Azure Networking

Adam Raffe
Cloud Solution Architect
@adamraffe

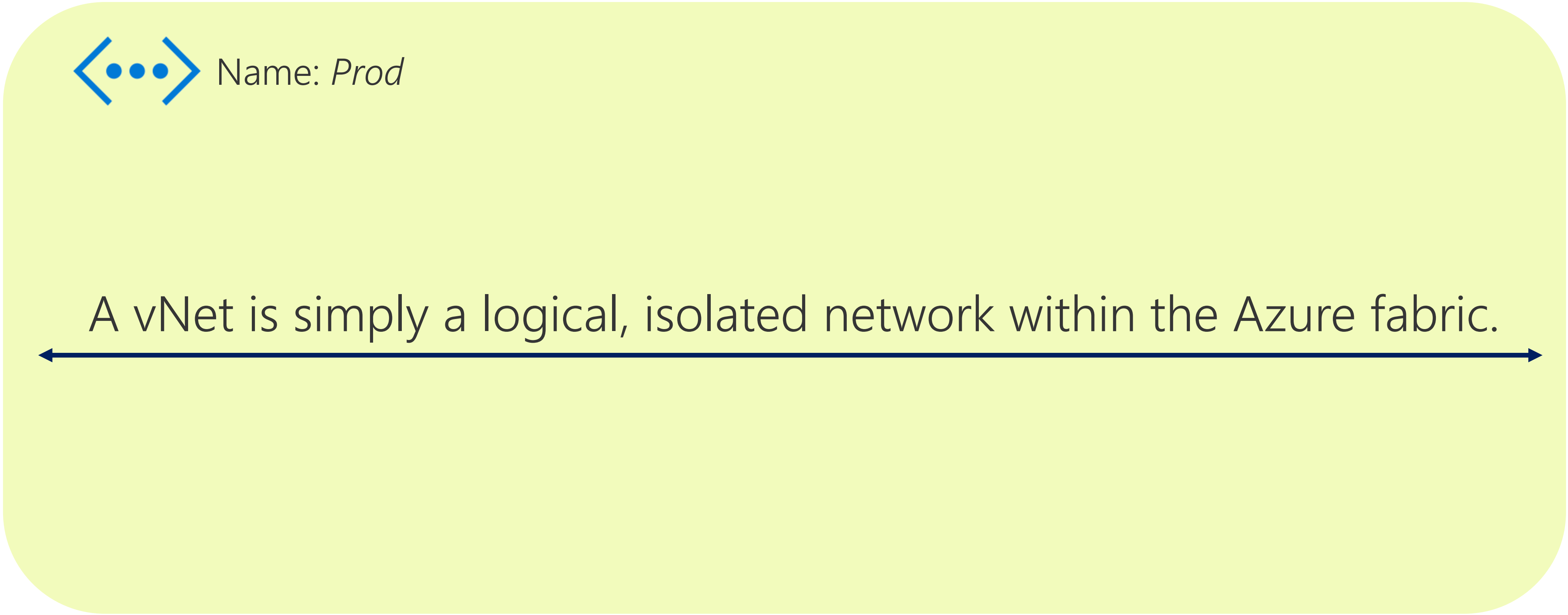
Microsoft

Virtual Networks

Azure networking is built around the concept of *Virtual Networks* (vNets).

Name: *Prod*

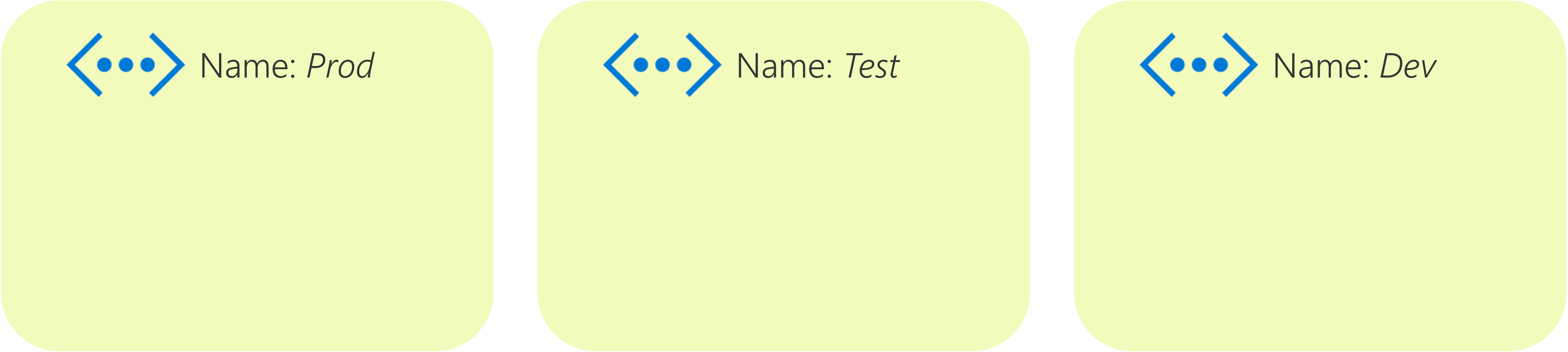
Azure networking is built around the concept of *Virtual Networks* (vNets).



Name: *Prod*

A vNet is simply a logical, isolated network within the Azure fabric.

vNets are (by default) completely isolated from each other.

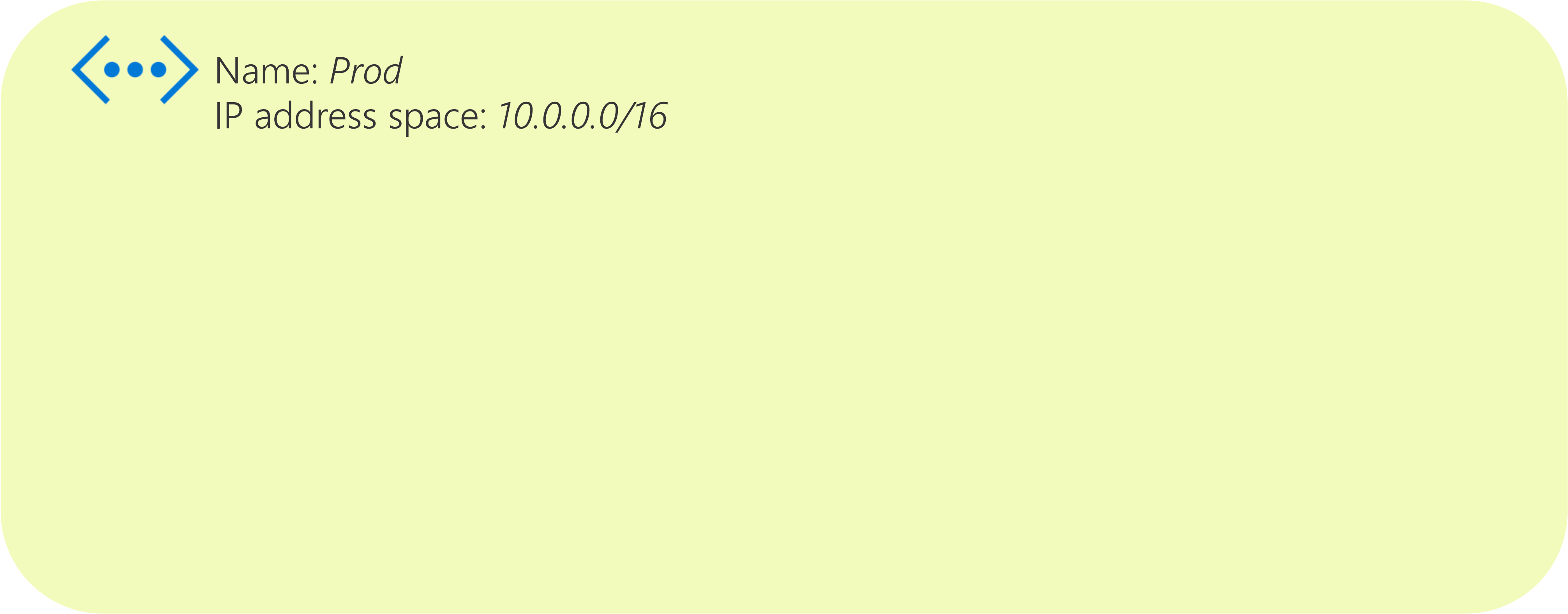


Name: *Prod*

Name: *Test*

Name: *Dev*

A vNet must be configured with at least one IP address space.



Name: *Prod*

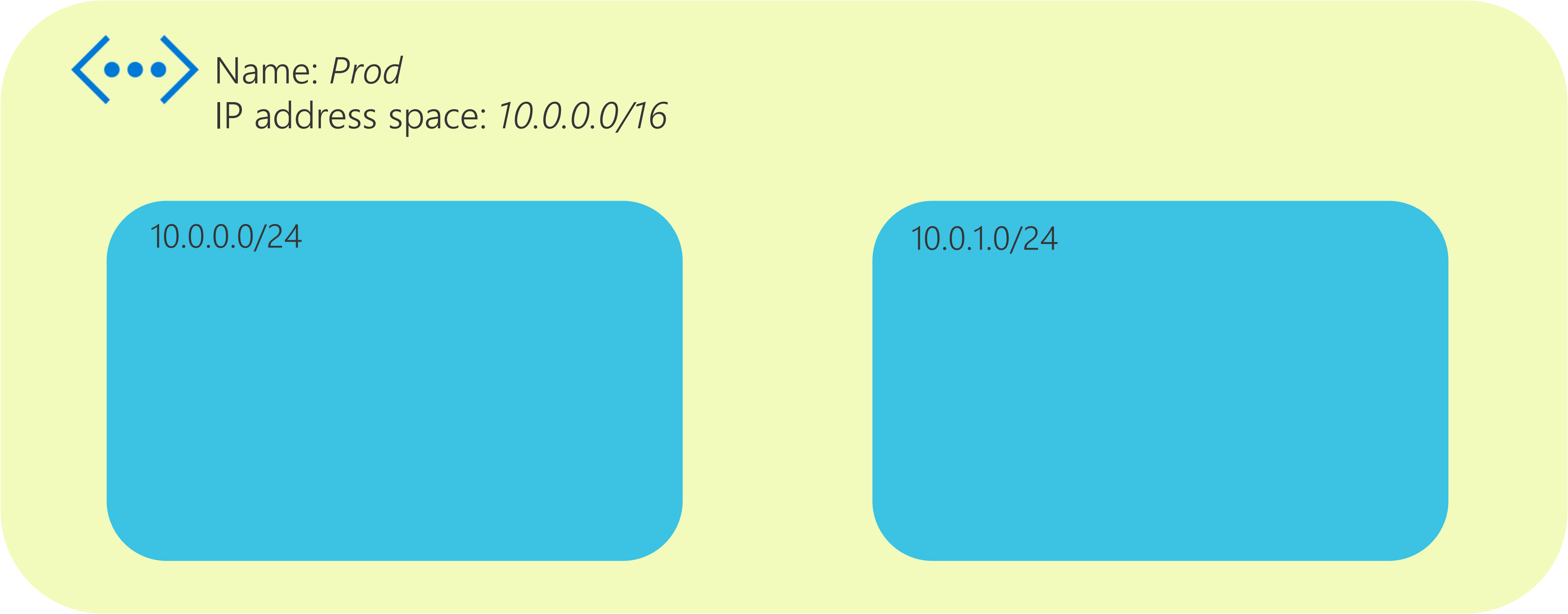
IP address space: *10.0.0.0/16*

A vNet must be configured with at least one IP address space.

Name: *Prod*
IP address space: *10.0.0.0/16, 192.168.1.0/24*

It is possible for a vNet to have more than one address space assigned to it.

We then configure one or more *subnets* inside the vNet.

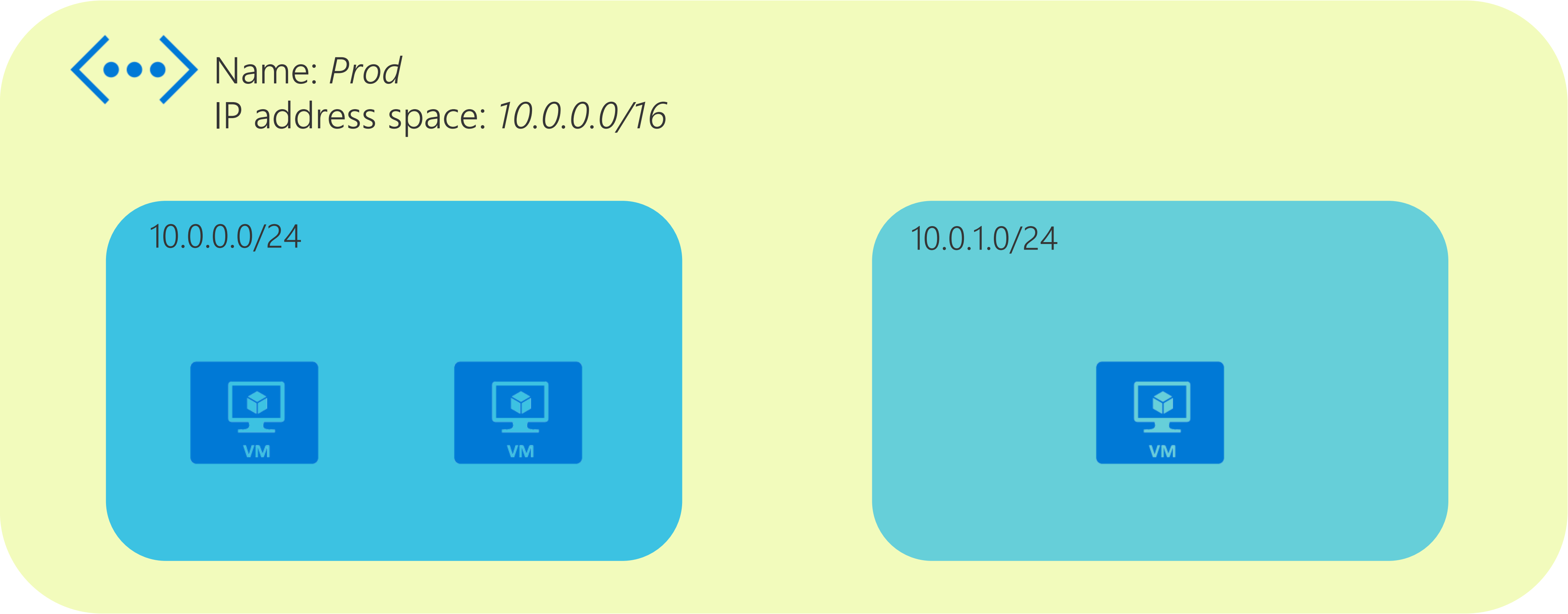


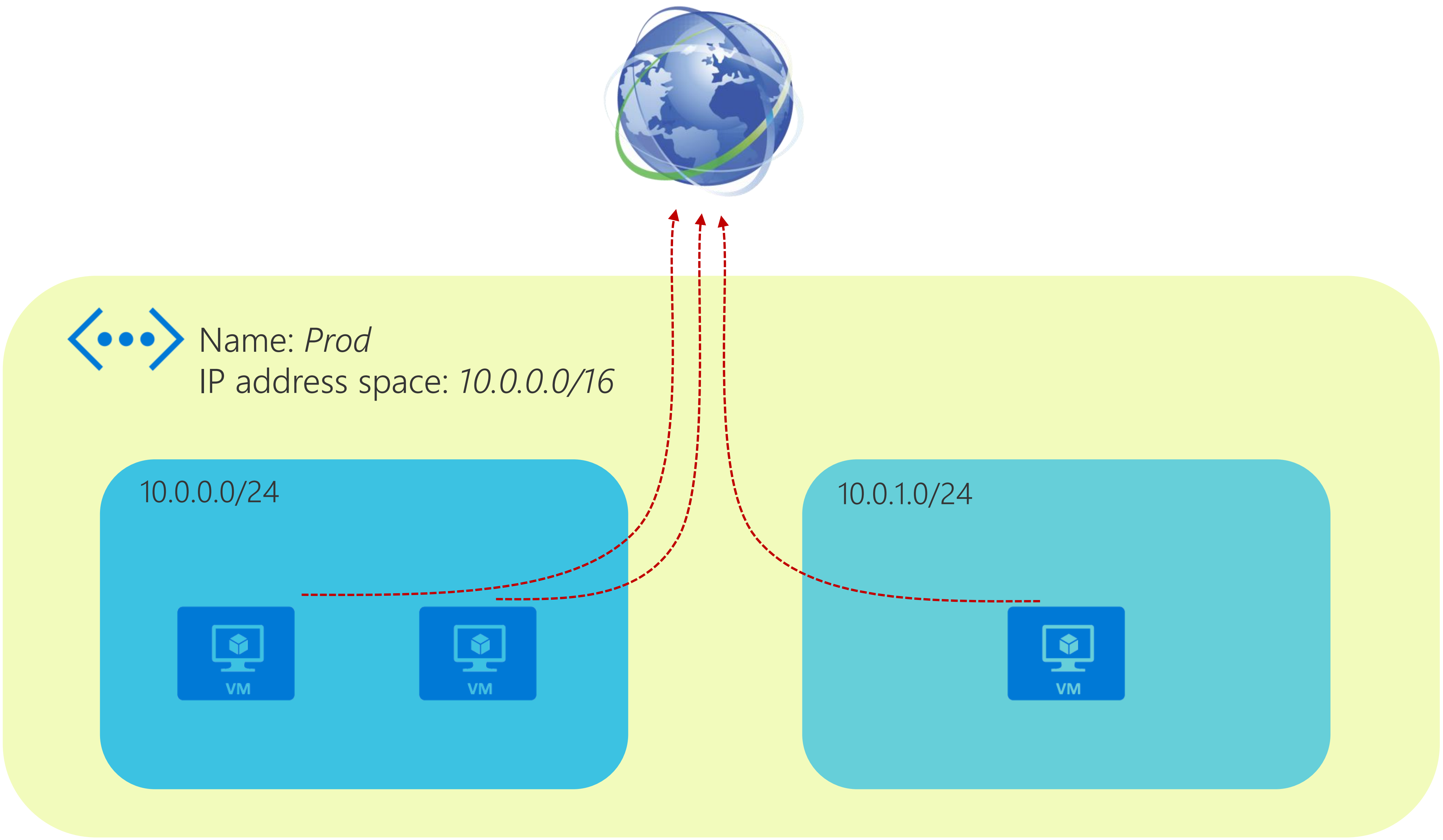
Name: *Prod*
IP address space: *10.0.0.0/16*

10.0.0.0/24

10.0.1.0/24

Virtual machines are deployed into a subnet.





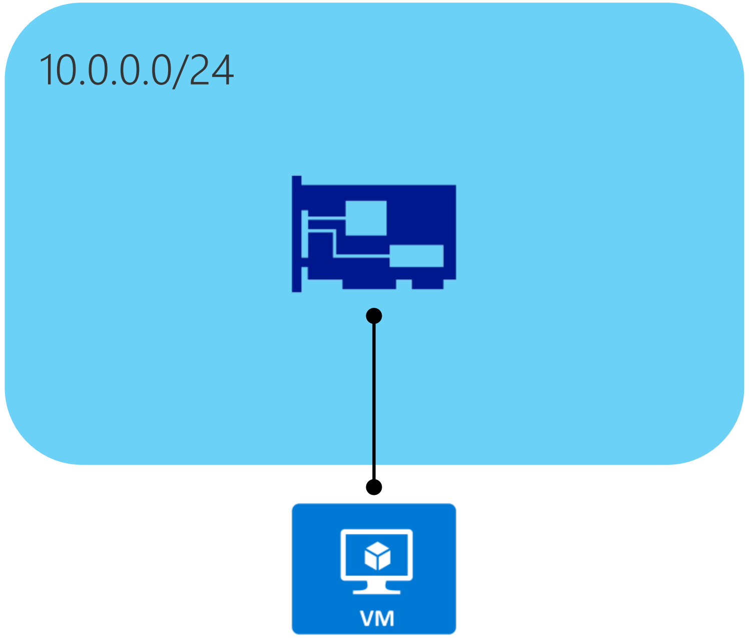
Name: *Prod*
IP address space: *10.0.0.0/16*

10.0.0.0/24

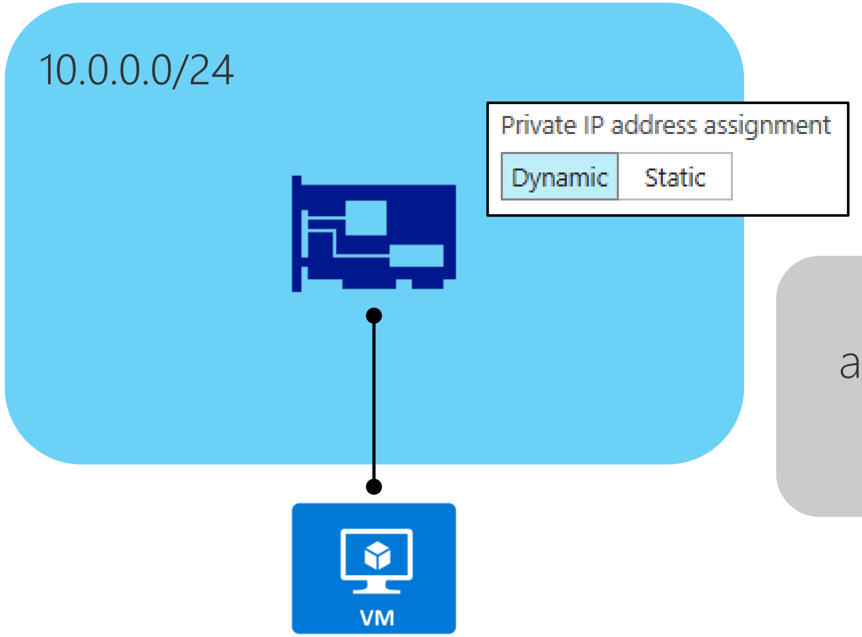
10.0.1.0/24

VMs have outbound Internet connectivity *by default.*

Technically, it is the *Network Interface* that connects a VM to a subnet.



Private IP addresses (i.e. taken from the subnet range) can be allocated either *dynamically* or *statically*.

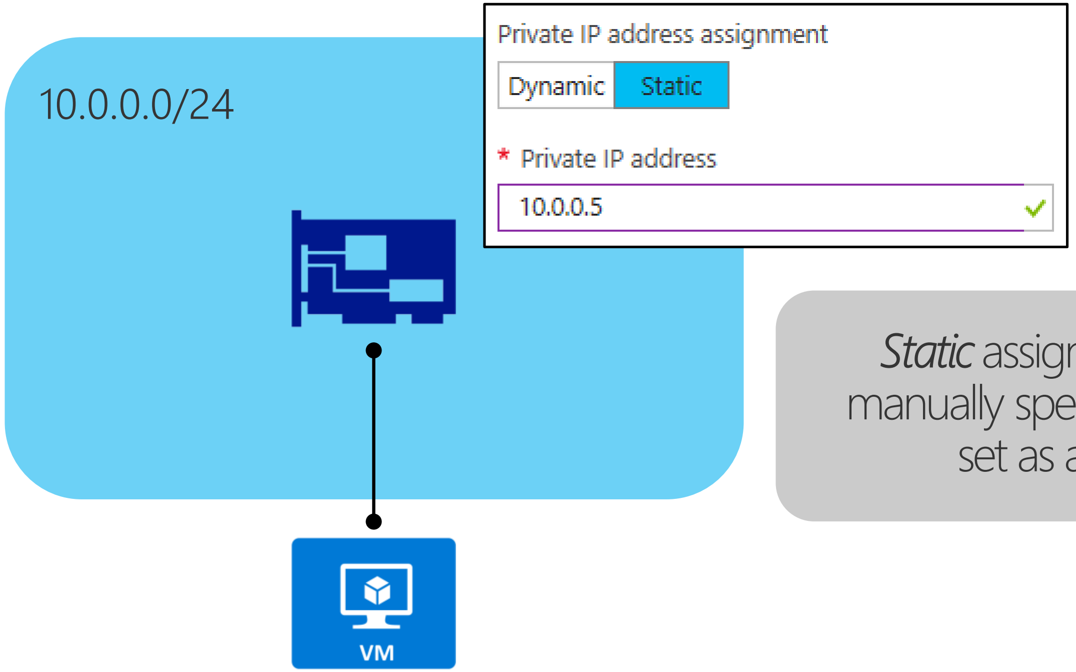


Private IP address assignment

[✓ Dynamic] Static

With *dynamic* assignment, addresses are automatically allocated by the DHCP server when the VM starts and **may not remain the same** when the VM reboots.

Private IP addresses (i.e. taken from the subnet range) can be allocated either *dynamically* or *statically*.



Private IP address assignment

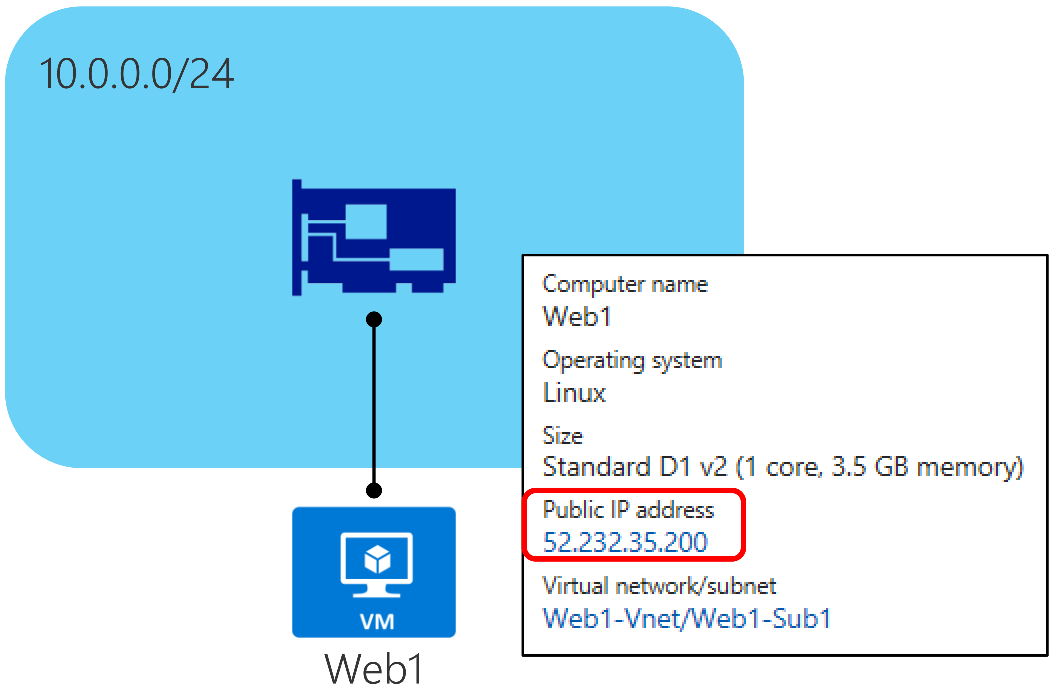
Dynamic [✓ Static]

* Private IP address

10.0.0.5 ✓

Static assignment means that you can manually specify the address and it will be set as a reservation by DHCP.

