Securing Applications in Azure

REFERENCE ARCHITECTURE GUIDE

AUGUST 2020

paloalto® NETWORKS

Table of Contents

Preface

GUIDE TYPES

Overview guides provide high-level introductions to technologies or concepts.

Reference architecture guides provide an architectural overview for using Palo Alto Networks® technologies to provide visibility, control, and protection to applications built in a specific environment. These guides are required reading prior to using their companion deployment guides.

Deployment guides provide decision criteria for deployment scenarios, as well as procedures for combining Palo Alto Networks technologies with third-party technologies in an integrated design.

DOCUMENT CONVENTIONS

Notes provide additional information.

Cautions warn about possible data loss, hardware damage, or compromise of security.

Blue text indicates a configuration variable for which you need to substitute the correct value for your environment.

In the **IP** box, enter `10.5.0.4/24`, and then click **OK**.

Bold text denotes:

- Command-line commands.

  ```
  # show device-group branch-offices
  ```

- User-interface elements.

  In the **Interface Type** list, choose **Layer 3**.

- Navigational paths.

  Navigate to **Network** > **Virtual Routers**.

- A value to be entered.

  Enter the password **admin**.

Italic text denotes the introduction of important terminology.

> An *external dynamic list* is a file hosted on an external web server so that the firewall can import objects.

Highlighted text denotes emphasis.

> Total valid entries: 755

ABOUT PROCEDURES

These guides sometimes describe other companies' products. Although steps and screen-shots were up-to-date at the time of publication, those companies might have since changed their user interface, processes, or requirements.

GETTING THE LATEST VERSION OF GUIDES

We continually update reference architecture and deployment guides. You can access the latest version of this and all guides at this location:

https://www.paloaltonetworks.com/referencearchitectures

WHAT'S NEW IN THIS RELEASE

Palo Alto Networks made the following changes since the last version of this guide:

- Clarified that a second virtual router in the common design model is required only if you are using an Azure public load balancer and is not required when you are using an application gateway
- Modified the VM-Series and Panorama™ capacities and requirements tables to reflect PAN-OS® 9.1
- Modified the PAYG content to reflect changes to capacity licenses based on instance sizing
- Made minor changes to improve readability and technical accuracy

Purpose of This Guide

This guide describes reference architectures for securing applications in Microsoft Azure by using Palo Alto Networks VM-Series virtual next-generation firewalls.

This guide:

- Provides an architectural overview for using VM-Series firewalls to provide visibility, control, and protection to your applications built on Microsoft Azure.
- Links the technical design aspects of Microsoft Azure and the Palo Alto Networks solutions and then explores several technical design models. The design models include two options for enterprise-level operational environments that span across multiple VNets.
- Provides a framework for architectural discussions between Palo Alto Networks and your organization.
- Is required reading prior to using the Palo Alto Networks deployment guides for Microsoft Azure. The deployment guides provide decision criteria for deployment scenarios, as well as procedures for enabling features of Microsoft Azure and the Palo Alto Networks VM-Series firewalls in order to achieve an integrated design.

AUDIENCE

This design guide is written for technical readers, including system architects and design engineers, who want to deploy the Palo Alto Networks VM-Series firewalls and Panorama within a public cloud data center infrastructure. It assumes the reader is familiar with the basic concepts of applications, networking, virtualization, security, and high availability, as well as a basic understanding of network and data center architectures.

To be successful, you must have a working knowledge of networking and policy in PAN-OS.

RELATED DOCUMENTATION

The following documents support this architecture guide:

- Azure—Transit VNet Design Model: Deployment Guide—Details deployment scenarios and step-by-step guidance for the Transit VNet design model on Azure
- Azure—Transit VNet Design Model (Common Firewall Option): Deployment Guide—Details deployment scenarios and step-by-step guidance for the Transit VNet design model (common firewall option) on Azure

Introduction

For new applications and service deployment, many organizations are moving to the public cloud. Instead of developing new applications and running them on their on-premises hardware, these organizations are increasingly using infrastructure hosted and maintained by remote vendors. These Infrastructure-as-a-Service (IaaS) environments, originally used by startups or for niche purposes by enterprises, are increasingly being used for applications that provide business differentiation. Applications deployed in public cloud IaaS environments are becoming more prevalent because they offer several productivity and scale benefits to an organization.

Although IaaS providers are responsible for ensuring the security and availability of their infrastructure, ultimately, organizations are still responsible for the security of the applications and data. This requirement does not differ from on-premises deployments. What does differ are the specific implementation details of how to properly deploy and configure security technology in a public cloud environment such as Azure.

Public Cloud Concepts

Organizations generally move to the public cloud with the goals of increasing scale and reducing time to deployment. Achieving these goals requires application architectures built specifically for the public cloud. Before you can architect for the public cloud, you must understand how it is different from traditional on-premises environments.

SCALING METHODS

Traditionally, organizations scale on-premises deployments through the purchase of devices that have increased performance capacity. Scaling up an on-premises deployment in this method makes sense because organizations typically purchase the devices to last year's requirements and must size the devices to satisfy the performance requirements during their lifetime.

Public cloud environments focus on scaling out the deployment instead of scaling up. This architectural difference stems primarily from the capability of public cloud environments to dynamically increase or decrease the number of resources you have allocated. In the public cloud, infrastructure used to satisfy performance requirements can have a lifetime in minutes instead of years. Instead of purchasing extra capacity for use at some time in the future, the dynamic nature of the public cloud allows you to allocate just the right amount of resources required to service the application.

In practice, to architect an application for the cloud, you need to distribute functionality, and you should build each functional area to scale out as necessary. Typically, this means a load balancer distributes traffic across a pool of identically configured resources. When changes occur in the application traffic, the number of resources you have allocated to the pool can dynamically increase or decrease. This design method provides scale and resiliency. However, the application architecture must take into account that the resources are transient. For example, you should not store the application state in the networking infrastructure or in the front-end application servers. Instead, store state information on the client or persistent storage services.

The ability to scale a cloud architecture extends not only to the capacity of an application but also capacity to deploy applications globally. Scaling an application to a new region in a traditional on-premises deployment requires significant investment and planning. Public cloud architectures are location-agnostic, and you can deploy them globally in a consistent amount of time.

REDUCED TIME TO DEPLOYMENT

To achieve the goals of a reduced time to deployment, you must have a development and deployment process that is repeatable and reacts to changes quickly. DevOps workflows are the primary method for implementing this process. DevOps workflows are highly dependent on the ability to automate, as much as possible, the process of deploying a resource or application. In practice, this means you must be able

to programmatically bootstrap, configure, update, and destroy the cloud infrastructure, as well as the resources running on it. Compared to traditional on-premises deployments where device deployment, configuration, and operation happen manually, automated workflows in a public cloud environment can significantly reduce time to deployment.

In fact, automation is so core to cloud design that many cloud application architectures deploy new capabilities through the automated build-out of new resources instead of updating the existing ones. This type of cloud architecture provides a number of benefits, not the least of which is the ability to phase in the changes to a subset of the traffic as well as the ability to quickly roll back the changes by redirecting traffic from the new resources to the old.

SECURITY INTEGRATION

VM-Series firewalls enable you to securely implement scalable cloud architectures and reduce time to deployment. Capabilities of VM-Series firewalls leveraged to achieve this include:

- **Application visibility**—VM-Series firewalls natively analyze all traffic in a single pass to determine the application, content, and user identity. The application, content, and user are used as core elements of your security policy and for visibility, reporting, and incident investigation.
- **Prevent advanced attacks at the application level**—Attacks, much like many applications, can use any port, rendering traditional prevention mechanisms ineffective. VM-Series firewalls allow you to use Threat Prevention and the WildFire® cloud-based threat analysis service to apply application-specific threat prevention policies that block exploits, malware, and previously unknown threats from infecting your cloud.
- **Consistent policy and management**—Panorama network security management enables you to manage your VM-Series deployments across multiple cloud environments, along with your physical security appliances, thereby ensuring policy consistency and cohesiveness. Rich, centralized logging and reporting capabilities provide visibility into virtualized applications, users, and content.
- **Automation features to reduce time to deployment**—VM-Series firewalls include management features that enable you to integrate security into your public cloud development projects. You can use bootstrapping to automatically provision a firewall with a working configuration, complete with licenses and subscriptions, and then auto-register itself with Panorama. You can publish firewall performance metrics and health information to Azure Application Insights, so you can create automated actions based on performance and usage patterns. To automate policy updates when workloads change, a fully documented XML API and dynamic address groups allow VM-Series firewalls to consume external data in the form of tags that can drive policy updates dynamically. The result is that you can deploy new applications and next-generation security simultaneously in an automated manner.

CLOUD INFRASTRUCTURE PROTECTION

Azure provides basic infrastructure components with a responsibility to ensure that each customer's workloads are appropriately isolated from other workloads and that the underlying infrastructure and physical environment are secure. However, the customer has the responsibility for securely configuring the instances, operating systems, and any necessary applications, as well as maintaining the integrity of the data processed and stored by each virtual machine. This shared-responsibility model is often a point of confusion for consumers of cloud services.

Services have default configurations that might be secure upon implementation, but it is up to the customer to make the assessment and lock those service configurations down to ensure the integrity of the data itself.

Security and compliance risks in cloud computing threaten an organization's ability to drive digital business. The dynamic nature of the cloud, coupled with the potential complexity of having multiple cloud service providers in the environment and massive volume of cloud workloads, makes security and compliance cumbersome.

Public cloud environments use a decentralized administration framework that often suffers from a corresponding lack of any centralized visibility. Additionally, compliance within these environments is complex to manage. Incident response requires the ability to rapidly detect and respond to threats. However, public cloud capabilities are limited in these areas.

Prisma™ Cloud offers comprehensive and consistent cloud infrastructure protection that enables organizations to effectively transition to the public cloud by managing security and compliance risks within their public cloud infrastructure.

Prisma Cloud threat defense enables your organization to:

- Improve the visibility of assets and applications.
- Provide security and compliance posture reporting.
- Enforce DevOps best practices, implemented using policy guardrails.
- Implement DevOps threat monitoring, which identifies risky configurations, network intrusions, and host vulnerabilities for the management plane. This complements the capabilities of the VM-Series firewall to secure the in-line data plane.
- Perform anomaly detection to identify compromised accounts and insider threats.
- Gain forensic capabilities that permit the investigation of current threats or past incidents to quickly determine the root cause.
- Use contextual alerting in order to prioritize issues and respond appropriately.

By using industry best practices for proactive security assessment and configuration management, Prisma Cloud makes cloud-computing assets harder to exploit. Prisma Cloud enables organizations to implement continuous monitoring of the Azure infrastructure and provides an essential, automated, up-to-date status of security posture that they can use to make cost effective, risk-based decisions about service configuration and vulnerabilities inherent in cloud deployments.

Organizations can also use Prisma Cloud to prevent the Azure infrastructure from falling out of compliance and to provide visibility into the actual security posture of the cloud in order to avoid failed audits and subsequent fines associated with data breaches and non-compliance.

Azure Concepts and Services

When deployed on Azure, VM-Series firewalls rely upon underlying Azure services and functionality to integrate into the application traffic flow and protect the workload. The concepts covered in this section give an overview of Azure services relevant to VM-Series firewalls. Microsoft Azure documentation is the definitive source of information on these topics and should be consulted for additional detail.

RESOURCE MANAGER

You use Azure Resource Manager to deploy, manage, and monitor resources. In Azure, *resources* are the managed components used to build applications and services. Resources include, but are not limited to, virtual machines, virtual networks, load balancers, storage services, and non-Microsoft services.

Azure Resource Manager provides a variety of interface options for deploying and managing resources. Azure PowerShell and CLI provide command-line control, the Azure portal provides a graphical front end, and Azure's Representational state transfer (REST) API allows programmatic interaction. While all of these interfaces use a common API to interact with Resource Manager, their capabilities can vary. For example, when new features are released, for a short period it is common that feature deployment and configuration is available only through the CLI. Command-line options also allow for scriptable interactions that are impossible in the portal.

Although those who frequently work with on-premises equipment feel the most familiar with the direct deployment of resources through the portal, Resource Manager templates are the most efficient way to deploy repeatable and tested designs in Azure. Templates define resources and their dependencies in a JavaScript Object Notation (JSON) file. Azure does not process templates in a step-by-step order but does ensure that dependencies are complete before deploying a resource. For example, before firewall deployment, Azure ensures the existence of the required networks that separate private and public zones. Thankfully, you don't have to create templates from scratch. When you deploy resources manually in the portal, Azure sets up a model template. These automatically created templates are good starting points for transforming initial proof-of-concepts that you manually build, test, and validate into scalable and repeatable deployments.

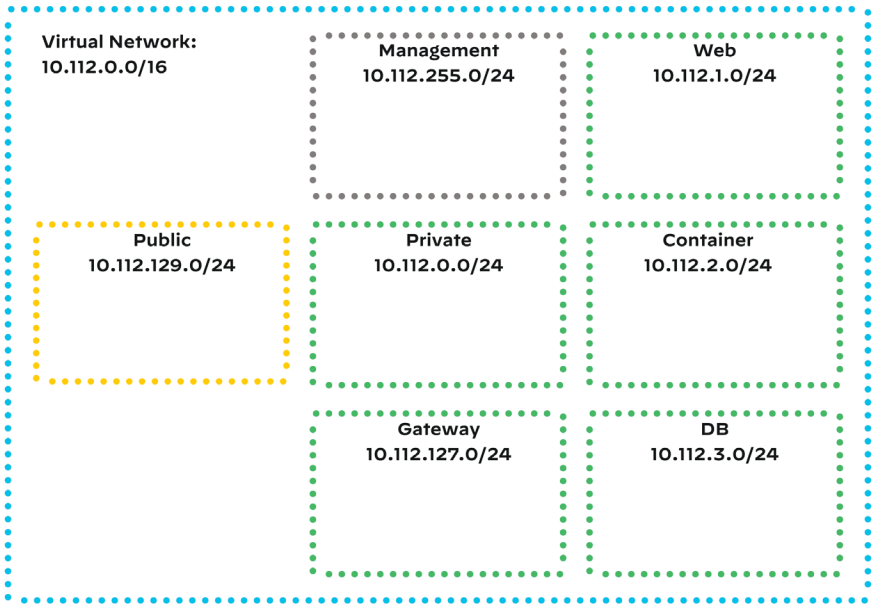
RESOURCE GROUPS

Resource groups are logical containers that define who can manage and control resources. All resources (virtual machines, virtual networks, load balancers, etc.) belong to a resource group. Importantly, though, resources can only belong to one resource group. Proof-of-concepts and personal labs often use a single resource group that contains all of the resources deployed in Azure. When deployments grow, most organizations want to implement role-based access control (RBAC) to control who can create and manage resources. Resource groups provide easy implementation of RBAC. Every organization has unique requirements in determining how to separate resources into RBAC domains, but a common technique is to group resources based on function (infrastructure, application). Separating application resources from the infrastructure resources that allow them to communicate with each other is possible because resource groups do not control communication between resources.

VIRTUAL NETWORKS

A *virtual network* (VNet) is a logically segmented network within Azure that allows connected resources to communicate with each other. You define a VNet by one or more public or private IP address ranges that you can then divide into subnets (/29 or larger). VNet IP address space, both public and private, is reachable only within the VNet or through services connected to the VNet, such as a VPN. Because VNets are isolated from each other, you can overlap IP network definition across VNets. When you want direct communication of resources in separate VNets, the VNets can be connected whether they are in the same Azure location or not, as long as there is no overlap in the IP network definition.

Figure 1 Single VNet



Virtual machine network interfaces receive IP addresses, default gateways, and DNS servers from the Azure DHCP service. By default, when you start a virtual machine, DHCP dynamically assigns the first available IP address in the subnet to the virtual machine's interface. Stopping a virtual machine releases any IP addresses allocated to it. The Azure DHCP service does not retain the IP address reservation until the lease time expires. The next time you start a virtual machine, DHCP again assigns the first available IP address in the subnet to the virtual machine's interface. In environments where virtual machines change state often, you should not expect consistent IP address assignment.

Static IP addressing is available when a consistent IP address is required. When there is only one IP on an interface, you do not need to configure static IP addresses in the operating system running on the virtual machine. Instead, you set up that static IP address in the Azure portal or the template. When you configure a static IP address, the virtual machine still receives the IP address through DHCP. However, unlike dynamic IP address allocation, when started, the virtual machine uses the configured IP address, and when stopped, the IP address is not released. The next time the virtual machine starts, the IP address remains the same.

Note

Azure reserves five internal IP addresses from each subnet. You cannot configure these IP addresses on a resource: the first and last addresses of the address space (for the subnet address and multicast) and three addresses reserved for internal use (for DHCP and DNS purposes).

An alternative to static IP addressing for consistent connectivity is the use of name resolution to communicate between resources within a VNet. By default, you configure resources through DHCP to point to Azure DNS servers. The Azure DNS servers provide not only public name resolution but also internal name resolution within the VNet. The addition or state change of a virtual machine automatically updates Azure name resolution. When a virtual machine has multiple internal IP addresses, its name resolves to its primary IP address.

Although VNets do not contain publicly routable IP addresses, you can associate resources within a VNet with a publicly routable IP address. You can map public IP addresses one-to-one to internal IP addresses, and Azure networking automatically translates the IP addressing of the traffic as it enters and leaves the VNet.

Similar to internal IP addresses, public IP addresses can change as a virtual machine changes state. You can configure a public IP address to be static, but in most situations, configuring a DNS name label on the public IP address is the preferred way to ensure consistent connectivity when the underlying IP address can change.

If you need multiple public IP addresses on a single network interface, you can configure a network interface with one or more secondary IP addresses. You can then associate a public IP address to each secondary IP address on the interface. Because DHCP cannot assign multiple IP addresses to a single interface, you should statically define secondary IP addresses in Azure and configure them in the virtual machine operating system.

Note

Even when they do not have a public IP address assigned, by default, all resources deployed in a VNet have unfiltered outbound access to the internet. Azure translates the IP addresses of outbound traffic to the internet automatically. Use of public IP addresses enables inbound connectivity to the resource.

VNet Peering

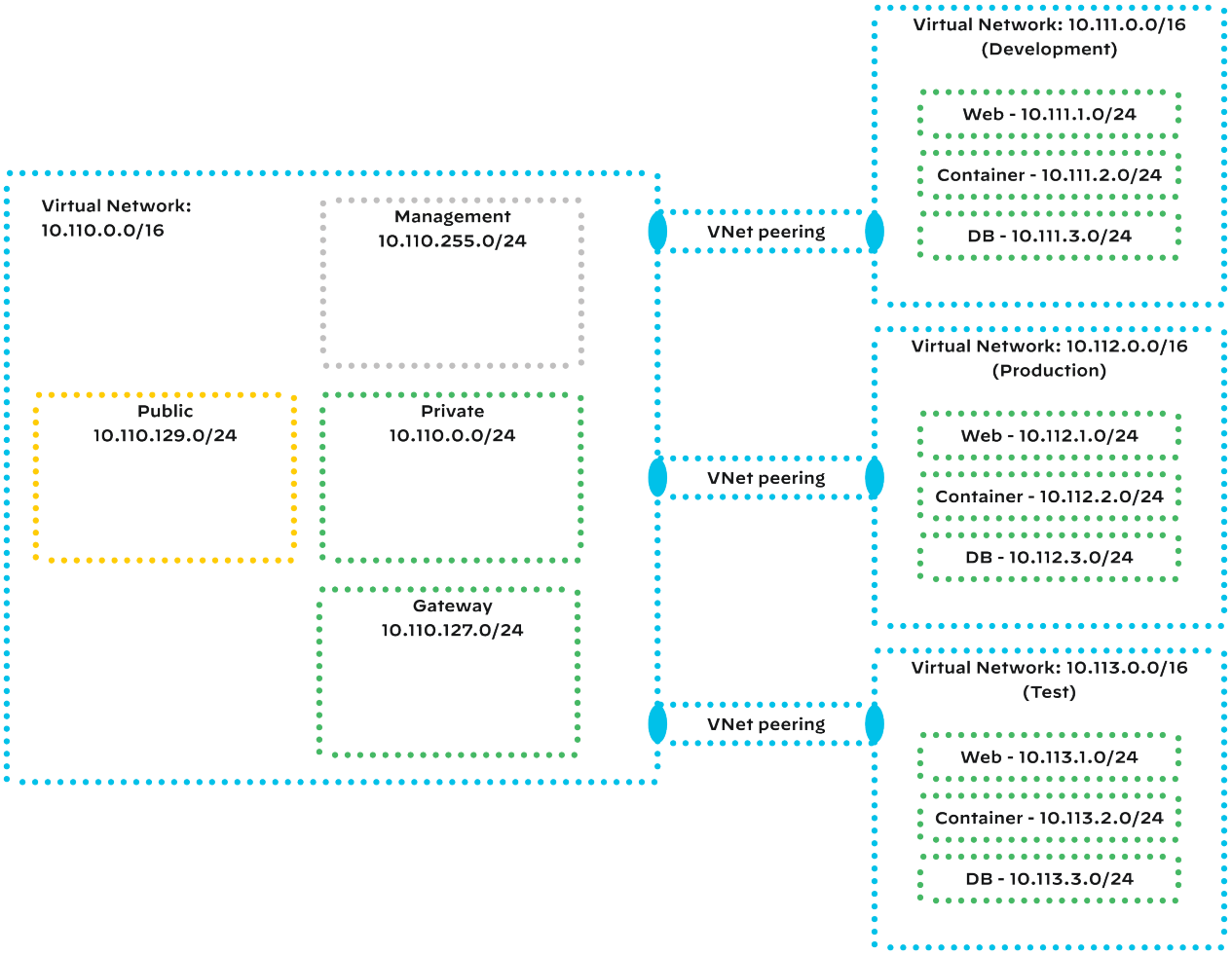
Virtual network peering allows you to logically connect Azure VNets. The use of multiple VNets allows you separate workloads across functional environments or administrative domains. However, you cannot use overlapping IP address space within the set of peered VNets.

Two types of peering are supported:

- **VNet peering**—You have deployed the connected VNets within the same Azure region (example: West US).
- **Global VNet peering**—You have deployed the connected VNets across multiple Azure regions (example: West US and East US). Some Azure networking capabilities are restricted when using Global VNet peering. For more information, see the Azure documentation for Virtual Network Peering.

You achieve full, seamless IP reachability across VNets after you establish the peer connection. All network traffic using VNet peering remains within the Azure backbone. Azure supports 1000 VNets within a subscription and supports up to 100 VNet peering connections for each VNet.

Figure 2 Multiple peered VNets



Traffic Forwarding

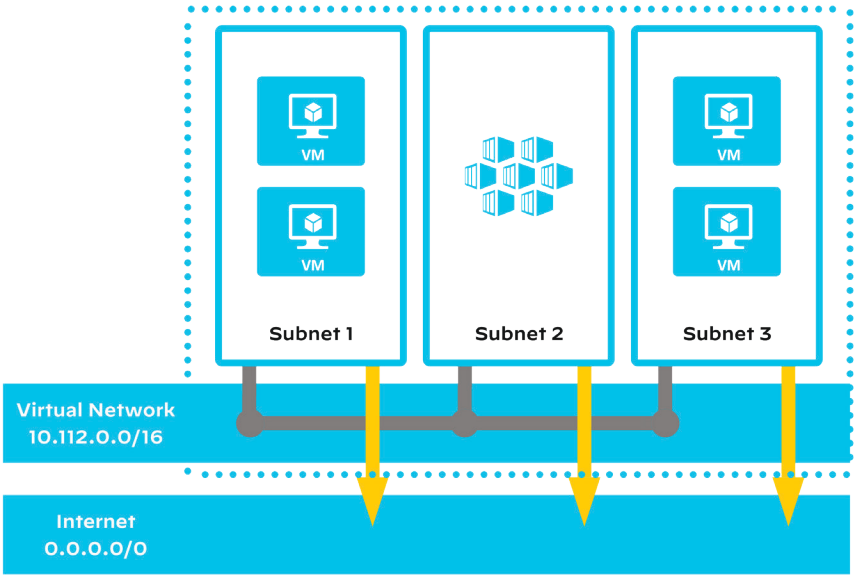
It is important to understand the differences between how Azure forwards traffic in a VNet and between VNets and how a traditional Layer 2/Layer 3 network forwards traffic. In traditional networking, when there are multiple Layer 2 subnets and a device wants to communicate with a device on a different subnet, it must send traffic through a Layer 3 router. Devices use Layer 2 packet forwarding to communicate to the Layer 3 router. Azure networking is a Layer 3 overlay network fabric and does not use traditional Layer 2 packet forwarding, including ARP and broadcast. The Azure network infrastructure attempts to make its differences transparent. For example, even though you cannot ping the default router or use traceroute during troubleshooting, virtual machines still have default routes defined.

By default, all resources connected to the VNet communicate directly, even when they are on different subnets. When you add a subnet, Azure automatically defines system routes to facilitate communication within the VNet as well as to the internet. The effect of this default behavior is that any resource that needs to be in the middle of a traffic flow, such as a firewall, won't see traffic. Azure networking applies additional system routes within the VNet to control RFC-1918 and RFC-6598 reserved routes.

Table 1 Azure system routes

Address space	Address prefix	Nexthop type
VNet defined (Example)	10.110.0.0/16	Virtual network
Peer VNet (Example)	10.111.0.0/16	VNet peering
Peer VNet (Example)	10.112.0.0/16	VNet peering
Peer VNet (Example)	10.113.0.0/16	VNet peering
Default (Azure defined)	0.0.0.0/0	Internet
RFC-1918 (Azure defined)	10.0.0.0/8	None
RFC-1918 (Azure defined)	172.16.0.0/12	None
RFC-1918 (Azure defined)	192.168.0.0/16	None
RFC-6598 (Azure defined)	100.64.0.0/10	None

Figure 3 Azure networking default behavior



Traffic-forwarding behavior with peered VNets is essentially the same as within a single VNet. Azure installs system routes for the defined address space of the peered VNets into the active forwarding table for each subnet.

