the Azure infrastructure costs for the VMs on which you will be running the software. There are no additional costs for VMSSs, but you do pay for the underlying compute resources. The actual Azure infrastructure price might vary if you have enterprise agreements or other discounts.

Installing Manually on VMs

You can manually install NGINX and NGINX Plus on VMs or even containers within Azure. The process would be no different than installing other network virtual appliances (NVAs) if you need an OS version or a version of the NGINX software not available in Azure Marketplace. You can use this VM image to create more servers or in scale sets.

Installing via Azure Resource Manager and PowerShell

Azure Resource Manager is the deployment and management service for Azure. It provides a consistent management layer that enables you to create, update, and delete resources in your Azure subscription. You can use its access control, auditing, and tagging features to secure and organize your resources after deployment.

There are no prebuilt Resource Manager templates or PowerShell scripts available from NGINX currently. However, there is nothing preventing the creation of a Resource Manager template and PowerShell script based on your custom deployment requirements for Azure using your previously created custom VM images.

The following provides an example of creating an Ubuntu 16.04 LTS marketplace image from Canonical along with the NGINX web server using Azure Cloud Shell and the Azure PowerShell module.

Open Azure Cloud Shell and perform the following steps in Azure PowerShell:

1. Use `ssh-keygen` to create an Secure Shell (SSH) key pair. Accept all the defaults by pressing the Enter key:

```
# Created in directory: '/home/azureuser/.ssh'
# RSA private key will be saved as id_rsa
# RSA public key will be saved as id_rsa.pub
ssh-keygen -t rsa -b 2048
```

2. Create an Azure resource group by using `New-AzResourceGroup`:

```
New-AzResourceGroup `
-Name "nginx-rg" `
-Location "EastUS2"
```

3. Create a virtual network (`New-AzVirtualNetwork`), subnet (`New-AzVirtualNetworkSubnetConfig`), and a public IP address (`New-AzPublicIpAddress`):

```
# Create a subnet configuration
$subnetConfig = New-AzVirtualNetworkSubnetConfig `
-Name "nginx-Subnet" `
-AddressPrefix 192.168.1.0/24

# Create a virtual network
$vnet = New-AzVirtualNetwork `
-ResourceGroupName "nginx-rg" `
-Location "EastUS2" `
-Name "nginxVNET" `
-AddressPrefix 192.168.0.0/16 `
-Subnet $subnetConfig

# Create a public IP address
# and specify a DNS name
$pip = New-AzPublicIpAddress `
-ResourceGroupName "nginx-rg" `
-Location "EastUS2" `
-AllocationMethod Static `
-IdleTimeoutInMinutes 4 `
-Name "nginxpublicdns$(Get-Random)"
```

4. Create an Azure Network Security Group (NSG) (`New-AzNetworkSecurityGroup`) and traffic rules using `New-AzNetworkSecurityRuleConfig`:

```
# Create an inbound NSG rule for port 22
$nsgRuleSSH = New-AzNetworkSecurityRuleConfig `
-Name "nginxNSGRuleSSH" `
-Protocol "Tcp" `
-Direction "Inbound" `
-Priority 1000 `
-SourceAddressPrefix * `
-SourcePortRange * `
-DestinationAddressPrefix * `
-DestinationPortRange 22 `
-Access "Allow"

# Create an inbound NSG rule for port 80
$nsgRuleWeb = New-AzNetworkSecurityRuleConfig `
-Name "nginxNSGRuleWWW" `
-Protocol "Tcp" `
-Direction "Inbound" `
-Priority 1001 `
-SourceAddressPrefix * `
-SourcePortRange * `
-DestinationAddressPrefix * `
-DestinationPortRange 80 `
-Access "Allow"

# Create a network security group (NSG)
$nsg = New-AzNetworkSecurityGroup `
  -ResourceGroupName "nginx-rg" `
  -Location "EastUS2" `
  -Name "nginxNSG" `
  -SecurityRules $nsgRuleSSH,$nsgRuleWeb
```

5. Create a virtual network interface card (NIC) by using `New-AzNetworkInterface`. The virtual NIC connects the VM to a subnet, NSG, and public IP address:

```
# Create a virtual network card and
# associate it with the public IP
# address and NSG
$nic = New-AzNetworkInterface `
-Name "nginxNIC" `
-ResourceGroupName "nginx-rg" `
-Location "EastUS2" `
-SubnetId $vnet.Subnets[0].Id `
-PublicIpAddressId $pip.Id `
-NetworkSecurityGroupId $nsg.Id
```