A VM can also have a *Public IP* assigned to it – by doing so, the VM will be accessible from the Internet.



Computer name
Web1

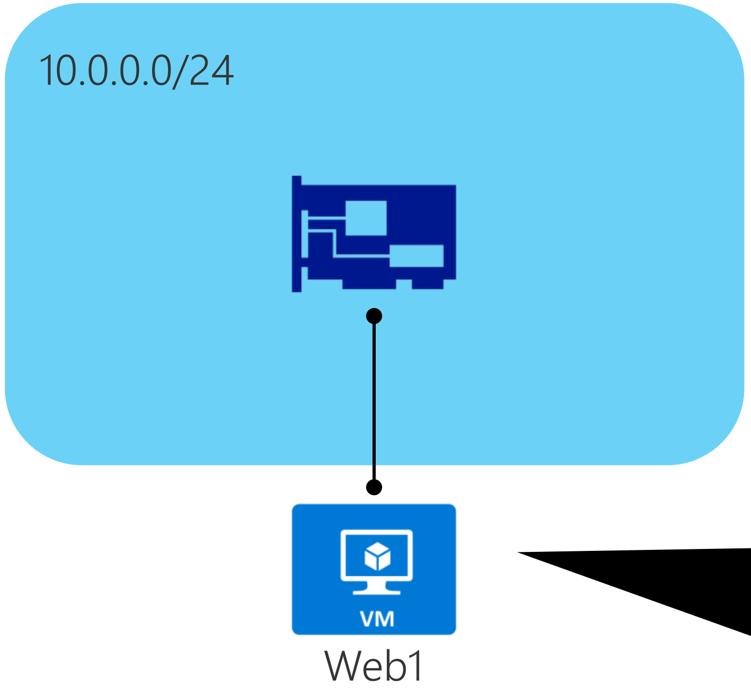
Operating system
Linux

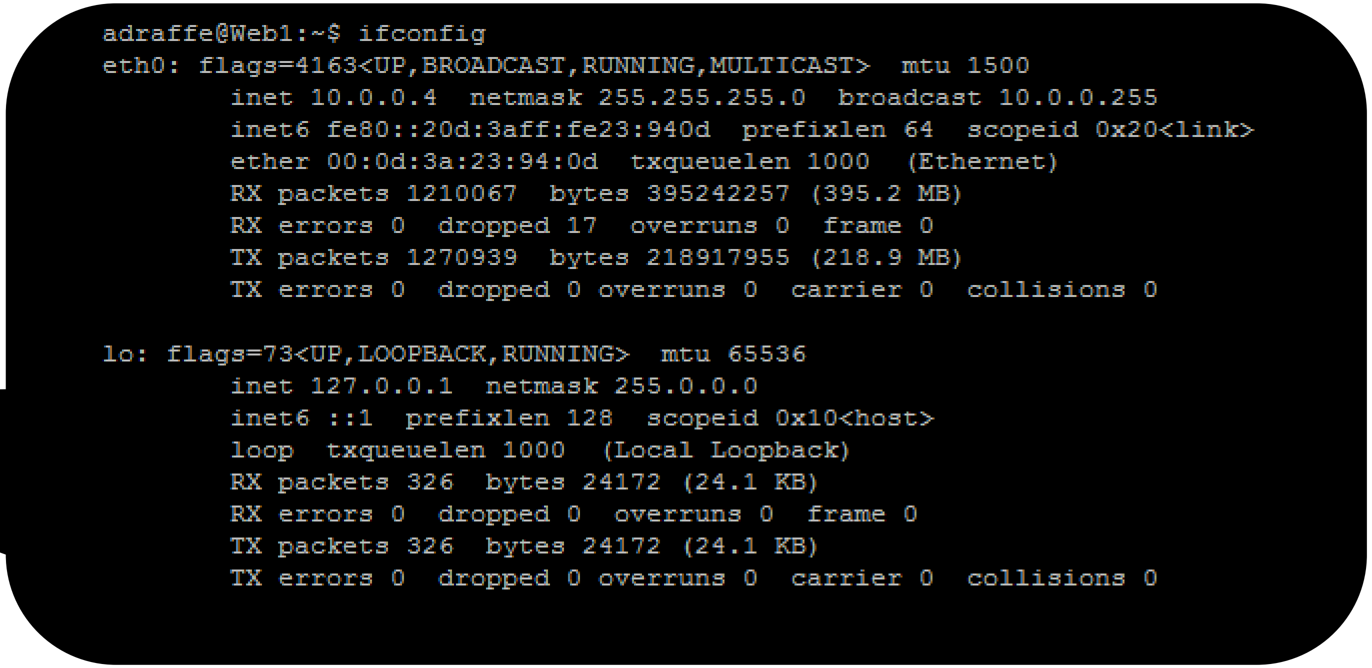
Size
Standard D1 v2 (1 core, 3.5 GB memory)

Public IP address
52.232.35.200

Virtual network/subnet
Web1-Vnet/Web1-Sub1

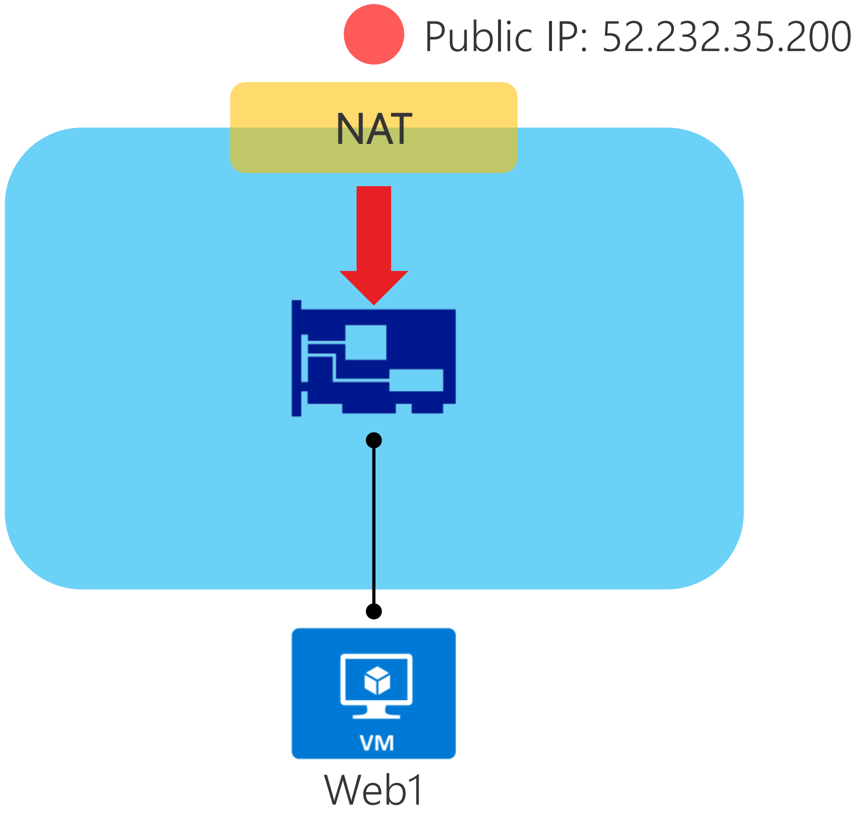
But…when I look at the interfaces on my VM, I don’t see my public IP?!