Azure networking supports system routes with the following next-hop types:

- **Virtual network**—System route to a destination prefix within the local VNet address space. Azure automatically creates this route type by default when an address space is defined. You may manually create additional routes of this type as necessary.
- **VNet peering**—System route to a destination prefix within a peered VNet address space. Azure automatically creates this route type by default when a peer connection is established.
- **Internet**—System route to the internet. When you create a VNet, Azure automatically creates a default system route that matches any destination prefix using a wildcard match (0.0.0.0/0). You can manually create additional, more-specific routes to the internet as necessary.
- **None**—Special system route to drop traffic for a specified destination prefix. Azure automatically creates these system routes for RFC-1918 and RFC-6598. You can manually create additional routes of this type in order to drop traffic to other destination prefixes as necessary.
- **Virtual appliance**—Manually created system route for a specified destination prefix with a specified next-hop IP address. You must assign the next-hop address to a virtual device deployed within a VNet (or peered VNet).
- **Virtual network gateway**—System route to a destination prefix assigned to a virtual network gateway connection. Azure automatically creates this system route when you configure the connection.

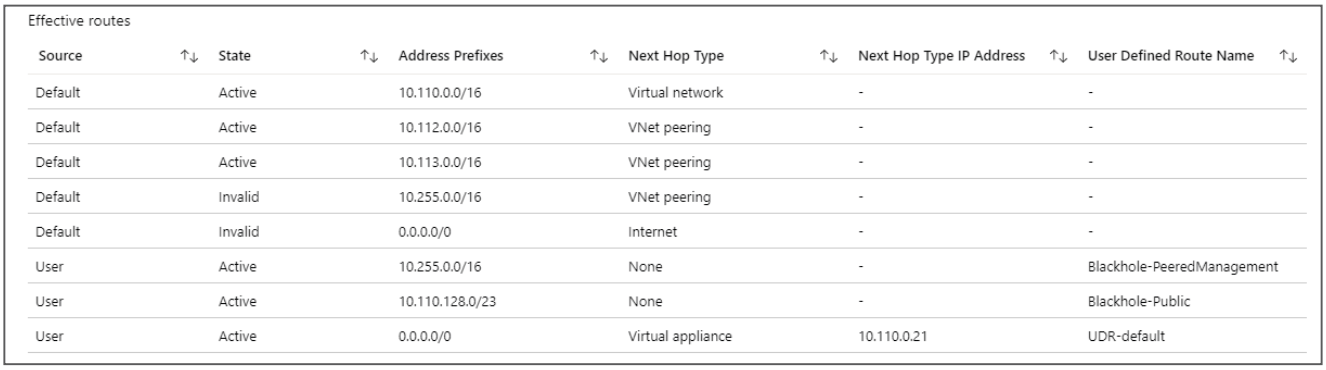
User-defined routes (UDRs) modify the default traffic-forwarding behavior of Azure networking. You configure a UDR on a per-subnet basis, and it applies to traffic that virtual machines send within the subnet. The destination for a route can be a different subnet in the VNet, a different subnet in another VNet (with an existing peer connection), anywhere on the internet, or a private network connected to the VNet. The next hop for the route can be any resource in the VNet or in a peered VNet in the same region. You primarily use a UDR to direct traffic to a resource, such as a load balancer or a firewall, within the VNet or in a peered VNet. It can also discard (or *blackhole*) traffic or send it across a VPN connection. UDR can affect traffic within a subnet, including host-to-host traffic within a subnet if the applied UDR is active for the subnet prefix.

Note

The use of UDR summary routes can have unexpected consequences. If you apply a UDR summary route to a subnet that falls within the summary but does not have a more specific UDR or system route, UDR controls traffic within the subnet (host to host).

By using the Effective routes troubleshooting tool, you can view the active system routes for a route table that you have applied.

Figure 4 Effective routes troubleshooting tool



Effective routes

Source	State	Address Prefixes	Next Hop Type	Next Hop Type IP Address	User Defined Route Name
Default	Active	10.110.0.0/16	Virtual network	-	-
Default	Active	10.112.0.0/16	VNet peering	-	-
Default	Active	10.113.0.0/16	VNet peering	-	-
Default	Invalid	10.255.0.0/16	VNet peering	-	-
Default	Invalid	0.0.0.0/0	Internet	-	-
User	Active	10.255.0.0/16	None	-	Blackhole-PeeredManagement
User	Active	10.110.128.0/23	None	-	Blackhole-Public
User	Active	0.0.0.0/0	Virtual appliance	10.110.0.21	UDR-default

Network Security Groups

Network security groups (NSGs) filter traffic into and out of subnets and virtual machine network interfaces. An NSG can be associated to the network interface of a virtual machine or to a subnet. The limit is one association for each interface and one association for each subnet. For ease of configuration when you apply the same policies to more than one resource, you can associate network security groups with multiple subnets and virtual machine network interfaces. An NSG associated to the subnet with a common policy can be easier to manage than unique NSGs applied at the interface level.

Note

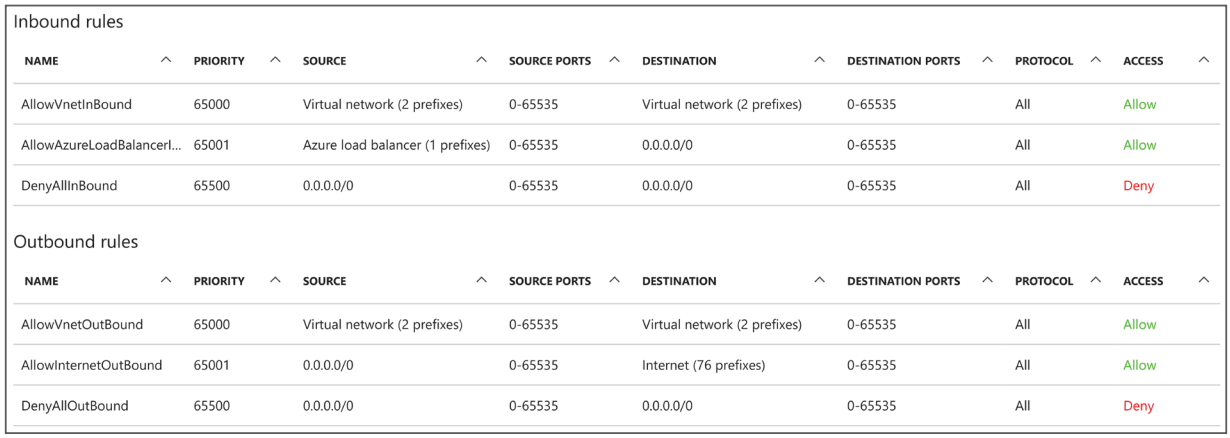
When you are using Standard SKU IP addresses, NSGs are required on the subnet or NIC in order for traffic to reach the resource.

A prioritized list of rules defines the policies of a network security group. There are separate policies for inbound and outbound. Rules are defined and matched by the traffic source, destination, port, and protocol. In addition to IP addressing, you can set the source and destination of a rule by using Azure tags or application security groups.

Network security groups are pre-configured with default security rules that:

- Allow all traffic within the VNet.
- Allow outbound traffic to the internet.
- Allow inbound traffic that is originating from Azure's load-balancer probe (168.63.129.16/32).
- Deny all other traffic.

Figure 5 Default security rules



Inbound rules

NAME	PRIORITY	SOURCE	SOURCE PORTS	DESTINATION	DESTINATION PORTS	PROTOCOL	ACCESS
AllowVnetInBound	65000	Virtual network (2 prefixes)	0-65535	Virtual network (2 prefixes)	0-65535	All	Allow
AllowAzureLoadBalancerI...	65001	Azure load balancer (1 prefixes)	0-65535	0.0.0.0/0	0-65535	All	Allow
DenyAllInBound	65500	0.0.0.0/0	0-65535	0.0.0.0/0	0-65535	All	Deny

Outbound rules

NAME	PRIORITY	SOURCE	SOURCE PORTS	DESTINATION	DESTINATION PORTS	PROTOCOL	ACCESS
AllowVnetOutBound	65000	Virtual network (2 prefixes)	0-65535	Virtual network (2 prefixes)	0-65535	All	Allow
AllowInternetOutBound	65001	0.0.0.0/0	0-65535	Internet (76 prefixes)	0-65535	All	Allow
DenyAllOutBound	65500	0.0.0.0/0	0-65535	0.0.0.0/0	0-65535	All	Deny

The default security rules do not show up in the network security group configuration. They cannot be modified or removed. To change the behavior of a default rule, you must precede it with a custom rule that has a higher priority. The default rules have priority values that begin at 65000, so you must assign a priority value less than 65000 in order to override the default rules. You can view the active rules through the **Effective security rules** troubleshooting tool in the network security group.

On-Premises Network Connectivity

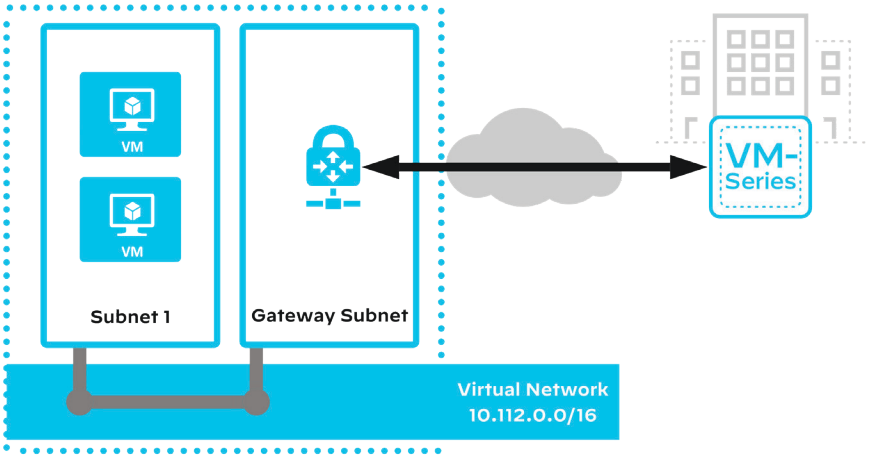
The Azure *virtual network gateway* (VNG) provides connectivity between an Azure virtual network and your on-premises networks and data centers. Site-to-site connectivity through a VNG can either be through IPSec VPN (also known as a *VPN gateway*) or a dedicated private connection (also known as an *ExpressRoute gateway*). When you deploy a virtual network gateway as a VPN gateway, it supports the configuration of IPSec VPN tunnels to one or more of your locations across the internet. When deployed as an ExpressRoute gateway, the VNG provides connectivity to on-premises locations through a dedicated private circuit facilitated by a connection provider.

Note

You can only deploy one virtual network gateway of each type in a virtual network, and you deploy them in a dedicated gateway subnet.

A VPN gateway is composed of a resilient pair of virtual machines deployed, by default, in an active/standby configuration. You may configure IP routing between Azure and the on-premises network to be static or dynamically exchanged through IKEv2 or BGP. For increased connection resiliency for deployments with resilient on-premises VPN devices, a VPN gateway supports configurations with multiple tunnels to a single location. Active/active configuration is also possible on select VPN gateway sizes.

Figure 6 VPN gateway



Connections through an Azure ExpressRoute gateway do not go over the public internet. Instead, Azure resources are accessed directly through colocation facilities, point-to-point Ethernet, or connectivity into your MPLS WAN. Microsoft recommends ExpressRoute connections for all enterprise customer connectivity and to support a range of bandwidth options from 50 Mbps to 10 Gbps.

For resiliency, Azure terminates ExpressRoute connections on a pair of edge routers. Two BGP connections provide resilient, dynamic IP routing between Azure and your on-premises networks. You can share ExpressRoute circuits across multiple VNets. In each virtual network that requires backhaul over the ExpressRoute gateway, a VNG connects the VNet to the ExpressRoute circuit.

RESILIENCY CONSTRUCTS

Availability Sets and Managed Disks

To ensure that maintenance and failures within the Azure data center do not affect the availability of an application or service, you can place the virtual machines used for load-sharing and resiliency into availability sets. *Availability sets* distribute virtual machines across multiple Azure fault and update domains. Separating fault domains places virtual machines that are members of the availability set on hypervisors that do not share power, physical switching, or other Azure data center infrastructure. Separating update domains stops concurrent reboots of hypervisors hosting the virtual machines in the availability set.

Note

You can only configure an availability set on a virtual machine during its initial deployment. You can't modify a virtual machine's availability set configuration after you have deployed the virtual machine.

Managed disks work in conjunction with virtual machine availability sets to provide enhanced resiliency. With managed disks, Azure ensures that each member of the availability set uses a unique hardware for their back-end disk store. This limits the number of virtual machines that a hardware or software failure in the Azure storage system can affect.

Load Balancing

Load balancing distributes traffic to a set of resources based on the traffic's DNS, Layer 7, or Layer 4 information. Azure offers the following types of load balancers:

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

Azure Traffic Manager

Azure Traffic Manager uses DNS to distribute traffic across multiple data centers. Traffic Manager integrates into DNS requests through DNS CNAME records that alias the application to Traffic Manager.

Traffic Manager offers multiple traffic routing methods, including:

- **Priority**—Defines a set of resources to receive the traffic and a backup set in the event the primary endpoints are unavailable
- **Weighted**—Distributes traffic across a set of resources based on a configured weight. You can define weights to distribute traffic among the endpoints equally.
- **Performance**—Uses latency to direct traffic to the closest resource location
- **Geographic**—Directs traffic to endpoints based on the source of the user's DNS query

After Traffic Manager determines which resource to route traffic toward, it sends a DNS response to the client, which directs the traffic to the selected resource. Unlike the other load balancers, clients connect to the resource directly. Traffic Manager does not interact with traffic passing between the client and the resource.

To provide resiliency in case of resource failure, Traffic Manager monitors the health of the resources through HTTP/HTTPS GET requests. If Traffic Manager does not receive an expected response from the resource (return of a 200 OK), Traffic Manager changes the resource state to *degraded* and removes the resource from the pool of available resources. Traffic Manager continues to monitor degraded resources so that when they return to a healthy state, they return to the pool of available resources.

Azure Application Gateway

Azure Application Gateway uses HTTP/HTTPS information to distribute traffic across resources within a data center. Application gateways have a single public IP address but can host up to 20 websites, each with a back-end pool. Primarily, Application Gateway relies on HTTP host headers to differentiate between websites. When you enable SSL offload on Application Gateway, it can also use server name indication to distinguish between websites. When SSL offload is enabled, you can choose to pass cleartext traffic to the back-end pool or re-encrypt the traffic before passing it to the back-end pool.

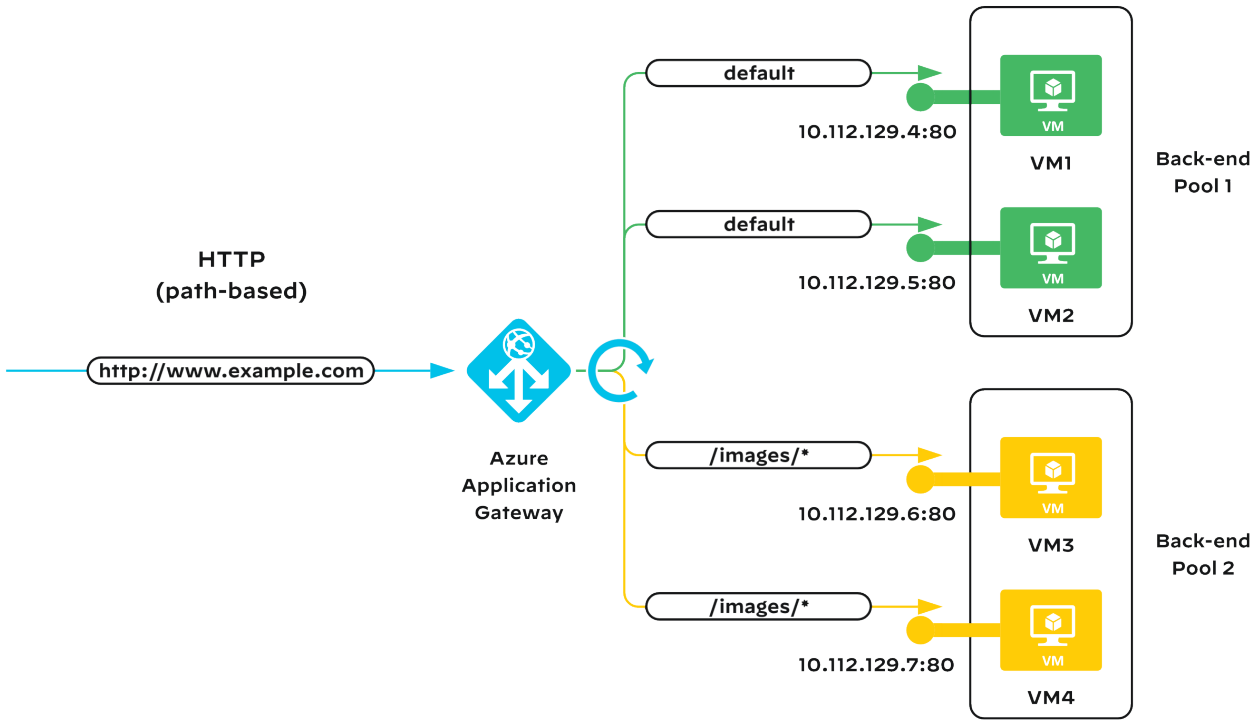
Also, for each website, URL path-based routing allows you to select a back-end pool to serve content, based on the folder in the URL path. For example, you could service the URLs www.example.com/images/ and www.example.com/video/ on two different back-end pools even though it is a single website.

Both the inbound and the return traffic must flow through Application Gateway. To ensure bi-directional traffic flow through Application Gateway, you must apply address translation to both the source and the destination IP addresses. The destination NAT forwards traffic to the selected back-end pool resource, and source NAT ensures the return traffic from the selected resource returns to Application Gateway.

Because Application Gateway uses system routes to direct traffic, you must deploy it in a dedicated subnet. If it shares a subnet with other resources, traffic from virtual machines in the subnet will not route to Application Gateway.

Back-end pools can be composed of network interfaces, public IPs, internal IPs, and fully qualified domain names (FQDNs). After Application Gateway chooses a back-end pool, Application Gateway uses round-robin distribution to the resources in the pool. To provide resiliency in case of resource failure, Application Gateway monitors the health of the resources through HTTP/HTTPS requests. If Application Gateway does not receive an expected response from the resource (HTTP response status code between 200 and 399), the Application Gateway removes the resource from the pool of available resources. Application Gateway continues to monitor unhealthy resources so that when they return to a healthy state, they return to the pool of available resources.

Figure 7 Application Gateway with multiple back-end pools

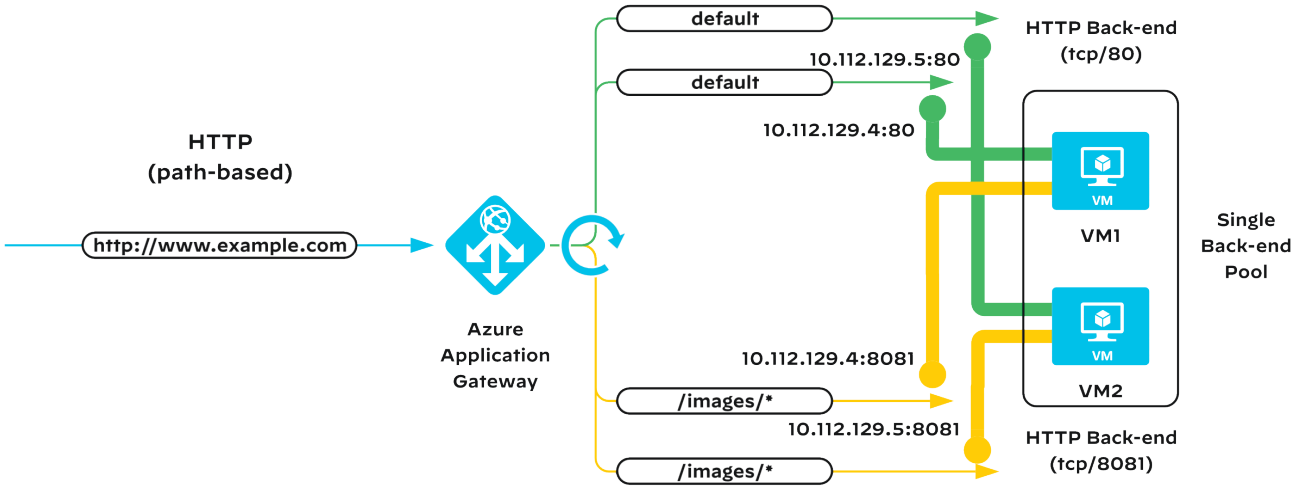


Each application gateway rule must include an HTTP setting, which corresponds to the protocol (HTTP or HTTPS) and TCP port of the back-end pool targets. Basic rules have a single back-end pool with an HTTP setting. Path-based rules have multiple rule entries, with a default rule entry that is identical to a basic rule. Each additional rule entry includes a path-matching expression that maps to a back-end pool and HTTP setting.

The path-based rules provide the most flexibility when you are creating your application gateway policy. Any of the following policies are valid, as long as the combination of back-end pool and HTTP setting are unique for each additional rule:

- Multiple back-end pools, each with default HTTP setting (HTTP/80)
- Single back-end pool with multiple HTTP settings (example: default HTTP/80 and HTTP/8081)
- Multiple back-end pools, each with one or more HTTP settings

Figure 8 Application gateway with single back-end pool and multiple HTTP settings



Optionally, you can deploy Application Gateway with Web Application Firewalls (WAF) functionality in addition to load-balancing. Azure implements Open Web Application Security Project (OWASP) core rule sets 2.29 and 3.0 by default, and you can also write your own rules. The WAF functionality can run in either detection or prevention mode and integrates with Azure Monitor and Azure Security Center.

Azure Load Balancer

Azure Load Balancer distributes flows that arrive on the load balancer's front end to back-end pool instances and allows you to scale your applications and provide high availability for services. The load balancer is available in both a Basic and Standard SKU. The Standard SKU load balancer has an expanded feature set and increased scale and is the recommended resource for this design guide.

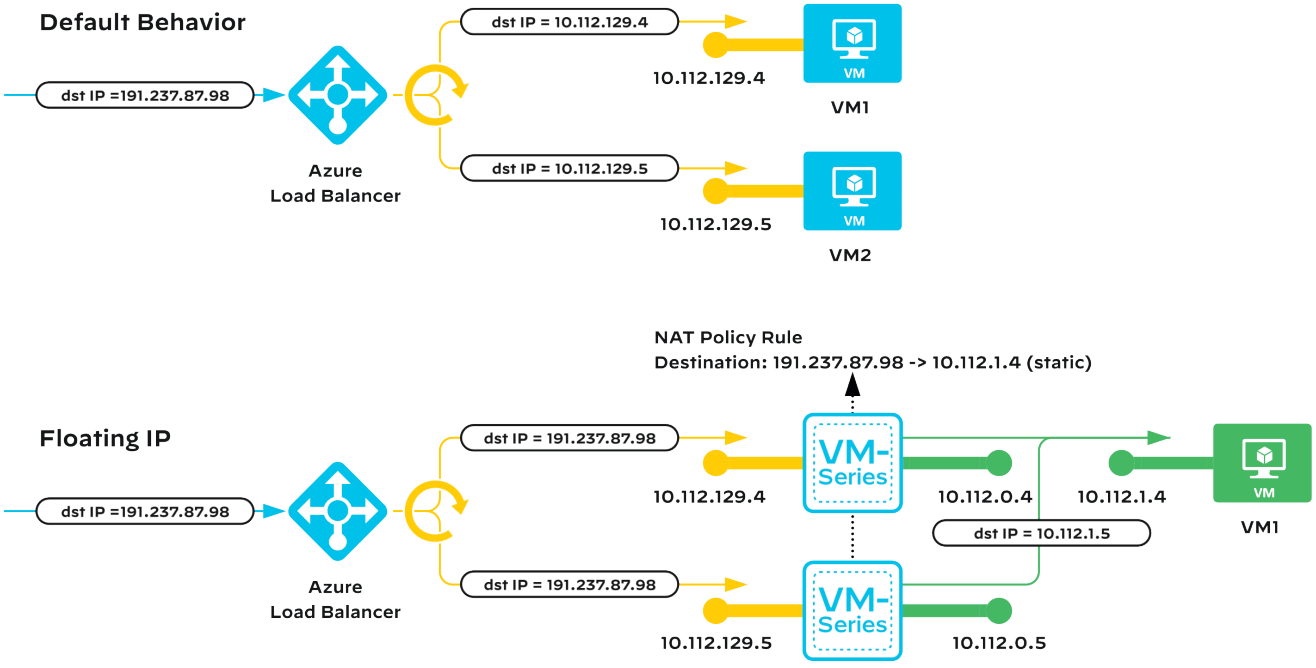
Azure Load Balancer distributes traffic based on TCP/UDP information. It listens on one or more front-end virtual IP addresses (VIPs). You configure rules defined by a protocol and port number to distribute traffic to a healthy back-end pool resource. The load balancer comes in two types, internal and public. The difference between them is the source of the traffic. You use the internal load balancer only for traffic originating within the Azure cloud or coming across a point-to-site VPN terminating within Azure. The public load balancer is reachable from any device on the internet. The default load-balancing algorithm uses a 5-tuple hash consisting of the source IP address and port number, destination IP address and port number, and protocol. Alternate algorithms include a 3-tuple and 2-tuple hash.