6. To create a VM in PowerShell, you create a configuration that has settings like the image to use, size, and authentication options. Then the configuration is used to build the VM:

```
# Define a credential object
$securePassword = ConvertTo-SecureString `
' ' -AsPlainText -Force
$cred = New-Object `
System.Management.Automation.PSCredential("azureuser",
$securePassword)

# Create a virtual machine configuration
$vmConfig = New-AzVMConfig `
-VMName "nginxVM" `
-VMSize "Standard_D1" | `
Set-AzVMOperatingSystem `
-Linux `
-ComputerName "nginxVM" `
-Credential $cred `
-DisablePasswordAuthentication | `
Set-AzVMSourceImage `
-PublisherName "Canonical" `
-Offer "UbuntuServer" `
-Skus "16.04-LTS" `
-Version "latest" | `
Add-AzVMNetworkInterface `
-Id $nic.Id

# Configure the SSH key
$sshPublicKey = cat ~/.ssh/id_rsa.pub
Add-AzVMSshPublicKey `
-VM $vmconfig `
-KeyData $sshPublicKey `
-Path "/home/azureuser/.ssh/authorized_keys"
```

7. Now, combine the previous configuration definitions to create a new VM by using `New-AzVM`:

```
New-AzVM `
-ResourceGroupName "nginx-rg" `
-Location eastus2 -VM $vmConfig
```

8. Connect to the VM after it is created. Create an SSH connection with the VM using the public IP address. To see the public IP address of the VM, use the `Get-AzPublicIpAddress` cmdlet:

```
Get-AzPublicIpAddress `
-ResourceGroupName "nginx-rg" | `
Select "IpAddress"
```

9. In the Azure Cloud Shell or your local bash shell, paste the SSH connection command into the shell to create an SSH session. When prompted, the login user name is `azureuser`. If a passphrase is used with your SSH keys, you need to enter that when prompted:

```
ssh azureuser@vm-public-ip
```

10. From your SSH session, update your package sources and then install the latest NGINX package:

```
sudo apt-get -y update
sudo apt-get -y install nginx
```

11. When done, type `exit` to leave the SSH session. Use a web browser of your choice to view the default NGINX welcome page. Enter the public IP address of the VM as the web address.
12. Once you have completed this process, you can remove the Azure resources by using the Remove-AzResourceGroup cmdlet to remove the resource group, VM, virtual network and all other Azure resources to avoid incurring ongoing charges:

```
Remove-AzResourceGroup `
-Name "nginx-rg"
```

CHAPTER 4

NGINX Plus and Microsoft Azure Load Balancers

Microsoft Azure has three options for load balancing: NGINX Plus, the Azure load balancing services, or NGINX Plus in conjunction with the Azure load balancing services.[1] The following aims to give you enough information to decide which best works for you and shows you how using NGINX Plus with Azure Load Balancer can give you a highly available HTTP load balancer with rich Layer 7 (L7) functionality.

Azure gives its users two choices for a load balancer: Azure Load Balancer for basic TCP/UDP load balancing (at Layer 4 [L4], the network layer) and Azure Application Gateway for HTTP/HTTPS load balancing (at L7, the application layer). Although these solutions work for simple use cases, they do not provide many features that come standard with NGINX Plus.

Table 4-1 provides a comparison of NGINX features with Azure options.

1 Further reading: Using Microsoft Azure Load Balancers and NGINX Plus

Table 4-1. Comparisons of NGINX features with Azure options (from nginx.com)