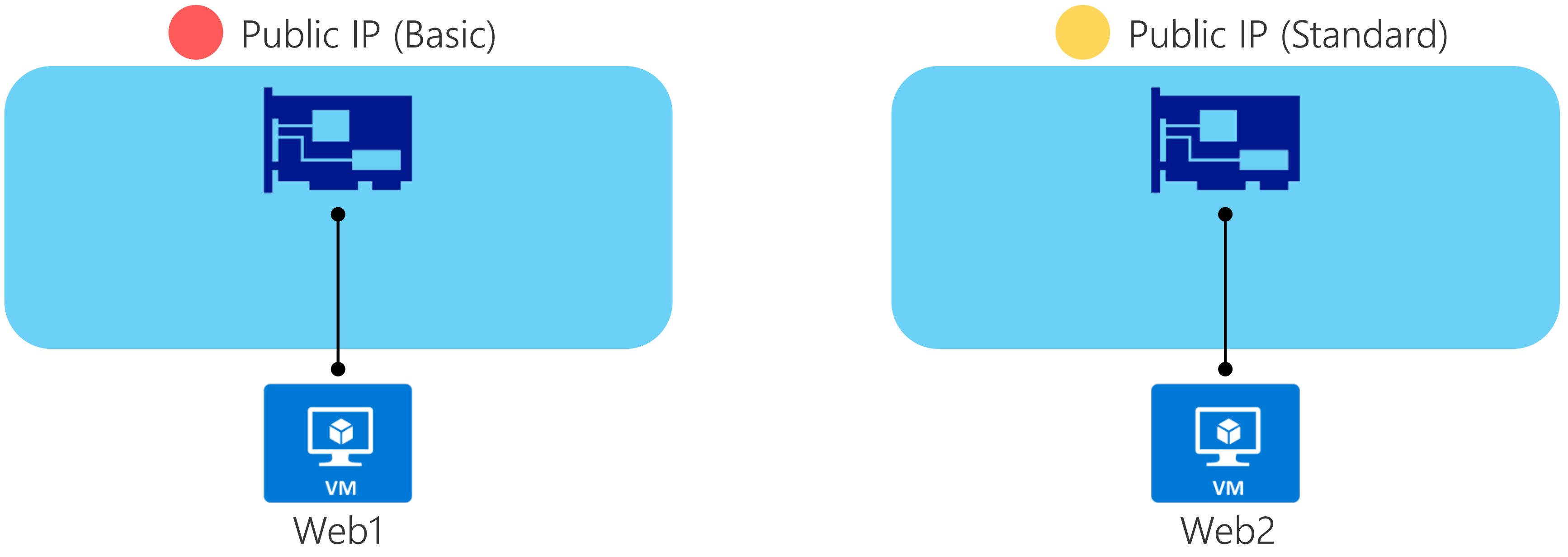
```
adraffe@Web1:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.4  netmask 255.255.255.0  broadcast 10.0.0.255
        inet6 fe80::20d:3aff:fe23:940d  prefixlen 64  scopeid 0x20<link>
        ether 00:0d:3a:23:94:0d  txqueuelen 1000  (Ethernet)
        RX packets 1210067  bytes 395242257 (395.2 MB)
        RX errors 0  dropped 17  overruns 0  frame 0
        TX packets 1270939  bytes 218917955 (218.9 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 326  bytes 24172 (24.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 326  bytes 24172 (24.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

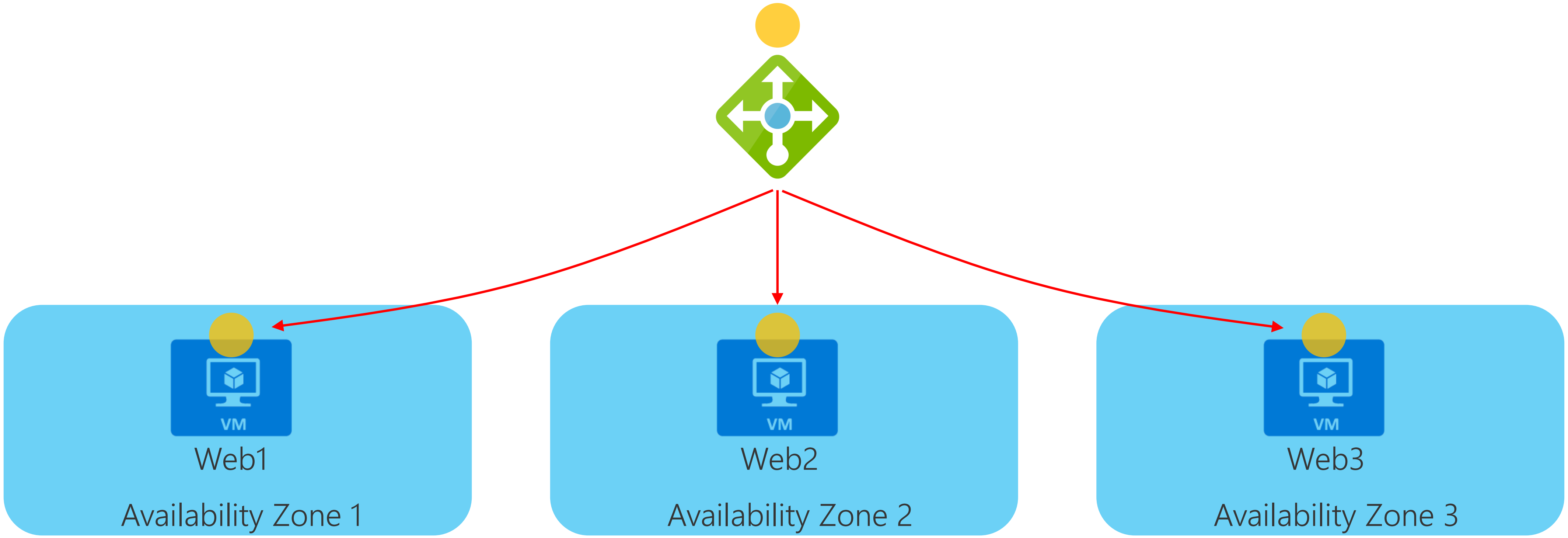


This is because the public IP actually exists as a NAT (Network Address Translation) entry on the Azure fabric that gets mapped to the VM.

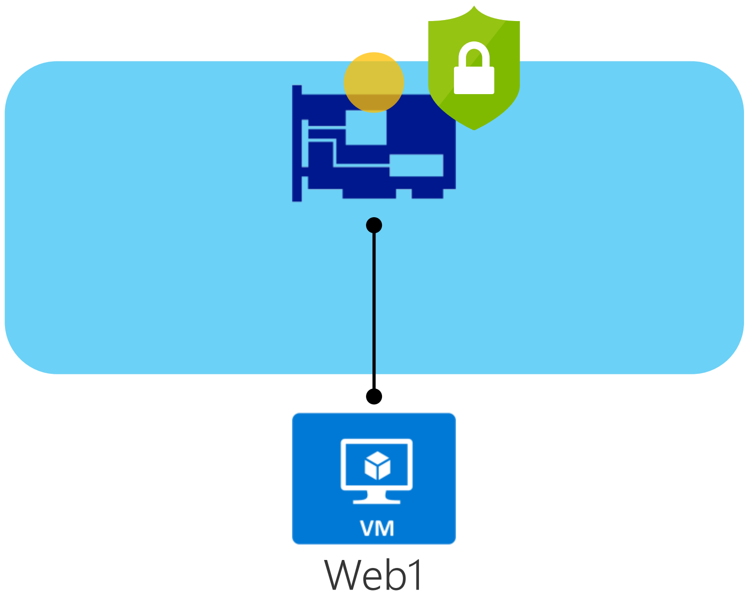
Public IP addresses are available in two “SKUs” – *Basic* or *Standard*.



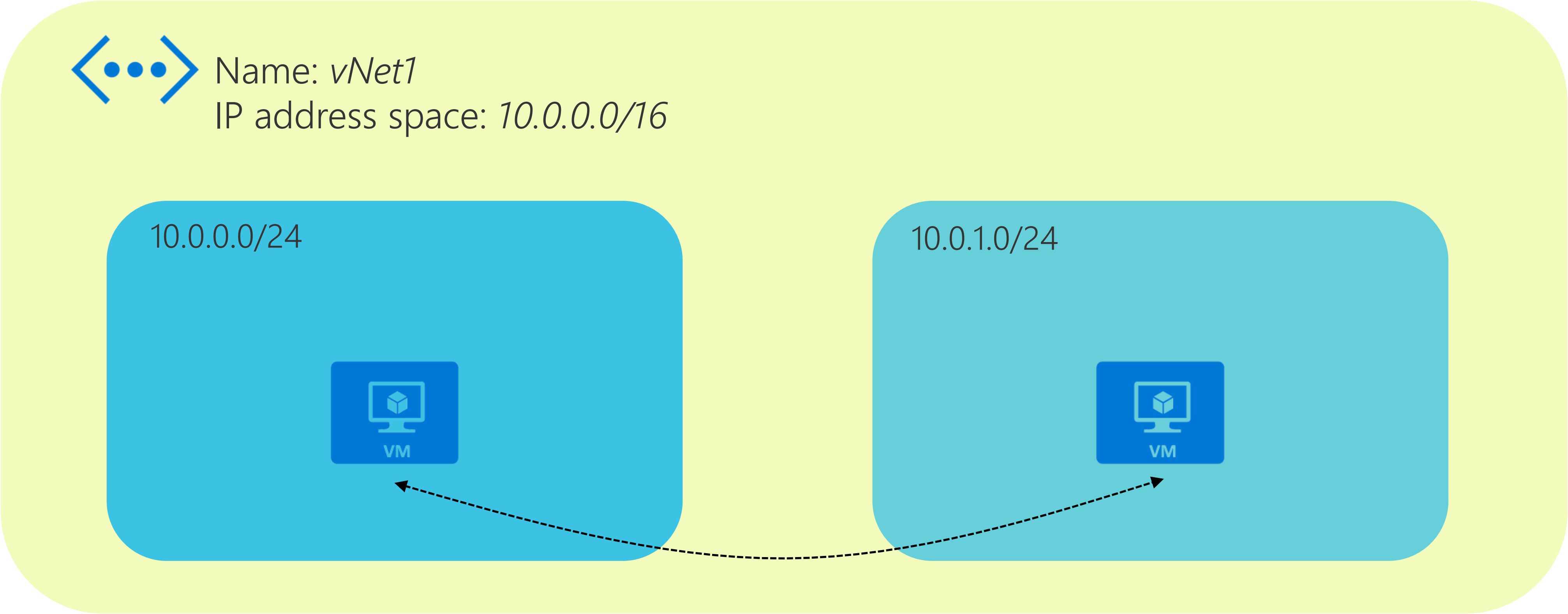
The main difference with Standard Public IP addresses are that they are *zone redundant*.



Caution: If attaching a “Standard SKU” public IP to a VM interface, you *must* apply a Network Security Group, otherwise you won’t be able to reach that VM.



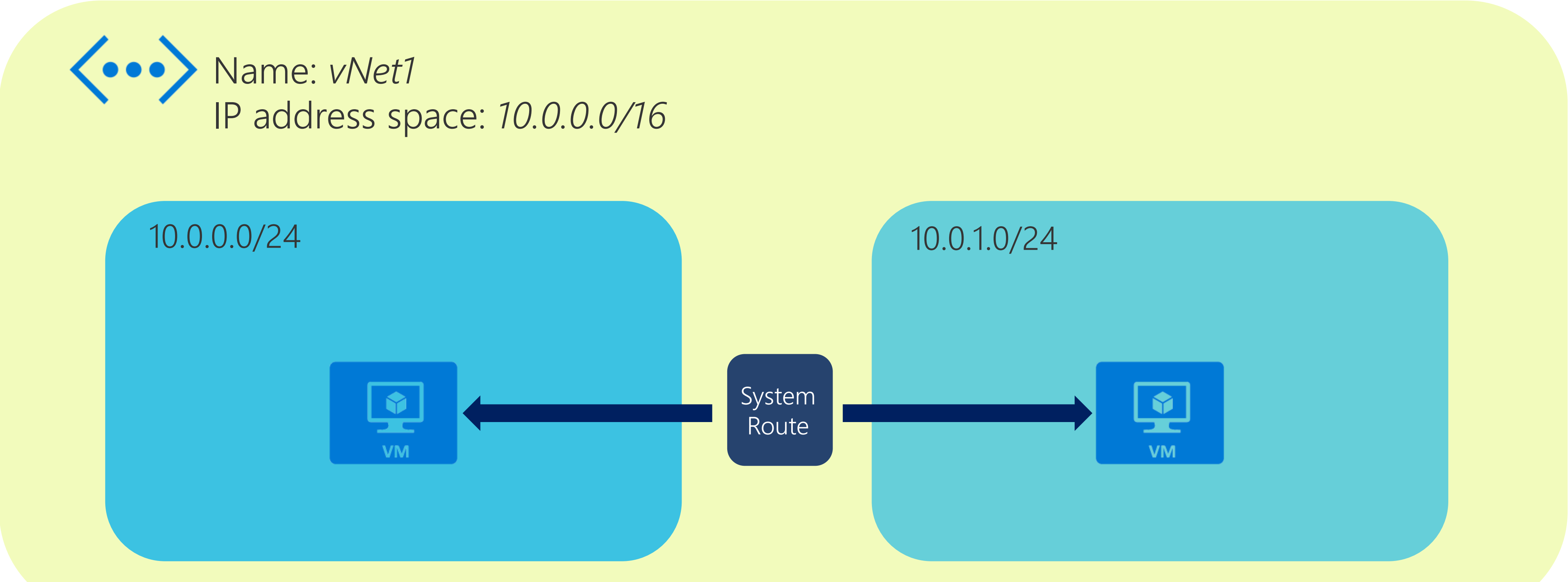
By default, VMs in different subnets within a vNet can route to each other directly.



Name: *vNet1*
IP address space: *10.0.0.0/16*

A *system route* is installed to allow traffic within the vNet.

System routes are also installed for vNet peerings, VM to Internet connectivity, VPN gateway connectivity, etc.

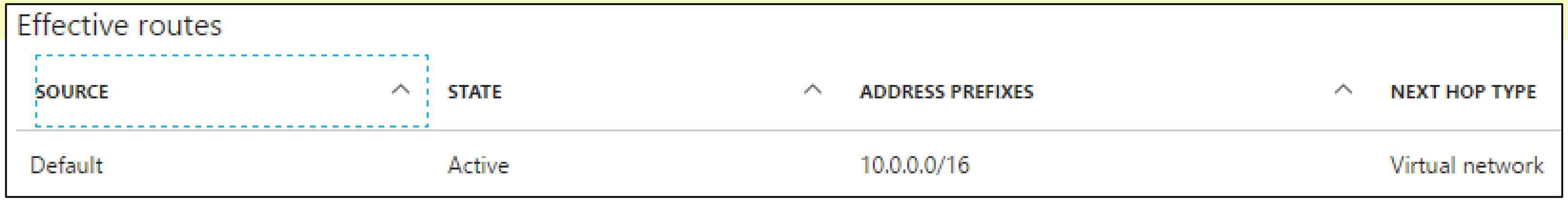


Name: *vNet1*
IP address space: *10.0.0.0/16*

10.0.0.0/24

10.0.1.0/24

System Route



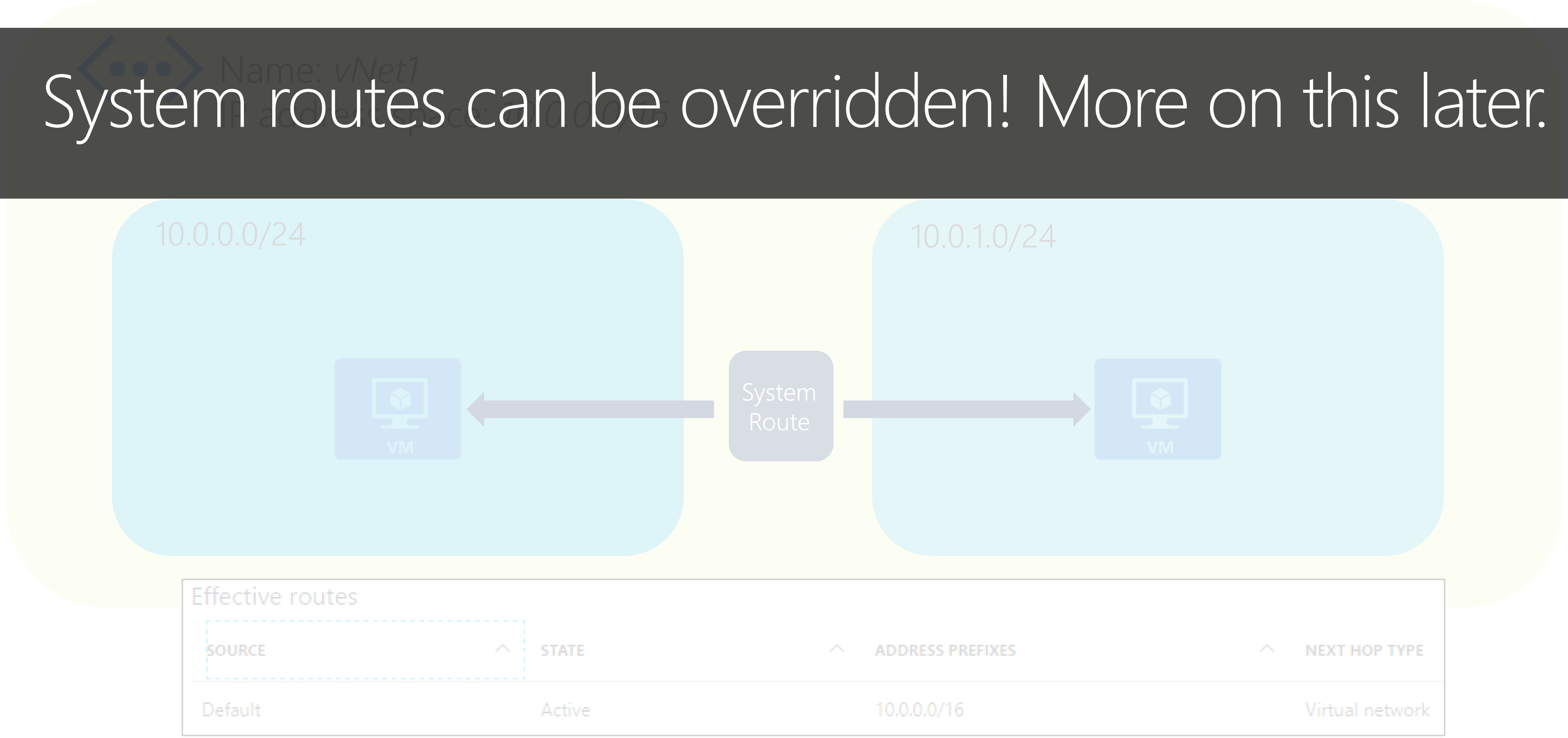
Effective routes

SOURCE	STATE	ADDRESS PREFIXES	NEXT HOP TYPE
Default	Active	10.0.0.0/16	Virtual network

A *system route* is installed to allow traffic within the vNet.

System routes are also installed for vNet peerings, VM to Internet connectivity, VPN gateway connectivity, etc.

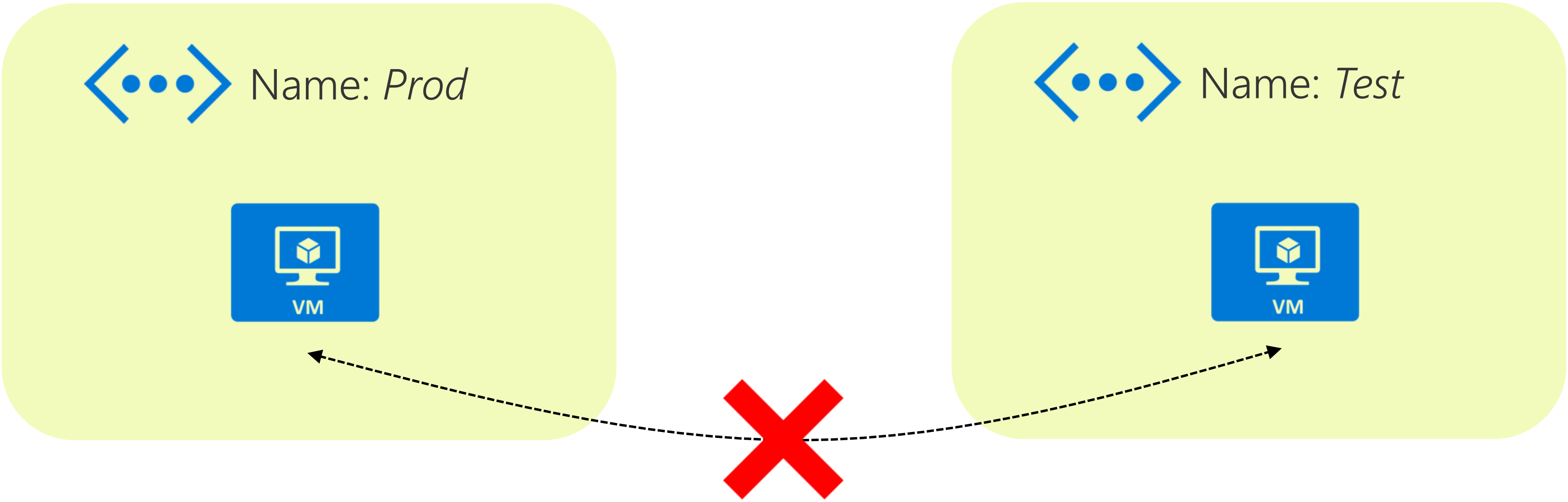
System routes can be overridden! More on this later.



Effective routes

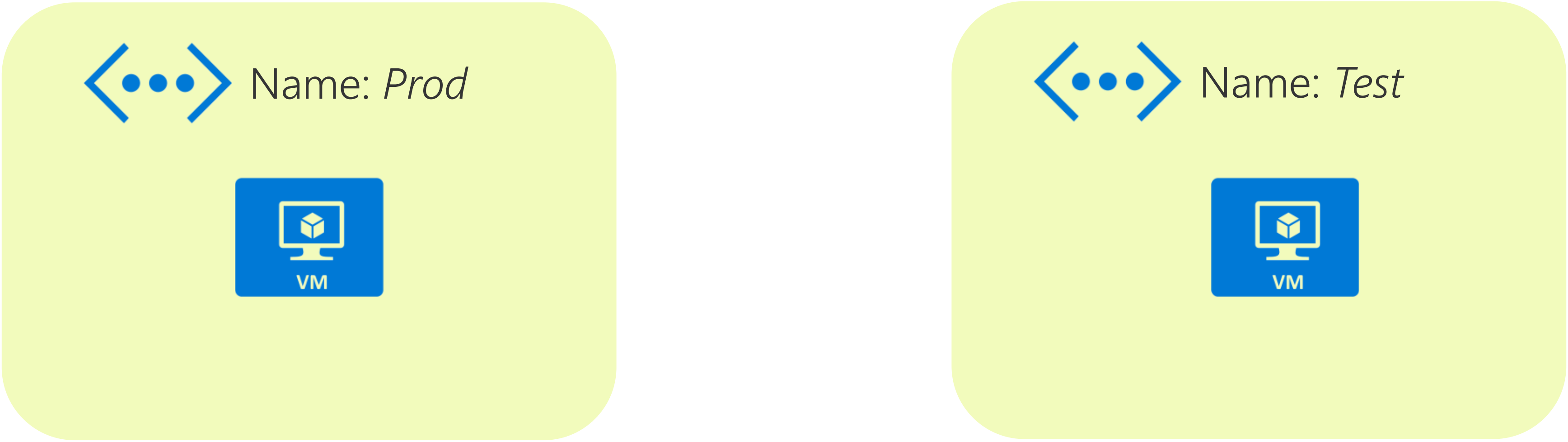
SOURCE	STATE	ADDRESS PREFIXES	NEXT HOP TYPE
Default	Active	10.0.0.0/16	Virtual network

VMs in different vNets *cannot* communicate, unless you configure this specifically (e.g. vNet peering or other method).



Connecting Virtual Networks Together

At some point, you'll want to allow connectivity between different virtual networks.



Name: *Prod*

VM

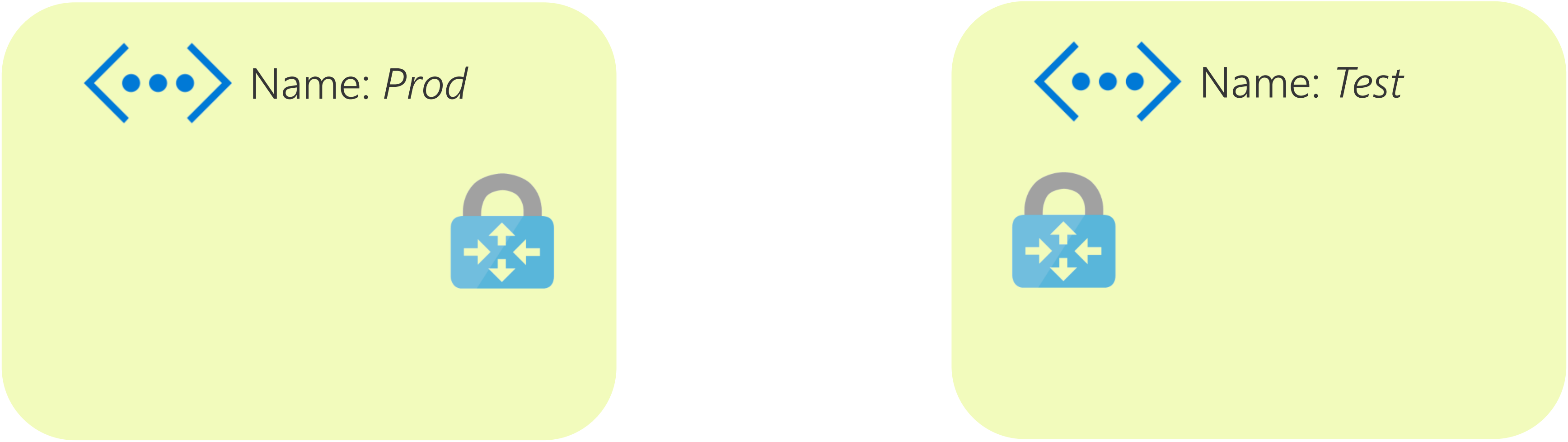
Name: *Test*

VM

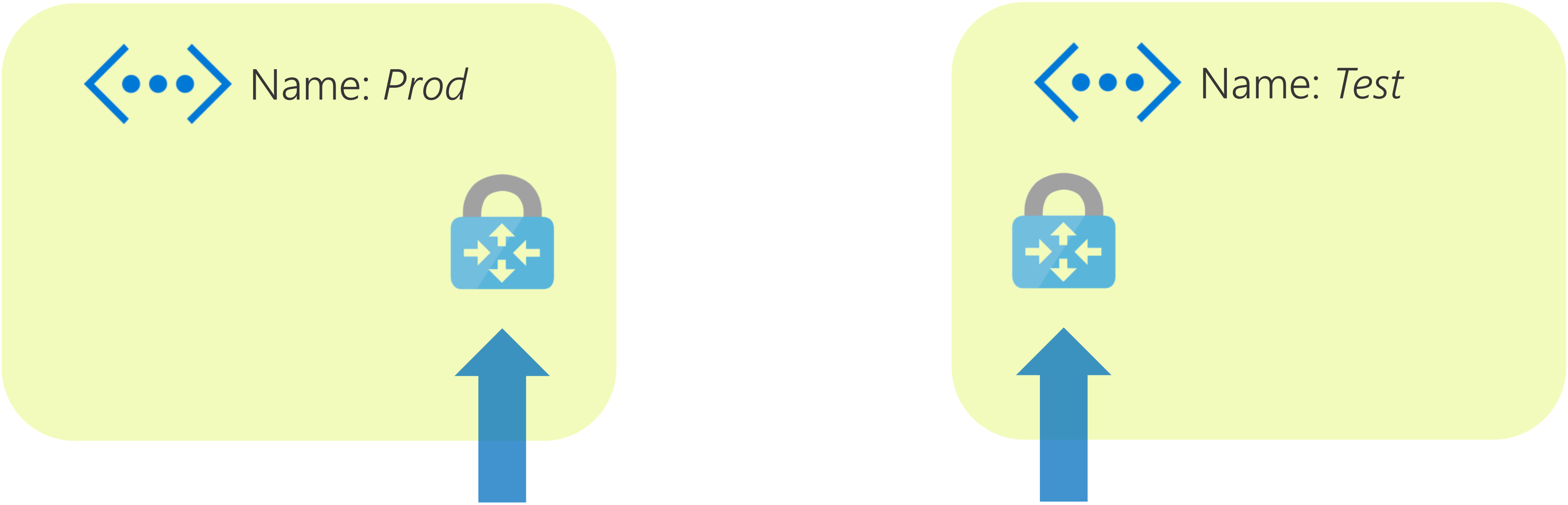
There are two main ways to achieve that: VPN Gateways or Virtual Network Peering.

Option 1: VPN Gateways

VPN gateways are appliances that can be used to connect two vNets together, or between on-premises networks and Azure.

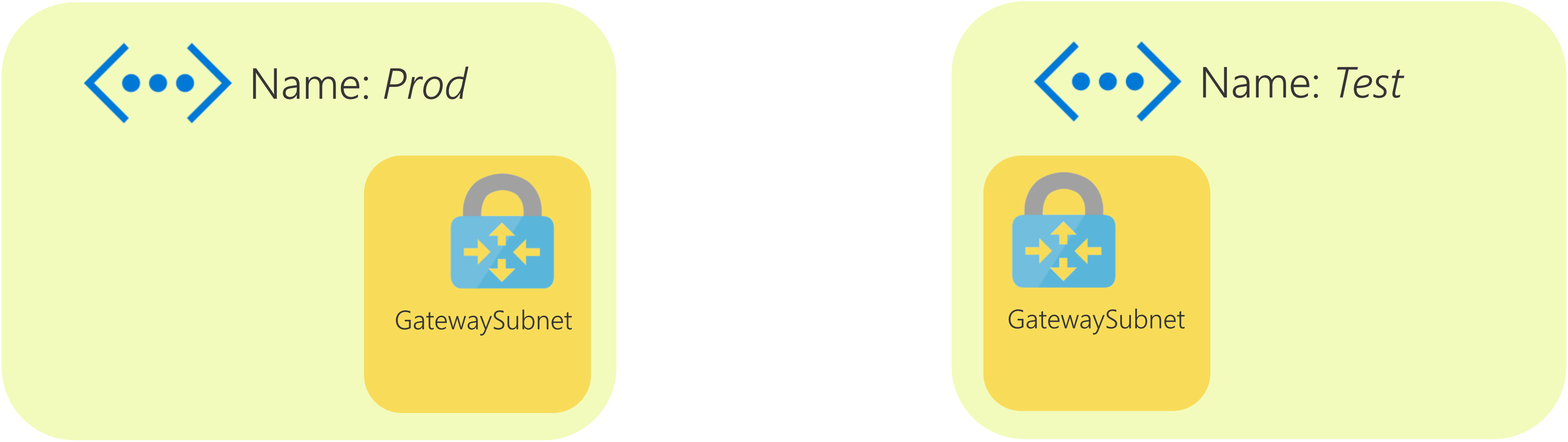


Option 1: VPN Gateways



To connect two vNets together,
you must create a VPN gateway in each vNet.

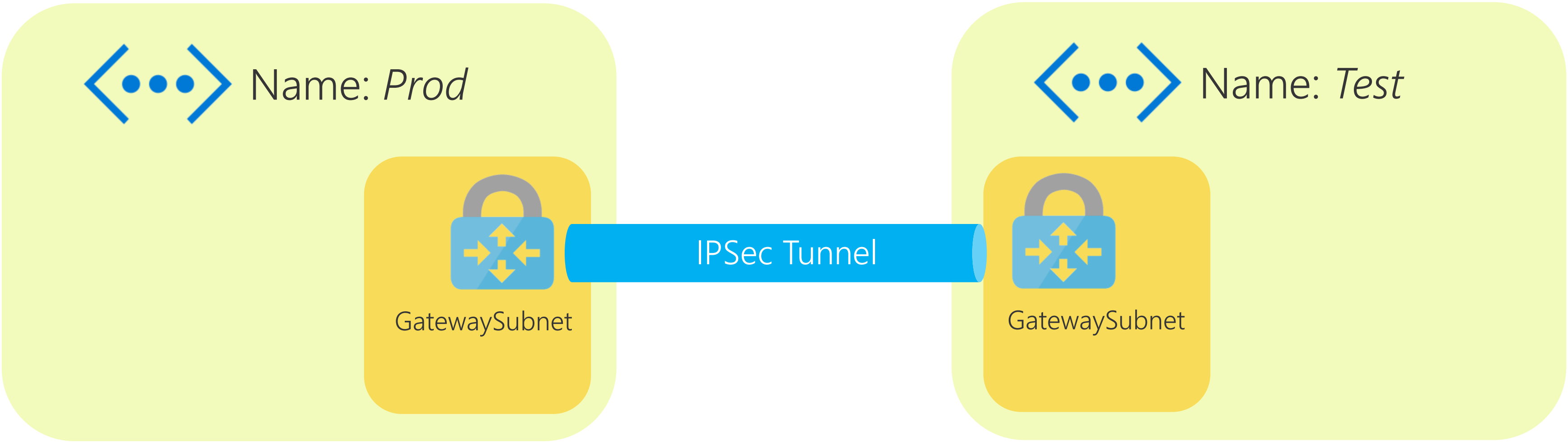
Option 1: VPN Gateways



VPN gateways always connect to a special subnet, called *GatewaySubnet* (this name is mandatory).

Option 1: VPN Gateways

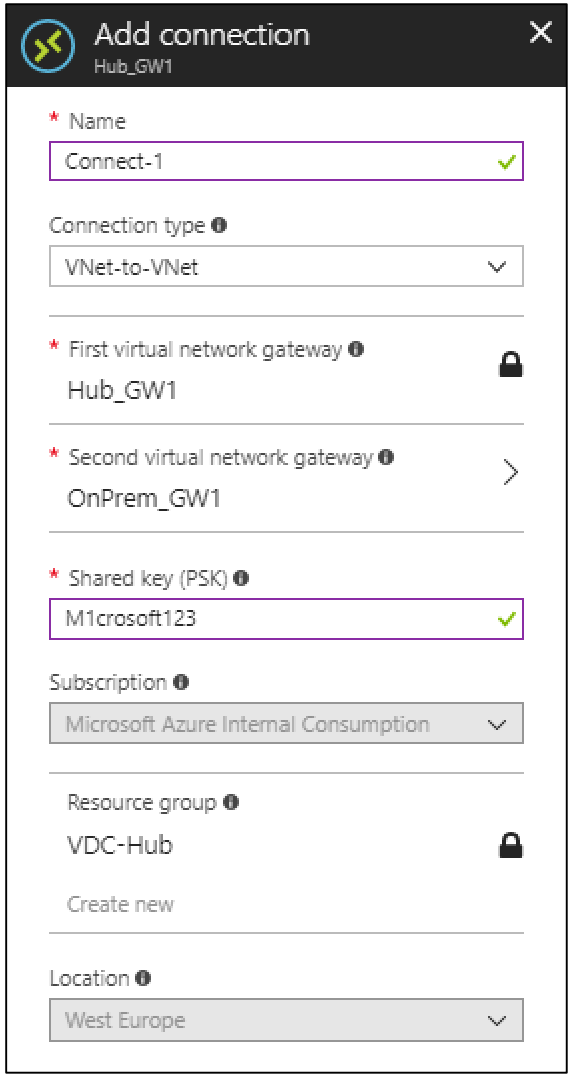
Once the gateways have been created,
the next step is to create a **connection** between them.



Option 1: VPN Gateways

To create a connection, simply specify the two VPN gateways and configure a shared key.

You need to do this from both sides, i.e. create a connection from each VPN gateway to the other one.



Add connection
Hub_GW1

* Name
Connect-1 ✓

Connection type
VNet-to-VNet

* First virtual network gateway
Hub_GW1

* Second virtual network gateway
OnPrem_GW1

* Shared key (PSK)
M1crosoft123 ✓

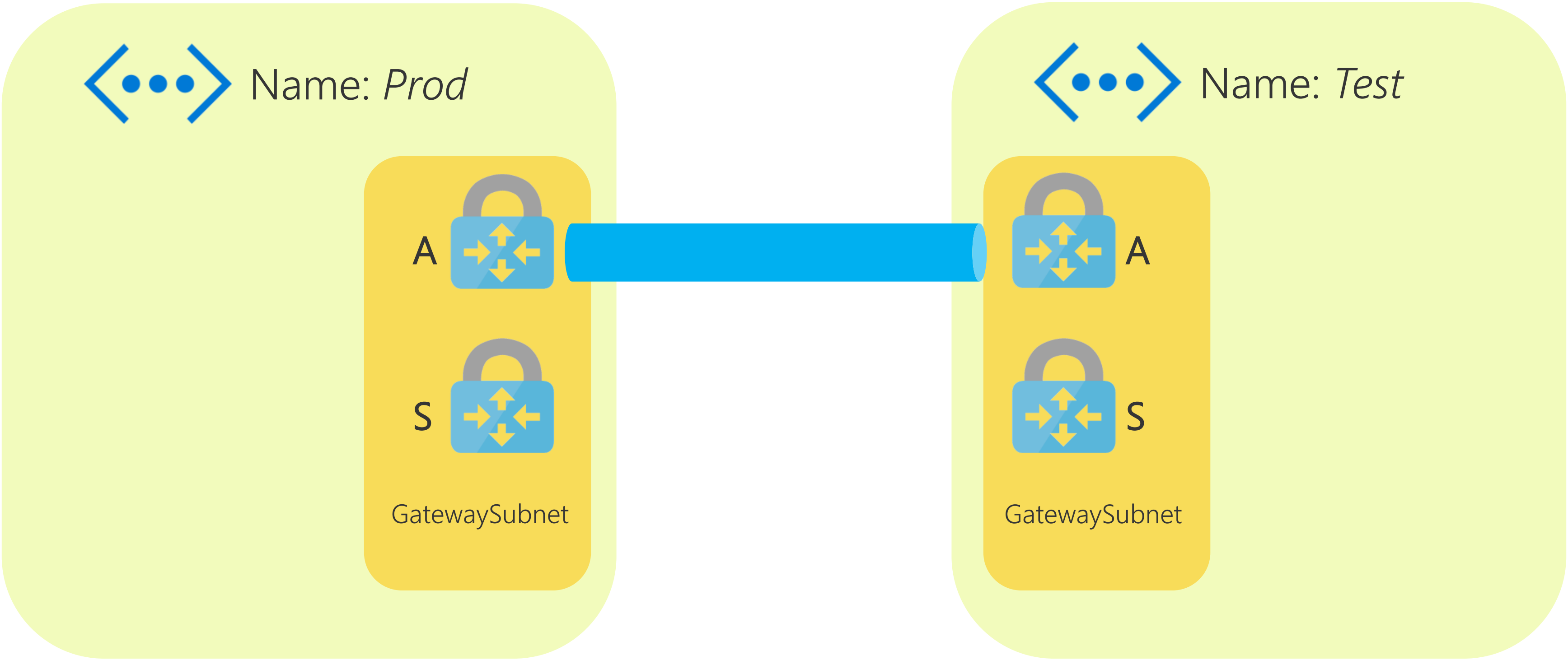
Subscription
Microsoft Azure Internal Consumption

Resource group
VDC-Hub
Create new

Location
West Europe

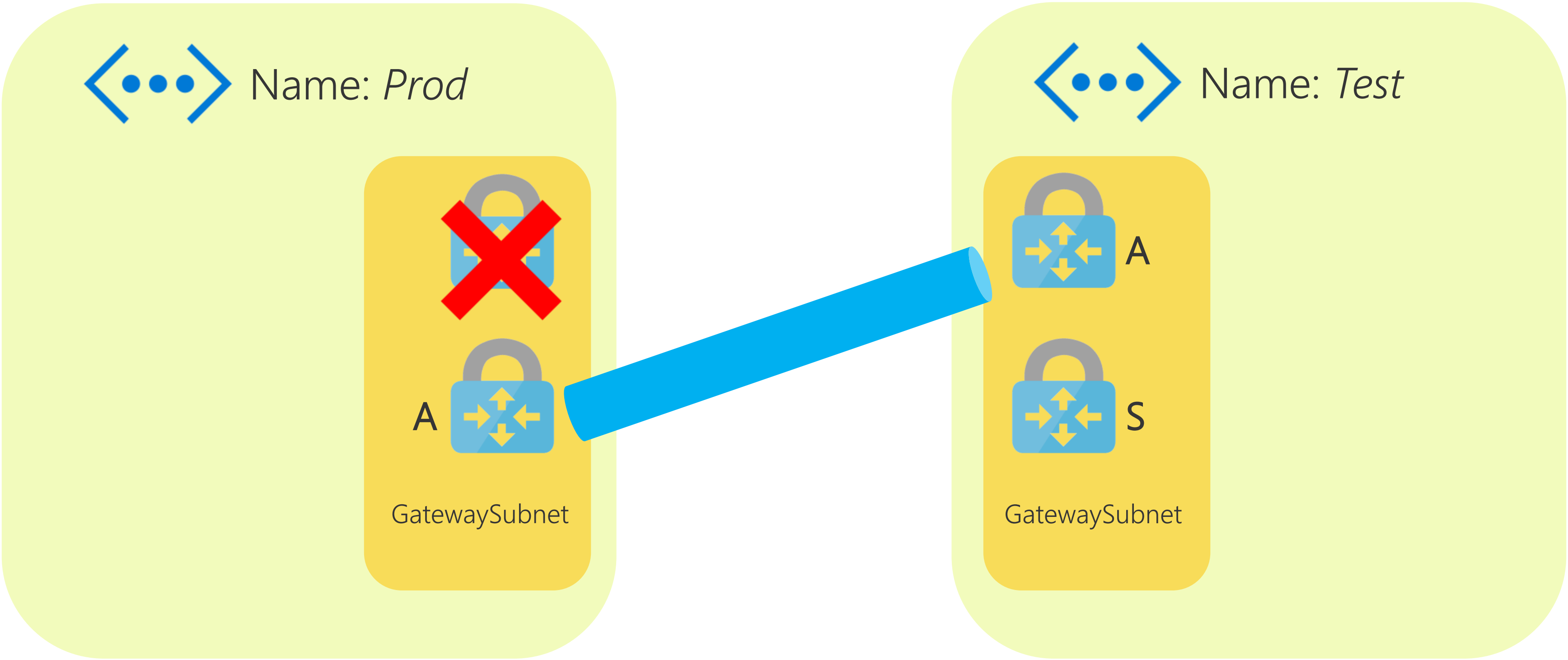
Option 1: VPN Gateways

Although this is hidden from the user, VPN gateways consist of two instances in an active / standby configuration.



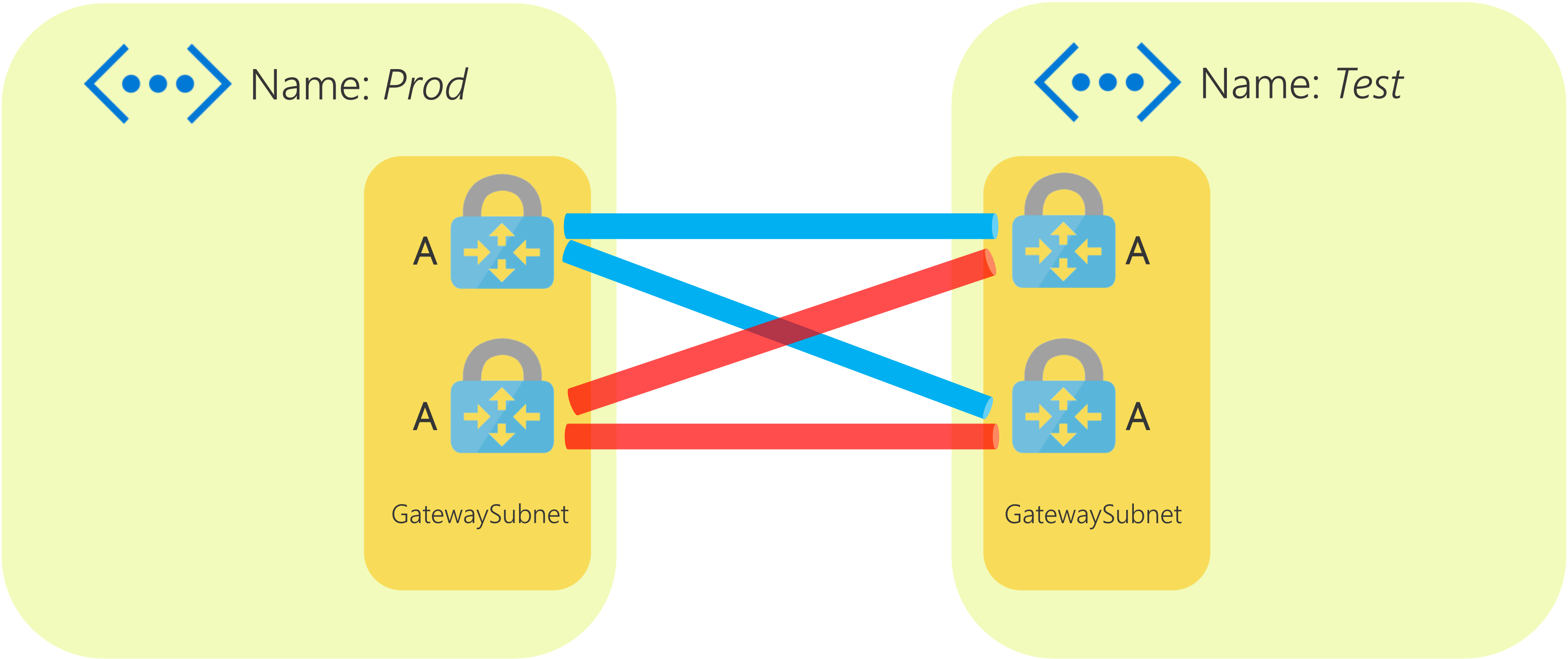
Option 1: VPN Gateways

Failure of a gateway will result in the standby taking over (worst case 1.5 mins failover time).



Option 1: VPN Gateways

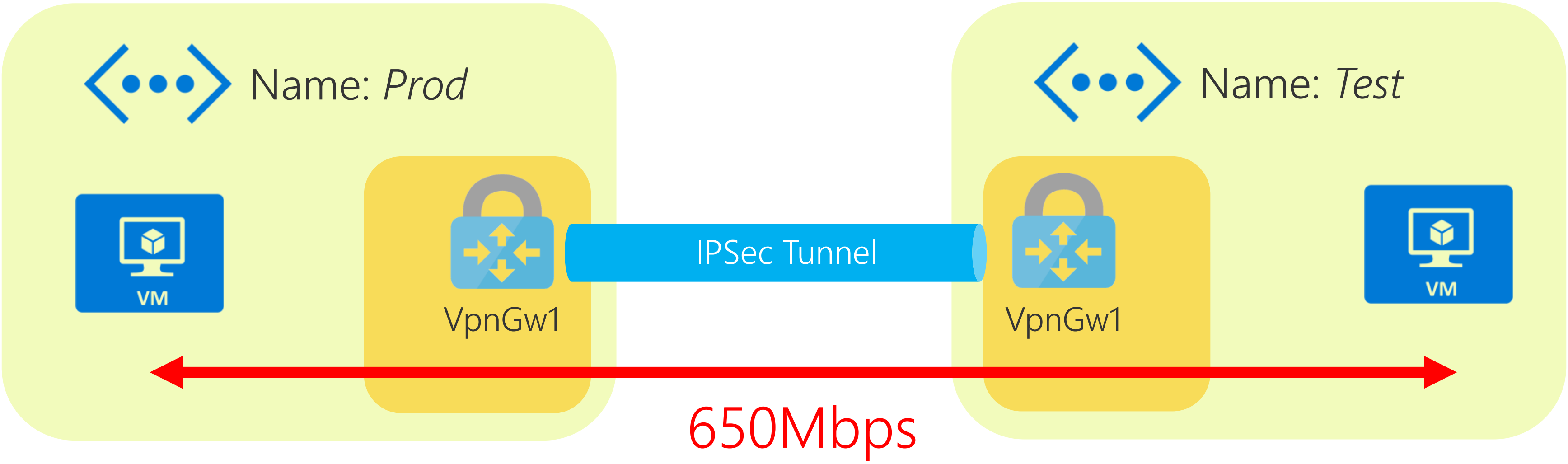
It's also possible to create VPN gateways in an active / active configuration, which will use a full mesh of IPSec tunnels.