Standard Load Balancer back-end pools are composed of any virtual machine in the VNet, but for the highest availability, the virtual machines should be members of availability sets. Basic Load Balancer back-end pools are composed only of the virtual machines that are members of availability sets. By default, the load balancer performs a destination NAT on the traffic before sending it to the back-end pool. It translates the destination IP address and port number of the incoming traffic to the IP address and port number of the virtual machine selected from the back-end pool. The load balancer does not translate the source IP address of the incoming traffic. The virtual machines in the back-end pool see the originating client's IP address as the source. However, even though the return traffic does not pass through the load balancer, Azure networking tracks the connection state and translates the source IP address and port number of the return traffic to the load balancer's VIP.

To support multiple applications with a single back-end pool, you must configure each application to have a unique port number in the back-end pool. Also, each application must either have a unique VIP or a unique port number on a VIP shared between multiple applications.

Alternatively, enabling floating IP on a load-balancing rule disables destination NAT and allows the virtual machines in the back-end pool to see the original destination. The back-end pool resources can use both the IP and the port in order to identify an application allowing for port reuse. However, a floating IP configuration requires that the back-end resources listen for the logical VIP address in addition to the interface address.

Figure 9 Azure Load Balancer configuration options



To provide resiliency in case of resource failure, Load Balancer monitors virtual machine health through guest agent, HTTP custom, and TCP custom probes. If Load Balancer does not receive an expected response (TCP Ack or HTTP 200) from the virtual machine, Load Balancer removes the virtual machine from the pool of available resources. Load Balancer continues to monitor unhealthy resources so that when they return to a healthy state, they return to the pool of available resources.

Guest agent probes determine if the virtual machine is in the ready state. Guest agent probes do not monitor the health of the services running on the virtual machine. Even when a service, such as a web server, is running on the resource, probe responses come from the guest agent and not from the web server. When you configure floating IP, the probe monitors the health of the back-end virtual machine via its interface IP address. Because the resource is listening for traffic on a logical VIP address, it is important to ensure the virtual machine interface IP address reflects the health of the service associated with the logical address.

Palo Alto Networks Design Details

VM-SERIES FIREWALL ON AZURE

The Palo Alto Networks VM-Series firewall is the virtualized form factor of a next-generation firewall that you can deploy in a range of private and public cloud computing environments. VM-Series firewalls protect public cloud deployments by using application enablement policies while simultaneously preventing known and unknown threats.

VM-Series Firewall Models

VM-Series firewalls on Azure are available in four primary models: VM-100, VM-300, VM-500, and VM-700. Varying only by capacity, all of the firewall models use the same image. A *capacity license* configures the firewall with a model number and associated capacity limits.

Table 2 VM-Series firewall capacities and requirements

Description	VM-100	VM-300	VM-500	VM-700
Firewall throughput (App-ID™ enabled)	1.5 Gbps	2.2 Gbps	2.5 Gbps	2.5 Gbps
Threat Prevention throughput	0.8 Gbps	1.9 Gbps	2.5 Gbps	2.5 Gbps
IPSec VPN throughput	0.5 Gbps	0.9 Gbps	1.3 Gbps	1.63 Gbps
Recommended Instance Size	DS3_v2	DS3_v2	DS4_v2	DS5_v2

Although the capacity license sets the VM-Series firewalls limits, the size of the virtual machine you deploy the firewall on determines its performance and functional capacity. In Table 2, the mapping of VM-Series firewall to Azure virtual machine size is based on VM-Series model requirements for CPU, memory, disk capacity, and network interfaces. When deployed on a virtual machine that provides more CPU than the model supports, VM-Series firewalls do not use the additional CPU cores. Conversely, when you deploy a large VM-Series model on a virtual machine that meets the minimum CPU requirements, it effectively performs the same as a lower model VM-Series.

In smaller VM-Series firewall models, it might seem that a virtual machine size smaller than those listed in Table 2 would be appropriate. However, smaller virtual machine sizes do not have enough network interfaces. Azure provides virtual machines with two, four, or eight network interfaces. Azure virtual machine sizes such the A2 might work if CPU, memory, and disk capacity were the only concern, but they are limited by only having two network interfaces. Because VM-Series firewalls reserve an interface for management functionality, two interface virtual machines are not a viable option. Four-interface virtual machines meet the minimum requirement of a management, public, and private interface. You can configure the fourth interface as a security interface for optional services such as a demilitarized zone (DMZ).

When deployed on larger Azure virtual machine sizes, all VM-Series firewall models support eight network interfaces. However, implementing a security policy between subnets in an Azure deployment might require fewer interfaces than a traditional implementation. Although traditional Layer 3 deployments require an interface on the firewall for each subnet, deployments on Azure do not. Azure networking UDR can send traffic leaving a subnet directly to an interface on a virtual machine, even when that interface isn't part of the subnet. UDR allows the firewall to be in the middle of traffic between subnets in a VNet without having a unique interface in each subnet as is required with a traditional appliance. However, in this case, the same security zone contains all subnets, and an intra-zone policy is required.

Although larger models of VM-Series firewalls offer increased capacities, on Azure, throughput is capped by Azure Network Flow limits. During testing of VM-Series firewalls in Azure, regardless of VM-Series model or the number of firewall interfaces, maximum throughput with App-ID and threat prevention enabled was approximately 2.5 Gbps. For the latest detailed information, see the VM-Series on Microsoft Azure document. Many factors affect performance, and Palo Alto Networks recommends you do additional testing in your environment to ensure the deployment meets your performance and capacity requirements. In general, public cloud environments are more efficient when scaling out the number of resources versus scaling up to larger virtual machine size.

License Options

You can license VM-Series firewalls on Azure with licenses purchased through the Azure Marketplace or regular Palo Alto Networks channels.

Note

Whichever licensing model you chose is permanent. After you deploy them, VM-Series firewalls cannot switch between the pay-as-you-go (PAYG) and bring-your-own-license (BYOL) licensing models. Switching between licensing models requires deploying a new firewall and migrating the configuration. Migration between evaluation, a regular license, and enterprise licensing agreement (ELA) is possible because they are all part of the BYOL licensing model.

Pay as you go (PAYG)—Also called a *usage-based* or *pay-per-use* license. You can purchase this type of license from the Azure public Marketplace, and you are billed hourly.

With the PAYG license, VM-Series firewalls are licensed and ready for use as soon as you deploy it. You do not receive a license authorization code. When the firewall is stopped or terminated in Azure, the usage-based licenses are suspended or terminated.

PAYG licenses support the VM-100, VM-300, VM-500, and VM-700 capacity licenses. A PAYG license applies a VM-Series capacity license based on the hardware allocated to the instance. The PAYG instance checks the amount of hardware resources available to the instance and applies the largest VM-Series firewall capacity license allowed for the resources available. For example, if the instance has 2 vCPUs and 16 GB of memory, and a VM-100 capacity license is applied based on the number of vCPUs. However, if the instance has 16 vCPUs and 16GB of memory, a VM-500 license is applied based on the amount of memory.

PAYG licenses are available in the following bundles:

- **Bundle 1**—Includes a VM-Series capacity license, Threat Prevention license (IPS, AV, malware prevention), and a premium support entitlement
- **Bundle 2**—Includes a VM-Series capacity license, Threat Prevention license (IPS, AV, malware prevention), DNS Security, GlobalProtect™, WildFire, PAN-DB URL Filtering licenses, and a premium support entitlement

Bring your own license (BYOL) and VM-Series ELA—A license that you purchase from a partner, reseller, or directly from Palo Alto Networks. VM-Series firewalls support all capacity, support, and subscription licenses in BYOL.

When using your own licenses, you license VM-Series firewalls like a traditionally deployed appliance, and you must apply a license authorization code. After you apply the code to the device, the device registers with the Palo Alto Networks support portal and obtains information about its capacity and subscriptions. Subscription licenses include Threat Prevention, PAN-DB URL Filtering, AutoFocus™, GlobalProtect, and WildFire.

To accelerate firewall deployment, the VM-Series enterprise licensing agreement (ELA) provides a fixed price licensing option that allows unlimited deployment of VM-Series firewalls with BYOL. Palo Alto Networks offers licenses in one and three-year term agreements with no true-up at the end of the term.

The VM-Series ELA includes four components:

- A license token pool that allows you to deploy any model of the VM-Series Firewall. Depending on the firewall model and the number of firewalls that you deploy, you deduct a specified number of tokens from your available license token pool. All of your VM-Series ELA deployments use a single license authorization code, which allows for easier automation and simplifies the deployment of firewalls.
- Threat Prevention, WildFire, GlobalProtect, DNS Security, and PAN-DB subscriptions for every VM-Series firewall deployed as part of the VM-Series ELA.
- Unlimited deployments of Panorama as a virtual appliance.
- Support that covers all the components deployed as part of the VM-Series ELA.

VM-SERIES FIREWALL INTEGRATION TO AZURE

This section describes the interaction between VM-Series firewalls and Azure services. It begins with a single firewall deployment and then expands with a discussion on resilient deployments.

You deploy VM-Series firewalls in Azure Resource Manager through templates. Predefined VM-Series templates are available through Azure Marketplace and code repositories such as GitHub. The templates available in the marketplace allow you to define settings including the administrator information, resource group, virtual network, subnet, virtual machine size, and VM-Series software version. The templates available in the marketplace default to three interfaces (management, private, and public). You can add additional interfaces to the virtual machine through the CLI.

Bootstrapping

At deployment, VM-Series firewalls have the factory default configuration and a base software image that varies based on which deployment method you have chosen. You can manually upgrade the software and update the configuration after deploying the virtual machine, or you can use bootstrapping to license (if using BYOL), configure, and update the firewall software at boot time.

Bootstrapping allows you to create a repeatable process of deploying VM-Series firewalls through a bootstrap package. The package can contain everything required to make the firewall ready for production or just enough information to get the firewall operational and connected to Panorama. In Azure, you implement the bootstrap package through an Azure file share that contains directories for configuration, content, license, and software. On the first boot, VM-Series firewalls mount the file share and use the information in the directories to configure and upgrade the firewall. After the firewall is out of the factory default state, it stops looking for a bootstrap package.

One of the fundamental design differences between traditional and public-cloud deployments is the lifetime of resources. One method of achieving resiliency in public cloud deployments is through the quick deployment of new resources and quick destruction of failed resources. One of the requirements for achieving quick resource build-out and tear-down is current and readily available configuration information for the resource to use during initial deployment. When the configuration is static, the simplest method of achieving this for VM-Series firewalls is to use bootstrapping to configure the firewall policies during firewall deployment.

Management

The first interface attached to the virtual machine (eth1/0) is the firewall's management interface. In most templates, this interface has a public IP address and DNS hostname attached to it in addition to the internal IP address in the VNet. The firewall's management interface obtains its internal IP address through DHCP. Azure networking translates the internal IP address to the public IP address when the traffic leaves the VNet.

Note

Because public IP addresses might change when virtual machines change state, it is recommended that you use the FQDN to manage the firewall.

You can use an Azure network's security group to restrict access to the firewall's management interface. Use a network security group (instead of limiting the connectivity in the firewall configuration) because it gives you the flexibility to modify the restriction even when the firewall isn't operational. When you have a connection from the VNet back to your on-premises network, such as an ExpressRoute or VPN connection, it might be best to manage the firewall through the internal IP address on the VNet instead of the public IP address. However, consider keeping the public IP address on the management interface to provide a second method of connecting to the firewall in case of configuration error or failure of the connection back to your network.

Managing Azure Deployments with Panorama

The best method for ensuring up-to-date firewall configuration is to use Panorama for central management of firewall policies. Panorama simplifies consistent policy configuration across multiple independent firewalls through its device group and template stack capabilities. When multiple firewalls are part of the same device group, they receive a common ruleset. Because Panorama enables you to control all of your firewalls—whether they be on-premises or in the public cloud, a physical appliance or virtual—device groups also provide configuration hierarchy. With device group hierarchy, lower-level groups include the policies of the higher-level groups. This allows you to configure consistent rulesets that apply to all firewalls, as well as consistent rulesets that apply to specific firewall deployment locations such as the public cloud.

As bootstrapped firewalls deploy, they can also automatically pull configuration information from Panorama. VM-Series firewalls use a VM authorization key and Panorama IP address in the bootstrap package to authenticate and register to Panorama on its initial boot. You must generate the VM authorization key in Panorama before creating the bootstrap package. If you provide a device group and template in the bootstrap package's basic configuration file, Panorama assigns the firewall to the appropriate device group and template so that the relevant rulesets are applied, and you can manage the device in Panorama going forward.

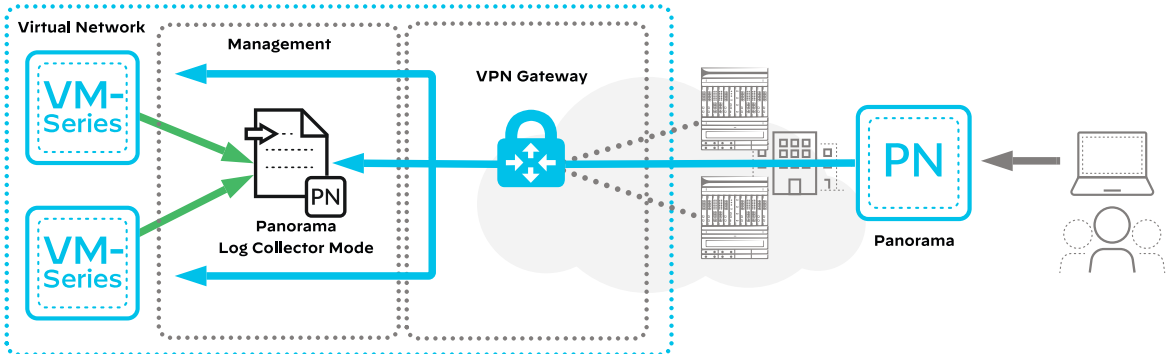
You can deploy Panorama in your on-premises data center or a public cloud provider such as Azure. When deployed in your on-premises data center, Panorama can manage all the physical appliances and VM-Series next-generation firewalls in your organization. If you want a dedicated instance of Panorama for the VM-Series firewalls in Azure, deploy Panorama on Azure.

When you have an existing Panorama deployment on-premises for firewalls in your data center and internet perimeter, you can use it to manage the VM-Series firewalls in Azure. Beyond management, you need to consider your firewall log collection and retention. Log collection, storage, and analysis is an important cybersecurity best practice that organizations perform to correlate potential threats and prevent successful cyber breaches.

On-Premises Panorama with Dedicated Log Collectors in the Cloud

Sending logging data back to the on-premises Panorama can be inefficient, costly, and pose data privacy and residency issues in some regions. An alternative to sending the logging data back to your on-site Panorama is to deploy Panorama dedicated log collectors on Azure and use the on-site Panorama for management. Deploying a dedicated log collector on Azure reduces the amount of logging data that leaves the cloud but still allows your on-premises Panorama to manage the VM-Series firewalls in Azure and have full visibility to the logs as needed.

Figure 10 Panorama Log Collector mode on Azure

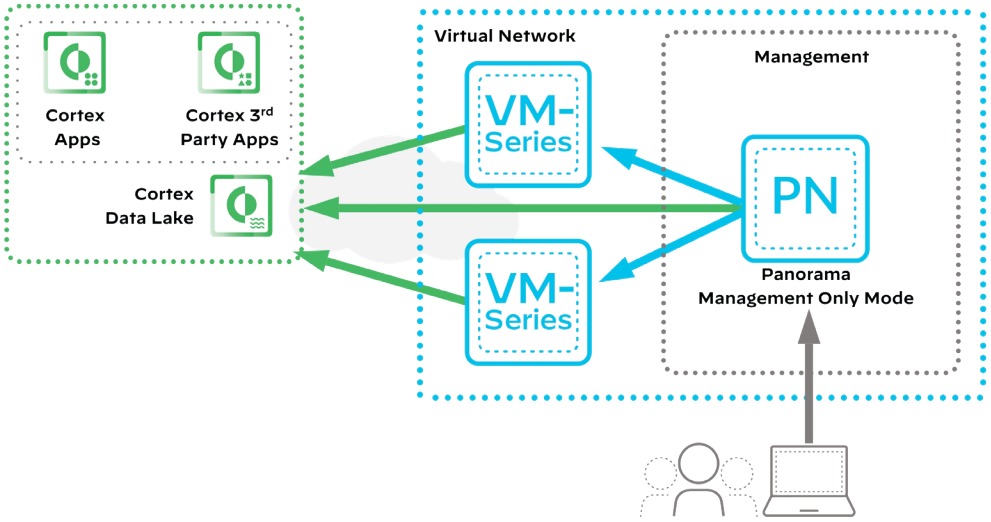


Panorama Management on Azure with Cortex Data Lake

There are two design options when deploying Panorama management on Azure. The first option is using Panorama for management only and using Palo Alto Networks Cortex™ Data Lake to store the logs generated by the VM-Series firewalls. Cortex Data Lake is a cloud-based log collector service that provides resilient storage and fast search capabilities for large amounts of logging data. Cortex Data Lake emulates a traditional log collector. The VM-Series firewalls send encrypted logs to Cortex Data Lake over TLS/SSL connections. Cortex Data Lake allows you to scale your logging storage as your Azure deployment scales because licensing is based on storage capacity and not the number of devices sending log data.

The benefit of using Cortex Data Lake goes well beyond scale and convenience when tied into the Palo Alto Networks Cortex AI based continuous security platform. Cortex is a scalable ecosystem of security applications that can apply advanced analytics in concert with Palo Alto Networks enforcement points in order to prevent the most advanced attacks. Palo Alto Networks analytics applications such as Cortex XDR and AutoFocus, as well as third-party analytics applications that you choose, use Cortex Data Lake as the primary data repository for all Palo Alto Networks offerings.

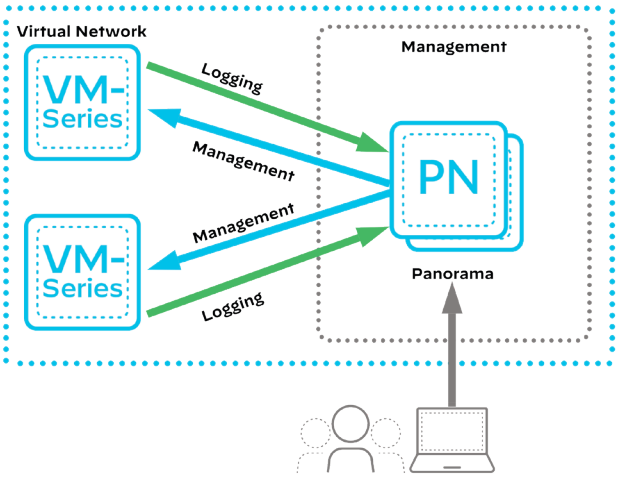
Figure 11 Panorama management and Cortex Data Lake



Panorama Management and Log Collection in the Cloud

The second design option when deploying Panorama management on Azure is using Panorama for both management and log collection. Panorama on Azure supports high-availability deployment as long as both virtual appliances are in the same VNet. You can deploy the management and log collection functionality as a shared virtual appliance or on dedicated virtual appliances. For smaller deployments, you can deploy Panorama and the log collector as a single virtual appliance. For larger deployments a dedicated log collector per region allows traffic to stay within the region and reduce outbound data transfers.

Figure 12 Panorama management and log collection in Azure



Panorama is available as a virtual appliance for deployment on Azure and supports Log Collector mode, Management-Only mode, and Panorama mode with the system requirements defined in Table 3. Panorama on Azure is available with only a BYOL licensing model.

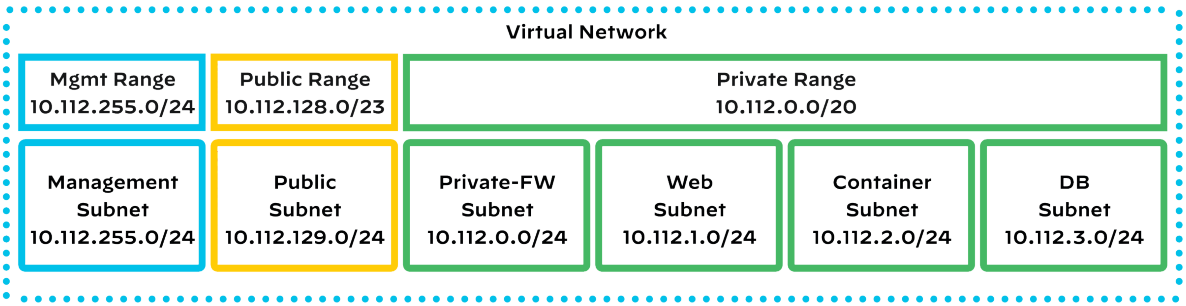
Table 3 Panorama virtual appliance on Microsoft Azure

	Management-Only	Panorama	Log Collector
Minimum system requirements	16 CPUs 32GB memory 81GB system disk	16 CPUs 32GB memory 2TB to 24TB log storage capacity	16 CPUs 32GB memory 2TB to 24TB log storage capacity

Networking and UDR

Each VNet supports multiple IP address ranges, and you can divide each IP address range into subnets. Although you can use a single large IP address range for the whole VNet, having three different ranges simplifies the configuration of traffic forwarding and network security groups. Consider using three ranges: one each for the management network, public network, and private network.

Figure 13 Virtual network IP address ranges and subnets

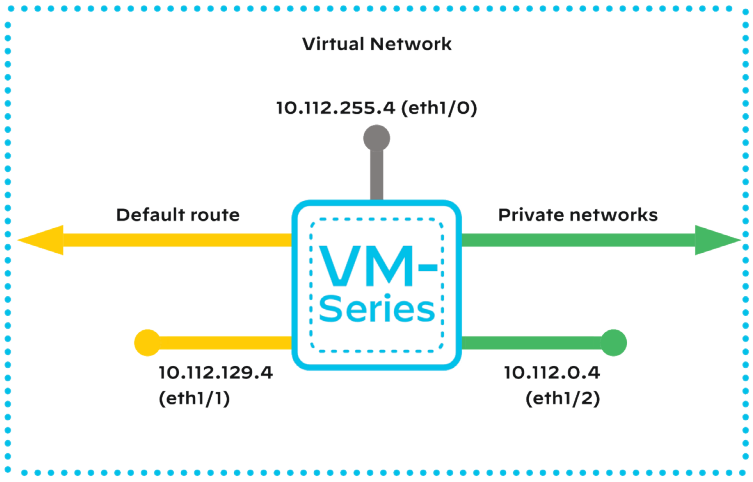


Although the next-generation firewall supports multiple interface deployment configurations such as virtual wire, Layer 2 and tap mode, on Azure, VM-Series firewall interfaces are always Layer 3 interfaces because of Azure's networking requirements.

In a Layer 3 deployment, you must assign each interface an IP address. In Azure, you should always configure VM-Series firewall interfaces to obtain their IP address through DHCP. Because user-defined route tables require a statically defined next-hop IP address, you should configure the firewall's virtual machine interfaces with static IP addresses in the Azure portal or template. The firewall continues to receive its IP address through DHCP, even when you configure a static IP.

By default, when a firewall interface obtains a default gateway from DHCP, it installs a default route. To ensure proper traffic flow, you should modify the firewall configuration so that default routes are static and not obtained through DHCP. To allow the firewall to reach virtual machines and services within the VNet, set up static routes to the VNet internal networks on the firewall's private interface. Even though Azure networking does not use traditional forwarding, you still configure the route's next hop as if the network has a default gateway. Azure reserves the first address in the subnet (example: .1 in a /24) as the subnet's default router address.

Figure 14 Firewall IP routing



Directing Traffic within the VNet

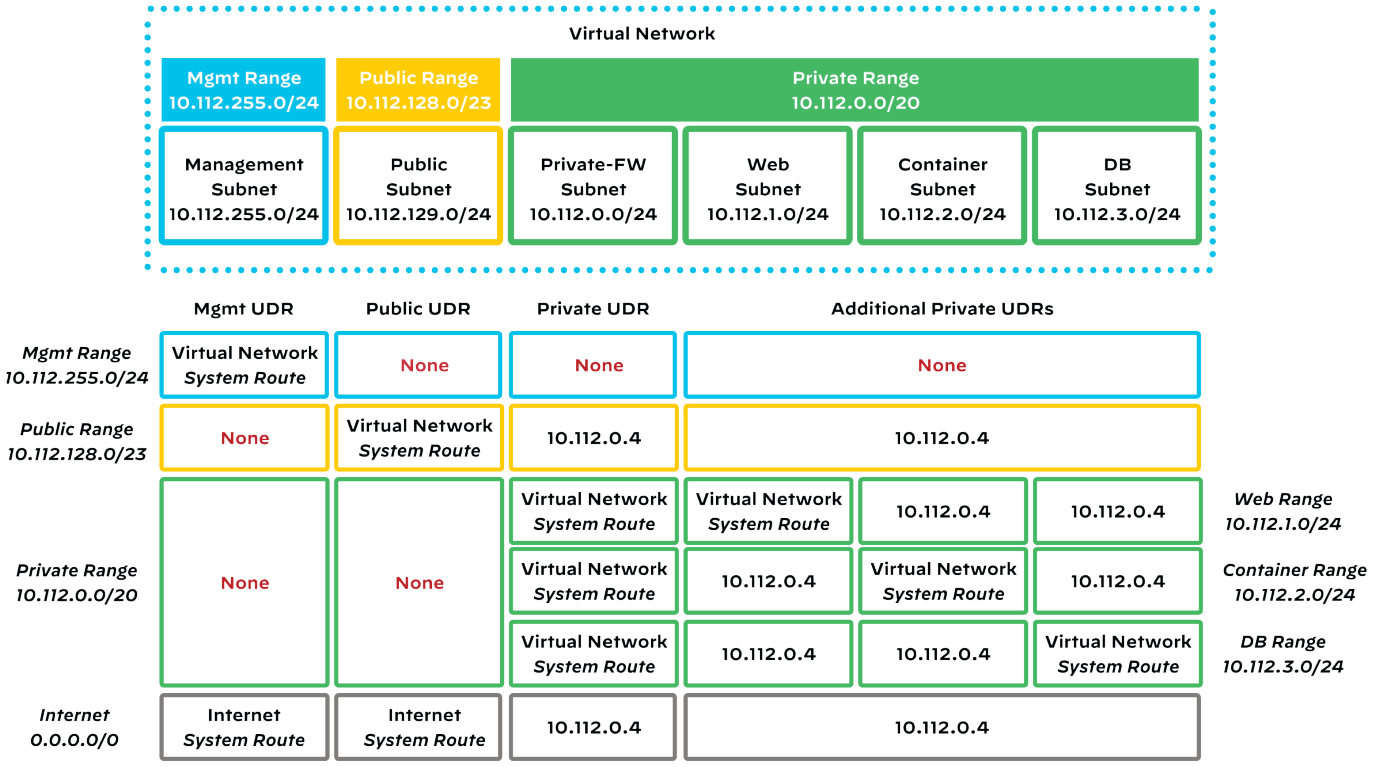
Create route tables to direct traffic through the firewall and stop traffic from flowing directly to the internet or between devices in the VNet. Each subnet in the VNet must have a user-defined route table applied. At the minimum, you need user-defined route tables that direct traffic appropriately within the VNet for the following:

- For the management subnet, the route table should discard all traffic destined to the private and public network ranges by routing the traffic to the next hop of *none*.
- For public subnets, the route table should direct all traffic destined to private network range to the firewall's public internal IP address. Discard traffic destined to the management network range by using the next hop of *none*.
- The subnet attached to the firewall's private interface should have a user-defined route table that directs traffic destined to the internet and public networks to the firewall's private IP address. Discard traffic destined to the management network range by using the next hop of *none*.
- For private subnets, the route table should direct all traffic (destined to the internet, to the private network range, and the public network range) to the firewall's private internal IP address. Discard traffic destined to the management network range by using the next hop of *none*.