Feature	Azure Application Gateway	Azure Load Balancer	NGINX Plus	Both Plus & Load Balancer
Mitigation capability	Application layer (Layer 7)		Application layer (Layer 7)	
HTTP-aware	✓	—	✓	✓
HTTP/2-aware	—	—	✓	✓
WebSocket-aware	—	—	✓	✓
TCP/UDP	—	✓	✓	✓
Load balancing methods	Simple	Simple	Advanced	Advanced
SSL/TLS termination	✓	—	✓	✓
SSL offloading	✓	—	✓	
URL request mapping	✓	—	✓	✓
URL rewriting and redirecting	—	—	✓	✓
HTTP health checks	Simple	Simple	Advanced	Advanced
TCP/UDP health checks	—	Simple	Advanced	Advanced
Session persistence	Simple	Simple	Advanced	Advanced
Active-active NGINX Plus cluster	—	—	—	✓
Limits	—	—	✓	✓
Routing capabilities	Simple decision based on request URL or cookie-based session affinity		Advanced routing capabilities	
IP address-based access control lists	— (must be defined at the web-app level in Azure)		✓	
Endpoints	Any Azure internal IP address, public internet IP address, Azure VM, or Azure Cloud Service		Any Azure internal IP address, public internet IP address, Azure VM, or Azure Cloud Service	
Azure VNet support	Both internet-facing and internal (VNet) applications		Both internet-facing and internal (VNet) applications	
WAF	✓		✓	
Volumetric attacks	Partial		Partial	
Protocol attacks	Partial		Partial	

Feature	Azure Application Gateway	Azure Load Balancer	NGINX Plus	Both Plus & Load Balancer
Application-layer attacks	✓		✓	
HTTP Basic Authentication	—		✓	
JWT authentication	—		✓	
OpenID Connect SSO	—		✓	

Comparing NGINX Plus and Azure Load Balancing Services

NGINX Plus offers a choice of several load-balancing methods. In addition to the default round-robin method there are the following:

Least connections
: A request is sent to the server with the lowest number of active connections.

Least time
: A request is sent to the server with the lowest average latency and the lowest number of active connections.

IP hash
: A request is sent to the server determined by the source IP address of the request.

Generic hash
: A request is sent to the server determined from a user defined key, which can contain any combination of text and NGINX variables, for example, the variables corresponding to the Source IP Address and Source Port header fields, or the URI.

You can extend all of the methods by adding different weight values to each backend server.

Azure Load Balancer offers one load-balancing method, Hash, which by default uses a key based on the 5-tuple of the header along with other information. The 5-tuple comprises the IP packets Source IP Address, Source Port, Destination IP Address, Destination Port, and Protocol. Customers can restrict the 5-tuple to a 3- or 2-tuple to enable source IP affinity.

Azure Application Gateway provides only a round-robin method.

Session persistence, also known as *sticky sessions* or *session affinity*, is needed when an application requires that all requests from a specific client continue to be sent to the same backend server because client state is not shared across backend servers. NGINX Plus supports three advanced session-persistence methods:

Sticky Cookie
: NGINX Plus adds a session cookie to the first response from the upstream group for a given client. This cookie identities the backend server that was used to process the request. The client includes this cookie in subsequent requests and NGINX Plus uses it to direct the client request to the same backend server.

Sticky Learn
: NGINX Plus monitors requests and responses to locate session identifiers (usually cookies) and uses them to determine the server for subsequent requests in a session.

Sticky Route
: You can configure a mapping between route values and backend servers so that NGINX Plus monitors requests for a route value and chooses the matching backend server.

NGINX Plus also offers two basic session-persistence methods, implemented as two of the aforementioned load-balancing methods:

IP Hash
: The backend server is determined by the IP address of the request.

Hash
: The backend server is determined from a user-defined key, for example Source IP Address and Source Port, or the URI.

Azure Load Balancer supports the equivalent of the NGINX Plus Hash method, although it is limited to 3- or 2-tuple for source IP affinity.

Azure Application Gateway supports the equivalent of the NGINX Plus Sticky Cookie method with the following limitations: you cannot configure the name of the cookie, when the cookie expires, the domain, the path, or the HttpOnly or Secure cookie attribute.

CHAPTER 5

Monitoring NGINX in Microsoft Azure

Azure Security Center with NGINX

Azure Security Center is a service that comes in a free tier with limited functionality and a fee-based standard tier with a complete set of security capabilities for organizations that need enhanced functionality. The free tier monitors compute, network, storage, and application resources in Azure. It also provides security policy, security assessment, security recommendations, and the ability to connect with other security partner solutions. The standard tier includes all of the capabilities of the free tier for on-premises environments (private cloud) as well as other public clouds such as Amazon Web Services (AWS) and Google Cloud Platform (GCP). The standard tier also includes many more security features along with the following critical security controls:

- Built-in and custom alerts
- Security event collection and advanced search
- Just-in-time virtual machine (VM) access
- Application whitelisting

The NGINX configuration deployed to Azure VMs and VMSSs can have the Microsoft Monitoring Agent installed to read various security-related configurations and event logs from the VM for

monitoring in Security Center. This provides a unified view of Azure resources including the NGINX resources.