```
New-AzureRmVirtualNetworkGateway -EnableActiveActiveFeature
```

Option 1: VPN Gateways

One of the issues with using VPN gateways to connect vNets is that you are limited to the bandwidth of the gateway (e.g. 650Mbps for VpnGw1).



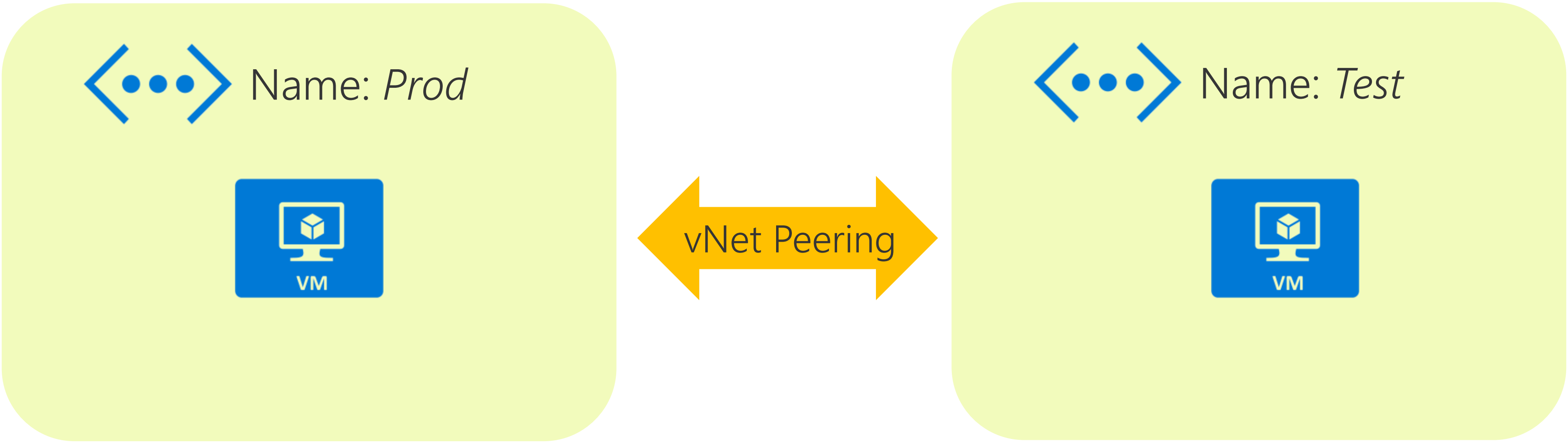
Option 2: vNet Peering

One of the issues in using VPN gateways to connect vNets is that you are limited to the bandwidth of the gateway (e.g. 650Mbps for VpnGw1).

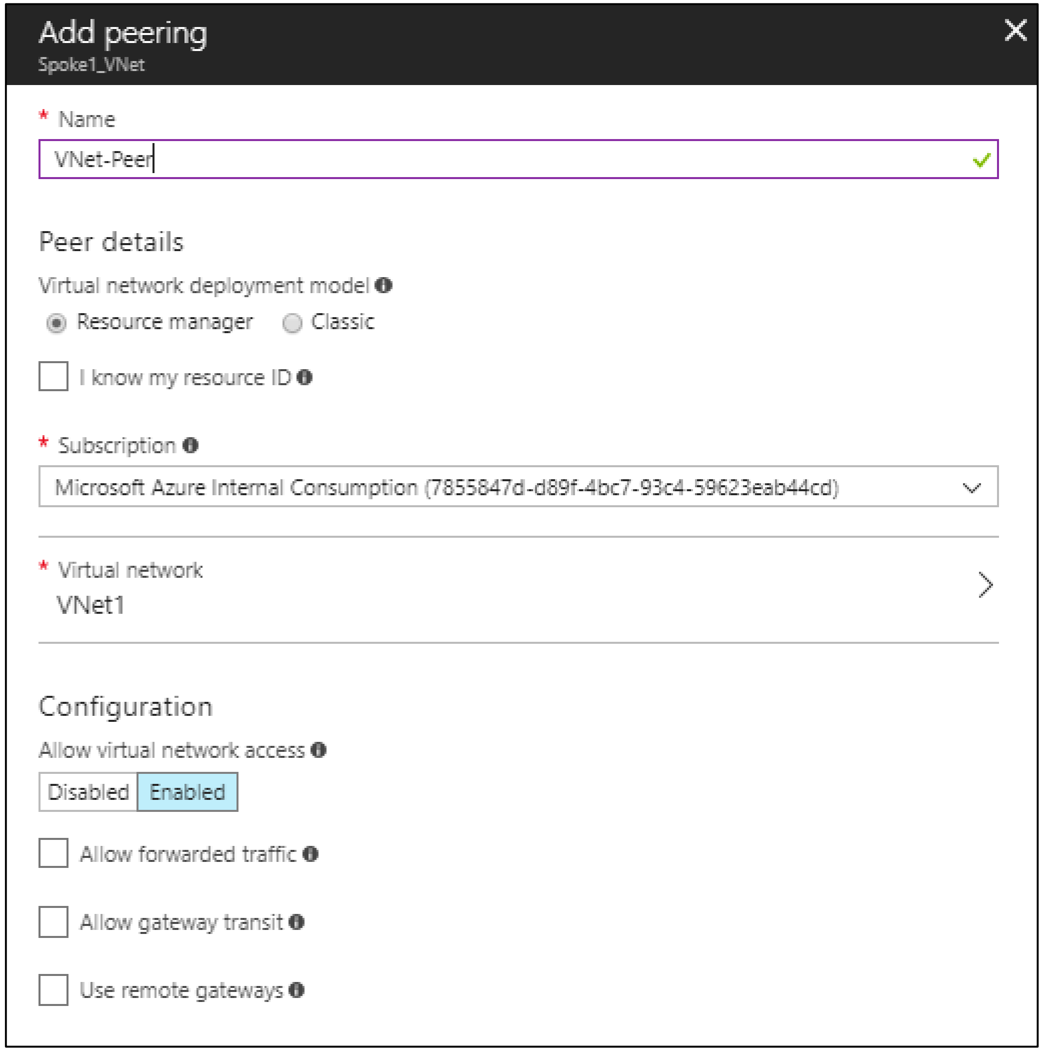
A better way to connect virtual networks together is by using **vNet Peering**.

Option 2: vNet Peering

vNet Peering uses the Microsoft backbone network to connect the vNets together – no gateways required!



Option 2: vNet Peering



Add peering ✕

Spoke1_VNet

* Name

VNet-Peer ✓

Peer details

Virtual network deployment model ℹ

● Resource manager ○ Classic

□ I know my resource ID ℹ

* Subscription ℹ

Microsoft Azure Internal Consumption (7855847d-d89f-4bc7-93c4-59623eab44cd)

* Virtual network

VNet1

Configuration

Allow virtual network access ℹ

Disabled [✓ Enabled]

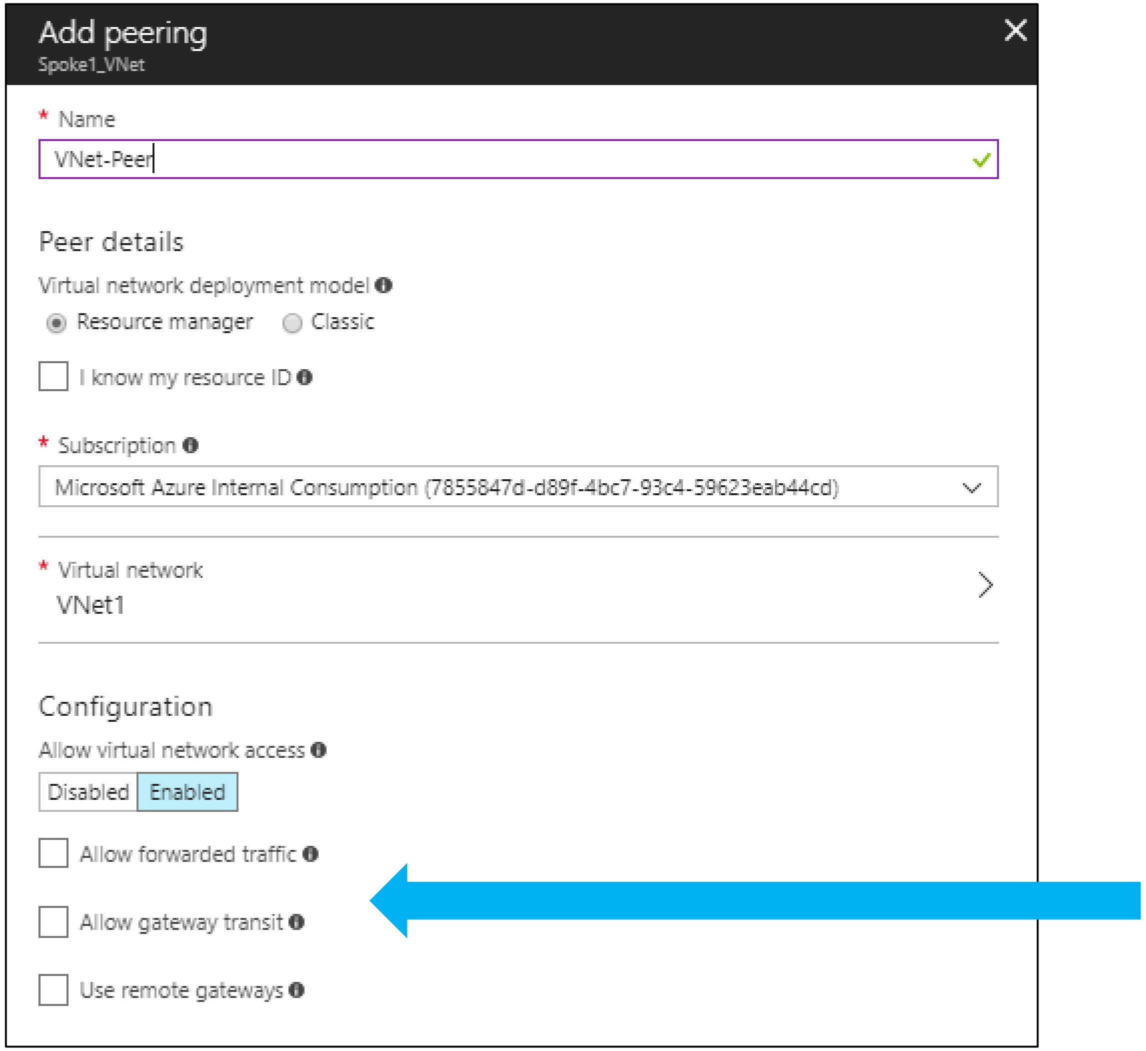
□ Allow forwarded traffic ℹ

□ Allow gateway transit ℹ

□ Use remote gateways ℹ

Setting up vNet Peering is easy – just specify the vNet you want to connect to.

Option 2: vNet Peering



Add peering

Spoke1_VNet

* Name

VNet-Peer ✓

Peer details

Virtual network deployment model

● Resource manager ○ Classic

□ I know my resource ID

* Subscription

Microsoft Azure Internal Consumption (7855847d-d89f-4bc7-93c4-59623eab44cd)

* Virtual network

VNet1

Configuration

Allow virtual network access

Disabled [✓ Enabled]

□ Allow forwarded traffic

□ Allow gateway transit

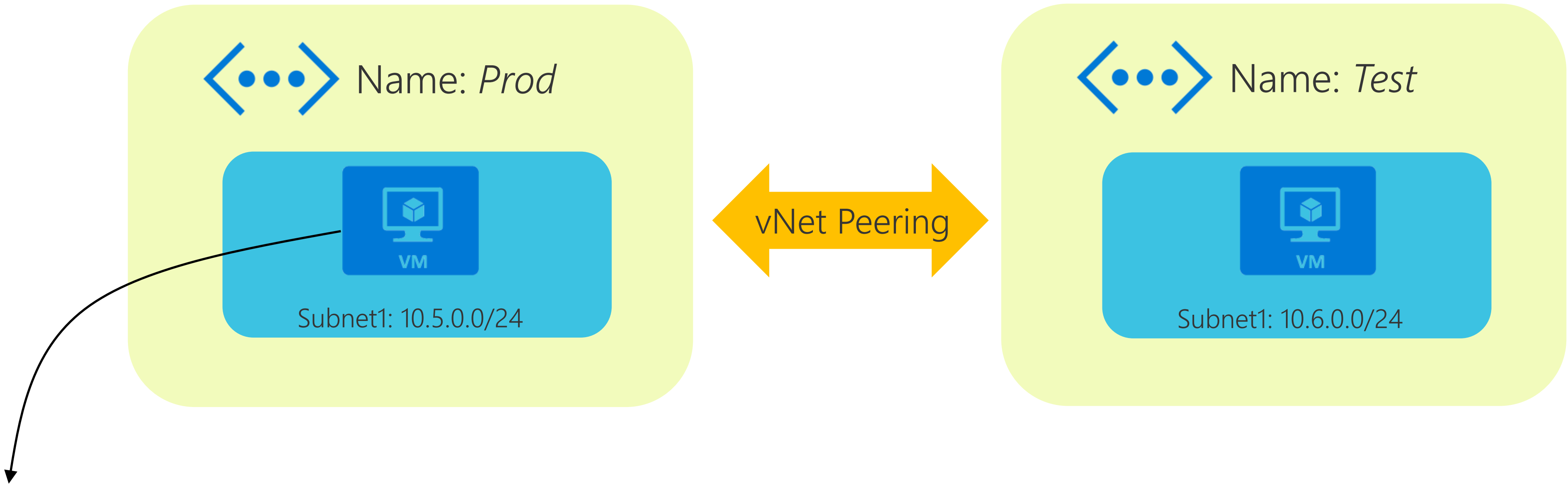
□ Use remote gateways

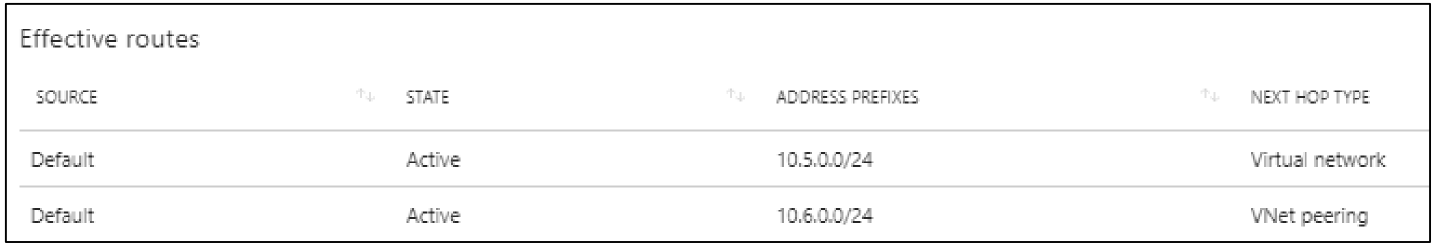
What are these options for?

These cover more advanced scenarios, such as sharing gateways, hub and spoke networks, etc – we'll come back to these later!

Option 2: vNet Peering