Note

If you do not want the firewall in the middle of all east-west traffic between private subnets, add routes only for the private subnets you want to protect. Do not use a summary route that includes multiple private subnets. The summary route can have the unintended consequence of enforcing a security policy on traffic within the subnet (host to host).

Figure 15 User-defined routes next-hop settings



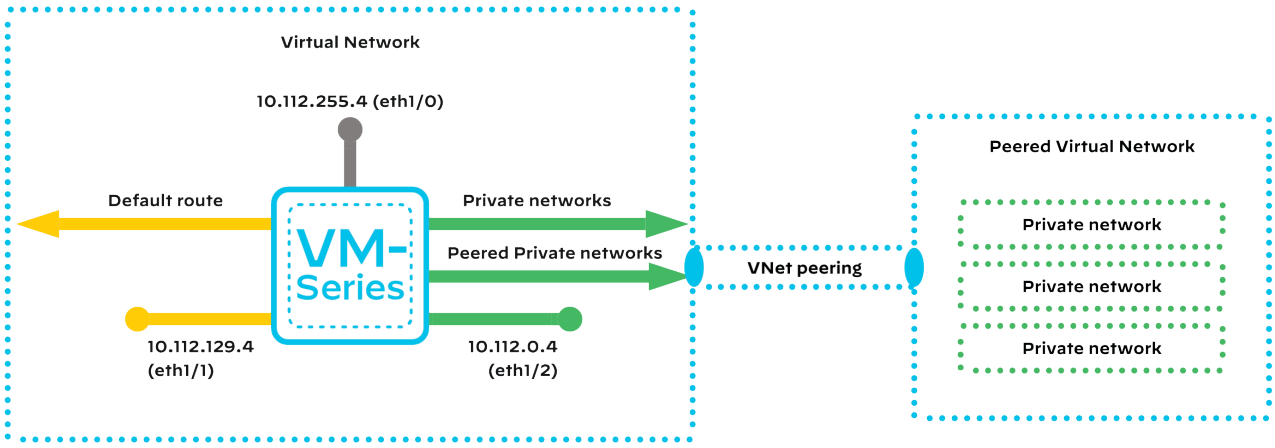
Deploying Firewalls with Multiple Peered VNets

No significant changes are required when you use VNet peering to extend the network to include multiple VNets.

> **Note**
>
> You must attach all of a firewall's interfaces to subnets within the same VNet.

To allow the firewall to reach virtual machines and services in a peered VNet, set up static routes to the peered VNet internal networks on the firewall's private interface. Even though Azure networking does not use traditional forwarding, you still configure the route's next hop as if the network has a default gateway.

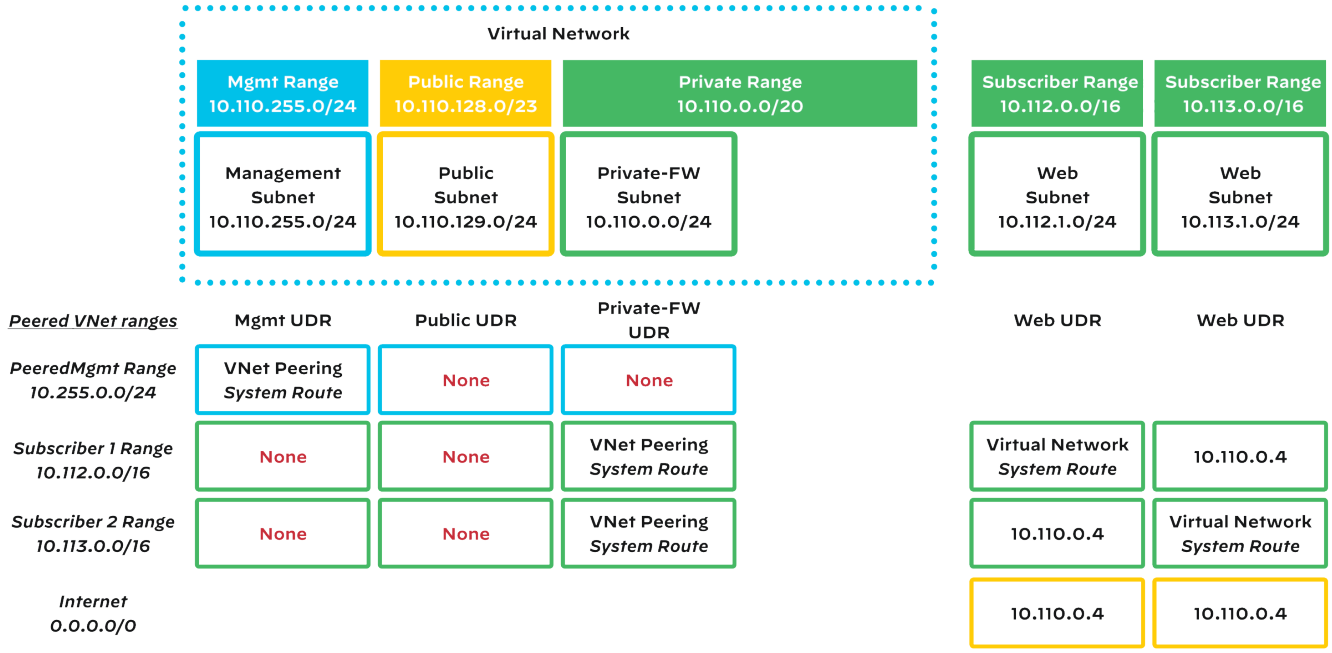
Figure 16 Firewall IP routing with peered VNets



You need to modify the user-defined route tables to include IP ranges in other VNets, as follows:

- For the management subnet, the route table should discard all traffic destined to the peered VNet private and public network ranges by routing the traffic to the next hop of *none*.
- For public subnets, the route table should direct all traffic destined to the peered VNet private network range to the firewall's public internal IP address. The route table discards traffic destined to the peered VNet management network range by using the next hop of *none*.
- The subnet attached to the firewall's private interface and other private subnets should discard traffic destined to the peered VNet management network range by using the next hop of *none*. This subnet should have a user-defined route table that directs traffic destined to the peered VNet public networks to the private IP address of a firewall in the peered VNet.

Figure 17 Additional user-defined routes next-hop settings for peered VNets



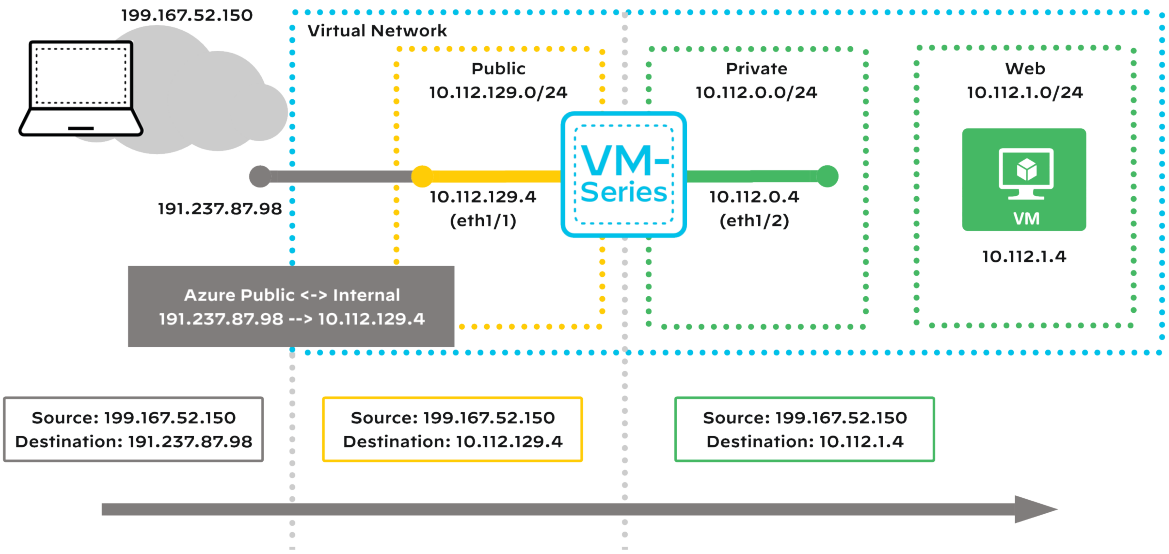
Traffic Flows

Inbound Traffic from the Internet

To allow a client on the internet to communicate with a resource behind the firewall, associate a public IP address with the resource. Although it is possible to associate public IP addresses directly to virtual machines in the private subnets, for the firewall to be able to protect the private resources, you must associate the public IP address with the firewall. The firewall then translates the destination IP address to the appropriate private resource.

Because Azure networking translates the destination IP address from the public to the internal IP address when the traffic enters the VNet, you must use internal IP addresses in the firewall's security and NAT policies. Although you can associate a public IP address to the primary internal IP address on the virtual machine's public interface, deploying this way requires port translation to support multiple private resources. To avoid port translation, you can have multiple secondary IP addresses assigned to the firewall's virtual machine public interface, one for each public IP address you associate with the firewall.

Figure 18 Inbound IP address translation



Outbound Traffic to the Internet

Traffic that originates from a virtual machine on a private subnet and is destined to the internet routes to the firewall through the user-defined route table applied to the virtual machine's subnet.

For virtual machines behind the firewall to communicate to devices on the internet, the firewall must translate the source IP address of the outbound traffic to an IP address on the public subnet. Azure then translates the source IP address again as the outbound traffic leaves the VNet. When you associate a public IP address with an internal IP address used in the NAT policy, Azure translates the outbound traffic to the public IP address.

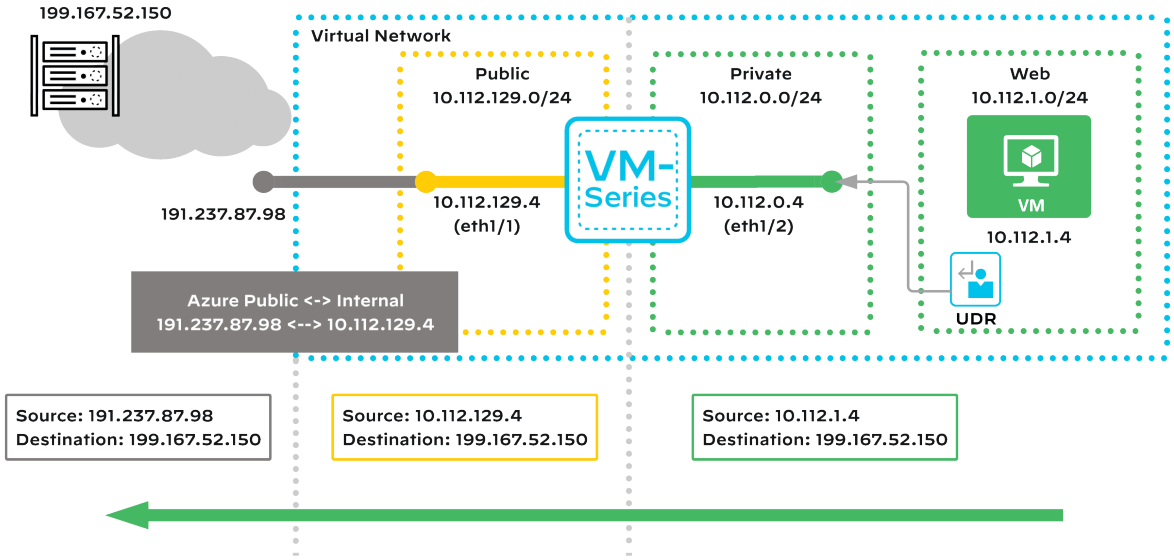
Note

The default behavior for Azure networking with outbound traffic is to use NAT to translate the source address to an automatically designated public IP address. If your virtual machine has an interface with an associated public IP address or has an interface that is a member of a load balancer or application gateway back-end pool, then the default behavior does not apply.

In all cases where the default behavior does not apply, then you must associate an Azure public IP address to the firewall's outbound interface.

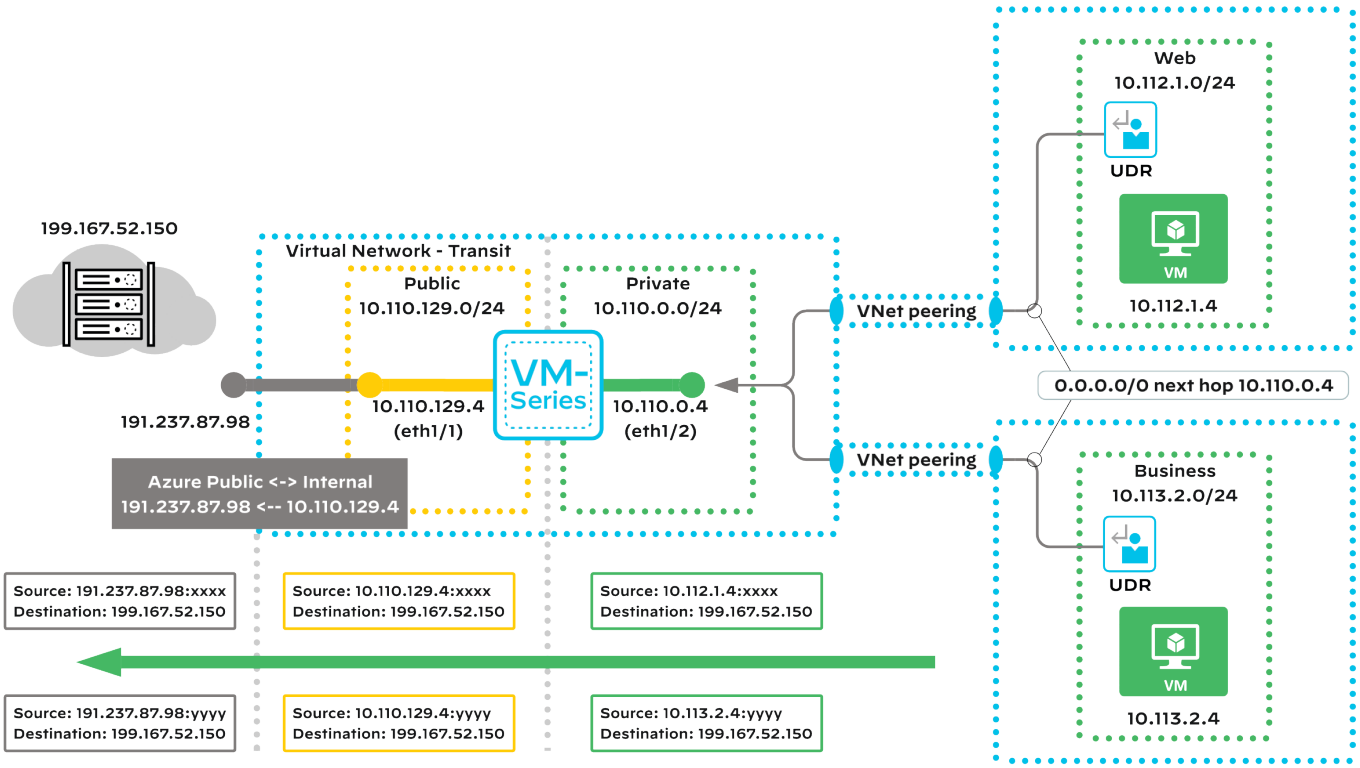
The IP address used in the NAT policy can either be the public interface IP address or any other IP address on the public subnet. When using non-interface IP addresses, ensure that there aren't any IP address conflicts by statically assigning all IP addresses used in the NAT policy as secondary IP addresses on the firewall's virtual machine public interface.

Figure 19 Outbound IP address translation



In large-scale deployments, you can provide outbound access through a peered VNet, commonly referred to as a *transit VNet*. This topology is similar to a hub-and-spoke design, with the transit VNet performing the hub role and the subscriber VNets as spokes. You share the firewall resources in the transit VNet across all of the subscriber VNets.

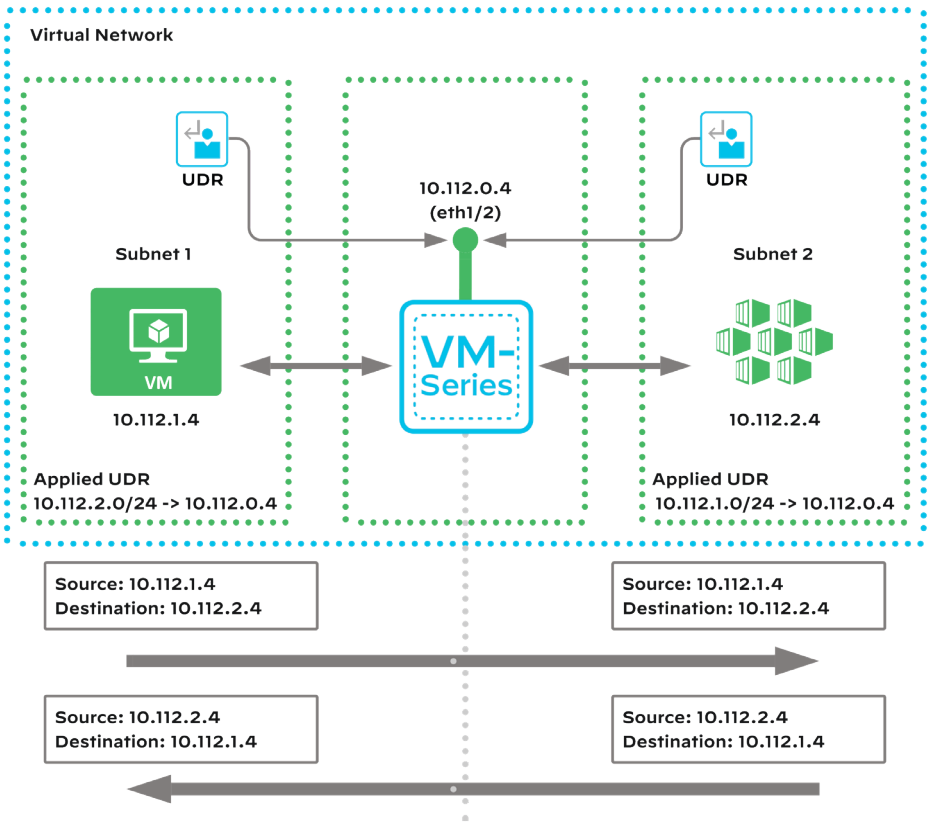
Figure 20 Outbound access with transit VNet



East-West Traffic between Private Subnets in the Same VNet

Traffic that originates from a virtual machine within a private subnet and is destined to a virtual machine in a different private subnet in the same VNet routes to the firewall through a user-defined route table applied to the virtual machine's subnet. Virtual machines that can communicate to each other without the need for a firewall to protect the traffic can be on the same subnet, and virtual machines that need traffic protection should be on different subnets. Because both ends of the communication are within the VNet, the firewall should not apply a NAT policy to traffic between private subnets.

Figure 21 Traffic flow between private subnets in the same VNet



Because you use a UDR to forward traffic, traffic between private subnets ingresses and egresses the firewall on the same interface and is assigned the same zone. To permit only limited traffic between private subnets, you must create a rule above the default intrazone security policy rule. You should then modify the default intrazone security policy to deny traffic, because it allows all traffic within a zone by default.

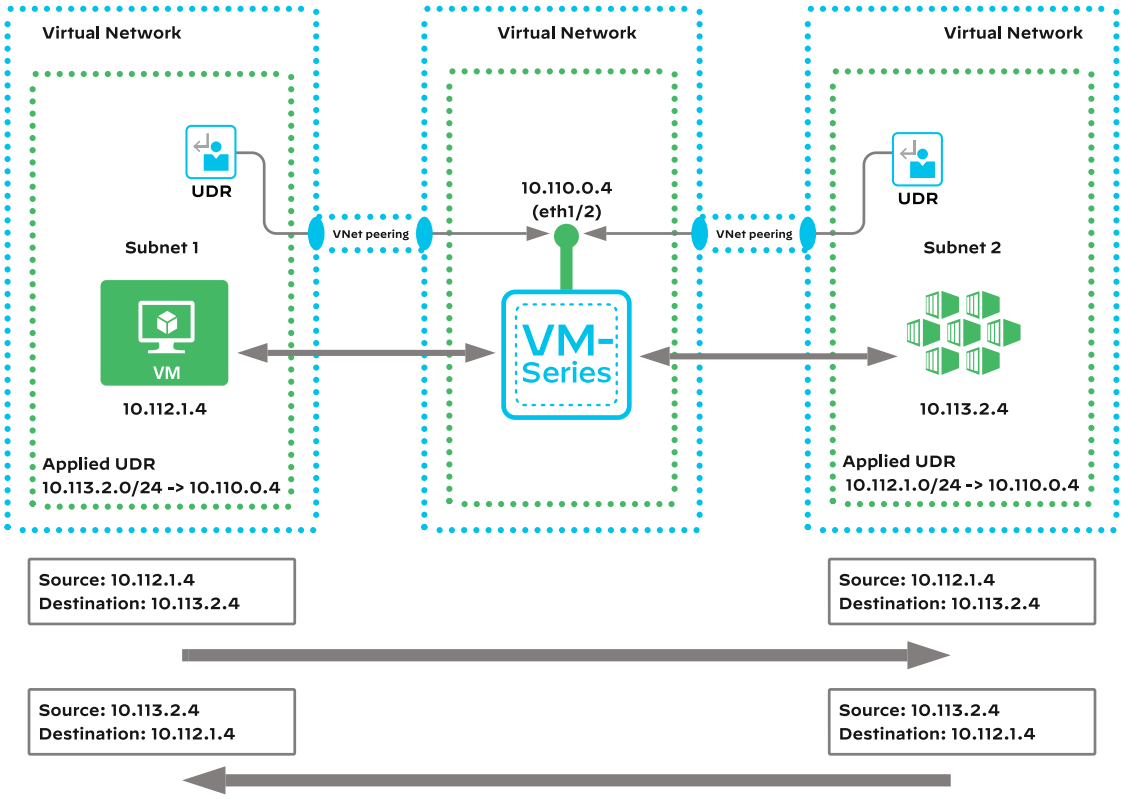
Note

To help with troubleshooting, consider modifying the default intrazone security policy to log traffic.

East-West Traffic between Private Subnets in Different VNets

Traffic that originates from a virtual machine within a private subnet in one VNet and is destined to a virtual machine in different private subnet in a different VNet routes to the firewall through a user-defined route table applied to the virtual machine's subnet. Because both ends of the communication are within peered VNets, the firewall should not apply a NAT policy to traffic between private subnets.

Figure 22 Traffic flow between private subnets in different VNets



Because you use a UDR to forward traffic, traffic between private subnets in different VNets ingresses and egresses the firewall on the same interface and is assigned the same zone. To permit only limited traffic between private subnets, you must create a rule above the default intrazone security policy rule. You should then modify the default intrazone security policy to deny traffic, because it allows all traffic within a zone by default.