Azure Monitor with NGINX

Meaningful metrics play a key role in helping to understand applications and the underlying services and infrastructure that they run to create nominal operational baselines as well as detect, investigate, and diagnose issues.

Azure Monitor integrates the capabilities of Log Analytics and Application Insights for end-to-end monitoring of applications that include NGINX as well the VMs and VMSSs hosting NGINX.

Syslog is an event logging protocol that is common to Linux and the best way to consolidate logs from multiple sources into a single location. The Microsoft Monitoring Agent (MMA) for Linux hosting NGINX configures the local Syslog daemon to forward messages to MMA, which then sends the message to Azure Monitor where a record is created.

Azure Governance and Policy Management for NGINX

Azure Management refers to the tasks and processes required to maintain business applications and the resources to support them. Azure Governance is one aspect of Azure Management. Azure Governance can be summarized by the following features and services that can be implemented across all your Azure environments:

- Create flexible hierarchies with Azure Management Groups for applying policies across multiple subscriptions.
- Azure policies enforce different rules and effects over your resources.
- Azure Blueprints allow the creation of fully compliant environments and the ability to apply group policies to new Azure subscriptions.
- Azure Resource Graph allows fast visibility into all your resources.

- Cost management allows the analysis of costs and the ability to monitor usage from a single dashboard.

NGINX as well the VMs and VMSSs hosting NGINX can be managed with the functionality provided in Azure Governance.

CHAPTER 6

Security

NGINX Management with NGINX Controller

NGINX Controller is a separate and optional product from NGINX, Inc. that manages the NGINX data plane and the entire life cycle of NGINX Plus under these configurations:

- Load balancer
- API gateway
- Proxy in a service mesh environment

This optional and separate NGINX product is fully functional within Azure and provides an additional or exclusive way to manage NGINX without the use of Azure Security Center, Azure Monitor, or Azure Portal or PowerShell.

NGINX Web Application Firewall with ModSecurity 3.0

NGINX Web Application Firewall (WAF) is a separate and optional product from NGINX, Inc. that protects applications against sophisticated Layer 7 (L7) attacks that might otherwise lead to systems being taken over by attackers, loss of sensitive data, and downtime. NGINX WAF is based on the widely used ModSecurity open source software.

ModSecurity is an open source, cross-platform WAF module. Known as the "Swiss Army Knife" of WAFs, it enables web application defenders to gain visibility into HTTP(S) traffic and provides a power rules language and API to implement advanced protections.

Microsoft Azure Firewall Integration into a Load-Balancing Solution

Azure Firewall is a managed, cloud-based network security service that protects your Azure Virtual Network resources. It is a fully stateful Firewall-as-a-Service with built-in high availability and unrestricted cloud scalability. You can centrally create, enforce, and log application and network connectivity policies across subscriptions and virtual networks.

You can integrate Azure Firewall in an end-to-end solution for a business application along with NGINX with the resulting data fed into Azure Monitor.

CHAPTER 7
Conclusion

Microsoft Azure, like other cloud service providers, offers the ability to instantly provision computing resources on demand. This includes support for fully managed Azure services such as load balancers as well as support for third-party network virtual appliance (NVA) load balancers such as NGINX.

You should now have a clear understanding of load balancing and how to design a solution for your Azure-based solution, whether you are using Azure-native load balancers or NGINX or a combination of both to create a resilient load-balancing solution. The solutions described in this book will enable you to improve load balancing, performance, security, and high availability for workloads on Azure.

About the Author

Arlan Nugara is a cloud solution architect who speaks widely on Azure and DevOps. Microsoft has awarded him an MVP (Most Valuable Professional) in Azure for the past two years for his expertise and contributions to the technical community across the United States and Canada. Arlan's original background is in software development with a specialization in enterprise software development and architecture for financial institutions over the previous 20 years.

Arlan's focus over the past two years has been the building of Azure Virtual Datacenters, where security is a key driving factor for a client's migration to the Azure cloud. A critical part of this approach is the building of a landing zone as a configured environment with a standard set of secured cloud infrastructure, policies, best practices, guidelines, and centrally managed services.