Once a vNet peering connection has been created, routes are automatically added to each vNet to point to the other.





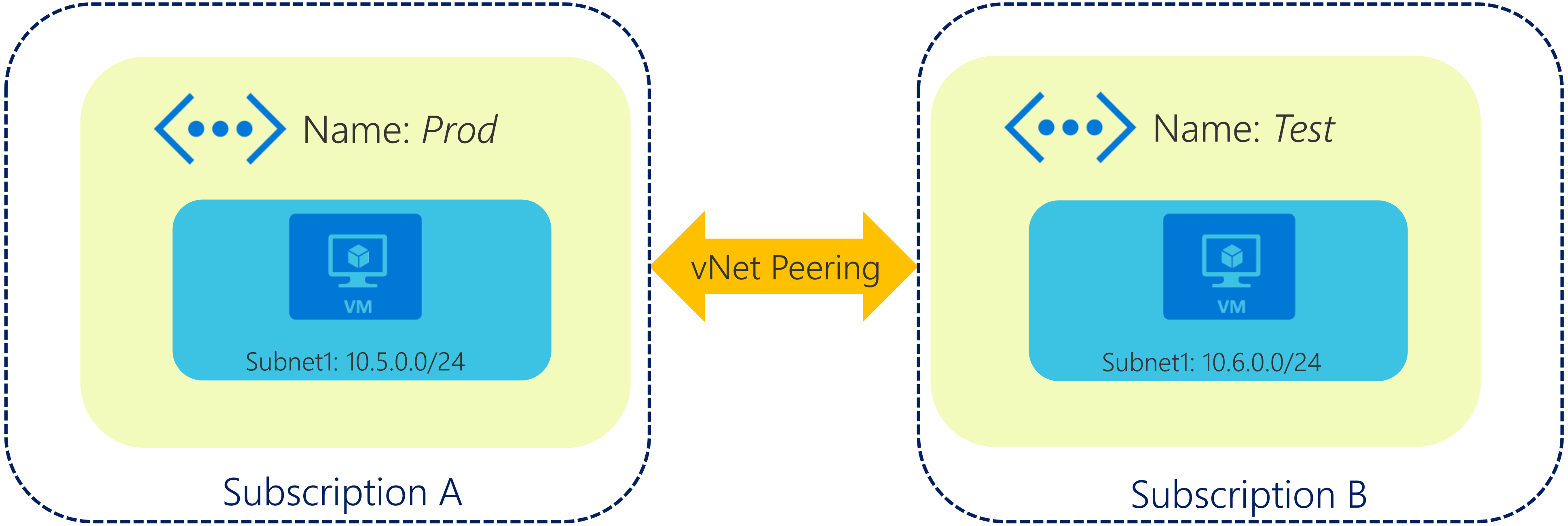
Effective routes

SOURCE	STATE	ADDRESS PREFIXES	NEXT HOP TYPE
Default	Active	10.5.0.0/24	Virtual network
Default	Active	10.6.0.0/24	VNet peering

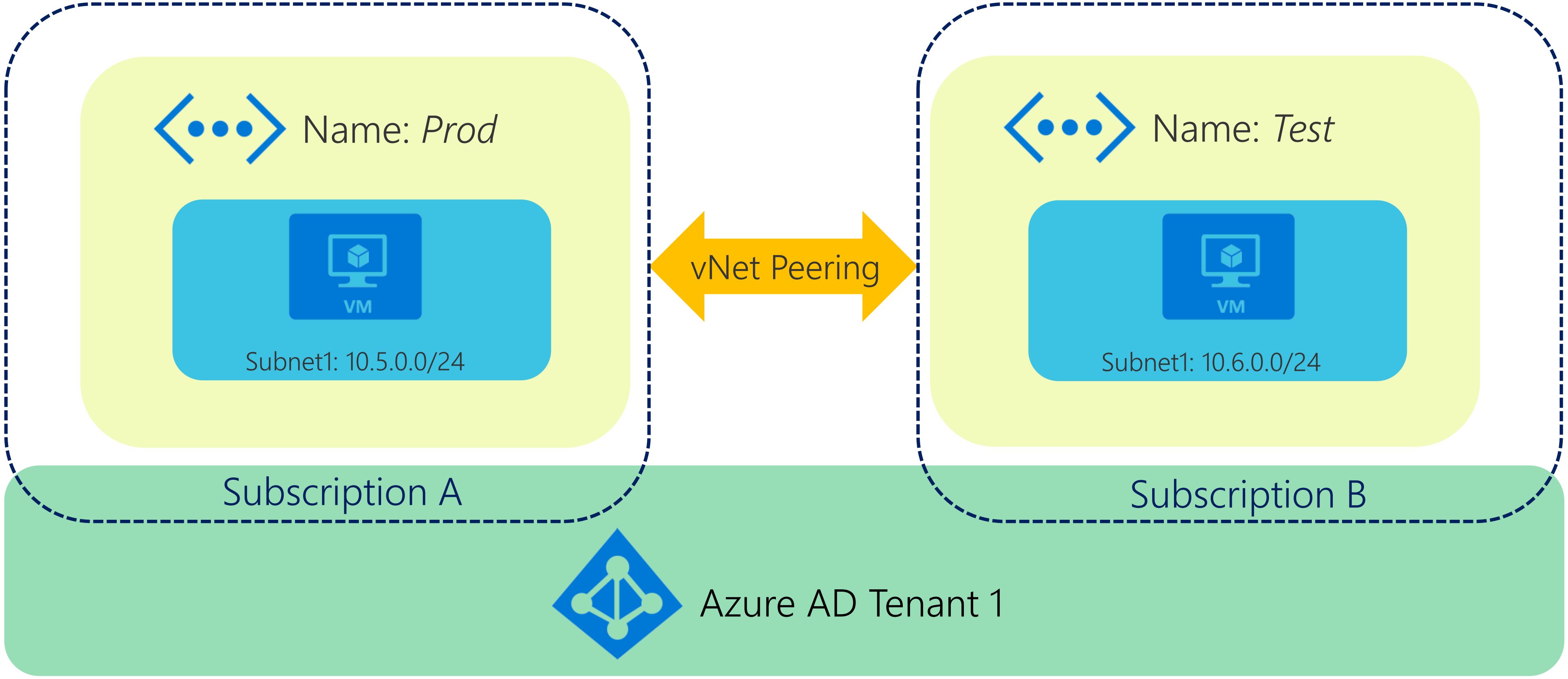
You can find this by selecting ‘Effective Routes’ under the NIC connected to a VM inside the vNet.

Option 2: vNet Peering

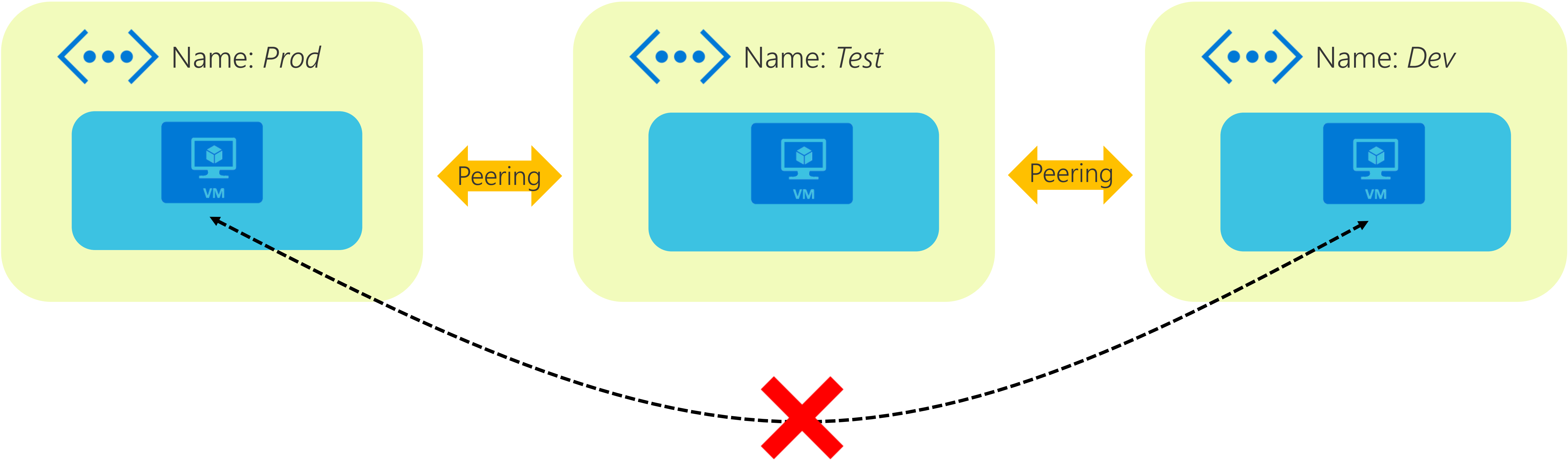
You can peer two vNets that reside in different subscriptions…



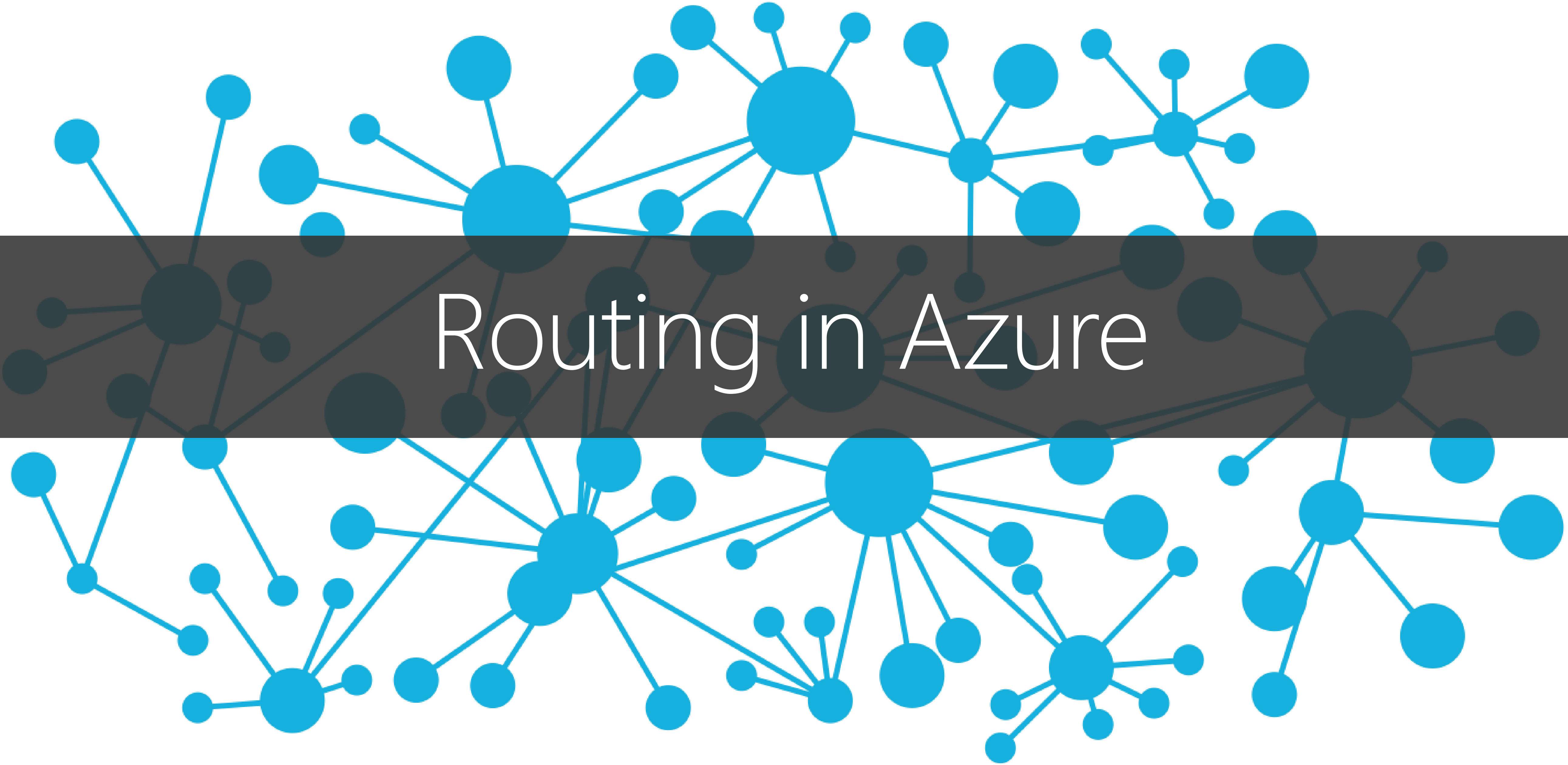
…but only if they are associated to the **same Azure AD tenant.**



vNet Peerings are **non-transitive** – this means that vNet *Prod* cannot communicate with vNet *Dev* through vNet *Test*.



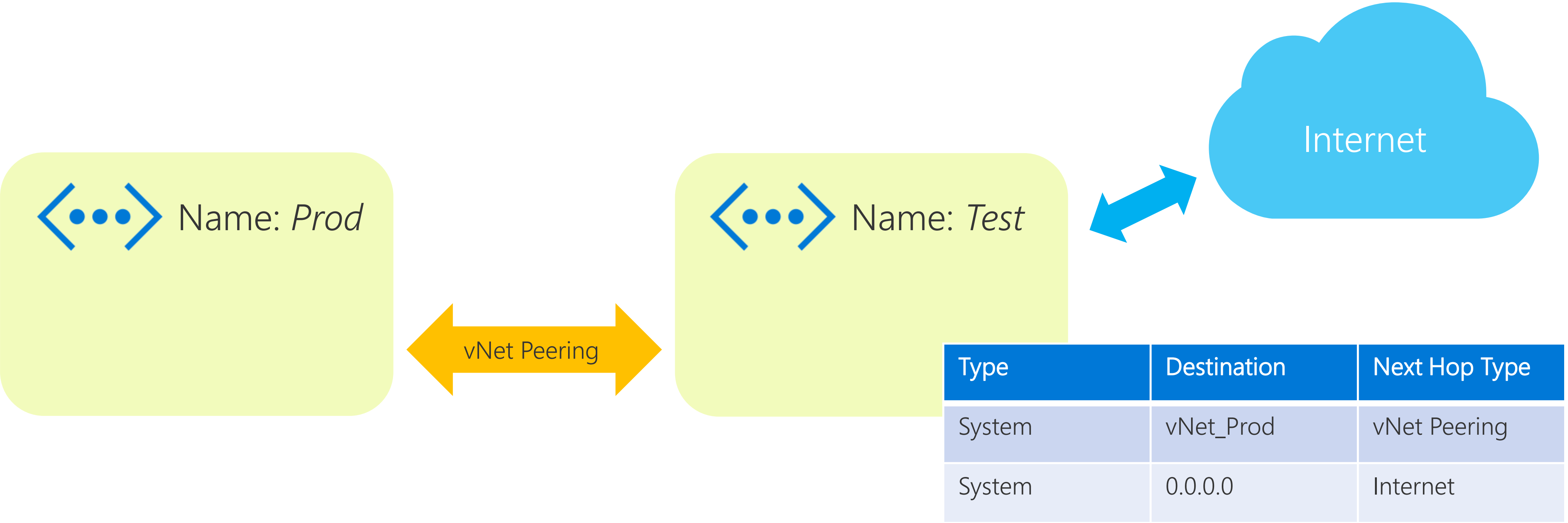
Routing in Azure



Routing in Azure…is just like routing anywhere else.

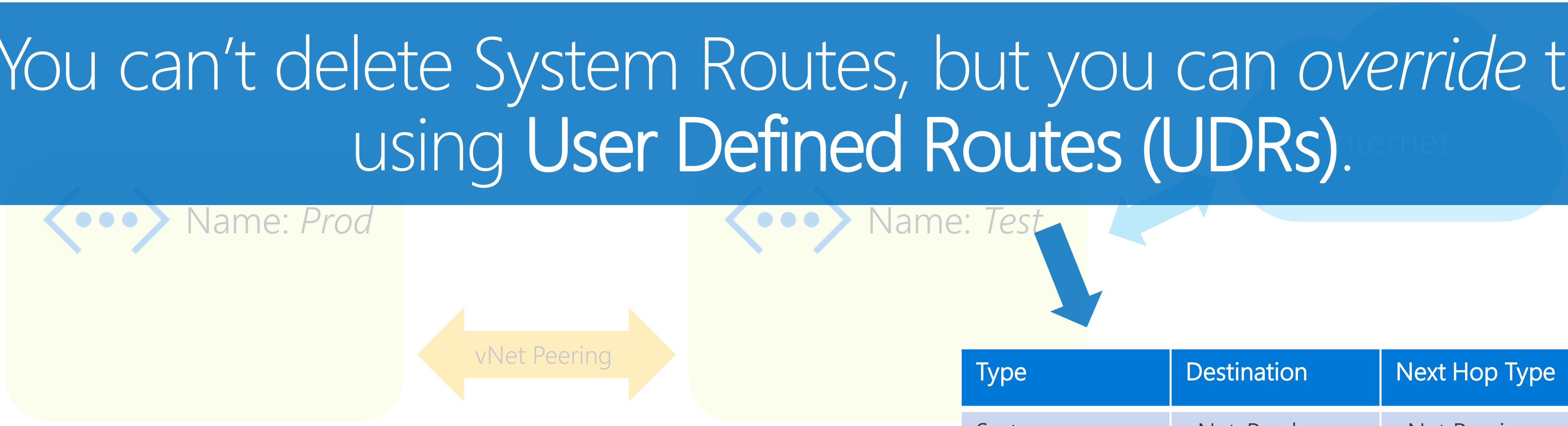
You still have destination prefixes, next hops, etc.

Azure configures **System Routes** when required – for example, for routing between subnets / vNets, routing to the Internet, etc.

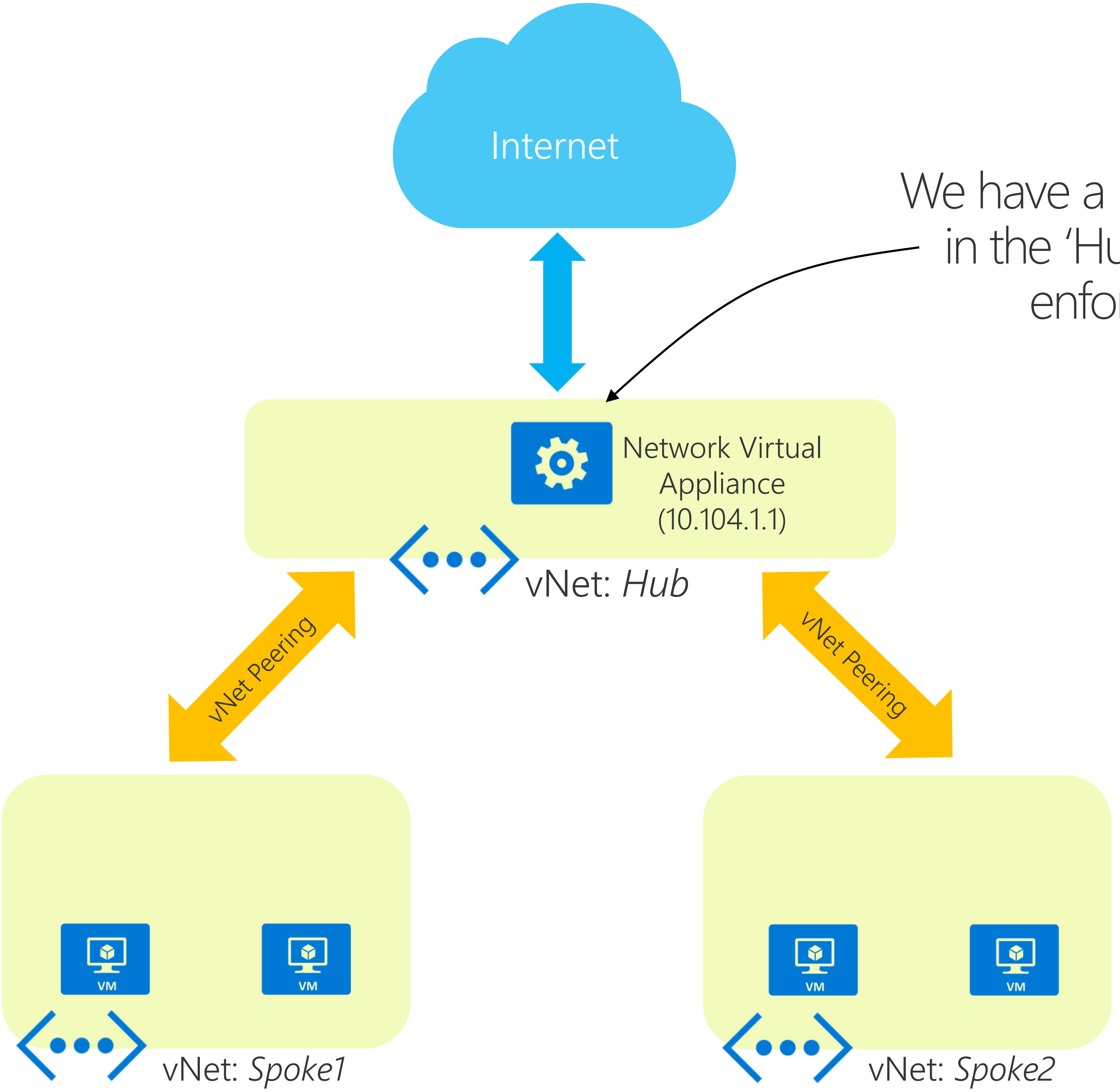


Type	Destination	Next Hop Type
System	vNet_Prod	vNet Peering
System	0.0.0.0	Internet

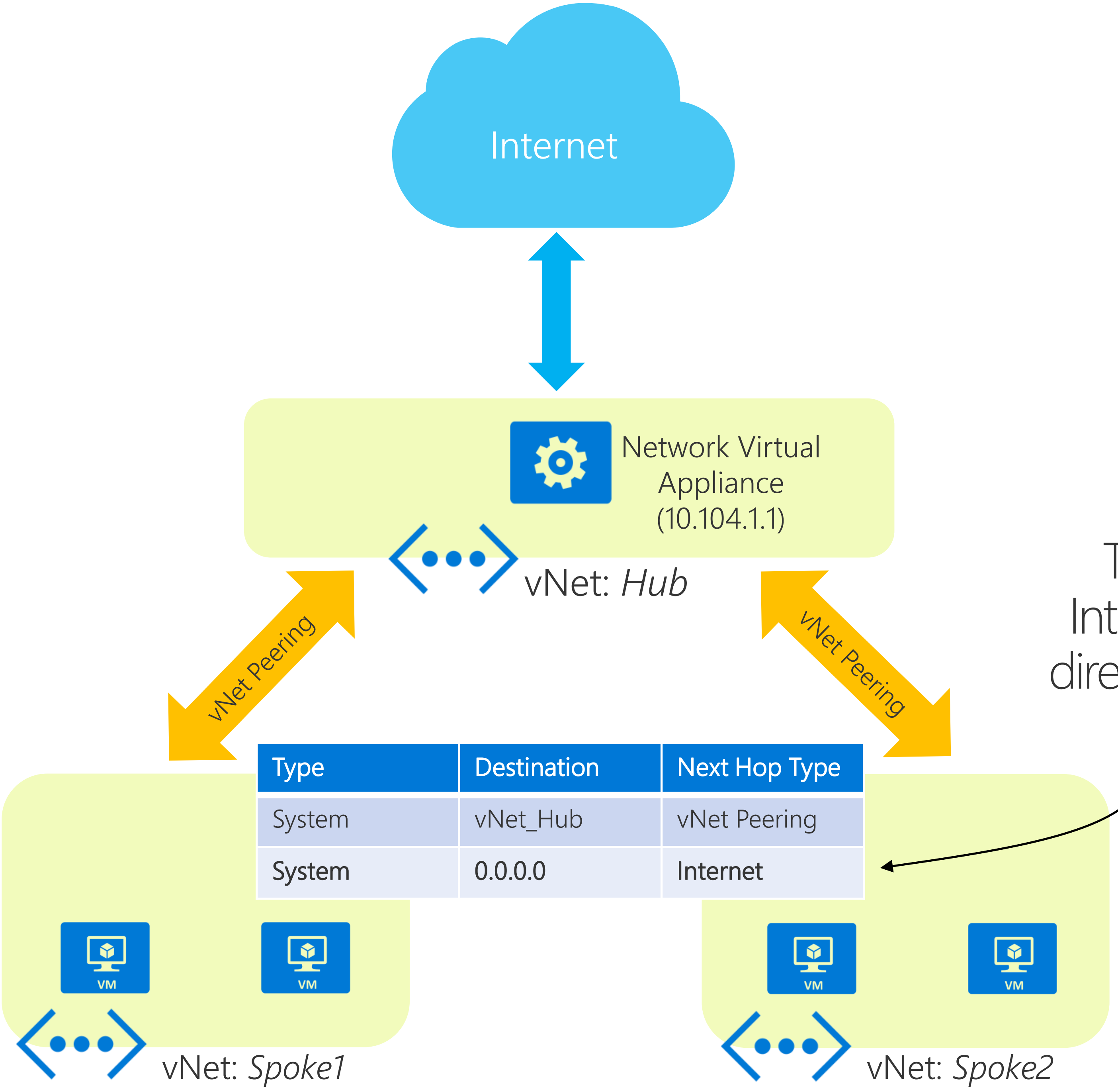
You can't delete System Routes, but you can *override* them, using **User Defined Routes (UDRs)**.



Type	Destination	Next Hop Type
System	vNet_Prod	vNet Peering
System	0.0.0.0	Internet

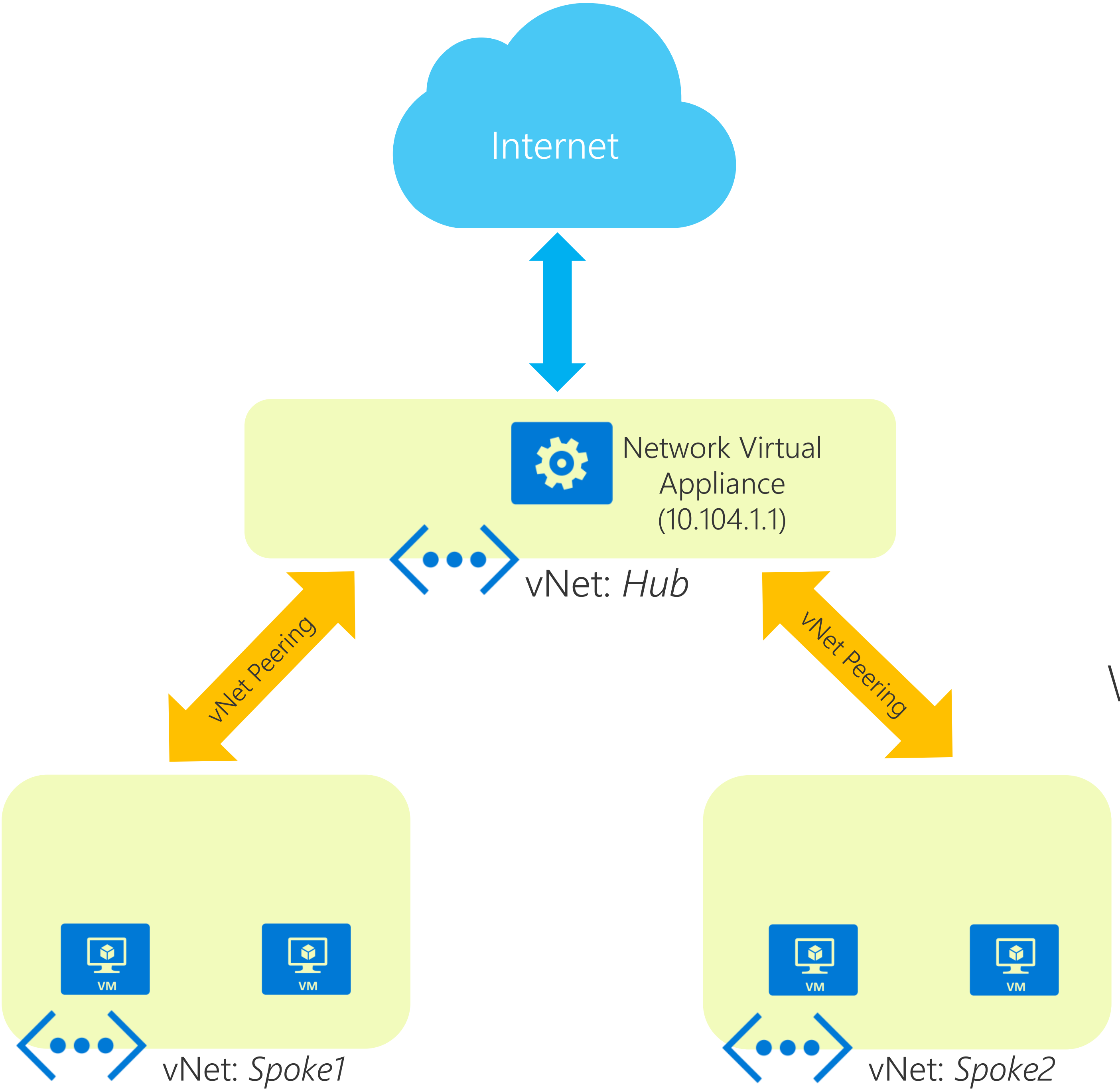


We have a Network Virtual Appliance (e.g. firewall) in the 'Hub' vNet, which we want to act as the enforcement point for Internet traffic.



Type	Destination	Next Hop Type
System	vNet_Hub	vNet Peering
System	0.0.0.0	Internet

The problem is, system routing means Internet traffic from the Spoke vNets goes directly out from the vNet (not via the NVA).



We configure a User Defined Route to point to the Internet via the virtual appliance.