Connectivity to On-Premises Networks Using a Virtual Network Gateway

The virtual network gateway connects on-premises networks to the Azure virtual network. You deploy the VNG in a dedicated gateway subnet. In Azure, you define the IP address ranges that route to the on-premises networks when you configure a connection between the virtual network gateway and the on-premises local network gateway, or Azure can learn them through the BGP routing protocol. By default, all

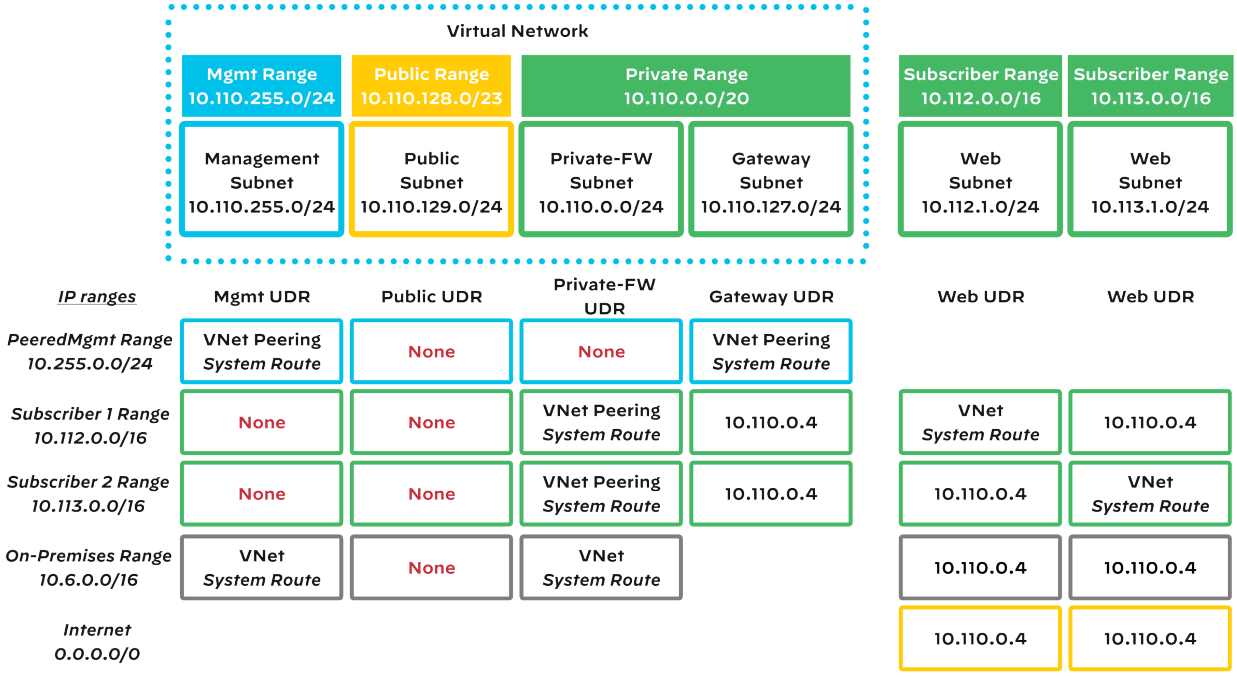
resources within the VNet can communicate with the on-premises network ranges. Azure automatically creates system routes to the on-premises network ranges during the deployment of the virtual network gateway and when dynamically learned. The system routes for the on-premises network ranges have a next hop of *virtual network gateway*.

You should override the system routes with user-defined routes that either direct the traffic to the firewall or discard the traffic with a destination of *none*. You should create a route table for the gateway subnet to direct traffic that is destined to the private subnets to the firewall. If you want to manage the firewalls from your on-premises network, you can also route traffic to the management subnet directly through the virtual network.

Note

Because you must override each on-premises network range in the route table, you should summarize the network ranges as much as possible. This reduces the amount of configuration required.

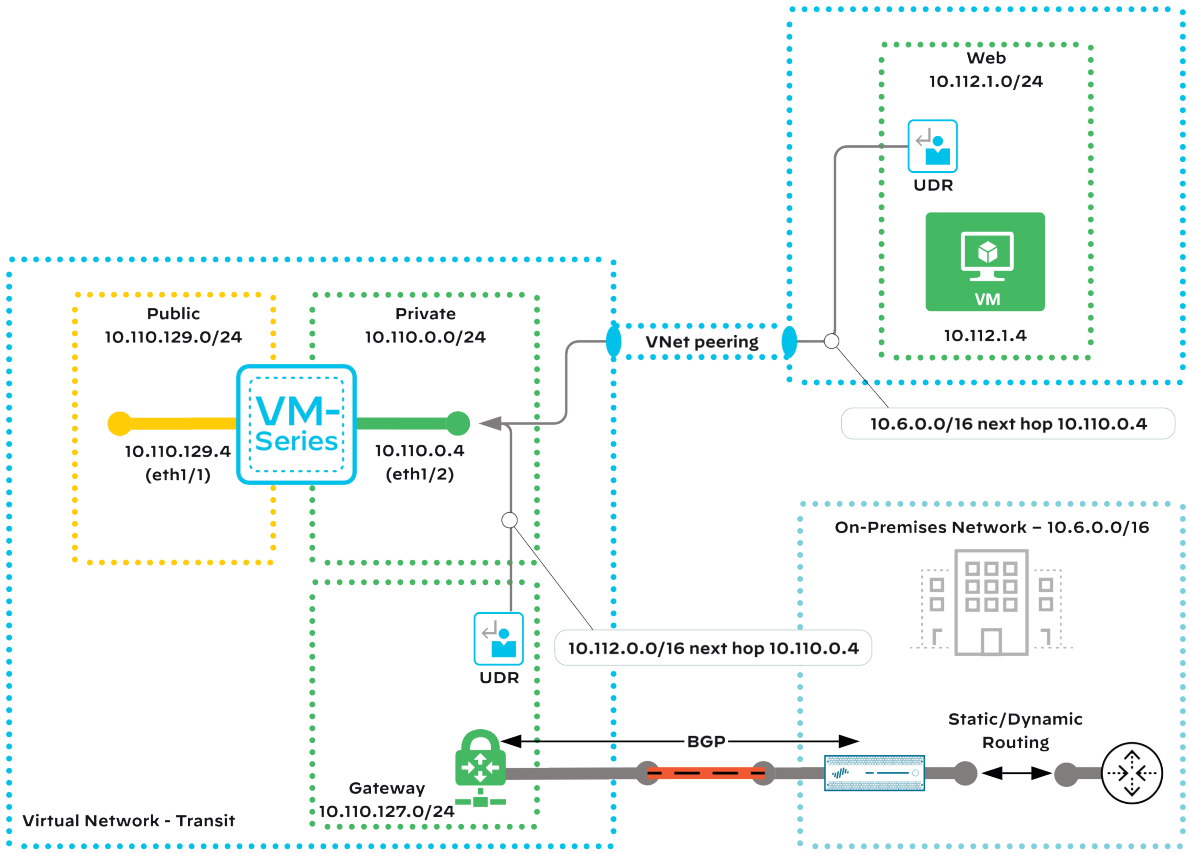
Figure 23 User-defined routes for on-premises networks



Note

When using VNet peering, you can configure only VNets without a VNG to use a remote VNG in a peered VNet.

Figure 24 Backhaul using VNG in peered VNets



When you are using the VNG for connectivity to on-premises networks, Azure automatically creates system routes for every network prefix that is configured or learned. The user-defined route configuration for on-premises network ranges must include discard routes with a destination of *none* applied to subnets in the public range to override the system routes. Managing the list of user-defined discard routes becomes cumbersome when the number of on-premises network system routes increases. If you enable BGP dynamic routing for your VNG, then you can disable automatic system route propagation for individual subnets instead of creating discard routes.

If the transit VNet is used for outbound traffic to the internet, then a user-defined route that matches internet destinations is already applied. This user-defined route might also be sufficient to direct traffic to the on-premises networks, as well. If not, then subnets in the private range might require an additional user-defined route to the transit VNet.

Resiliency

Traditionally, you achieve firewall resiliency through a high availability (HA) configuration on the firewall. In a high availability configuration, a pair of firewalls shares configuration and state information that allows the second firewall to take over for the first if a failure occurs. Although you can configure high availability so that both firewalls are passing traffic, in the majority of deployments, the firewalls operate as an active/passive pair where only one firewall is passing traffic at a time.

Unlike traditional implementations, this architecture achieves VM-Series resiliency in Azure through the use of native cloud services. The benefits of configuring resiliency through native public cloud services instead of firewall high availability are faster failover and the ability to scale out the firewalls as needed. However, in a public cloud resiliency model, configuration and state information is not shared between firewalls. Applications typically deployed in public cloud infrastructure, such as web- and service-oriented architectures, do not rely on the network infrastructure to track session state. Instead, they track session data within the application infrastructure, which allows the application to scale out and be resilient independent of the network infrastructure. For information about high availability in VM-Series on Azure, see Set up Active/Passive HA on Azure.

The Azure resources and services used to achieve resiliency for the firewall include:

- **Availability sets**—Ensure that a failure or maintenance event in Azure does not affect all VM-Series firewalls at the same time.
- **Load balancers**—Distribute traffic across two or more independent firewalls that are members of a common availability set. Every firewall in the load balancer's pool of resources actively passes traffic, allowing firewall capacity to scale out as required. The load balancer monitors the availability of the firewalls through TCP or HTTP probes and updates the pool of resources as necessary.

Note

Azure Load Balancer is available in both a Basic and Standard SKU. The Standard SKU load balancer has an expanded feature set and increased scale and is the recommended resource for this architecture guide.

- **Multiple application gateway instances**—Distribute traffic across two or more independent firewalls that are members of a common availability set. Every firewall in the application gateway's pool of resources actively passes traffic, allowing firewall capacity to scale out as required. The application gateway monitors the availability of the web server back-end resources through HTTP/HTTPS probes and updates the pool of resources as necessary.

Another way that firewall resiliency in Azure differs from traditional firewall high availability is that in Azure you do not implement firewall resiliency at a device level. Instead, you implement firewall resiliency based on the direction of the traffic. For example, it is possible to configure firewall resiliency for inbound traffic from the internet and its return traffic, but not for outbound traffic originating from private virtual machines. In fact, the resiliency for inbound traffic differs from that of outbound and east–west traffic.

Resiliency for Inbound Traffic

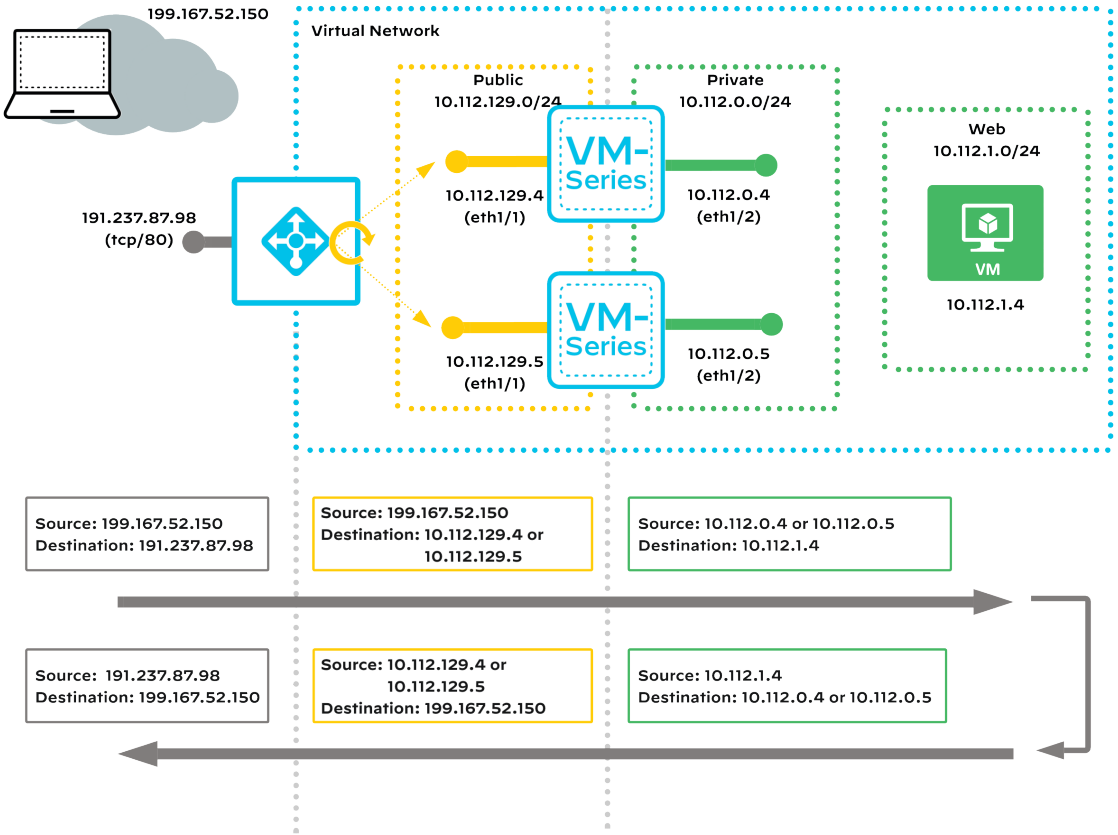
You implement resiliency for inbound traffic from the internet through the use of an Azure public load balancer or an Azure application gateway.

Resiliency for Inbound Traffic with Azure Public Load Balancers

Public load balancers have one or more public IP addresses configured on the front end and have a back-end pool associated to the public interfaces of the firewalls. Load-balancing rules direct traffic to the firewalls based on the destination IP address and TCP or UDP port numbers. By default, the load balancer translates the destination IP address to the public interface IP address of the firewall selected from the back-end pool.

To get the traffic to a resource in the private zone, a NAT policy rule on the firewall must translate the destination IP address from its public interface IP address to the resource IP address. To ensure traffic symmetry, or that return traffic from the resource leaves through the firewall that processed the incoming traffic, the firewall must also translate the source IP address to the IP address of its private interface. Without this source NAT, Azure uses routing to select the path out of the VNet.

Figure 25 Default load-balancer traffic flow

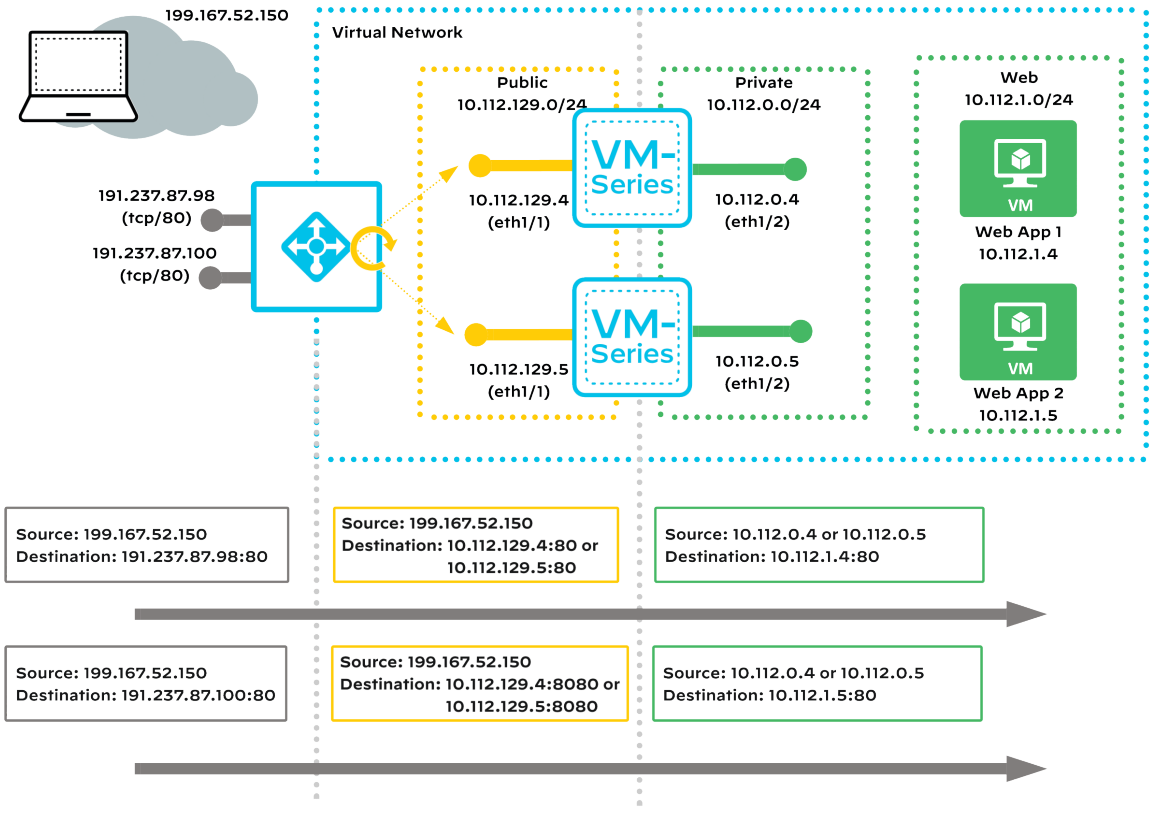


> **Note**
>
> Although you can use an interface in the source NAT configuration, NAT policy rules cannot use an interface to define destination IP address match criteria.
>
> Statically defining the firewall interface IP addresses in Azure is important to ensure that the NAT policy rule stays valid through virtual machine reboots.

There are a few design limitations when using the default load-balancer behavior. The first limitation is that, to support multiple applications with the same destination port number, you must use port translation. This requirement stems from the fact that each firewall in the back-end pool is limited to one IP address. There is no differentiation in destination IP address even when there are multiple front-end IP addresses receiving traffic. So, although the load balancer can receive traffic on two different public IP addresses listening on the same TCP port, it cannot forward that traffic to a common set of firewalls without translating the destination port. Load-balancing rules that share a back-end pool must have different back-end ports.

To support multiple applications while using the load balancer's default behavior, the firewall NAT policy rules must use the service (TCP port number) as part of the traffic match and translate the destination port to what the private resources expect. You should also configure the service for security policy rules to include the specific service ports in use instead of *application-default*.

Figure 26 Default load balancer with multiple applications



The second design limitation of the default Azure load-balancer behavior relates to health probes. Health probes determine the health of the firewalls in the back-end pool. The load balancer sends health probes to the IP address defined in the back-end pool, in this case, the primary internal IP address of the firewall's public interface. Although you can configure the health probes to monitor the full path to the private resources (because the firewall NAT and security policies use the public interface IP address), directly monitoring the health of the firewalls might be preferable in designs where the health probe ends up monitoring the same resource through multiple firewalls.

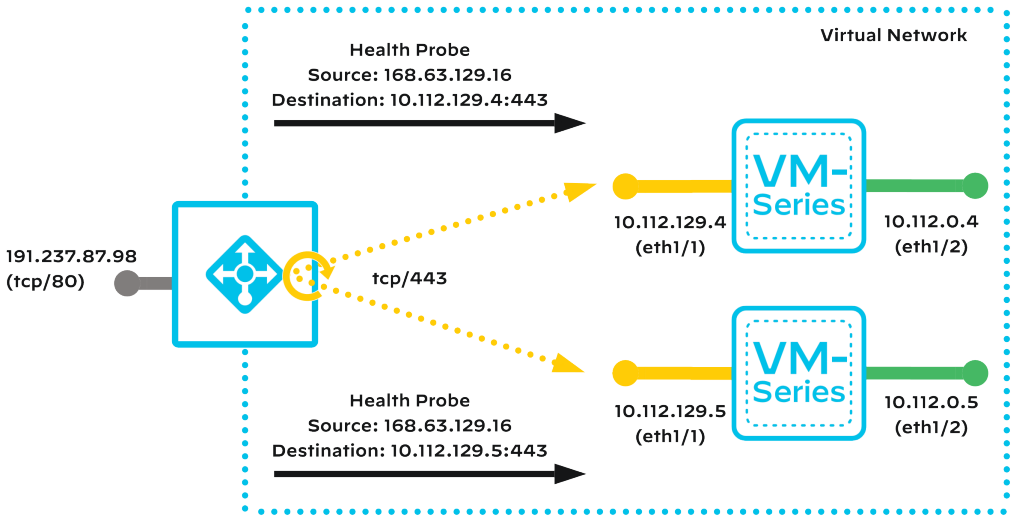
Because a TCP probe only expects an ACK, the simplest method of determining firewall health is to enable an interface management profile on the firewall interface that permits access to either the SSH or HTTPS service from the Azure health probe.

Note

Azure always sources health probes from an IP address of 168.63.129.16.

However, because the load balancer uses the firewall's public IP address as the destination for traffic as well as the health probes, you need to take care to ensure the services configured in the interface management profile don't overlap with the traffic sent to the firewall from the load balancer. For example, if the load balancer is sending HTTP traffic to the firewall, HTTP should not be enabled in the interface management profile.

Figure 27 Load-balancer health probes



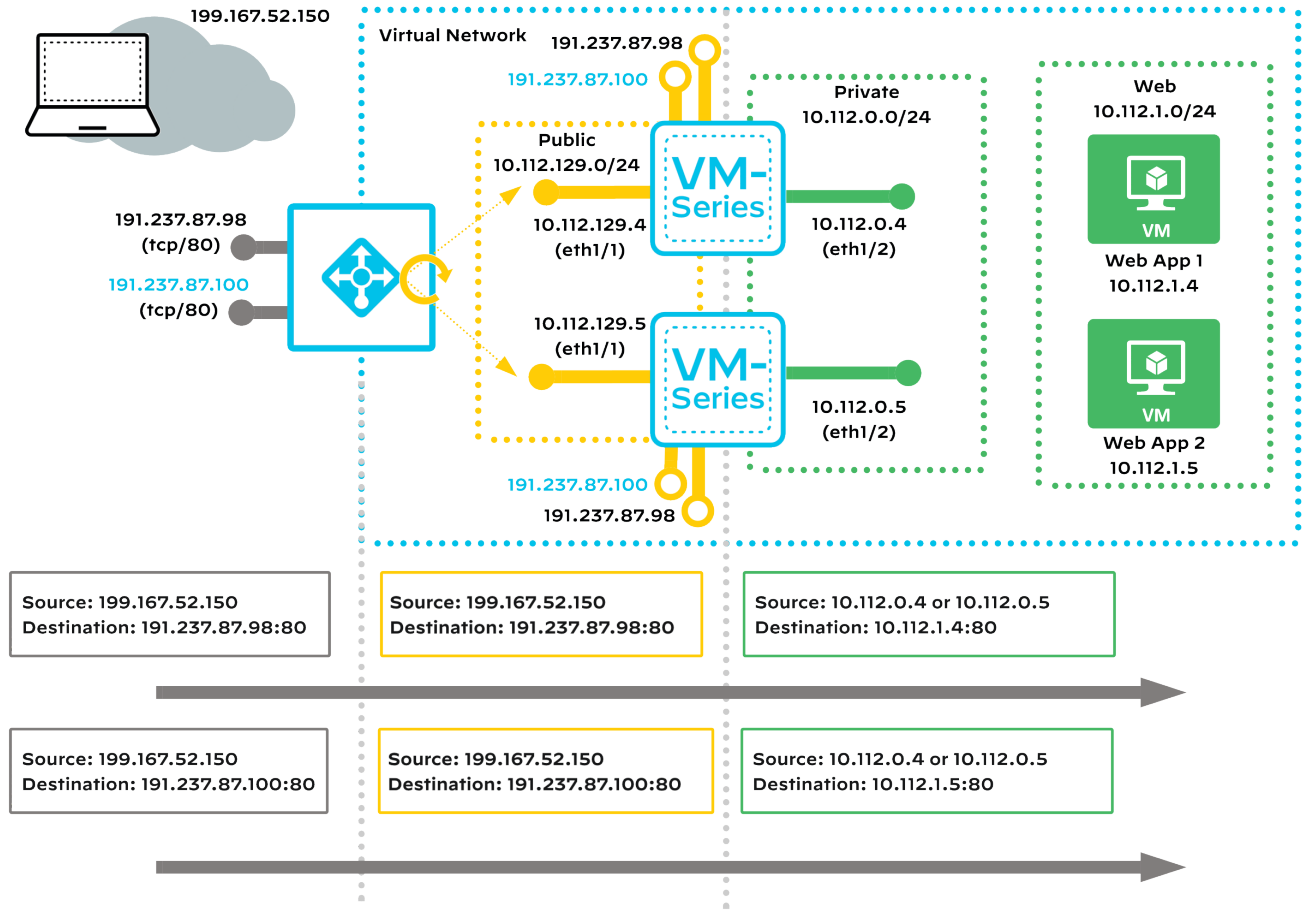
If you want to avoid the design limitations of the default load-balancer behavior, you can configure the load balancer by using floating IP. When set to use floating IP, the load balancer does not translate the destination IP address before sending the traffic to resources in the back-end pool.

To get the traffic to a resource in the private zone, a NAT policy rule on the firewall must translate the

destination IP address from the public IP address assigned to the load balancer to the private resource IP address. When using dynamic public IP address assignment, use FQDN address objects in the firewall NAT and security policies. FQDN address objects resolve hostnames to IP addresses on firewall boot up and periodically after that to keep firewall policies current.

Floating IP removes the design limitations of the load balancer. Supporting multiple applications that use the same destination port number no longer poses an issue when using floating IP because the firewall can use the unique destination IP address to differentiate applications. Also, when you use floating IP, there won't be any overlap between the services defined in the interface management profile and the services defined in the firewall policies.

Figure 28 Traffic flow with floating IP

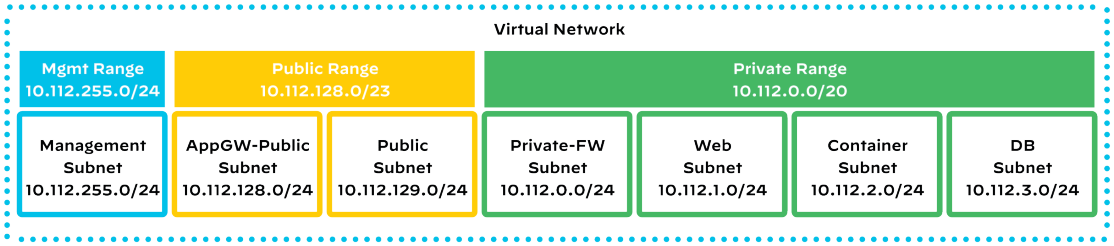


Finally, beyond removing design limitations, floating IP allows for simplified firewall configuration. Floating IP removes firewall interface IP addresses from the policies and replaces it with IP addresses that are consistent across all devices. This allows for consistent policies across firewalls.

Resiliency for Inbound Traffic with Azure Application Gateway

The application gateway requires that you deploy it in a dedicated subnet. You must create a new subnet in the public network range that you use for the application gateway. You cannot deploy other resource types in this subnet.

Figure 29 Virtual network IP address ranges and subnets with application gateway



Application gateways used for inbound traffic have one or more front ends, each associated to a public IP address. You can assign multiple FQDNs to a public IP address, and you can use information in the URL in the application gateway's forwarding rules. The application gateway uses a back-end pool associated to the public interfaces of the firewalls when used for inbound resiliency. You configure routing rules to associate a front end with a back-end pool.