Type	Destination	Next Hop Type	Next Hop
System	vNet_Hub	vNet Peering	
User Defined	0.0.0.0	Virtual Appliance	10.104.1.1

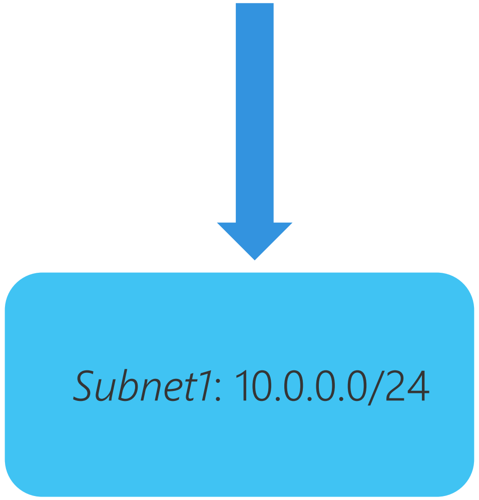
To configure UDRs, we first create a **Route Table**.

Type	Destination	Next Hop Type	Next Hop

We then create **routes**.

Type	Destination	Next Hop Type	Next Hop
User Defined	10.1.0.0/16	Virtual Appliance	10.104.1.1
User Defined	10.2.0.0/16	Virtual Appliance	10.104.1.1

Type	Destination	Next Hop Type	Next Hop
User Defined	10.1.0.0/16	Virtual Appliance	10.104.1.1
User Defined	10.2.0.0/16	Virtual Appliance	10.104.1.1

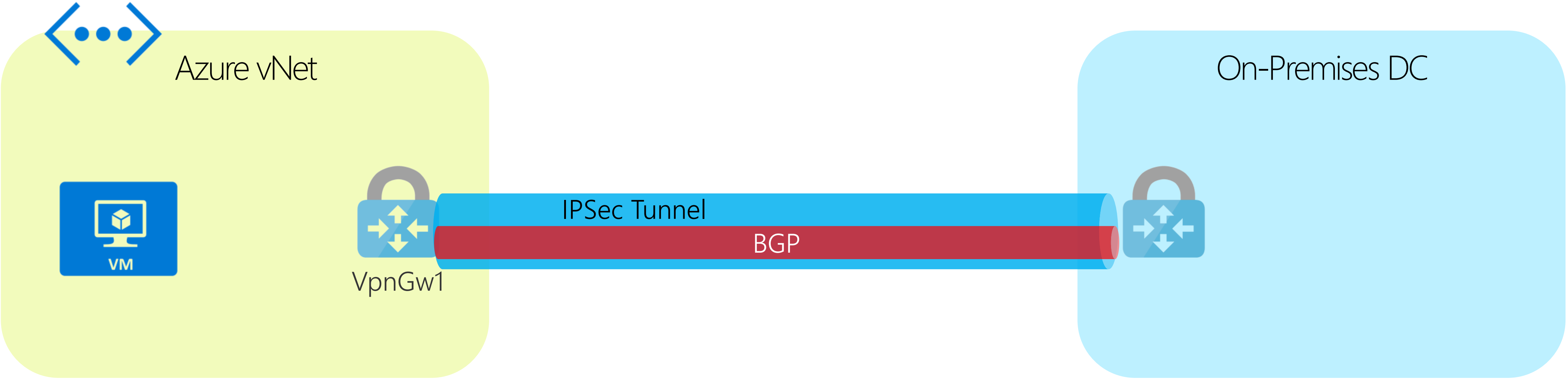


Subnet1: 10.0.0.0/24

Finally, we **associate** the route table with a **subnet**.

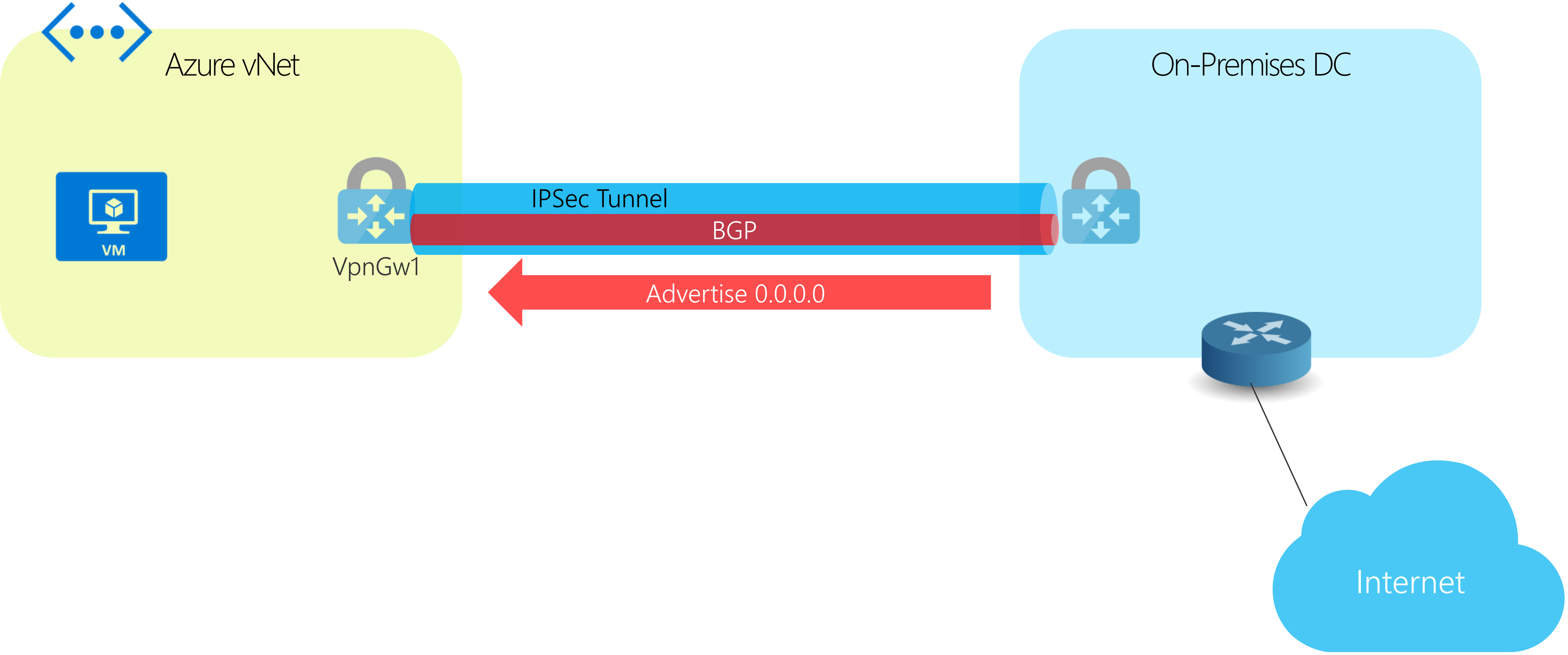
What about dynamic routing?
Is that supported?

BGP is supported on VPN Gateways*.

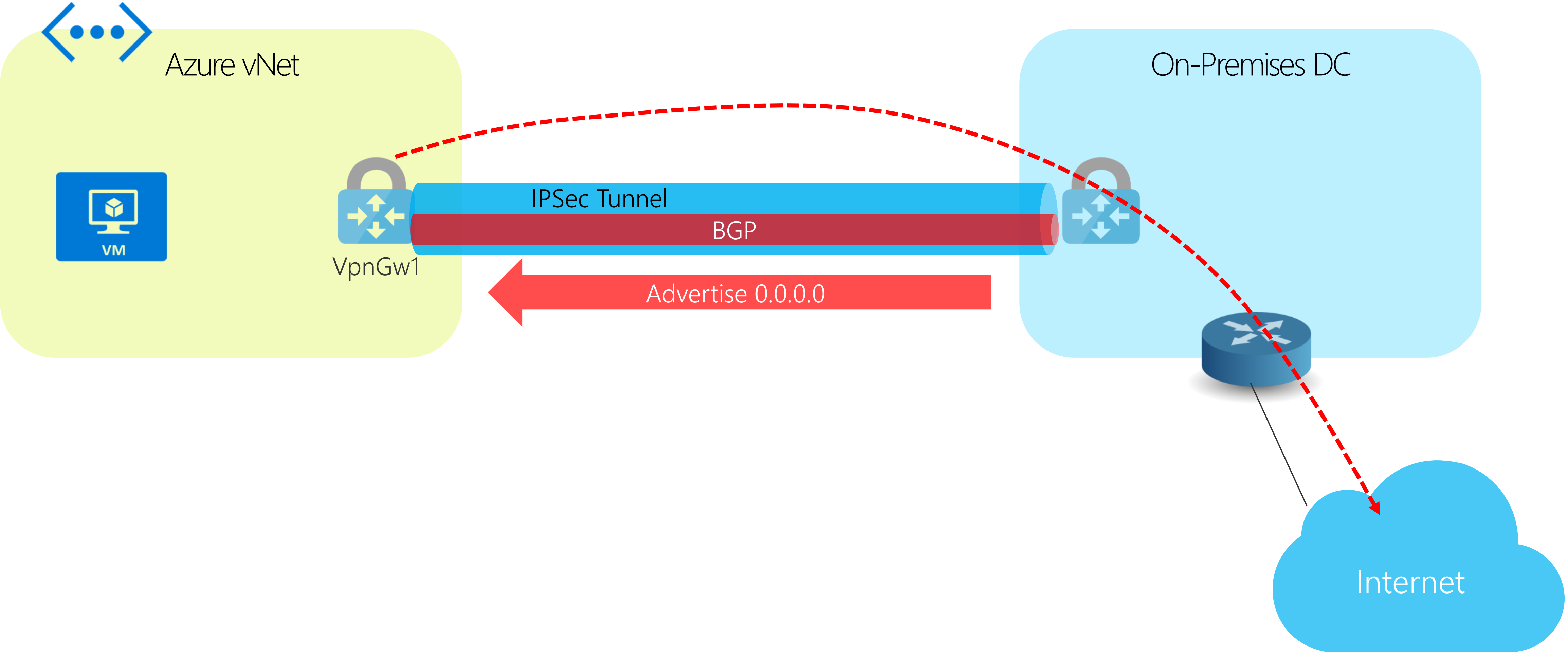


* Not on 'Basic' SKU or on 'policy-based' gateways

Advertising 0.0.0.0 to Azure forces all traffic via the VPN gateway.

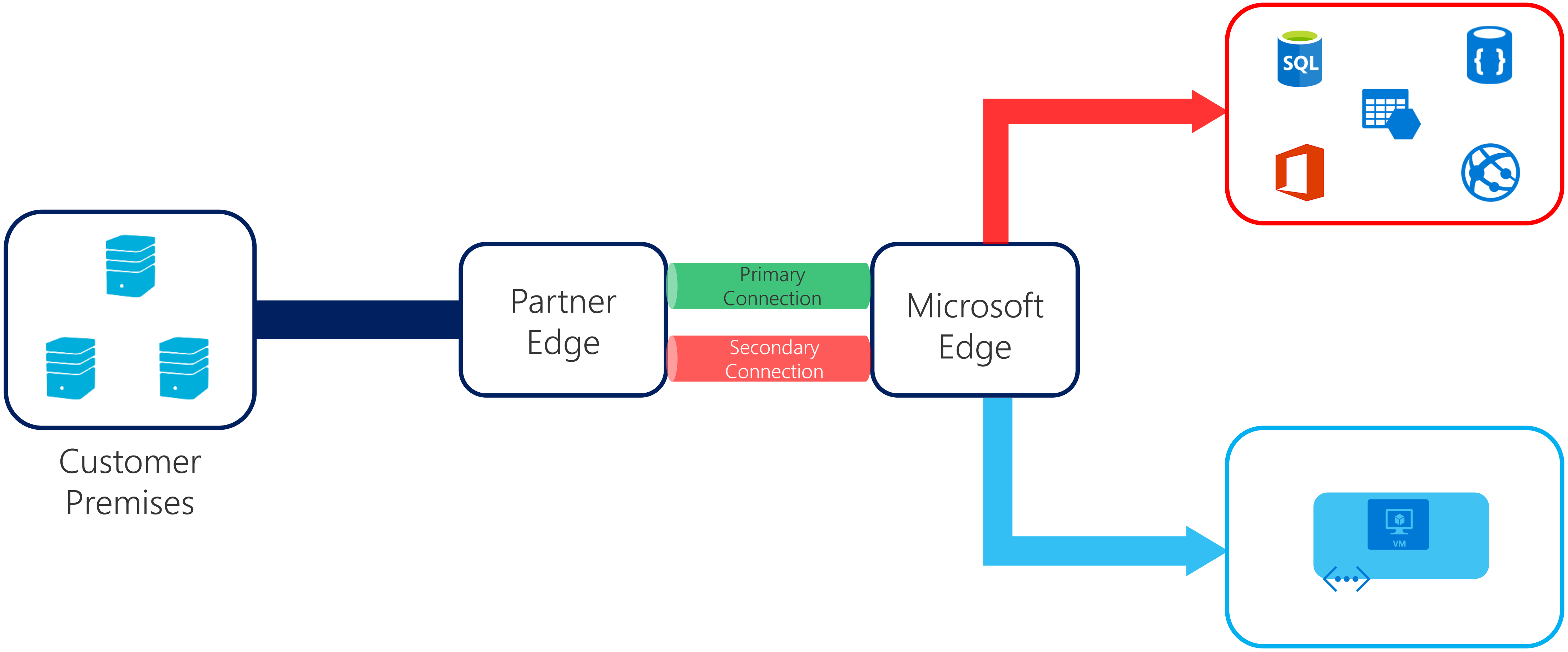


Advertising 0.0.0.0 to Azure forces all traffic via the VPN gateway.



Connecting to On-Premises Data Centres Using ExpressRoute

ExpressRoute extends On-Premises networks into the Microsoft Cloud using a *dedicated connection*.



Why ExpressRoute?

Predictable performance & latency

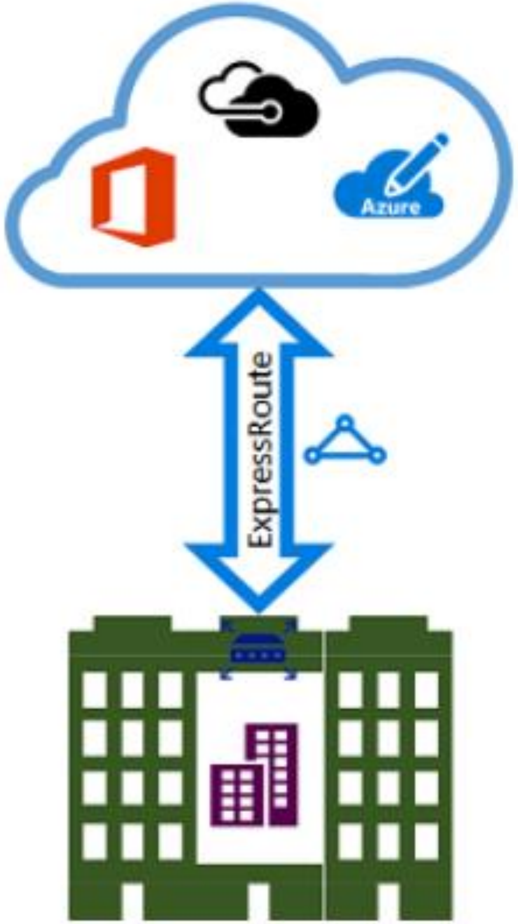
High throughput

Built-in redundancy

Three ER Connectivity Models

1. Co-located Cloud Exchange.

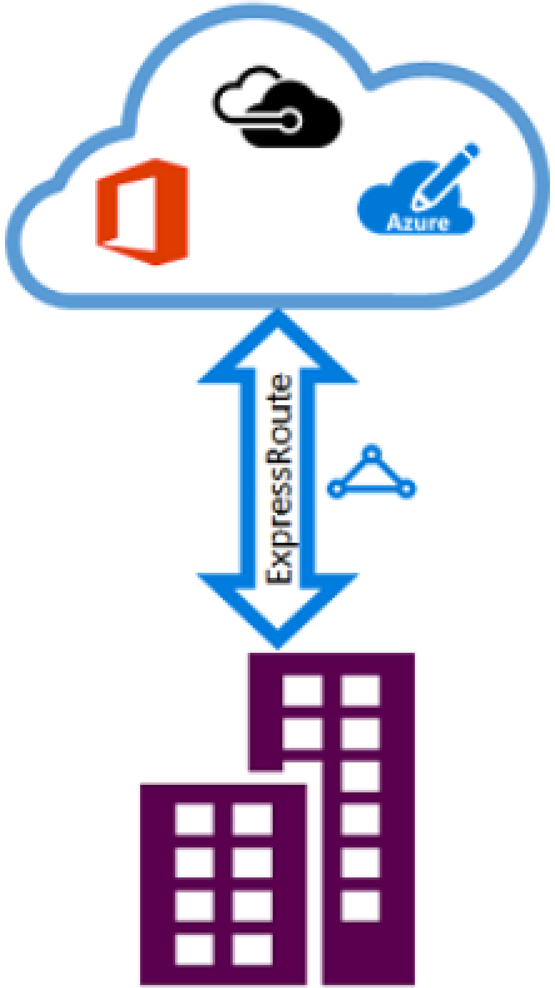
Virtual cross connect to MS cloud through provider's Ethernet exchange.



Three ER Connectivity Models

2. Point-to-point Ethernet.

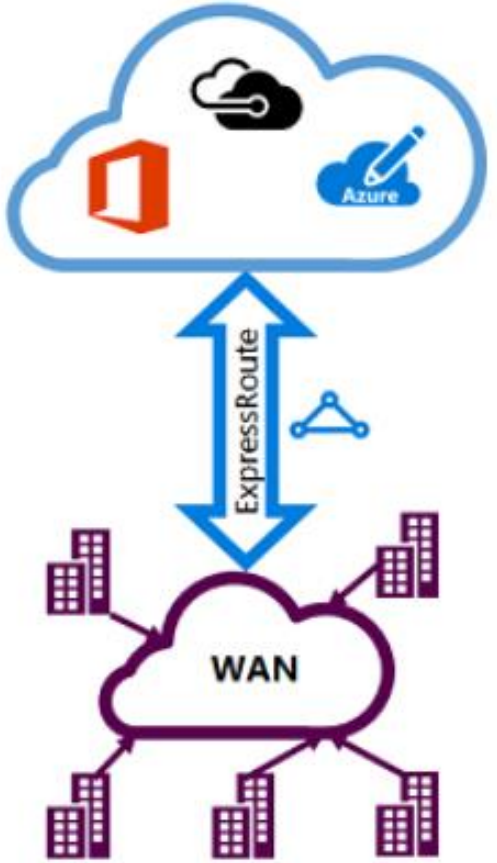
Connect on-premises datacenters/offices to the Microsoft cloud through point-to-point Ethernet links.



Three ER Connectivity Models

3. IPVPN (Any-to-Any).

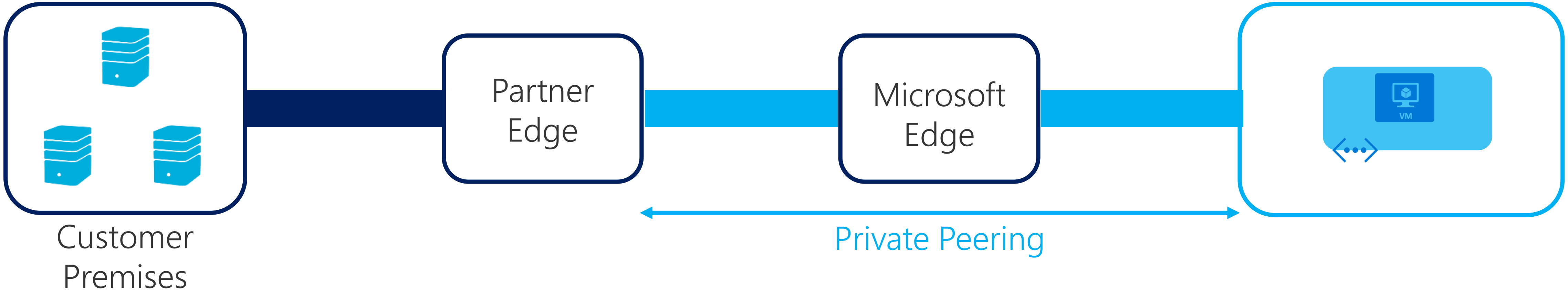
Integrate your WAN with the Microsoft cloud using any-to-any connectivity typically MPLS VPN.



Two Peering Types

1. Private Peering

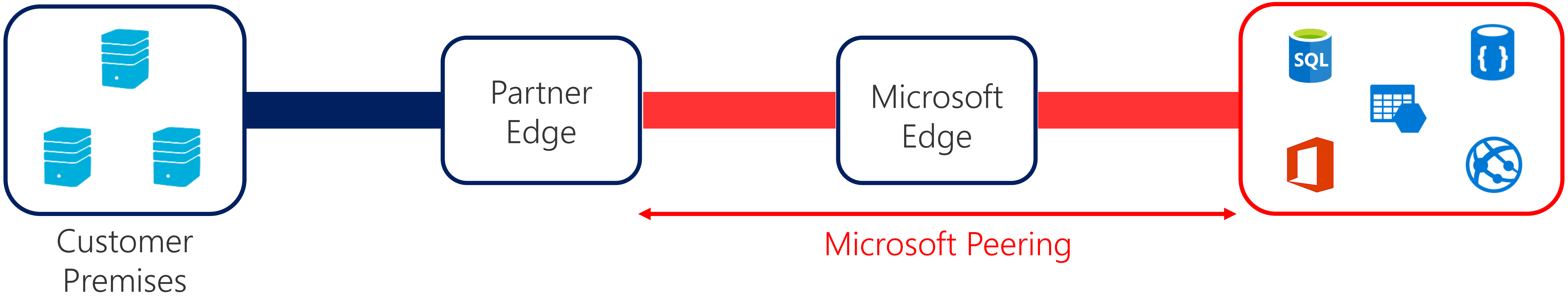
Connect to Azure services deployed within a virtual network.



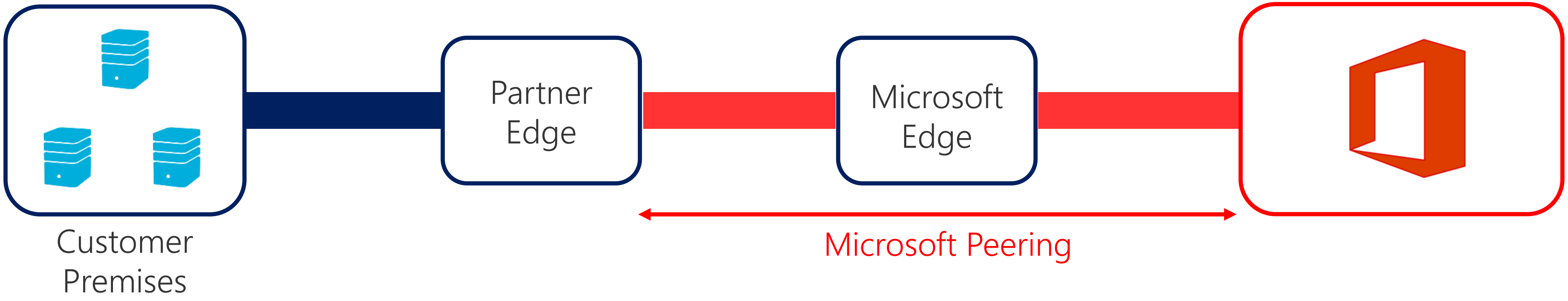
Two Peering Types

2. Microsoft Peering

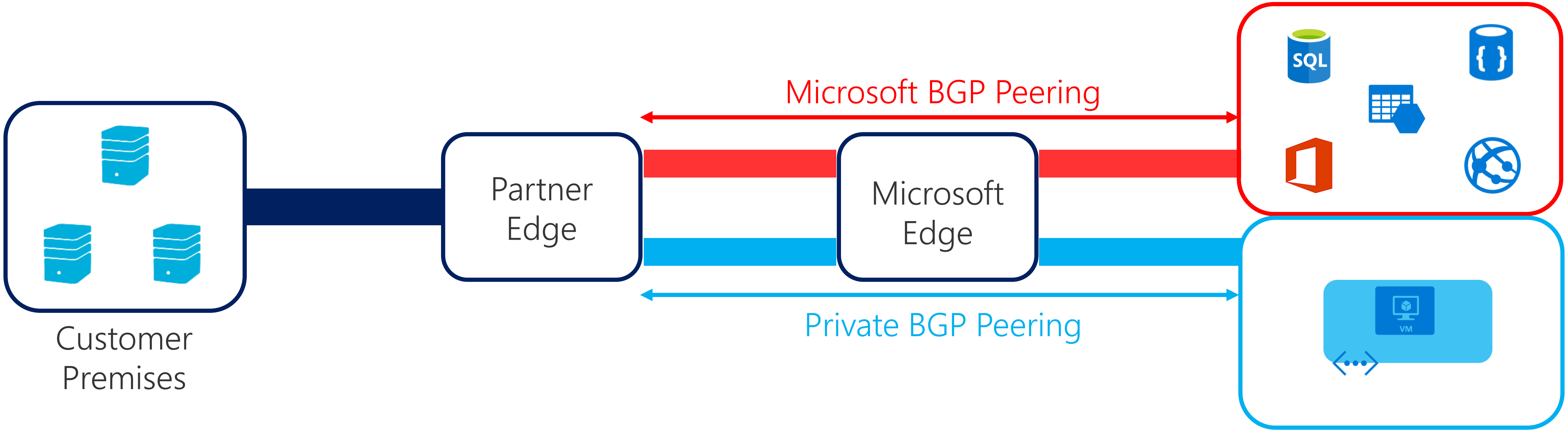
Connect to Azure PaaS services (storage, SQL, Web Apps, etc) and SaaS services (O365, etc).



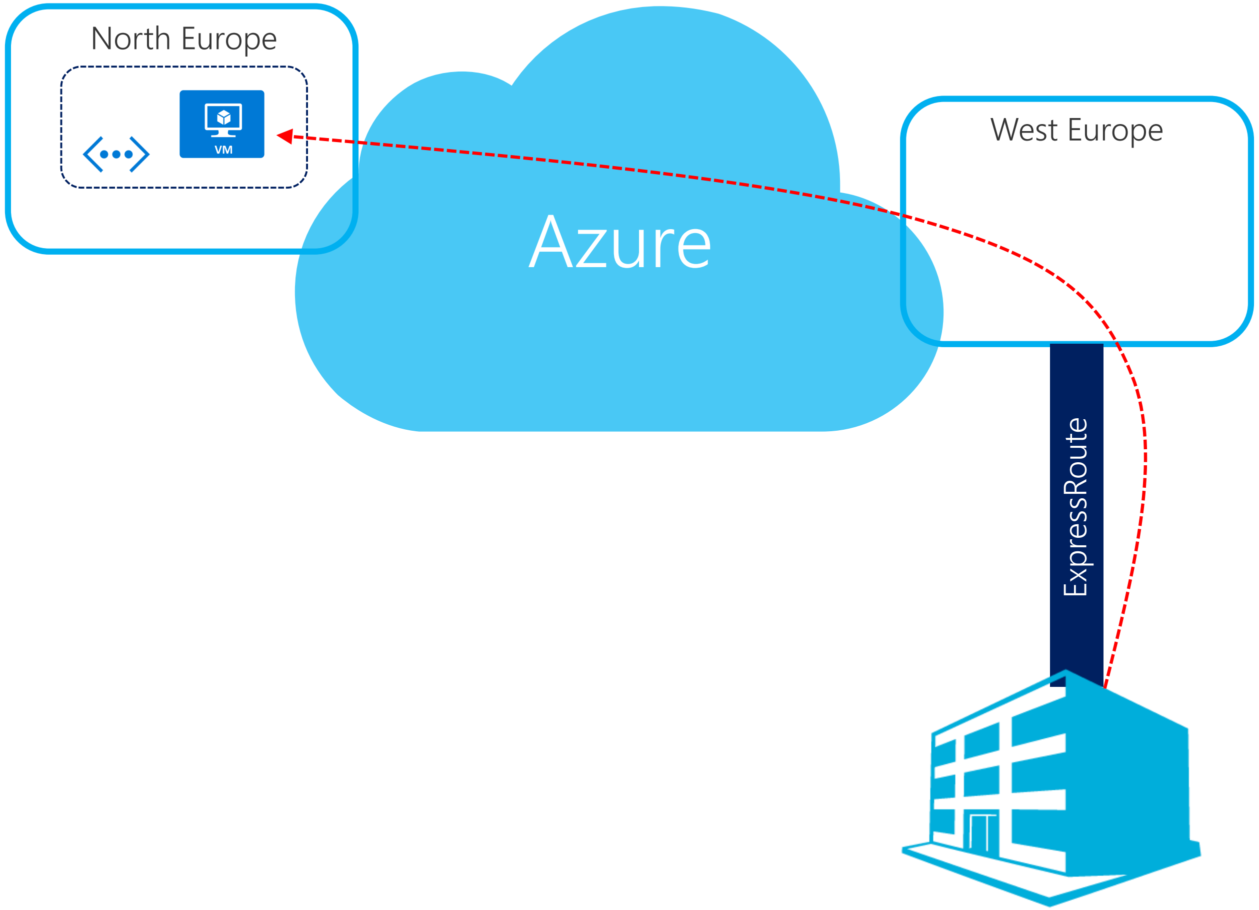
Important! Connecting to Office 365 via ExpressRoute Microsoft peering requires approval.



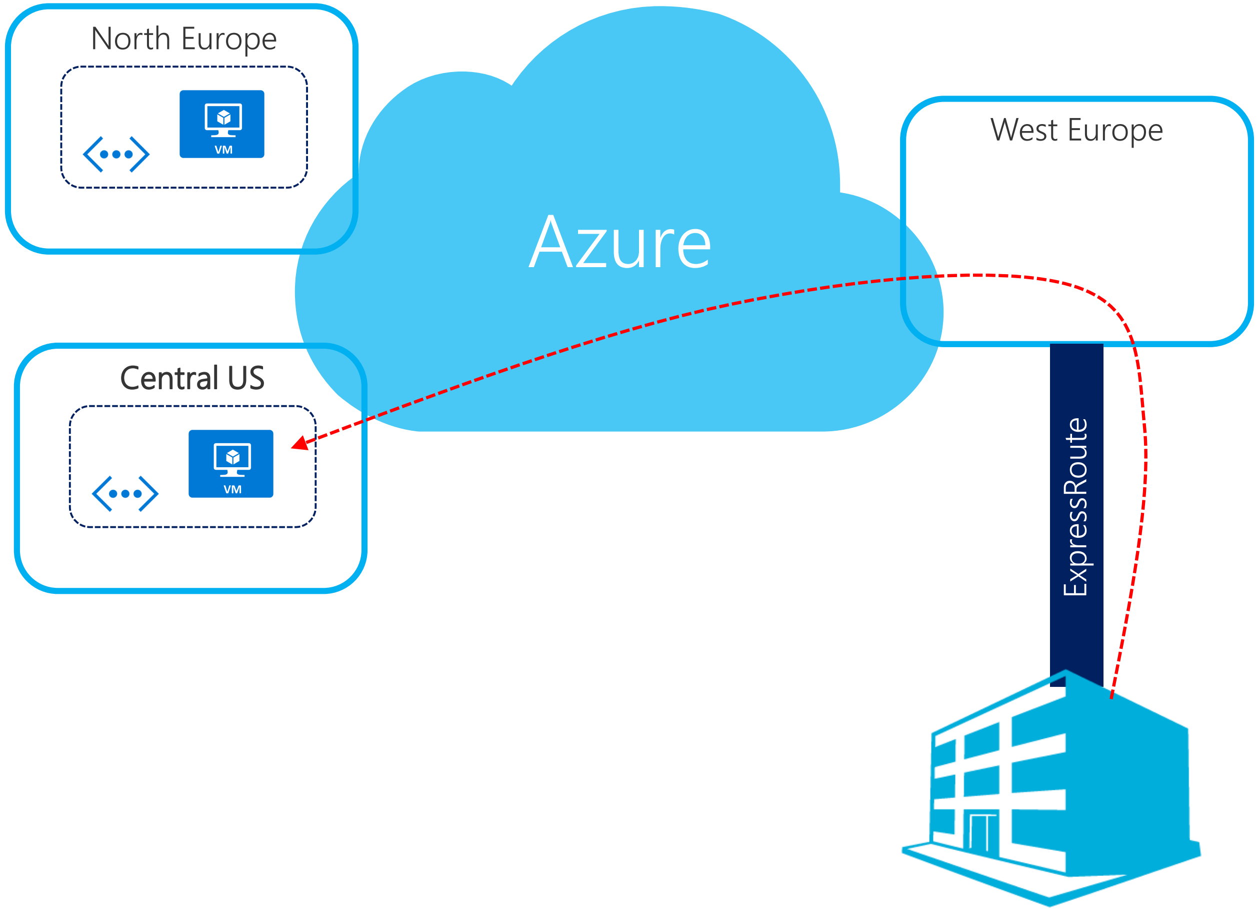
Each peering type requires a separate set of BGP peer sessions.



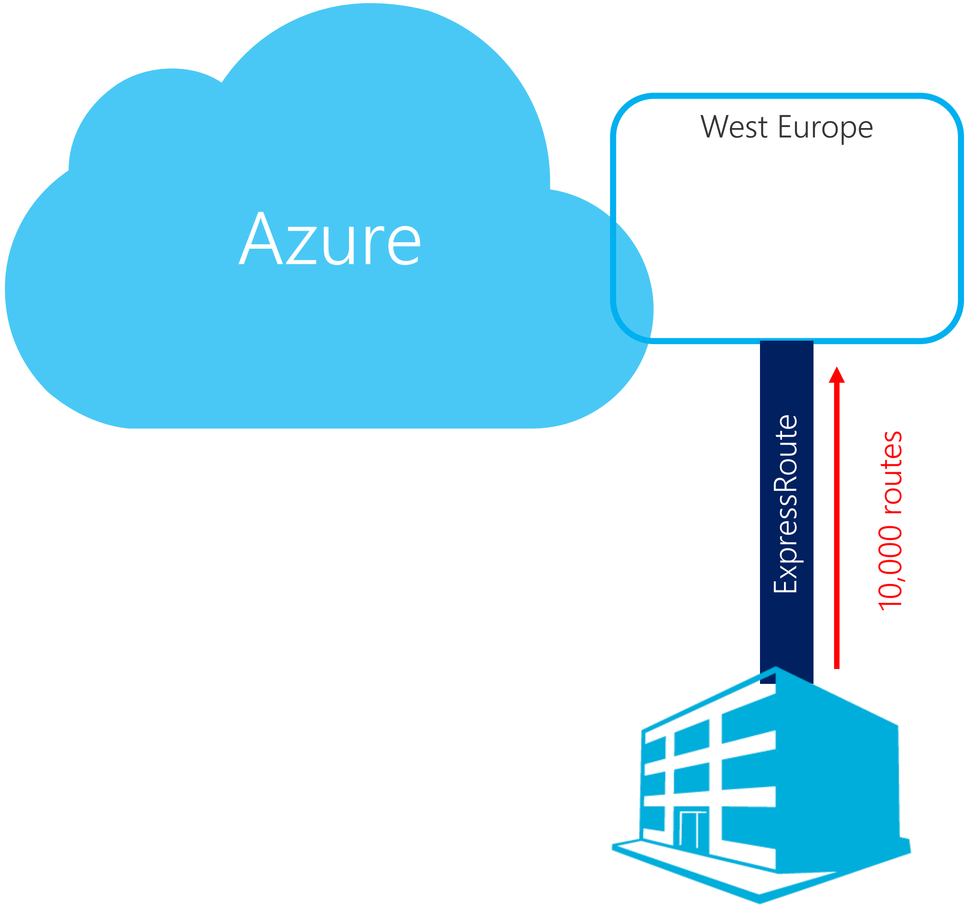
An ExpressRoute connection allows access to other regions within the same ‘geo-political area’.



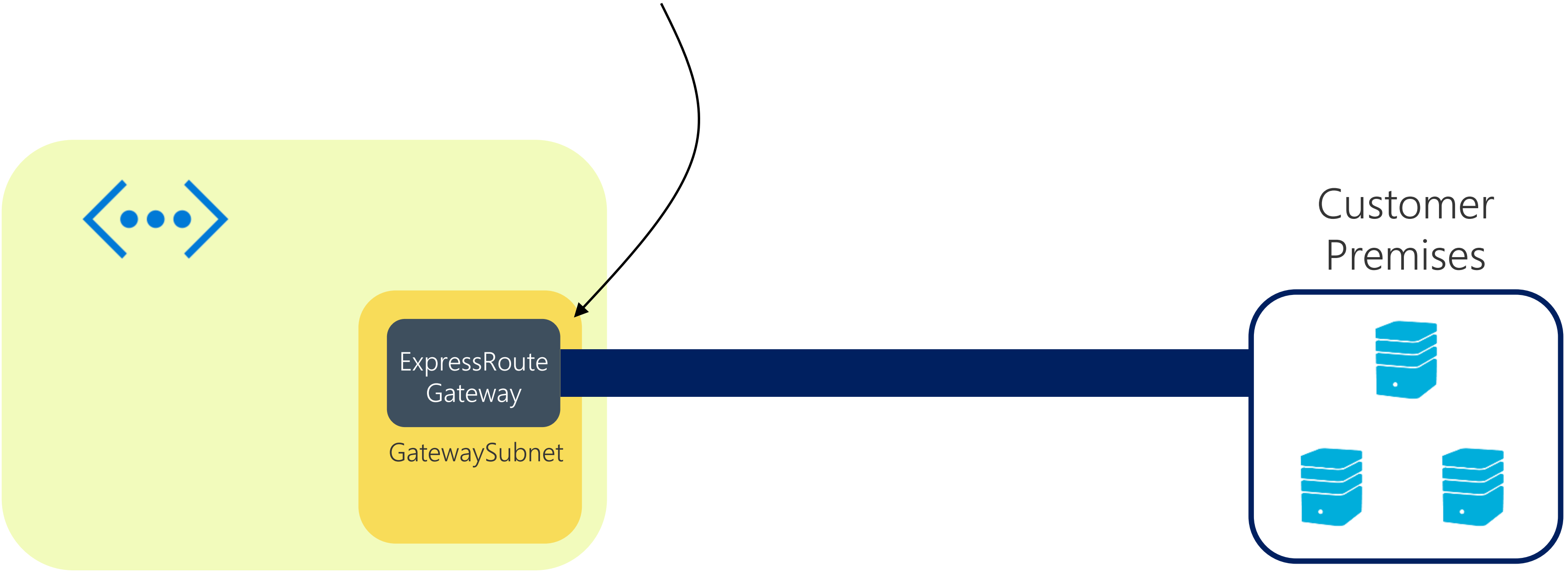
The ExpressRoute 'Premium' add-on enables connectivity across geo-political boundaries.



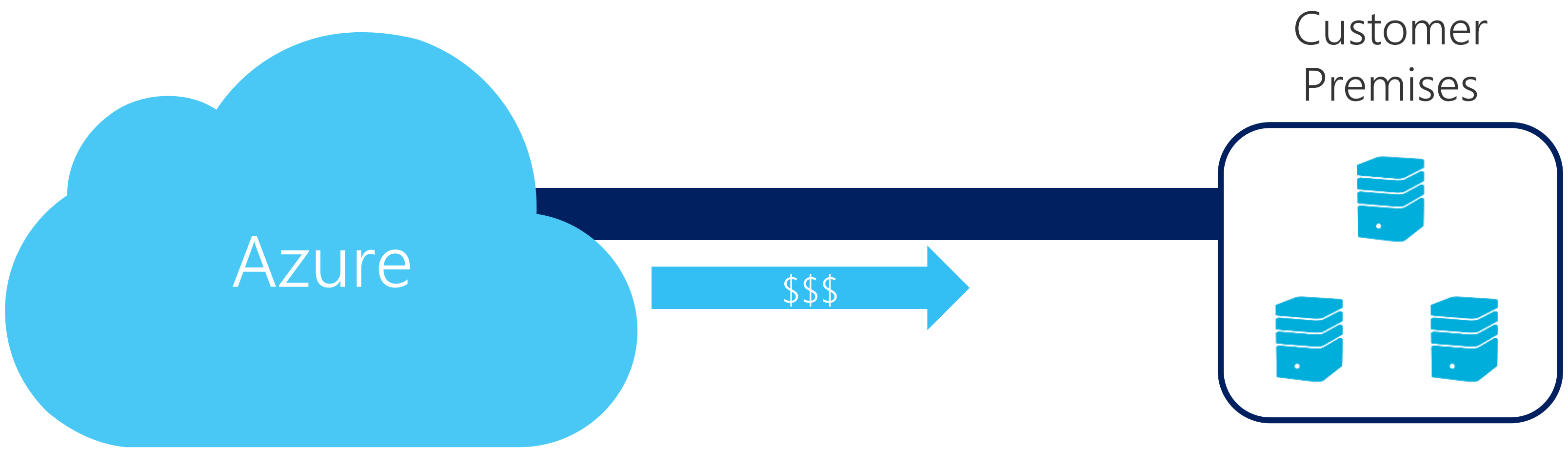
The 'Premium' add-on also gives you increased route limits (10,000).



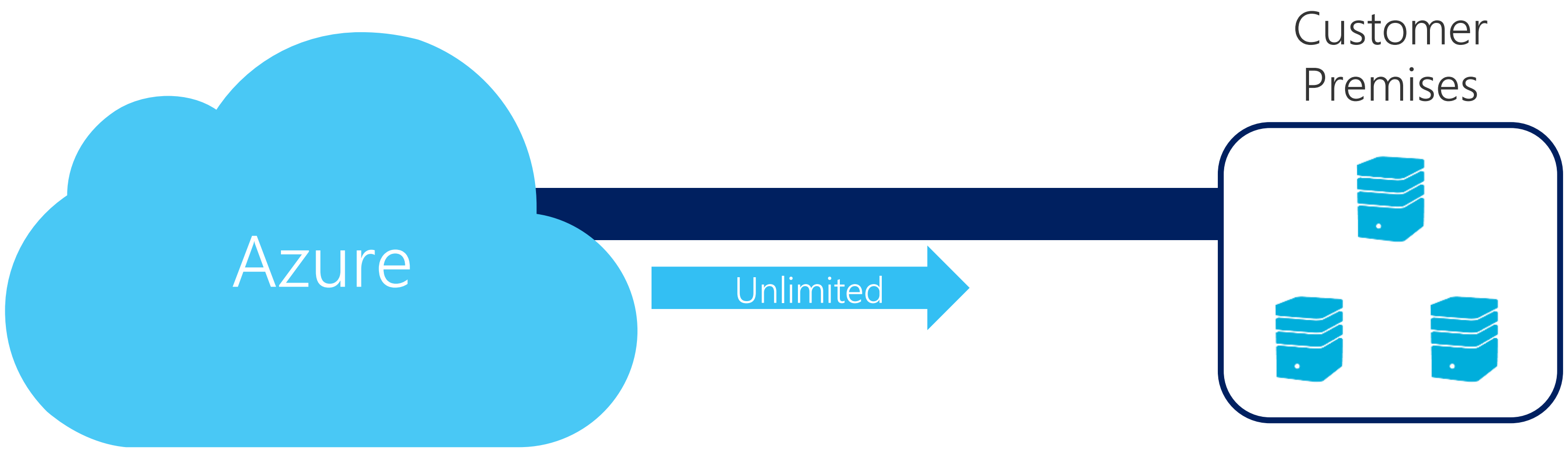
ExpressRoute uses a gateway that resides in the GatewaySubnet (similar to VPN gateways).



ExpressRoute has two types of billing: *Metered* and *Unmetered*.

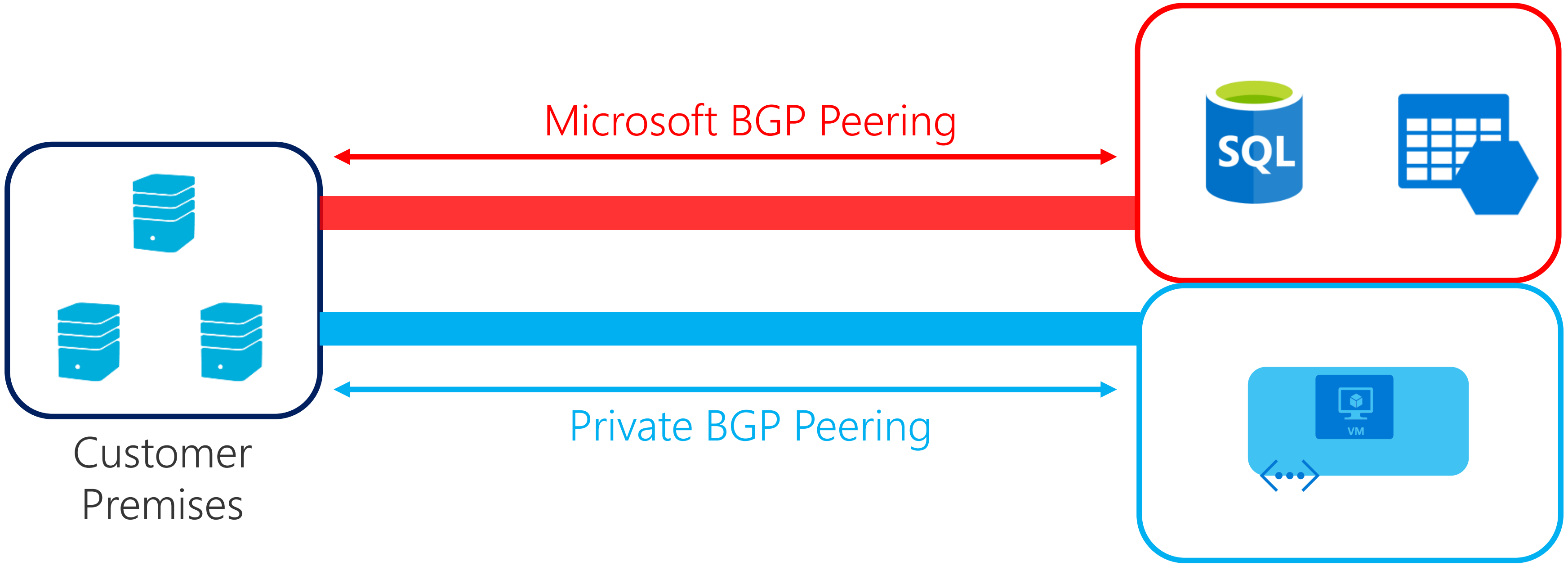


With Metered billing, you pay for outbound data transfers (per GB).

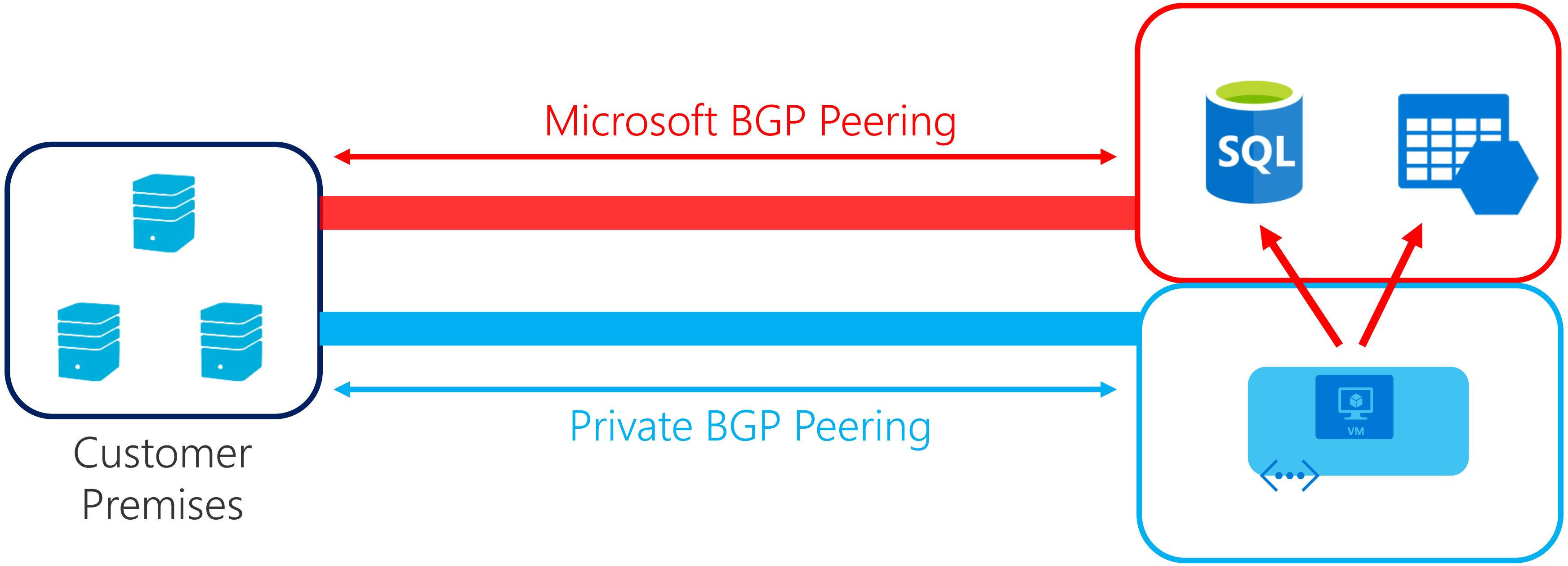


Unmetered billing gives you unlimited outbound data transfer (but of course is more expensive in the first place).

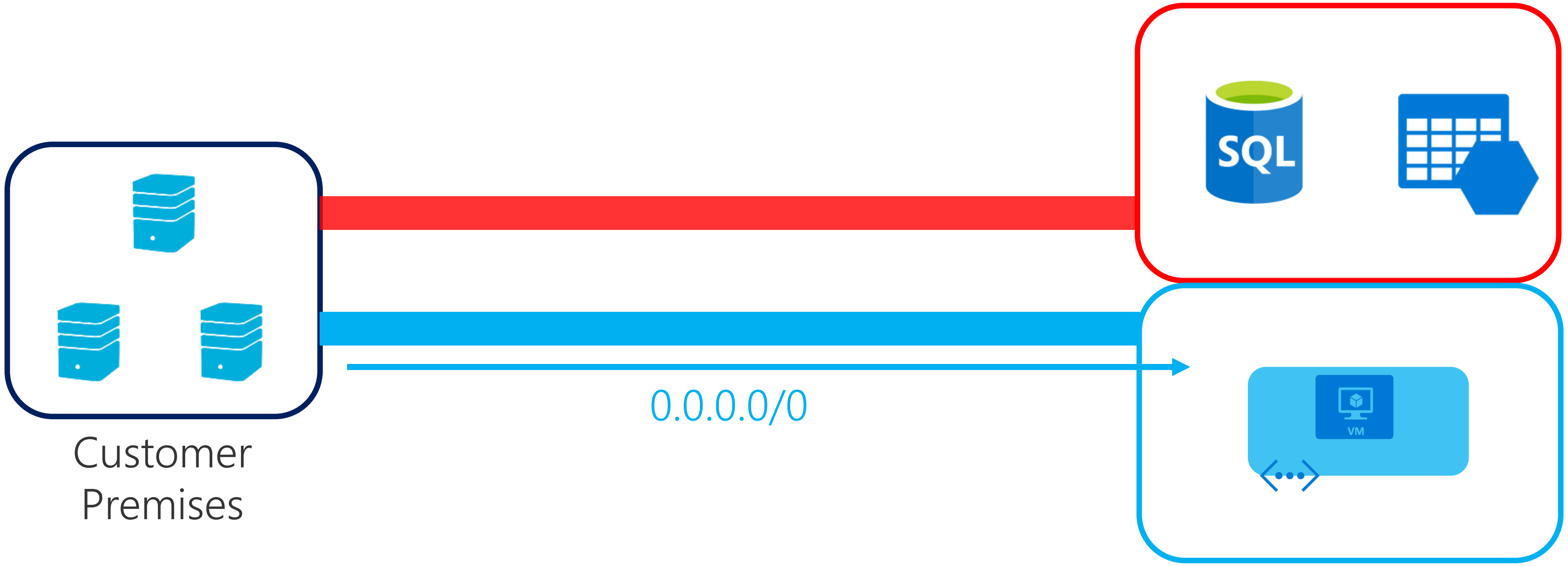
Let’s imagine a scenario where our Azure VMs need to communicate with public Azure services (e.g. Azure Storage or Azure SQL).



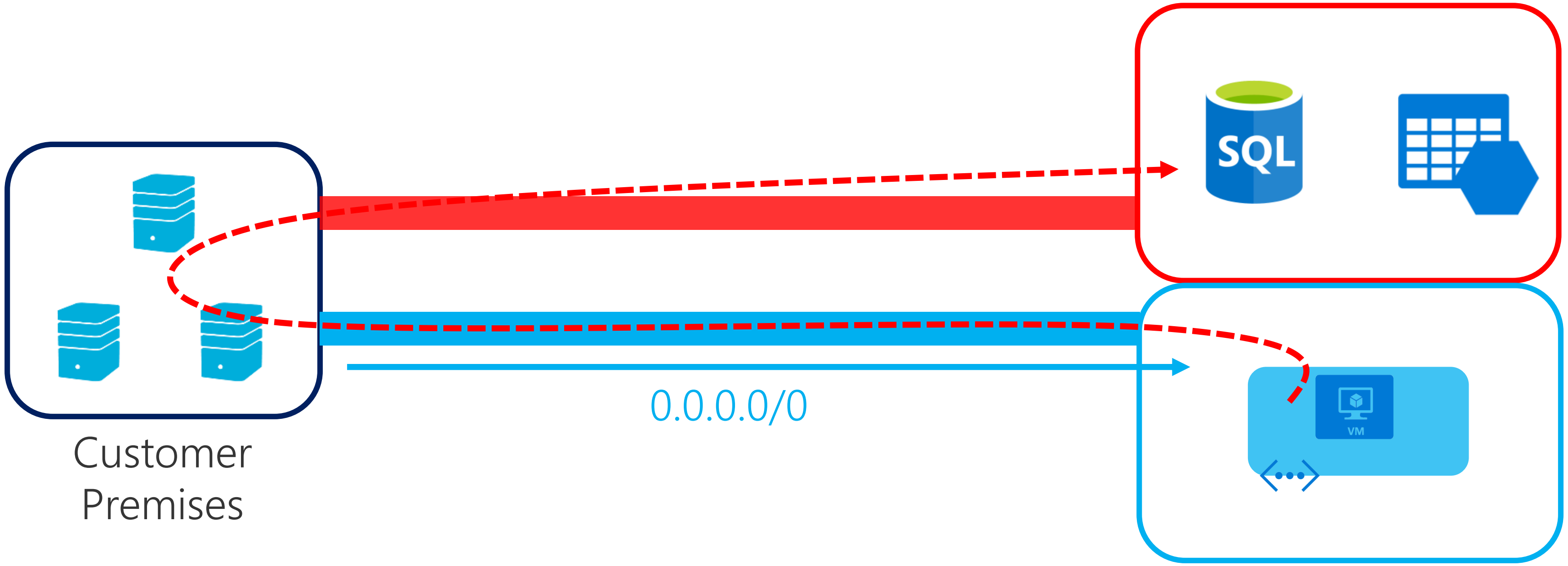
In most circumstances, traffic will be kept on the Microsoft network (i.e. it won’t traverse the ExpressRoute circuit).

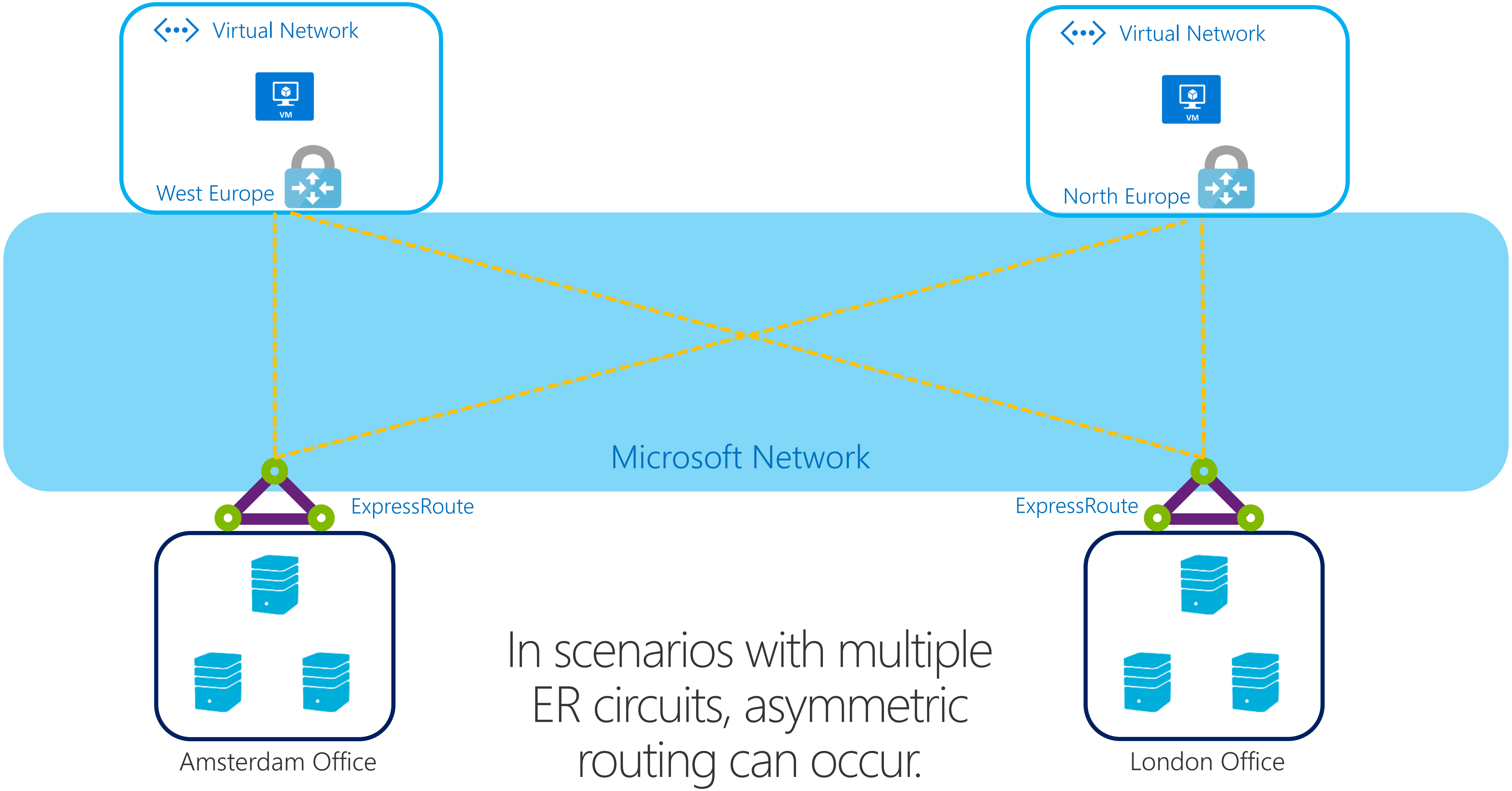


However, some customers might choose to advertise a default route to Azure via the private peering (to force all traffic via on-premises).