The application gateway is a proxy device and terminates inbound connections by using listeners. You configure the first routing rule when you initially deploy the application gateway. The routing rule includes a listener that you configure for HTTP or HTTPS on a TCP port you select. After the initial deployment, you can create additional routing rules and listeners using HTTP or HTTPS on other TCP ports. The application gateway initiates back-end connections sourced from its own instance IP addresses in the application gateway subnet. The web server resources in the private zone can be individual servers or servers configured as the back-end pool of a separate internal load balancer.

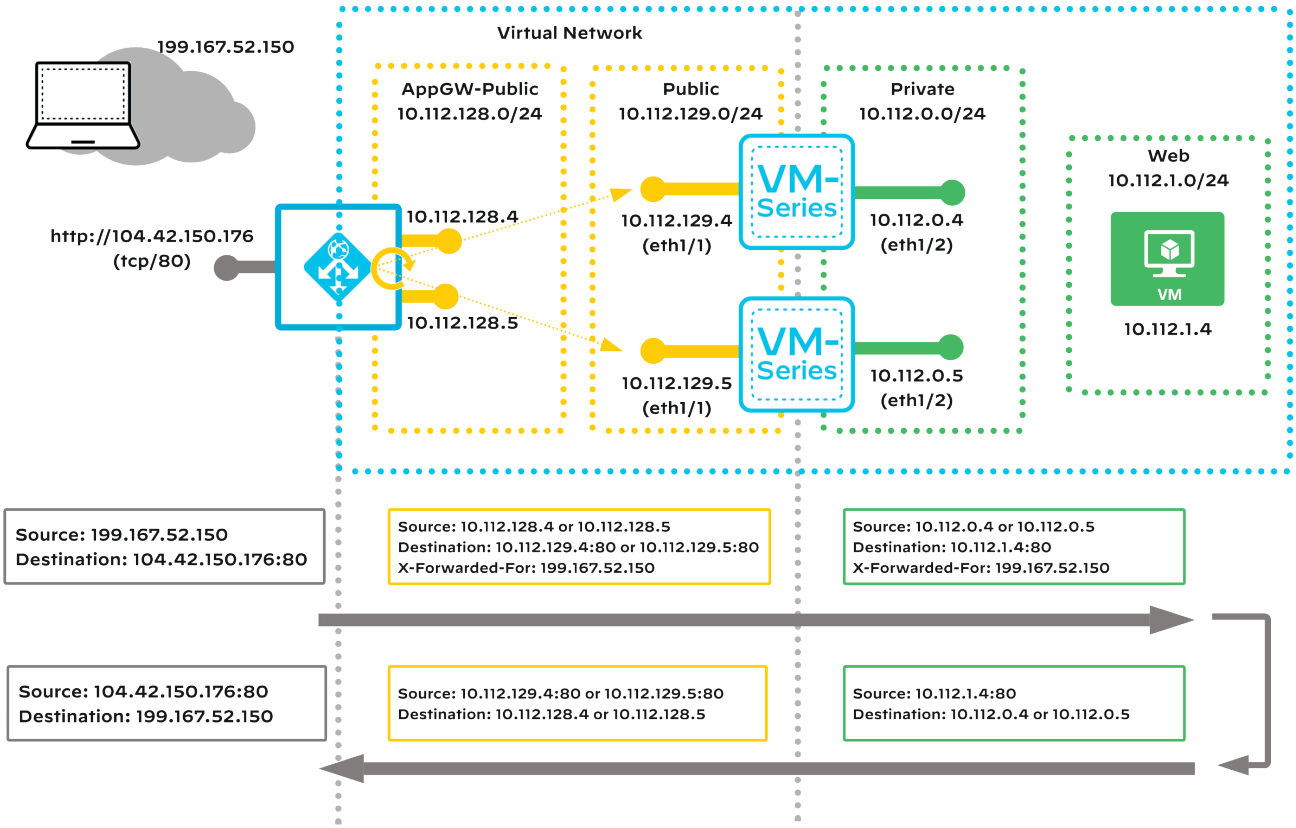
Note

The application gateway initiates sessions to the back-end pool by using its own source IP addresses (one per instance). The application gateway uses the X-Forwarded-For (XFF) HTTP header field in order to add the original source IP address of the web client to the HTTP packet header. The firewall logs the XFF information in addition to other session data in order to retain information about the original source IP address for each session.

Basic forwarding rules direct traffic to the back-end pool based on the port and protocol of the incoming connection established to the listener and based on the destination port and protocol of the back-end resource behind the firewall. Path-based forwarding rules are similar to the basic forwarding rules but also use regular expression matching within the URL to direct traffic.

To get the traffic to a resource in the private zone, a NAT policy rule on the firewall must translate the destination address of any traffic sourced from the application gateway from the firewall's public interface IP address to the back-end resource IP address. The back-end resource can be a server or the front-end IP of an internal load balancer. To ensure traffic symmetry, the firewall must also translate the source IP address to the IP address of its private interface.

Figure 30 Default application gateway traffic flow

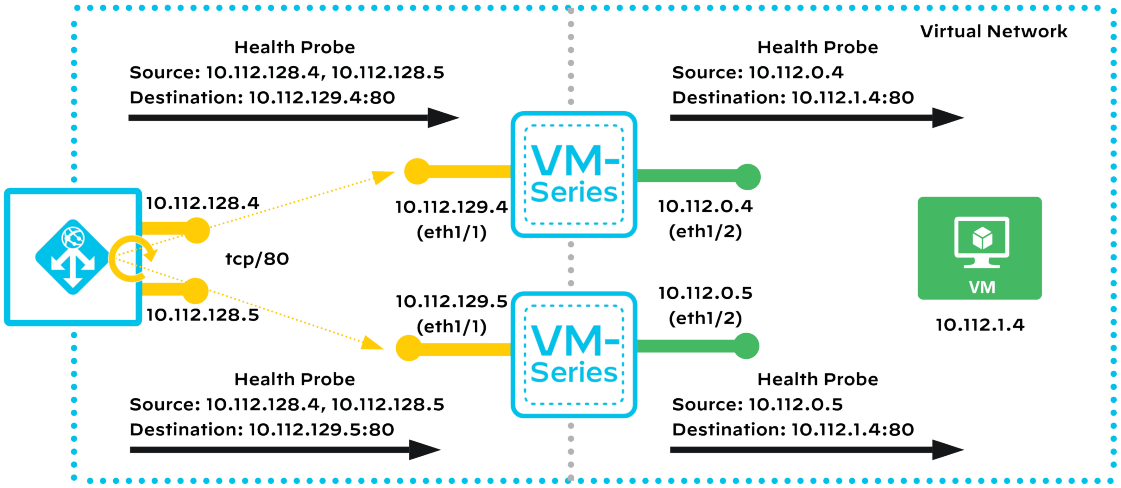


Health probes for the application gateway determine the health of the actual back-end resources. Unlike the Azure load balancer, the application gateway sources health probes from its IP addresses. You configure the firewall with NAT and security policies that permit the probes to pass directly to the back-end resources. A successful health probe verifies that both the firewall and the actual back-end resource are available.

Note

If you configure the firewall with a management profile on the public interface for HTTP, HTTPS, SSH or Telnet, then traffic to TCP/80, TCP/443, TCP/22 or TCP/23 does not pass to the back-end resources, and those ports cannot be used for the back end with the application gateway. If the management profile permits the application gateway source IPs, the health probes might succeed, but this does not verify health of the back-end resources.

Figure 31 Application gateway health probes



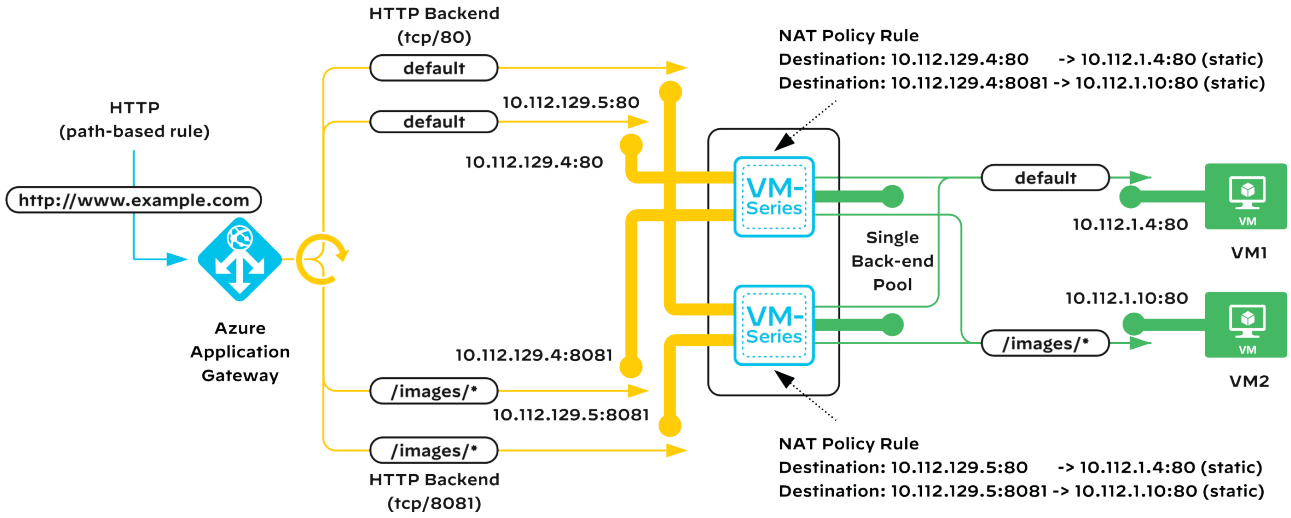
A limitation of the firewall design with the application gateway is that the back-end pool can only include the public interfaces of the firewalls and their corresponding IP addresses. To support multiple web server resources behind the firewalls requires the configuration of destination port NAT policy rules. Each application gateway HTTP or HTTPS back end requires the assignment of unique TCP port.

Table 4 Example application gateway configuration with destination port NAT

Front-end listener	Path	Usage	Back-end protocol/ port	Web server resource	Firewall destination port NAT policy rules
HTTP/80	All	Default	HTTP/80	Web-1 (http://10.112.1.4:80)	10.112.129.4:80 -> 10.112.1.4:80 10.112.129.5:80 -> 10.112.1.4:80
HTTP/80	/images/*	URL path-based routing	HTTP/8081	Web-2 (http://10.112.1.10:80)	10.112.129.4:8081 -> 10.112.1.10:80 10.112.129.5:8081 -> 10.112.1.10:80
HTTPS/443	All	Re-encrypt	HTTPS/443	Web-3 (https://10.112.1.20:443)	10.112.129.4:443 -> 10.112.1.20:443 10.112.129.5:443 -> 10.112.1.20:443
HTTPS/443	/images/*	SSL offload	HTTP/8443	Web-2 (http://10.112.1.10:8443)	10.112.129.4:8443 -> 10.112.1.10:8443 10.112.129.5:8443 -> 10.112.1.10:8443

The example NAT policy shown in Table 4 requires that you create a new NAT policy rule on each firewall for each HTTP/HTTPS back end on the application gateway. The example in the table assumes that two firewalls are in the application gateway back-end pool and each table entry lists a pair of rules. However, you configure each firewall with only the specific NAT policy rules that correspond to its public interface IP address.

Figure 32 Application gateway with single back-end pool (firewalls)



If you use an internal load balancer for the web server resources, then the NAT policy on the firewall references the front-end IP of the load balancer. You configure the internal load balancer to distribute load across web server back-end pool members and optionally, to perform destination port based NAT. A benefit of using the internal load balancer for NAT is that the web servers can continue to use standard web ports (80/443), and you do not need to configured the web servers to listen on non-standard ports (8081) unless the same web server resource has multiple usages.

The combination of the firewall NAT policy rules in Table 5 and the load balancer rules in Table 6 combines the capabilities of the firewall-only configuration with an internal load-balancer configuration.

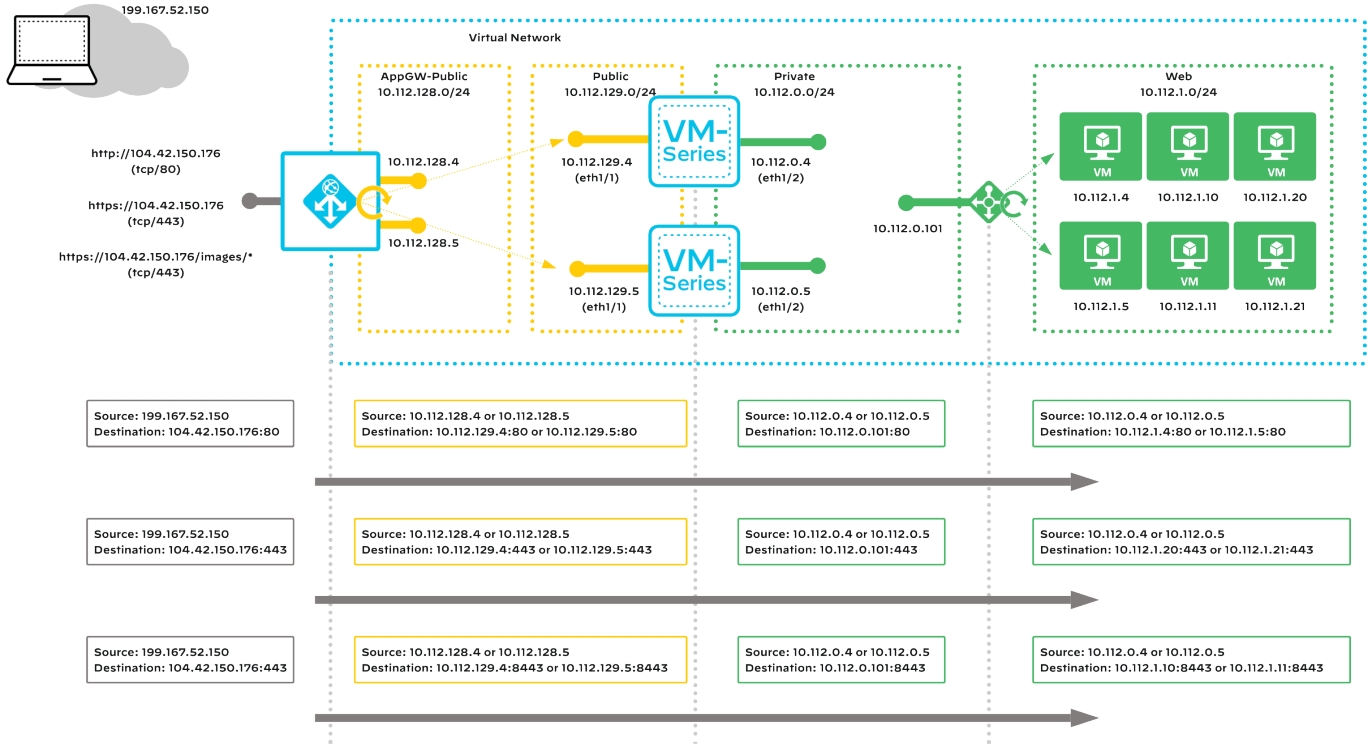
Table 5 Example application gateway configuration with destination NAT to load-balancer front end

Front-end listener	Path	Usage	Back-end protocol/ port	Web server resource	Firewall destination NAT policy rule
HTTP/80	All	Default	HTTP/80	Web-Pool-1	10.112.129.4:80 -> 10.112.0.101:80 10.112.129.5:80 -> 10.112.0.101:80
HTTP/80	/images/*	URL path-based routing	HTTP/8081	Image-Pool-2	10.112.129.4:8081 -> 10.112.0.101:8081 10.112.129.5:8081 -> 10.112.0.101:8081
HTTP/8000	All	Direct to Web Server (No ILB)	HTTP/8000	Web-1 (http://10.112.1.4:80)	10.112.129.4:8000 -> 10.112.1.4:80 10.112.129.5:8000 -> 10.112.1.4:80
HTTPS/443	All	Re-encrypt	HTTPS/443	SSL-Pool-3	10.112.129.4:443 -> 10.112.0.101:443 10.112.129.5:443 -> 10.112.0.101:443
HTTPS/443	/images/*	SSL offload	HTTP/8443	Image-Pool-2	10.112.129.4:8443 -> 10.112.0.101:8443 10.112.129.5:8443 -> 10.112.0.101:8443

Table 6 Example internal load-balancer rules

Front-end IP	Usage	Front-end port	Back-end pool	Pool members	Back-end port
10.112.0.101	Default	TCP/80	Web-Pool-1	10.112.1.4 10.112.1.5	TCP/80
10.112.0.101	URL path-based routing	TCP/8081	Image-Pool-2	10.112.1.10 10.112.1.11	TCP/80
10.112.0.101	Re-encrypt	TCP/443	SSL-Pool-3	10.112.1.20 10.112.1.21	TCP/443
10.112.0.101	SSL offload	TCP/8443	Image-Pool-2	10.112.1.10 10.112.1.11	TCP/8443

Figure 33 Application gateway flow with multiple listeners and an internal load balancer



Resiliency for Outbound and Internal Traffic

You implement resiliency for traffic that originates inside the VNet (outbound to the internet, east-west traffic between subnets, and backhaul traffic to on-premises networks) by using Azure user-defined routes and internal load balancers. Internal load balancers have a front end defined by one or more internal IP addresses and a back-end pool defined by the private interfaces of the firewalls.

To get internal traffic to the resilient firewalls, Azure user-defined routes direct traffic to the load balancer's internal IP address. To best support UDR, you should associate the load balancer's internal IP addresses with the subnet that contains the firewalls' private interfaces, and you should assign a static IP address.

All traffic that matches the user-defined route forwards to the load balancer. The Azure Standard Load Balancer with the HA ports feature supports a more effective deployment option than the Azure Basic Load Balancer. HA ports rules simplify the load-balancer configuration by supporting traffic on all TCP/UDP ports with a single rule.

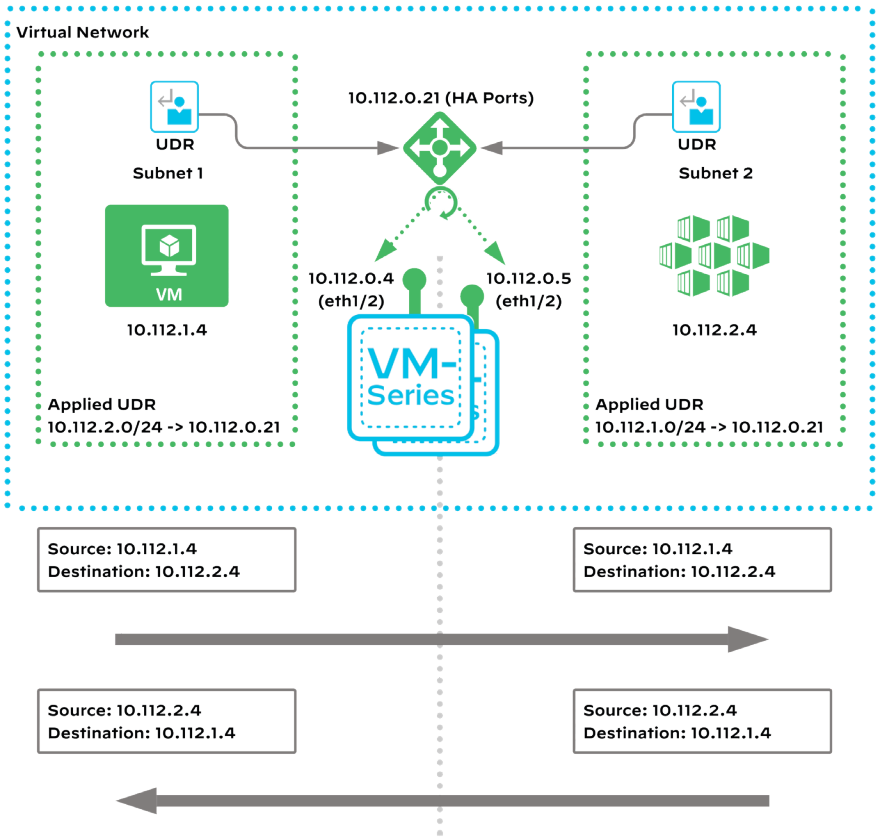
When you use a single load balancer to provide resiliency for outbound, east-west, and backhaul traffic, the load-balancer rules required to support one traffic profile might be too permissive for another. If this becomes an issue, you can use firewall security policies to limit applications, or you can divide the different traffic profiles across additional front-end IP addresses mapped to the back-end pool.

Internal load balancers do not translate the source or destination IP address of the traffic, and floating IP is not necessary. In most designs, the firewall does not need to translate the destination IP address. If the destination traffic is within the Azure VNet, then the load balancer maintains session state to ensure that return traffic to the resource enters through the firewall that processed the outgoing traffic. If the same front end and back-end pool of the load balancer see both directions of the traffic flow, the load balancer maintains the session state. For destinations outside the VNet, the firewall must translate the source IP address to the IP address of the egress interface. Without this source NAT, routing might send the return traffic to a different firewall.

Note

When east-west traffic between subnets traverses the firewall, both the incoming and outgoing traffic is on the same interface and assigned to the same zone. You must create a rule above the default intrazone security policy rule. You should then modify the default intrazone security policy to deny traffic, because it allows all traffic within a zone by default.

Figure 34 Standard SKU internal load-balancer traffic flow



PRISMA CLOUD INFRASTRUCTURE FOR AZURE

Prisma Cloud provides cloud infrastructure protection across the following areas:

- **Multi-cloud security**—Provide a consistent implementation of security best practices across your Azure and other public cloud environments. Prisma Cloud requires no agents, proxies, software, or hardware for deployment and integrates with a variety of threat intelligence feeds. Prisma Cloud includes pre-packaged policies to secure multiple public cloud environments.
- **Continuous compliance**—Maintain continuous compliance across CIS, NIST, PCI, FedRAMP, GDPR, ISO and SOC 2 standards by monitoring API-connected cloud resources across multiple cloud environments in real time. You can generate compliance documentation with one-click exportable, fully prepared reports.
- **Cloud forensics**—Go back in time to the moment a resource was first created and see chronologically every change and who made it. Prisma Cloud provides forensic investigation and auditing capabilities of potentially compromised resources across your Azure environment, as well as other public cloud environments. Historical information extends back to initial creation of each resource and the detailed change records includes who made each change.
- **DevOps and automation**—Enable secure DevOps without adding friction by setting architecture standards that provide prescribed policy guardrails. This methodology permits agile development teams to maintain their focus on developing and deploying apps to support business requirements.

Prisma Cloud connects to your cloud via APIs and aggregates raw configuration data, user activities, and network traffic to analyze and produce concise, actionable insights.

Azure integration requires that you create and register an Azure Application ID with a password-based key and reader permissions to your Azure subscription. Use this Application ID with the associated key and the Azure Active Directory ID and Azure Subscription ID to connect your cloud to Prisma Cloud. Additional permissions and configuration are required to ingest and analyze flow logs and collect other data. You perform these steps manually using Azure Resource Manager, but you can also use a Prisma Cloud onboarding script to automate the process.

Prisma Cloud performs a five-stage assessment of your cloud workloads. Contributions from each stage progressively improve the overall security posture for your organization:

- **Discovery**—Prisma Cloud continuously aggregates configuration, user activity, and network traffic data from disparate cloud APIs. It automatically discovers new workloads as soon as they are created.
- **Contextualization**—Prisma Cloud correlates the data and applies machine learning to understand the role and behavior of each cloud workload.
- **Enrichment**—The correlated data is further enriched with external data sources—such as vulnerability scanners, threat intelligence tools, and security information and event management (SIEM) services—to deliver critical insights.
- **Risk assessment**—Prisma Cloud scores each cloud workload for risk based on the severity of business risks, policy violations, and anomalous behavior. Aggregated risk scores enable you to benchmark and compare risk postures across different departments and across the entire environment.
- **Visualization**—Prisma Cloud enables you to visualize the entire cloud infrastructure environment with an interactive dependency map that moves beyond raw data to provide context.

Prisma Cloud Threat Defense

Prisma Cloud enables you to visualize your entire Azure environment, down to every component within the environment. The platform dynamically discovers cloud resources and applications by continuously correlating configuration, user activity, and network traffic data. Combining this deep understanding of the Azure environment with data from external sources, such as threat intelligence feeds and vulnerability scanners, enables Prisma Cloud to produce context around risks.

The Prisma Cloud platform comes with policies that adhere to industry-standard best practices. You can also create custom policies based on your organization's specific needs. The platform continuously monitors for violations of these policies by existing resources as well any new resources that you create. You can easily report on the compliance posture of your Azure environment to auditors.

Prisma Cloud automatically detects user and entity behavior within the Azure infrastructure and management plane. The platform establishes behavior baselines, and it flags any deviations. The platform computes *risk scores*—similar to credit scores—for every resource, based on the severity of business risks, violations, and anomalies. This quickly identifies the riskiest resources and enables you to quantify your overall security posture.

Prisma Cloud reduces investigation-time from weeks or months to seconds. You can use the platform's graph analytics to quickly pinpoint issues and perform upstream and downstream impact analysis. The platform provides you with a DVR-like capability to view time-serialized activity for any given resource. You can review the history of changes for a resource and better understand the root cause of an incident, past or present.

Prisma Cloud enables you to quickly respond to an issue based on contextual alerts. Alerts are triggered based on risk-scoring methodology and provide context on all risk factors associated with a resource. This makes it simple to prioritize the most important issues first. You can send alerts, orchestrate policy, or perform auto-remediation. You can send the alerts to Cortex XSOAR or third-party tools, such as Slack and Splunk, to remediate the issue.

Prisma Cloud provides the following visibility, detection, and response capabilities:

- **Host and container security**—Configuration monitoring and vulnerable image detection.
- **Network security**—Real-time network visibility and incident investigations. Suspicious/malicious traffic detection.
- **User and credential protection**—Account and access key compromise detection. Anomalous insider activity detection. Privileged activity monitoring.
- **Configurations and control plane security**—Compliance scanning. Storage, snapshots and image configuration monitoring. Security group and firewall configuration monitoring. IP address management configuration monitoring.

Continuous Monitoring

The dynamic nature of the cloud creates challenges for risk and compliance professionals tasked with measuring and demonstrating adherence to security and privacy controls.

View the collected continuous security-monitoring data collected in the Prisma Cloud portal and verify compliance of your resources to HIPAA, GPDR PCI, NIST, and SOC 2 standards. This capability eliminates the manual component of compliance assessment.

Prisma Cloud provides security and compliance teams with a view into the risks across all their cloud accounts, services, and regions by automating monitoring, inspection, and assessment of your cloud infrastructure services. With real-time visibility into the security posture of your environment, you can identify issues that do not comply with your organization's required controls and settings and send automated alerts.

Design Models

There are many ways to use the concepts discussed in the previous sections to achieve an architecture that secures application deployment in Azure. The design model and options in this section offer example architectures that secure inbound access to an application in Azure, the communication between private virtual machines and workloads, and the connection to your on-premises networks.

The primary difference between the design model options is resource allocation in Azure.

Consider which option best fits your needs and use it as a starting point for your design. The design model options in this reference design are the:

- **Transit VNet model**—In this model, you allocate the functions and resources across multiple VNets that are connected in a hub-and-spoke topology. The hub of the topology, or *transit VNet*, is the central point of connectivity for all inbound, outbound, east-west, and backhaul traffic. The spokes isolate workloads in their own VNets. This design model is highly scalable and highly resilient and is suitable for large-production deployments.

  The model separates inbound traffic flows onto a dedicated set of firewalls, allowing for greater scaling of inbound traffic loads. Outbound, east-west, and backhaul traffic flows through a common firewall set that is a shared resource. You deploy all firewalls in the transit VNet.

  This model consolidates resources that multiple workloads can share. This model also offers increased scale and operational resiliency and reduces the chances of high bandwidth use from the inbound traffic profile affecting other traffic profiles. We recommend this model for your production deployments.

- **Transit VNet model (Common Firewall Option)**—This option is identical to the Transit VNet model, except that in this option, all traffic flows through a single set of firewalls. The set of firewalls is a shared resource and has scale limitations compared to the Transit VNet model because additional traffic is flowing through a single set of firewalls and because performance degrades when traffic crosses virtual routers. This option keeps the number of firewalls low and is suitable for small deployments and proof-of-concepts. However, the technical integration complexity is high.

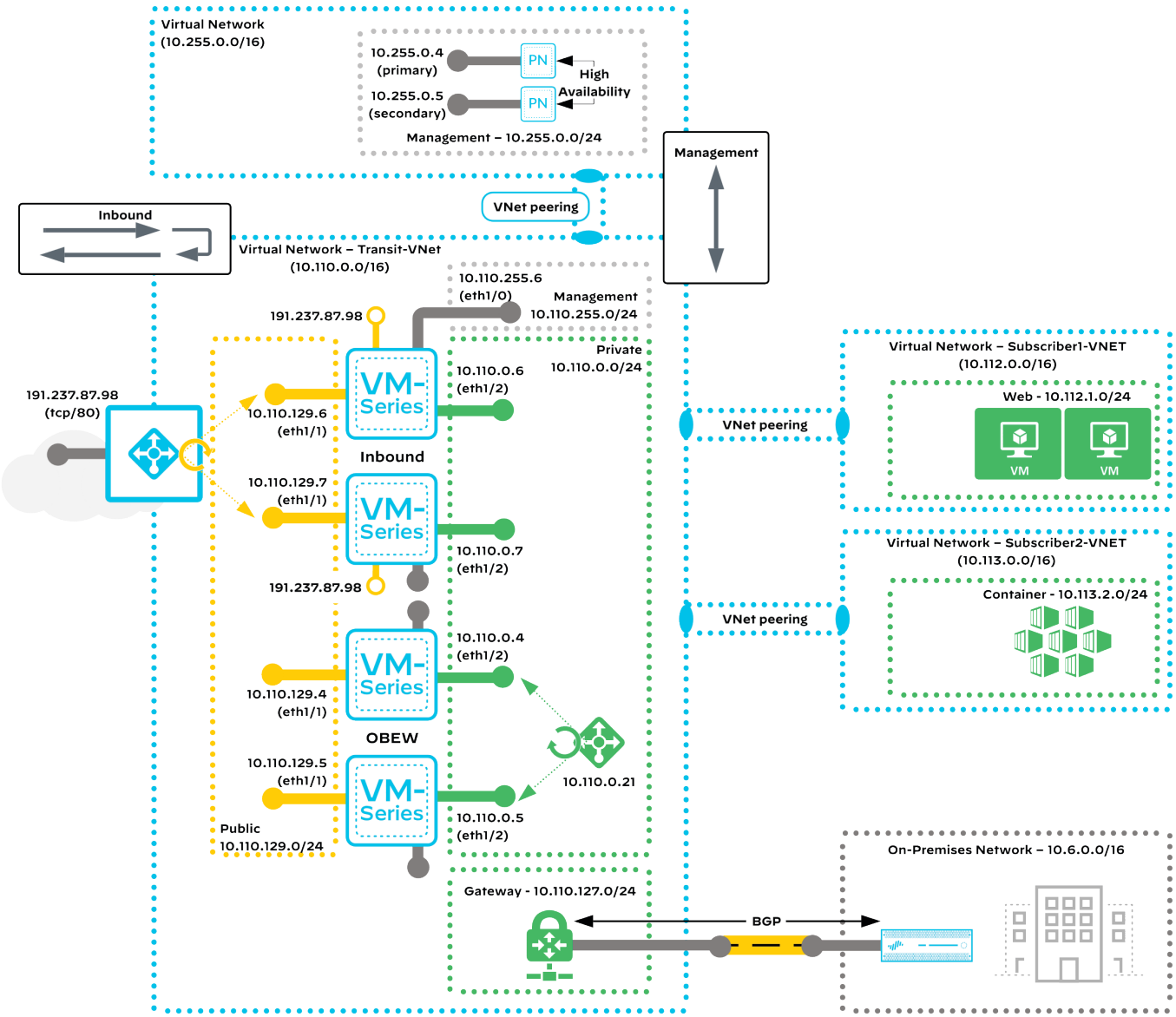
TRANSIT VNET MODEL

The Transit VNet model distributes the various traffic profiles across resources in multiple VNets and is highly scalable. This design model is composed of a single transit VNet and one or more subscriber VNets. The transit VNet does not typically contain any virtual compute resources and acts as a hub for interconnecting the subscriber (or *spoke*) VNets. A transit VNet provides inbound, outbound, and backhaul access for subscriber VNets as a shared service. The transit VNet also controls east-west traffic between subscribers.

A dedicated set of firewalls services inbound traffic from the internet. Another set of firewalls provides visibility and control of all other traffic profiles (outbound, east-west, and backhaul). The firewalls are members of availability sets that distribute their virtual machines across the Azure infrastructure in order to avoid downtime caused by infrastructure maintenance or failure. The separation of functions allows you to scale inbound traffic volume independently and simplifies the configuration of the inbound firewalls. You manage the firewalls in the transit VNet from Panorama deployed in a peered VNet.

The model requires that you establish a VNet peer connection between each subscriber VNet and the transit VNet. You cannot use overlapping IP address space within the set of peered VNets.

Figure 35 Transit VNet model



Inbound Traffic

There are two options for inbound traffic:

- **Azure public load balancer**—Choose this option if you require load balancing at Layer 4 only (TCP/UDP). Health probes in this design monitor the firewall resources and are not directly monitoring the health of the web server resources.
- **Azure application gateway**—Choose this option if you require load balancing at Layer 7 (application layer) for HTTP and HTTPS. Capabilities include URL path-based routing and SSL offload. Health probes in this design directly monitor the health of the web server resources.

Inbound Traffic with Azure Public Load Balancer

For inbound traffic, a public load balancer distributes traffic to the inbound firewalls. To simplify firewall configuration, the front-end public IP address is associated with a DNS name, and floating IP is enabled on the load-balancer rules. Load-balancer rules forward the required web service ports to the firewalls. Common ports required for inbound traffic include TCP/80 (HTTP) and TCP/443 (HTTPS). The public load balancer's health probes monitor firewall availability through the HTTPS or SSH services activated in the interface management profile. Only traffic sourced from the health probe IP address can connect to the HTTPS or SSH services.

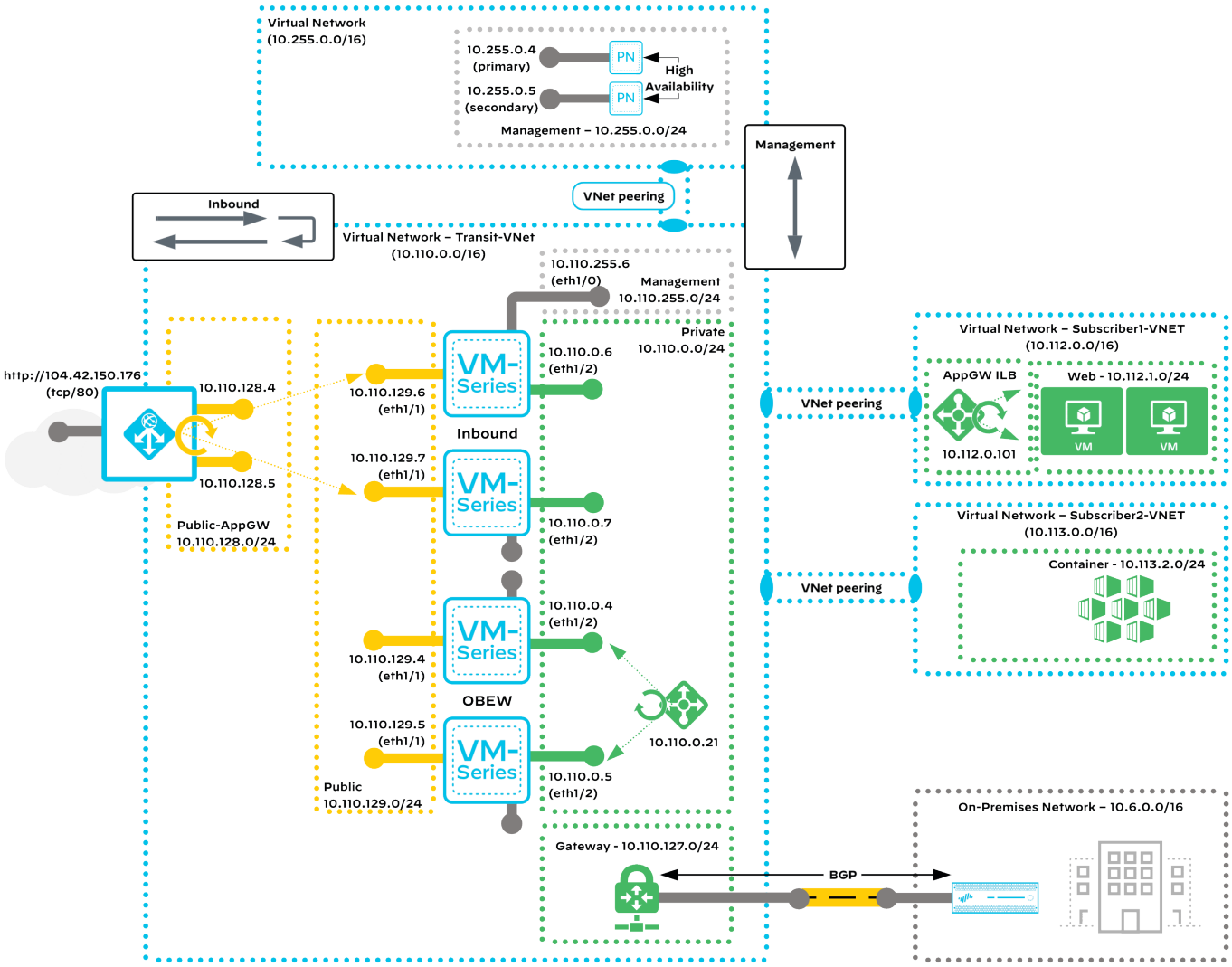
User-defined routes direct traffic from the subnet that contains the public interfaces destined for the other networks in the VNet to the next hop of *none*. This ensures that only inbound traffic forwarded through the public load balancer can communicate to private resources through the firewall.

The firewall applies both a destination and source NAT to inbound traffic. Destination NAT translates the FQDN address object associated with the load-balancer public DNS name to the virtual machine or load balancer on the subscriber network. The source NAT translates the source to be the IP address of the private interface of the firewall, ensuring return traffic flows symmetrically.

The firewall security policy allows appropriate application traffic to the resources in the private network while firewall security profiles prevent known malware and vulnerabilities from entering the network in traffic allowed by the security policy.

Inbound Traffic with Azure Application Gateway

Figure 36 Transit VNet model with application gateway



You create an additional public subnet for the application gateway. Specify a minimum of two application gateway instances in order to ensure that you distribute the instances across Azure update and fault domains.

For inbound traffic, an application gateway with a public front-end terminates incoming connections and initiates corresponding new connections to the configured HTTP/HTTPS back-ends. You assign unique TCP ports for all back ends. The application gateway sources all new connections from the private IP addresses of the application gateway instances and distributes the connections to the public interfaces of the inbound firewalls, which you have configured as the back-end pool targets for the application gateway. The application gateway's health probes monitor back-end availability on all specified HTTP/HTTPS ports.

You use application gateway destination NAT rules on the firewalls to map to back-end resources directly or through one or more internal load balancers.

This model supports any combination of the following methods:

- **Firewall destination port NAT to back-end resource**—No internal load balancer required, uses port-based NAT policy rules associated to the back-end resource. The firewall NAT policy contains all resource-mapping parameters.
- **Internal load balancer with one or more front-end IP addresses**—Uses port-based NAT policy rules associated to the load balancer's front-end IP addresses. You also configure port mapping on the load balancer. This option uses the load-balancer for resiliency and scaling of the back-end resources.
- **Multiple internal load balancers**—Uses port-based NAT policy rules associated to each load balancer's front-end IP addresses. This option supports more granular separation of both the load balancers and the back-end resources.

The firewall also applies a source NAT to inbound traffic. The source NAT translates the source to be the IP address of the private interface of the firewall, ensuring return traffic flows symmetrically.

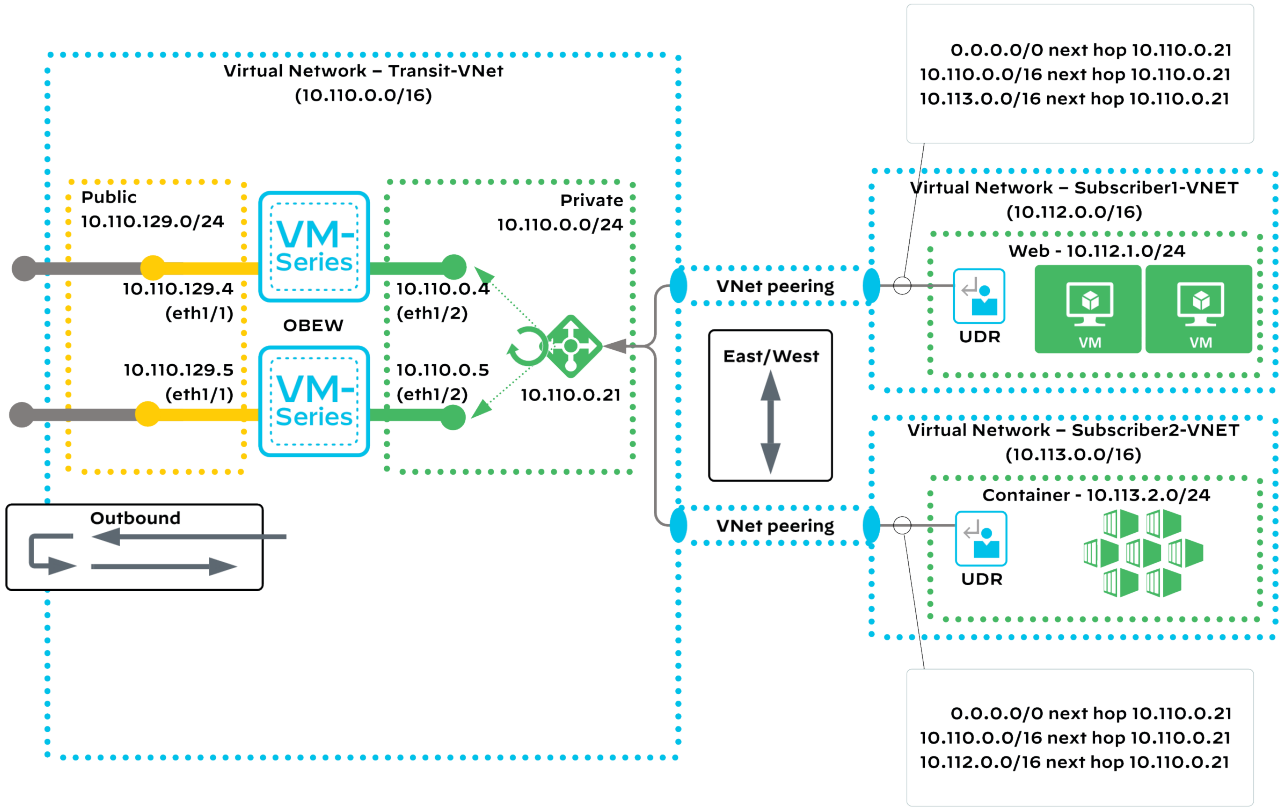
The firewall security policy allows HTTP/HTTPS application traffic from the application gateway instances to the resources in the private network, and firewall security profiles prevent known malware and vulnerabilities from entering the network in traffic allowed by the security policy. To support the use of HTTP/HTTPS back ends on ports other than 80/443, you should configure the service for security policy rules to include the specific service ports in use instead of *application-default*.

User-defined routes direct traffic from the subnets that contain the public interfaces destined for the other networks in the VNet to the next hop of *none*. This ensures that only inbound traffic forwarded through the application gateway can communicate to private resources through the firewall.

Outbound Traffic

User-defined routes on the private subnets in the subscriber VNets direct traffic to the load balancer's front-end IP address, which shares a subnet with the firewall private interfaces. The internal load balancer in the transit VNet distributes traffic to the set of firewalls. Load-balancer rules forward all TCP and UDP ports to the firewalls. The internal load balancer's health probes monitor firewall availability through the HTTPS service enabled in the interface management profile. Only traffic sourced from the health probe IP address can connect to the HTTPS service.

Figure 37 Outbound and east-west traffic



You define static routes for the health probe IP address and for the private network range out of the private interface because that is the interface that is receiving the health probes. Additionally, a static default route forwards traffic out the public interface.

The firewall applies source NAT to outbound traffic. The firewall translates the source address to its public interface. Azure automatically translates the interface IP address to the public IP address associated to the firewall's public interface when the traffic leaves the VNet.

The firewall security policy allows appropriate application traffic from the resources in the subscriber private networks to the internet. You should implement the security policy by using positive security policies (*whitelisting*). Security profiles prevent known malware and vulnerabilities from entering the network in return traffic allowed by the security policy. URL filtering, file blocking, and data filtering protect against data exfiltration.

East-West Traffic

The same internal load balancer that distributes outbound traffic to the firewalls also distributes the spoke-to-spoke *east-west traffic*, which is the traffic between private subnets within different subscribers. You apply user-defined routes for the private network subnets to the private subnets and direct traffic to the transit VNet's internal load balancer's front-end IP address. The existing load-balancer rules for outbound traffic also apply to east-west traffic and to all TCP/UDP ports.

The firewall should not translate the destination for traffic between private subnets. A positive control security policy should allow only appropriate application traffic between private resources and requires that you create corresponding security policy rules to permit specific traffic. You must then override the default intrazone security policy rule and modify it to deny traffic. Security profiles should also be enabled to prevent known malware and vulnerabilities from moving laterally in the private network through traffic allowed by the security policy.

Backhaul and Management Traffic

The same internal load balancer that distributes outbound and east-west traffic to the firewall also distributes traffic from on-premises networks. User-defined routes applied to the gateway subnet direct traffic that has a destination in the private network range to the internal load balancer. The existing load-balancer rules for outbound traffic also apply to backhaul and to all TCP and UDP ports.

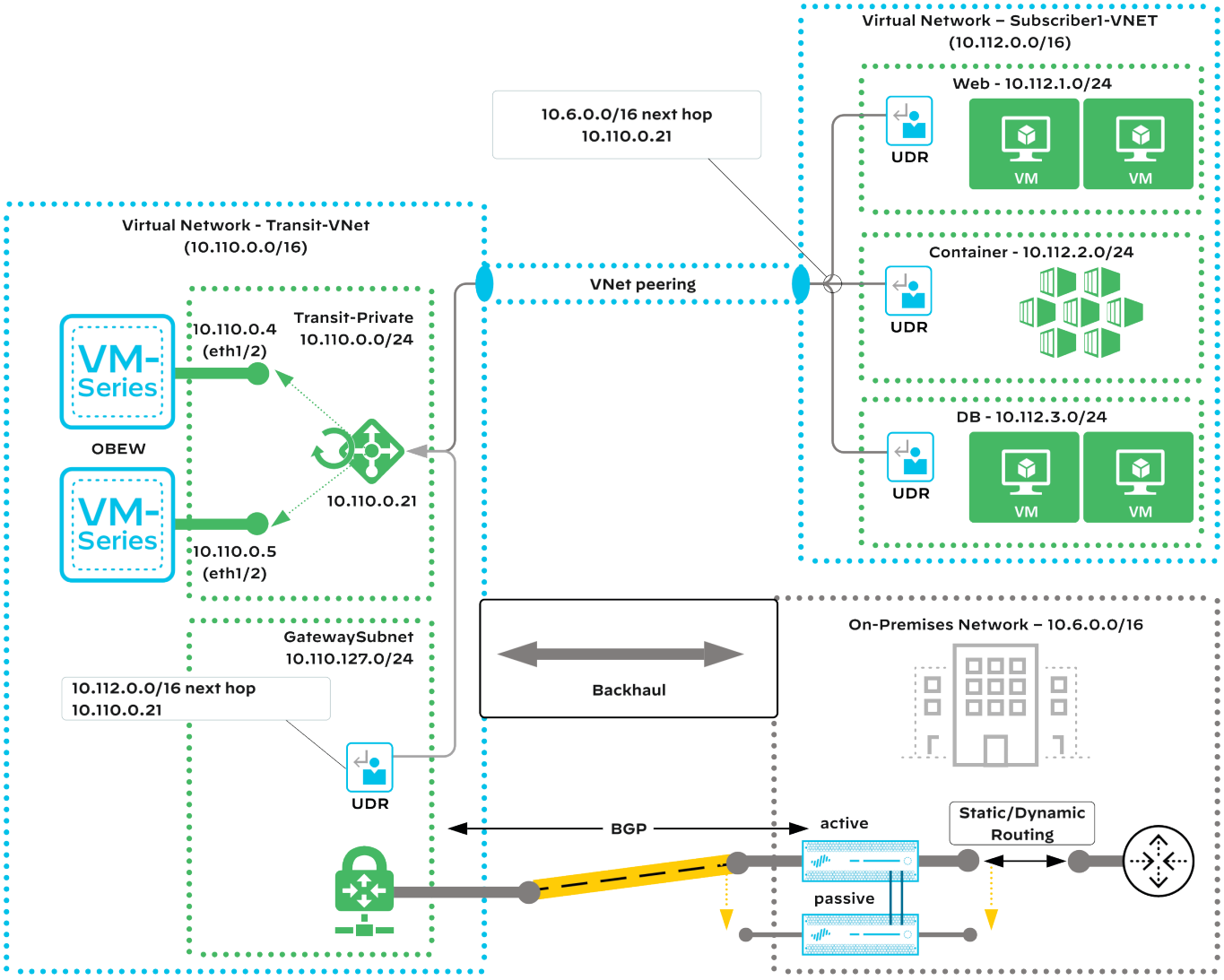
Traffic originating in private subnets and destined to on-premises networks follows the same path as outbound and east-west traffic. The only additional requirement is that you apply user-defined routes that forward on-premises network ranges to the internal load balancer's front end.

A VNG deployed in the transit VNet connects the Azure virtual networks to the on-premises networks. Enable BGP dynamic routing to the on-premises networks. Disable BGP route-propagation for public subnets. This eliminates the need for user-defined routes to discard the traffic to the on-premises networks.

Traffic from the on-premises networks communicates to the management subnets directly. This allows on-premises administrators to manage the firewalls even when a misconfiguration occurs in user-defined routes or load balancers.

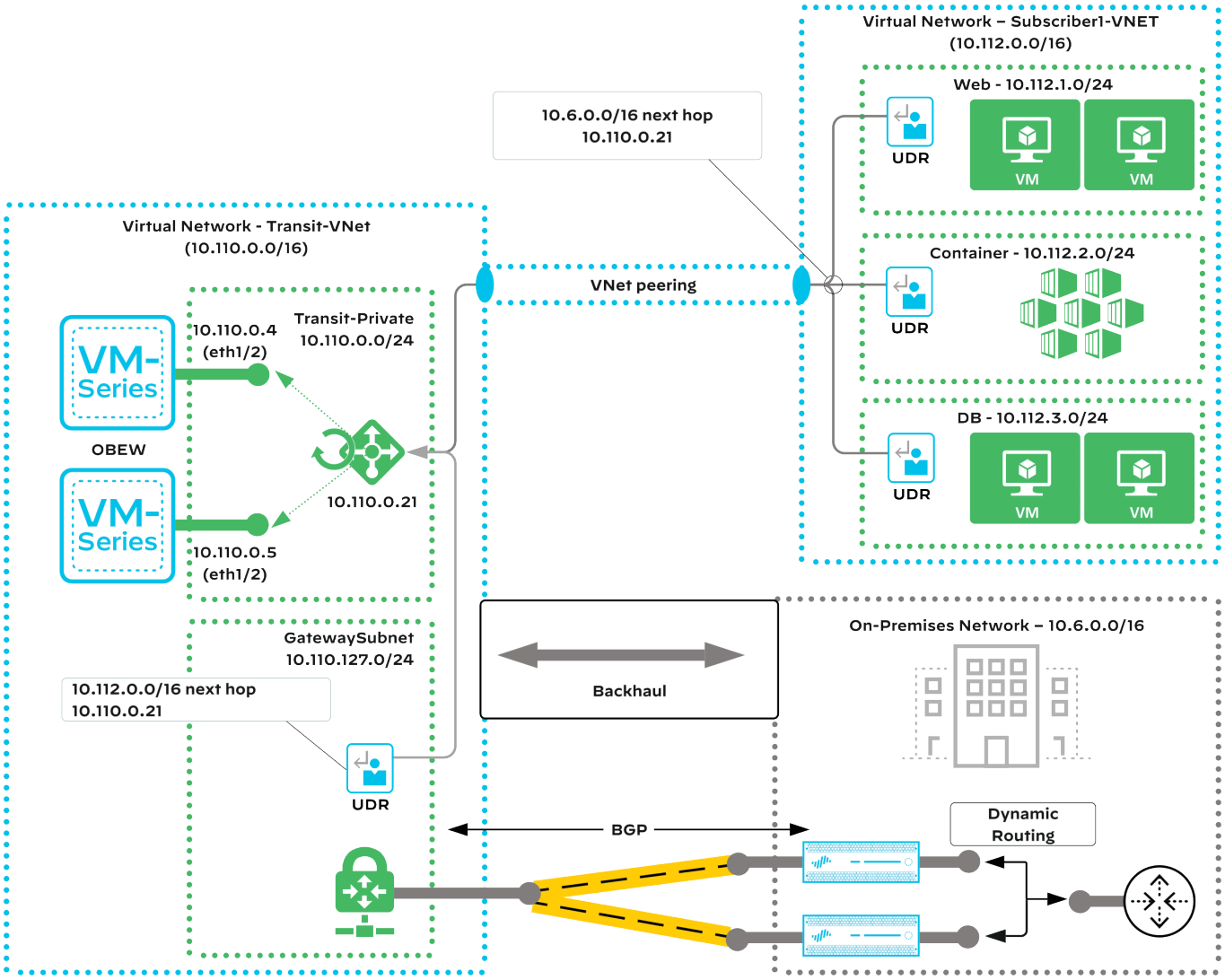
Deploy an active/passive firewall pair at your on-premises network and configure a single VPN tunnel from the VNG to the firewall pair. The high availability features of the firewall provide resiliency for this topology. If the active firewall fails, the passive firewall becomes active. In this configuration, only a single firewall is forwarding traffic.

Figure 38 Transit VNet backhaul with active/passive firewall pair



If you prefer to rely on dynamic routing protocols for resiliency, then deploy an active/active firewall pair at your on-premises networks and configure a VPN tunnel from the VNG to each firewall. You configure BGP to prefer one tunnel as the active path and the other tunnel as a backup path. If the firewall with the active path fails, BGP reroutes traffic to the backup path through the other active firewall. Traffic flows only over a single tunnel in both directions to ensure route symmetry.

Figure 39 Transit VNet backhaul with active/active firewall pair



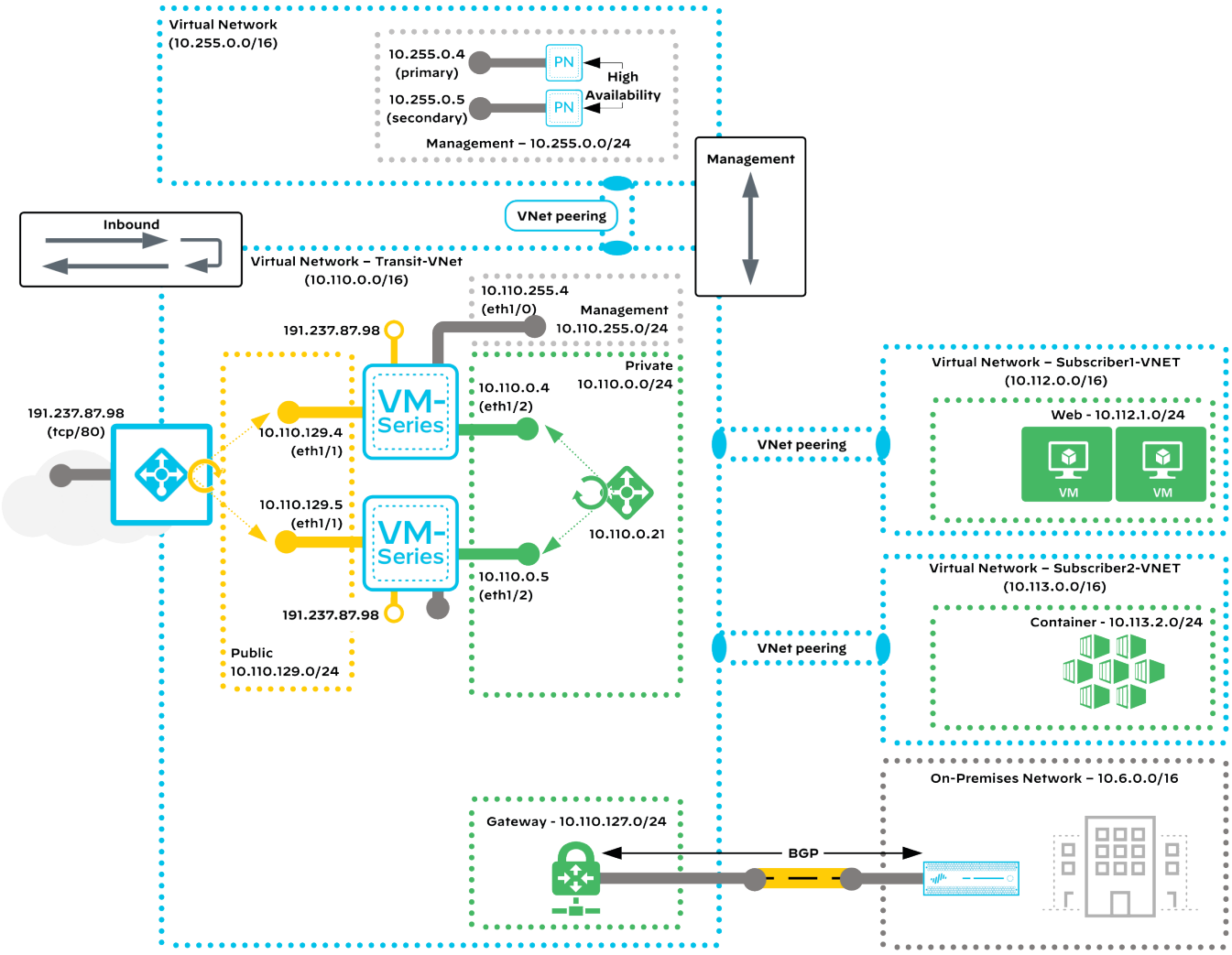
TRANSIT VNET MODEL—COMMON FIREWALL OPTION

In the common firewall option, a common set of firewalls provides visibility and control of all traffic profiles (inbound, outbound, east-west, backhaul). This option is essentially the same as the Transit VNet model, except as noted. This option is recommended for small-scale or proof-of-concept deployments.

Combining all traffic profiles onto a single set of firewalls reduces the total number of firewalls the design requires. However, this increases the complexity of the design and reduces the overall scalability.

The firewalls are members of an availability set that distributes their virtual machines across the Azure infrastructure in order to avoid downtime caused by infrastructure maintenance or failure. You establish a VNet peer connection between each subscriber VNet and the transit VNet. You cannot use overlapping IP address space within the set of peered VNets.

Figure 40 Transit VNet model—common firewall option



Differences from the Transit VNet model:

- **Inbound**—The public load balancer or application gateway forwards traffic to the same set of firewalls used for outbound, east-west, and backhaul traffic. This design requires dedicated virtual routers on the firewalls because both the public interface and private interface receive health probes.
- **Outbound**—There are no differences between this option and the Transit VNet model.
- **East-west**—There are no differences between this option and the Transit VNet model.
- **Backhaul**—There are no differences between this option and the Transit VNet model.

Inbound Traffic

There are two options for inbound traffic:

- **Azure public load balancer**—Choose this option if you require load balancing only at Layer 4 (TCP/UDP). Health probes in this design monitor the firewall resources and are not directly monitoring the health of the web server resources.
- **Azure application gateway**—Choose this option if you require load balancing at Layer 7 (application layer) for HTTP and HTTPS. Capabilities include URL path-based routing and SSL offload. Health probes in this design directly monitor the health of the web server resources.

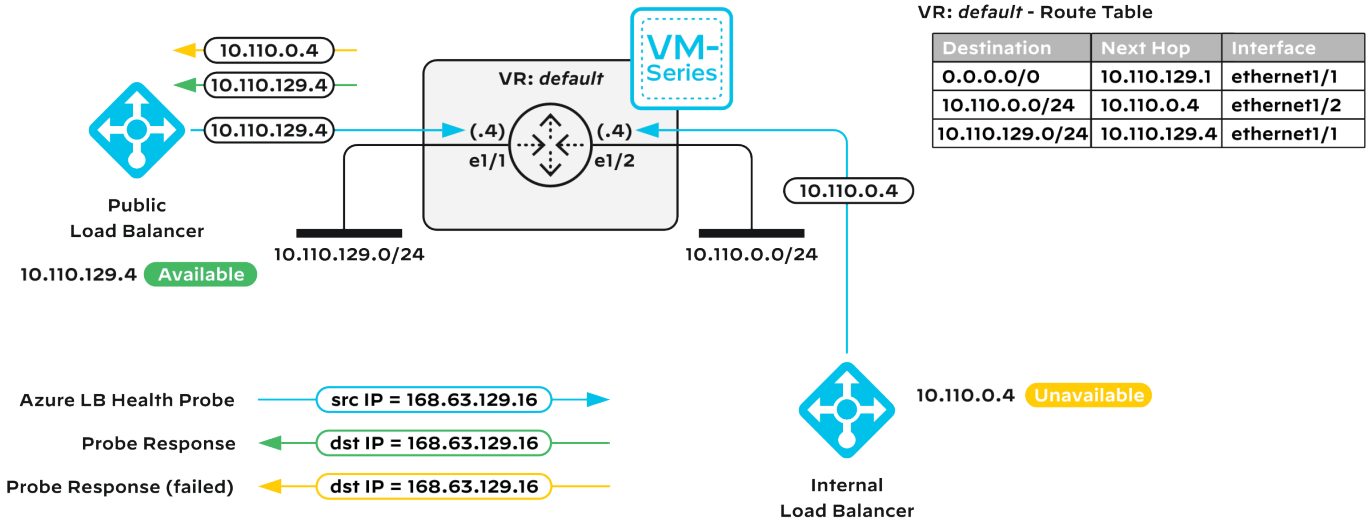
Inbound Traffic with Azure Public Load Balancer

For inbound traffic, a public load balancer distributes traffic to the firewalls. To simplify firewall configuration, the front-end public IP address is associated with a DNS name, and floating IP is enabled on the load-balancer rules. Load-balancer rules forward the required web service ports to the firewalls. Common ports required for inbound traffic include TCP/80 (HTTP) and TCP/443 (HTTPS). The public load balancer's health probes monitor firewall availability through the HTTPS or SSH services activated in the interface management profile. Only traffic sourced from the health probe IP address can connect to the HTTPS or SSH services.

User-defined routes direct traffic from the subnet that contains the public interfaces destined for the other networks in the VNet to the next hop of *none*. This ensures that only inbound traffic forwarded through the public load balancer can communicate to private resources through the firewall.

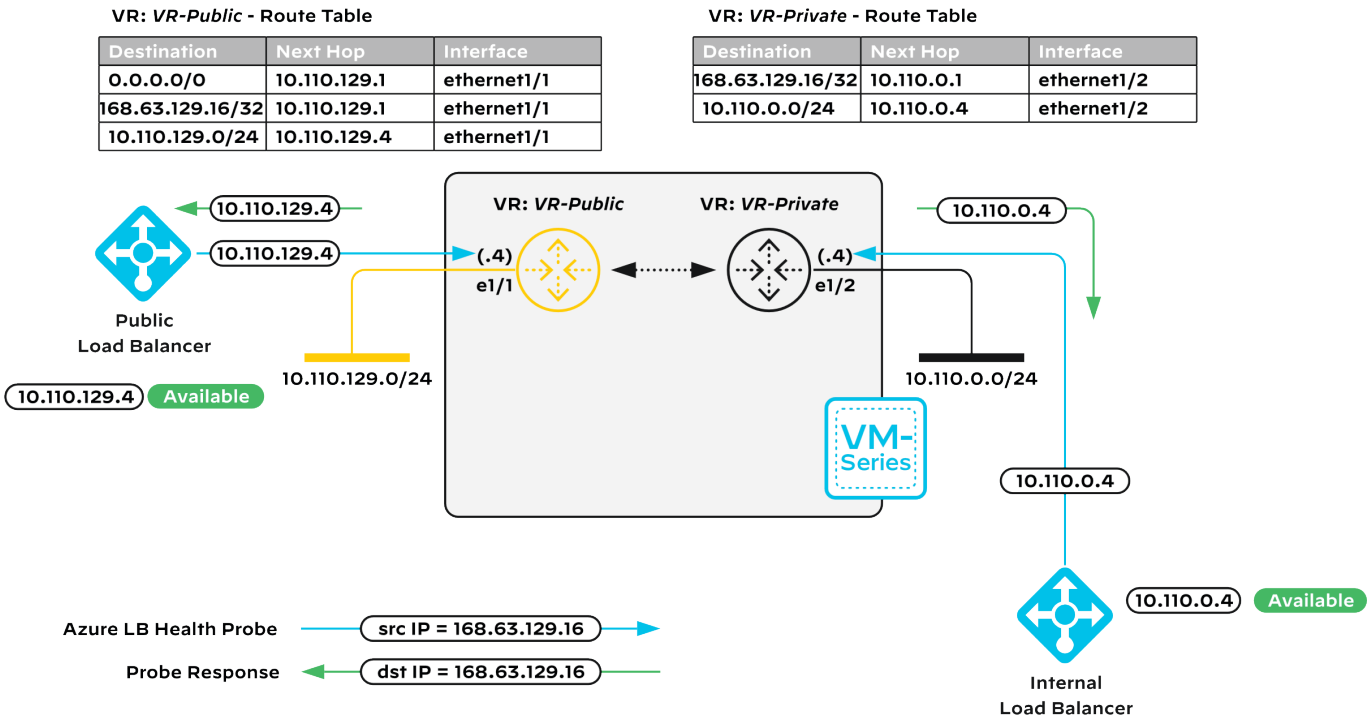
As discussed earlier in this guide, the Azure load-balancer always sources health probes from an IP address of 168.63.129.16. This implementation is incompatible with multi-homed devices such as routers or firewalls. Health probes succeed on one interface only and fail on the additional interface, because there is only one active IP route table entry for 168.63.129.16/32. You must configure multiple virtual routers to support health probes from the Azure load balancer on multiple interfaces.

Figure 41 Health probe failures with a single virtual router



The public interface uses a dedicated virtual router. Static routes define a default route out the public interface as well as a route to private networks through the virtual router dedicated to the private interface. Dedicated virtual routers allow the firewall interface that received the health probe to source responses.

Figure 42 Health probes with multiple virtual routers

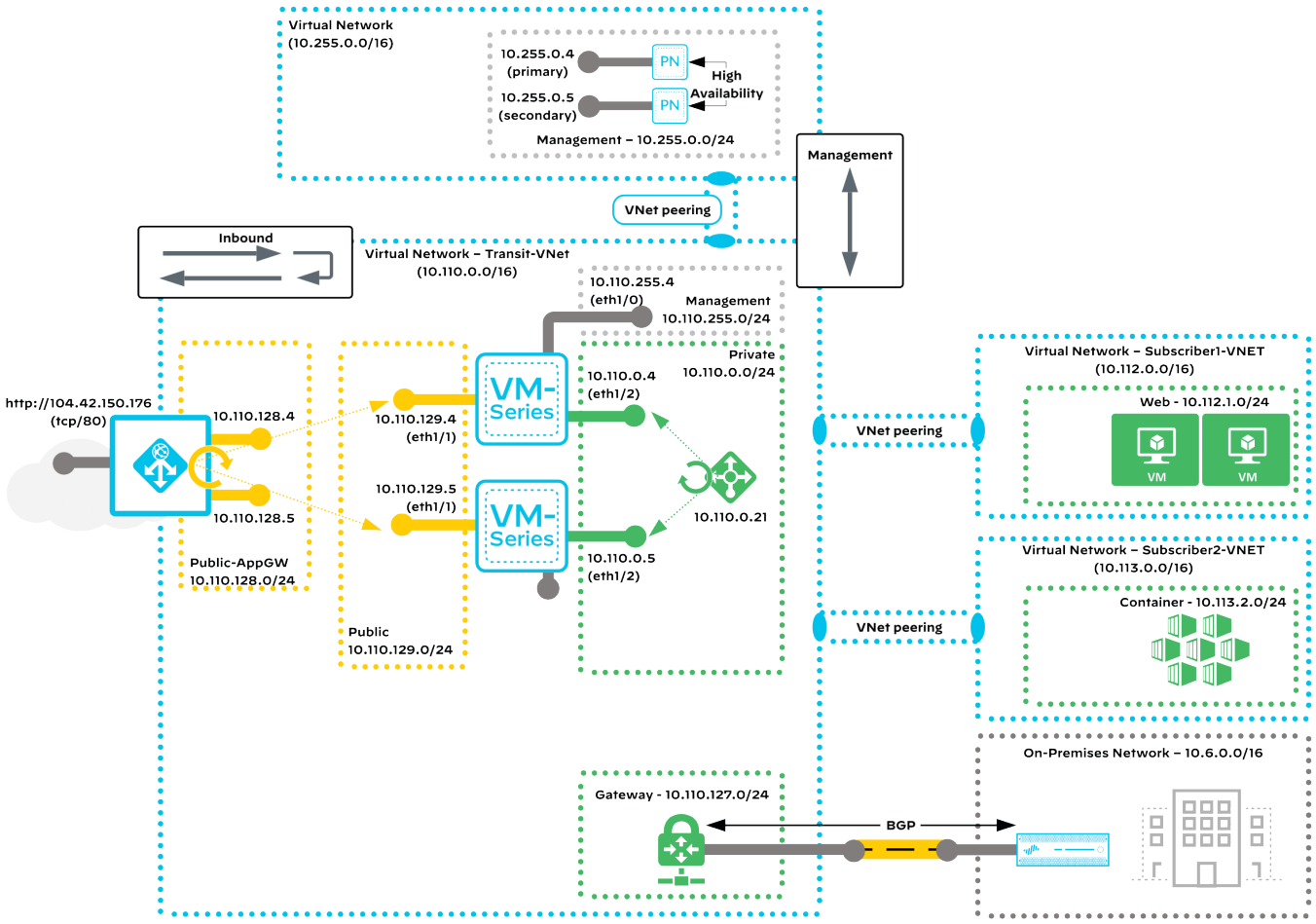


The firewall applies both a destination and source NAT to inbound traffic. Destination NAT translates the FQDN address object associated with the load-balancer public DNS name to the virtual machine or load balancer on the subscriber network. The source NAT translates the source to be the IP address of the private interface of the firewall, ensuring return traffic flows symmetrically.

The firewall security policy allows appropriate application traffic to the resources in the private network while firewall security profiles prevent known malware and vulnerabilities from entering the network in traffic allowed by the security policy.

Inbound Traffic with Azure Application Gateway

Figure 43 Common firewall option with application gateway



You create an additional public subnet for the application gateway. Specify a minimum of two application gateway instances in order to ensure that you distribute the instances across Azure update and fault domains.

For inbound traffic, an application gateway with a public front end terminates incoming connections and initiates corresponding new connections to the configured HTTP/HTTPS back ends. You assign unique TCP ports for all back ends. The application gateway sources all new connections from the private IP addresses of the application gateway instances and distributes the connections to the public interfaces of the firewalls, which you have configured as the back-end pool targets for the application gateway. The application gateway's health probes monitor back-end availability on all specified HTTP/HTTPS ports.

You use application gateway destination NAT rules on the firewalls to map to back-end resources directly or through one or more internal load balancers.

This model supports any combination of the following methods:

- **Firewall destination port NAT to back-end resource**—No internal load balancer required, uses port-based NAT policy rules associated to the back-end resource. The firewall NAT policy contains all resource-mapping parameters.
- **Internal load balancer with one or more front-end IP addresses**—Uses port-based NAT policy rules associated to the load balancer's front-end IP addresses. You also configure port mapping on the load balancer. This option uses the load-balancer for resiliency and scaling of the back-end resources.
- **Multiple internal load balancers**—Uses port-based NAT policy rules associated to each load balancer's front-end IP addresses. This option supports more granular separation of both the load balancers and the back-end resources.

The firewall also applies a source NAT to inbound traffic. The source NAT translates the source to be the IP address of the private interface of the firewall, ensuring return traffic flows symmetrically.

The firewall security policy allows HTTP/HTTPS application traffic from the application gateway instances to the resources in the private network, and firewall security profiles prevent known malware and vulnerabilities from entering the network in traffic allowed by the security policy. To support the use of HTTP/HTTPS back ends on ports other than 80/443, you should configure service for security policy rules to include the specific service ports in use instead of *application-default*.

User-defined routes direct traffic from the subnets that contain the public interfaces destined for the other networks in the VNet to the next hop of *none*. This ensures that only inbound traffic forwarded through the application gateway can communicate to private resources through the firewall.

Outbound Traffic

User-defined routes on the private subnets in the subscriber VNets direct traffic to the internal load balancer's front-end IP address, which shares a subnet with the firewall private interfaces. The internal load balancer in the transit VNet distributes traffic to the set of firewalls. Load-balancer rules forward all TCP and UDP ports to the firewalls. The internal load balancer's health probes monitor firewall availability through the HTTPS service enabled in the interface management profile. Only traffic sourced from the health probe IP address can connect to the HTTPS service.

You define static routes for the health probe IP address and for the private network range out of the private interface because that is the interface that is receiving the health probes. Additionally, a static default route forwards traffic out the public interface.

The firewall applies source NAT to outbound traffic. The firewall translates the source address to its public interface. Azure automatically translates the interface IP address to the public IP address associated to the firewall's public interface when the traffic leaves the VNet.

The firewall security policy allows appropriate application traffic from the resources in the subscriber private networks to the internet. You should implement the security policy by using positive security policies (*whitelisting*). Security profiles prevent known malware and vulnerabilities from entering the network in return traffic allowed by the security policy. URL filtering, file blocking, and data filtering protect against data exfiltration.

East-West Traffic

The same internal load balancer that distributes outbound traffic to the firewalls also distributes the spoke-to-spoke *east-west traffic*, which is the traffic between private subnets within different subscribers. You apply user-defined routes for the private network subnets to the private subnets and direct traffic to the transit VNet's internal load balancer's front-end IP address. The existing load-balancer rules for outbound traffic also apply to east-west traffic and to all TCP/UDP ports.

The firewall should not translate the destination for traffic between private subnets. A positive control security policy should allow only appropriate application traffic between private resources and requires that you create corresponding security policy rules to permit specific traffic. You must then override the default intrazone security policy rule and modify it to deny traffic. Security profiles should also be enabled to prevent known malware and vulnerabilities from moving laterally in the private network through traffic allowed by the security policy.

Backhaul and Management Traffic

You establish a VNet peer connection between each subscriber VNet and the transit VNet. You cannot use overlapping IP address space within the set of peered VNets.

The same internal load balancer that distributes outbound and east-west traffic to the firewall also distributes traffic from on-premises networks. User-defined routes applied to the gateway subnet direct traffic that has a destination in the private network range to the internal load balancer. The existing load-balancer rules for outbound traffic also apply to backhaul and to all TCP and UDP ports.

Traffic originating in private subnets and destined to on-premises networks follows the same path as outbound and east-west traffic. The only additional requirement is that you apply user-defined routes that forward on-premises network ranges to the internal load balancer's front end.

A VNG deployed in the transit VNet connects the Azure virtual networks to the on-premises networks. Enable BGP dynamic routing to the on-premises networks. Disable BGP route-propagation for public subnets. This eliminates the need for user-defined routes to discard the traffic to the on-premises networks.

Traffic from the on-premises networks communicates to the management subnets directly. This allows on-premises administrators to manage the firewalls even when a misconfiguration occurs in user-defined routes or load balancers.

Deploy an active/passive firewall pair at your on-premises network and configure a single VPN tunnel from the VNG to the firewall pair. The high availability features of the firewall provide resiliency for this topology. If the active firewall fails, the passive firewall becomes active. In this configuration, only a single firewall is forwarding traffic.

If you prefer to rely on dynamic routing protocols for resiliency, then deploy an active/active firewall pair at your on-premises networks and configure a VPN tunnel from the VNG to each firewall. You configure BGP to prefer one tunnel as the active path and the other tunnel as a backup path. If the firewall with the active path fails, BGP reroutes traffic to the backup path through the other active firewall. Traffic flows over only a single tunnel in both directions to ensure route symmetry.

Summary

The shared security responsibility model states that protecting your Microsoft Azure applications and data is your responsibility. VM-Series firewalls on Azure enable you to protect your applications and data from known and unknown threats as vigilantly as you protect on-premises applications and data.

VM-Series firewalls on Azure complement default security features with complete application visibility and control, resulting in a reduction in the threat footprint and the ability to prevent known and unknown threats.

You can use the feedback form to send comments about this guide.

HEADQUARTERS

Palo Alto Networks
3000 Tannery Way
Santa Clara, CA 95054, USA
http://www.paloaltonetworks.com

Phone: +1 (408) 753-4000
Sales: +1 (866) 320-4788
Fax: +1 (408) 753-4001
info@paloaltonetworks.com

B-000115P-1-20b