In that case, all traffic between your vNet and Azure public services will *hairpin via on-premises*.

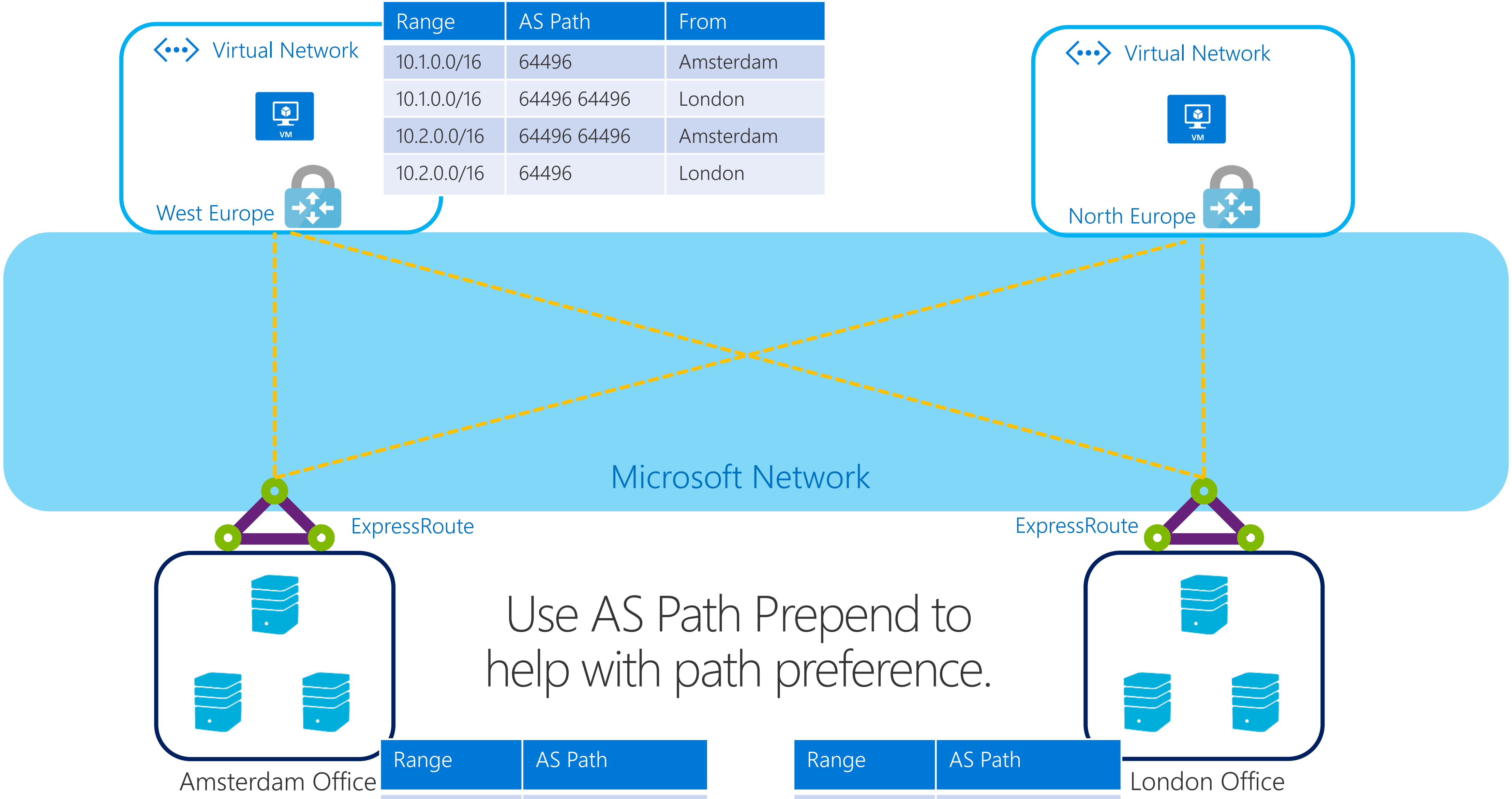




In scenarios with multiple ER circuits, asymmetric routing can occur.

West Europe Routing Table

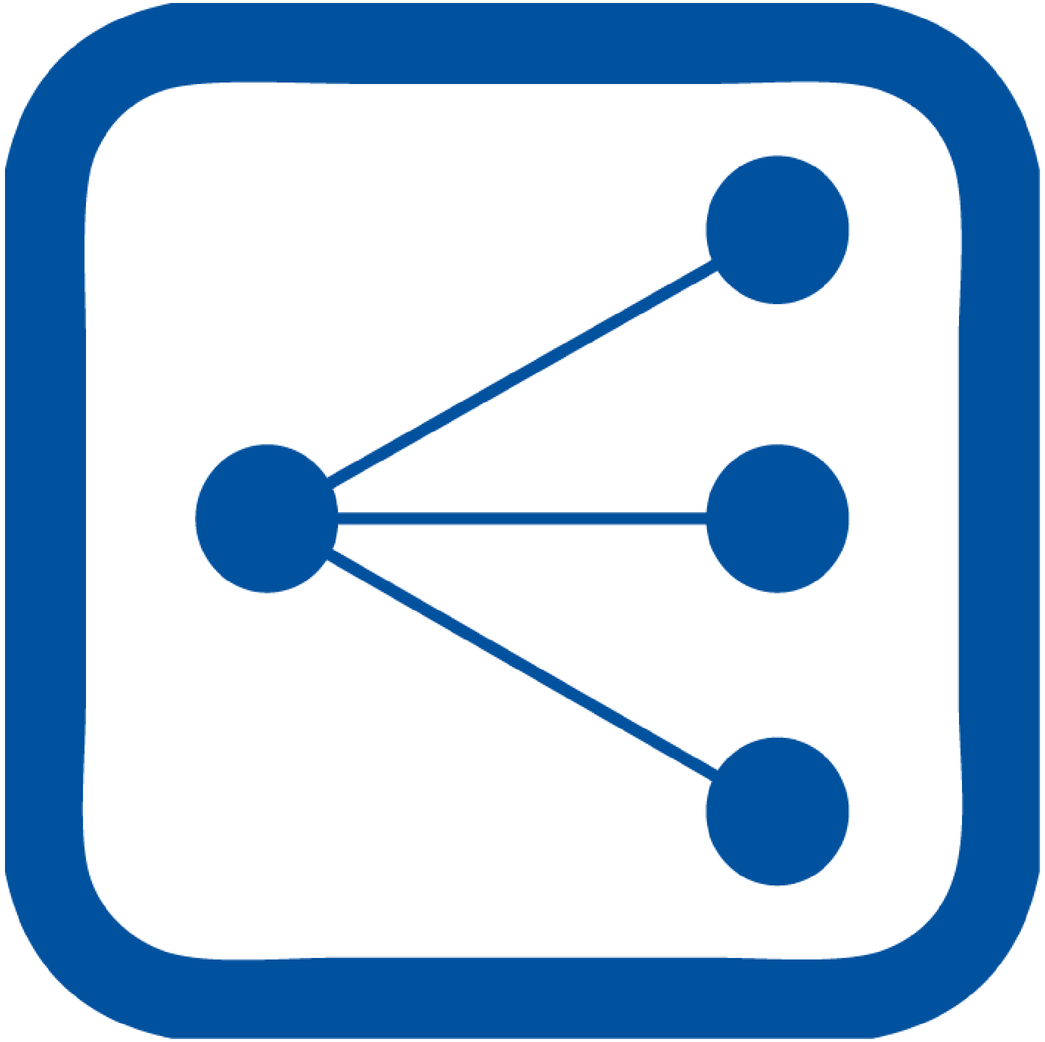
Range	AS Path	From
10.1.0.0/16	64496	Amsterdam
10.1.0.0/16	64496 64496	London
10.2.0.0/16	64496 64496	Amsterdam
10.2.0.0/16	64496	London



Use AS Path Prepend to help with path preference.

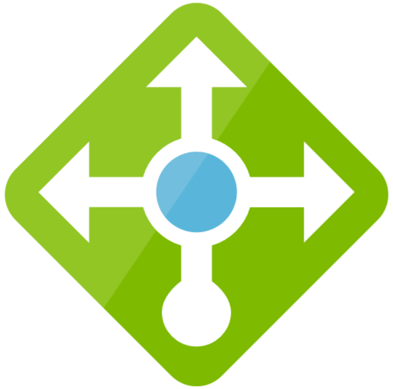
Range	AS Path
10.1.0.0/16	64496
10.2.0.0/16	64496 64496

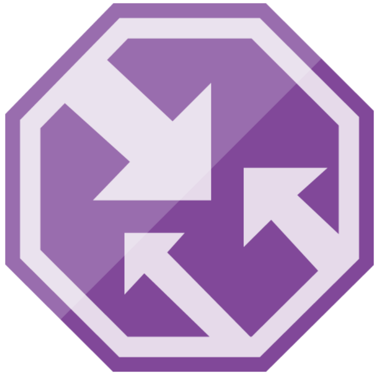
Range	AS Path
10.1.0.0/16	64496 64496
10.2.0.0/16	64496



Load Balancing in Azure

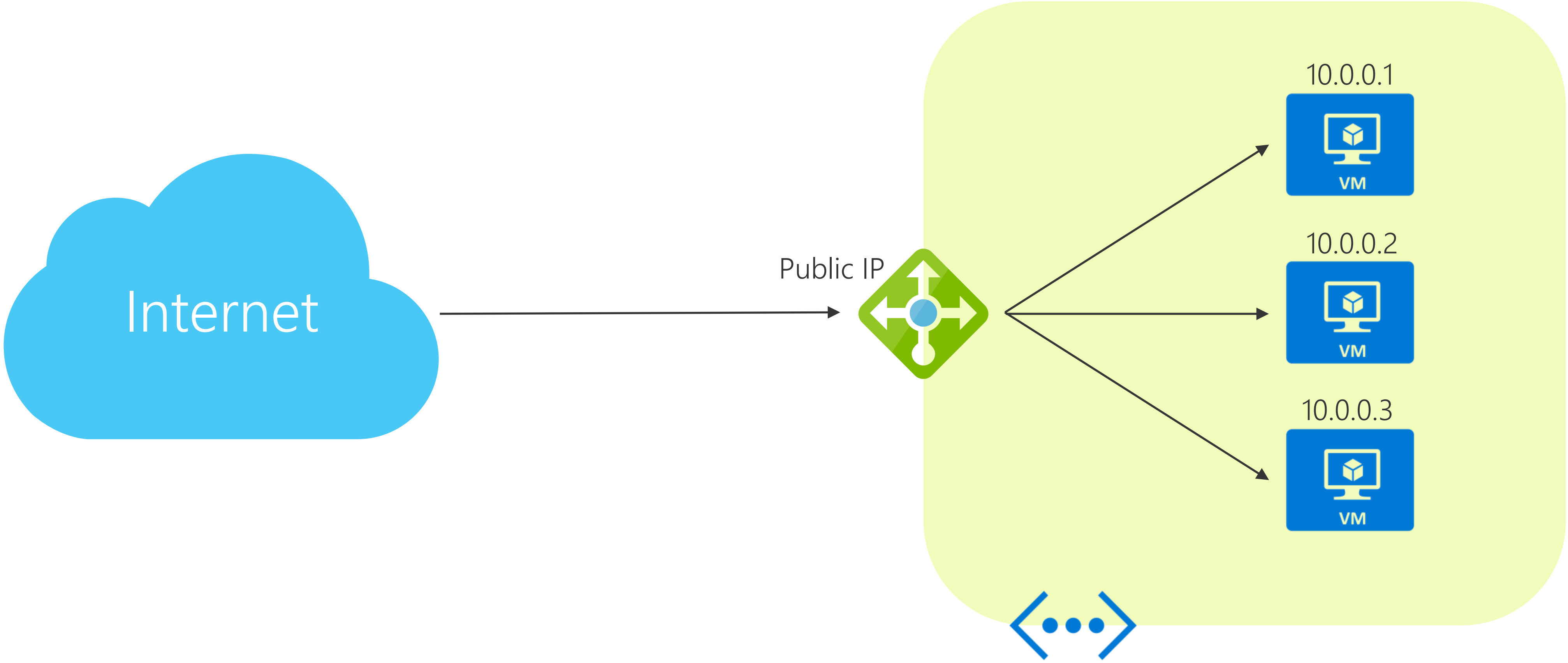
Azure has a few different types of load balancer available.

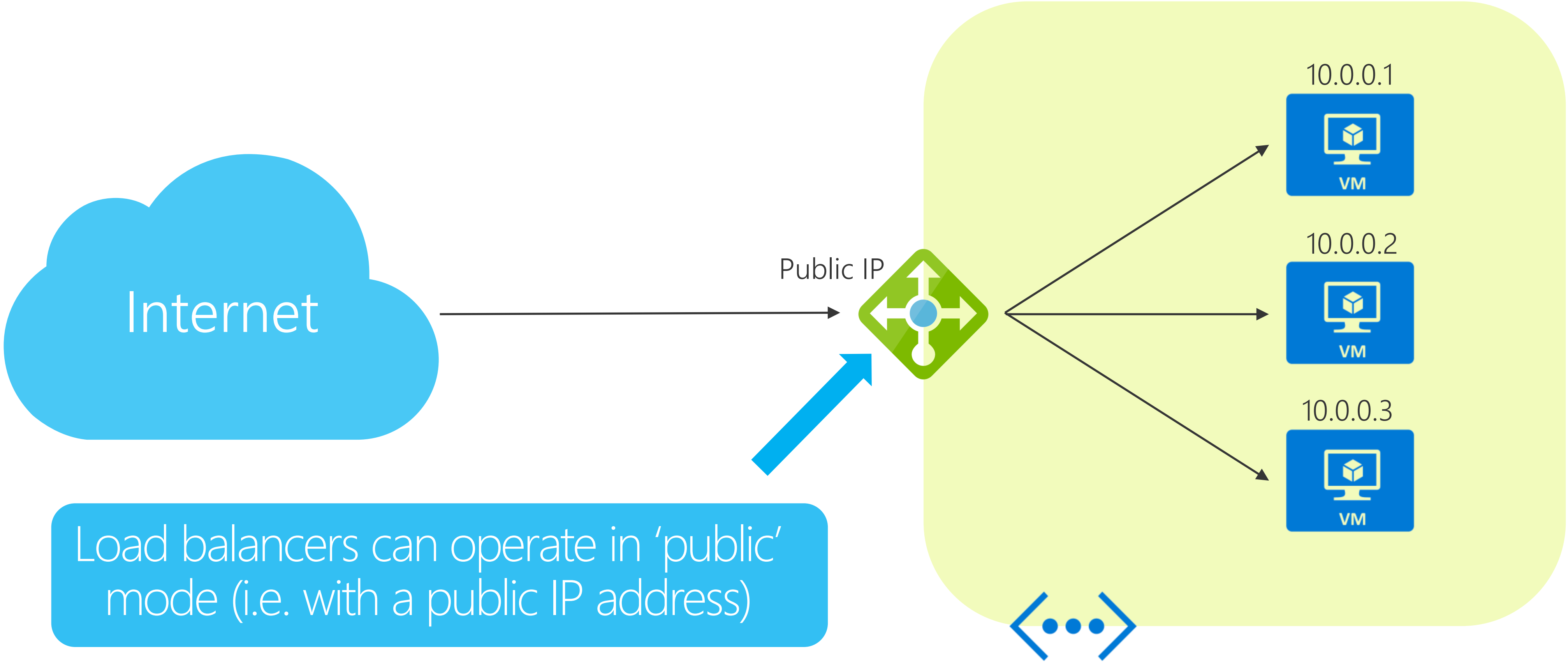




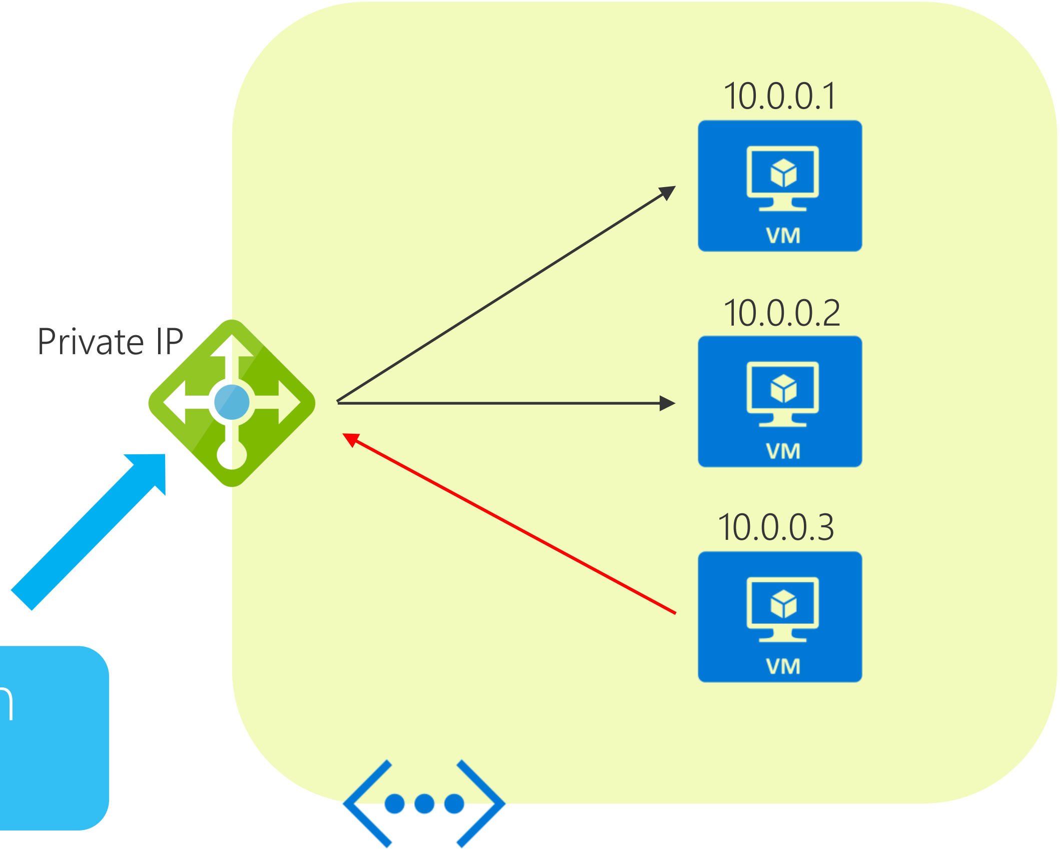
First, let’s look at the basic / standard load balancer.

The Azure load balancer offers basic L4 load balancing.

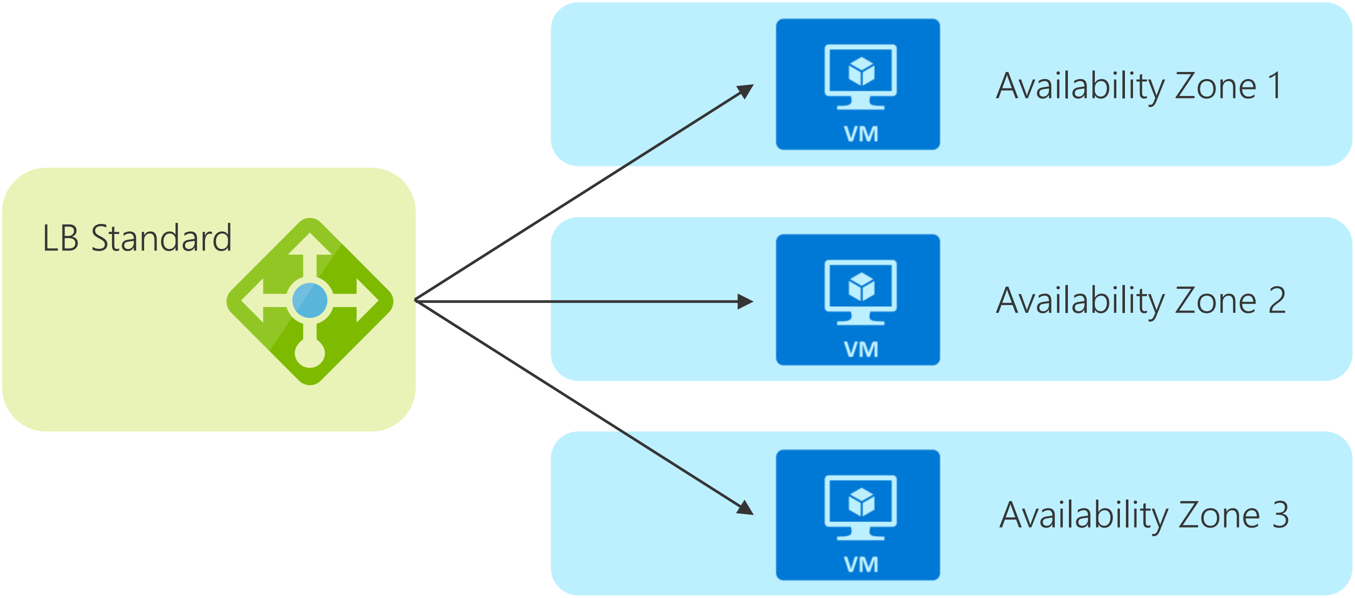




Load balancers can operate in ‘public’ mode (i.e. with a public IP address)



Load balancers can also operate in ‘internal’ mode (i.e. private IP).



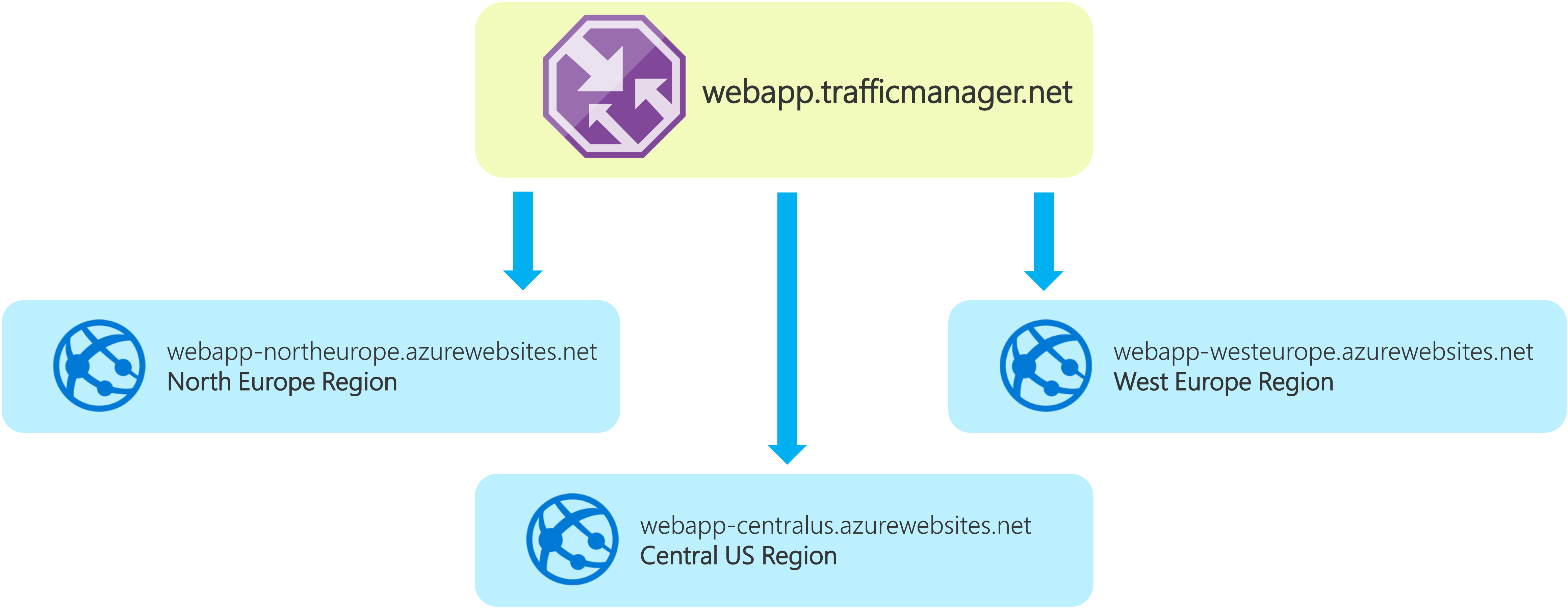
The 'Standard SKU' load balancer adds support for zone redundancy.

Azure Application Gateway is a fully featured application delivery controller working at layer 7.

Web Application Firewall
HTTP load balancing
Cookie based session affinity
SSL offload

Multi-site routing
URL based content routing
Advanced diagnostics
Azure Web App support

Traffic manager is a DNS based global load balancing service.



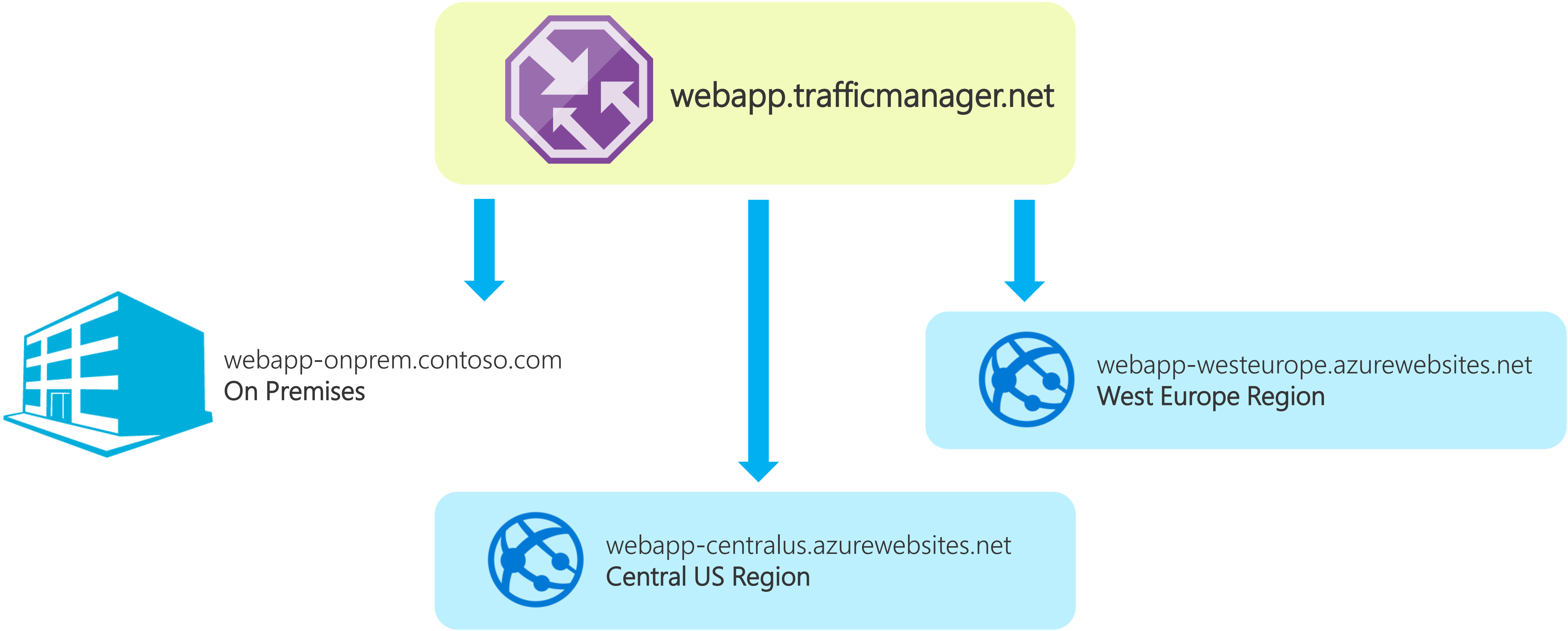
webapp.trafficmanager.net

webapp-northeurope.azurewebsites.net
North Europe Region

webapp-westeurope.azurewebsites.net
West Europe Region

webapp-centralus.azurewebsites.net
Central US Region

Traffic manager can also work with non-Azure based end points.



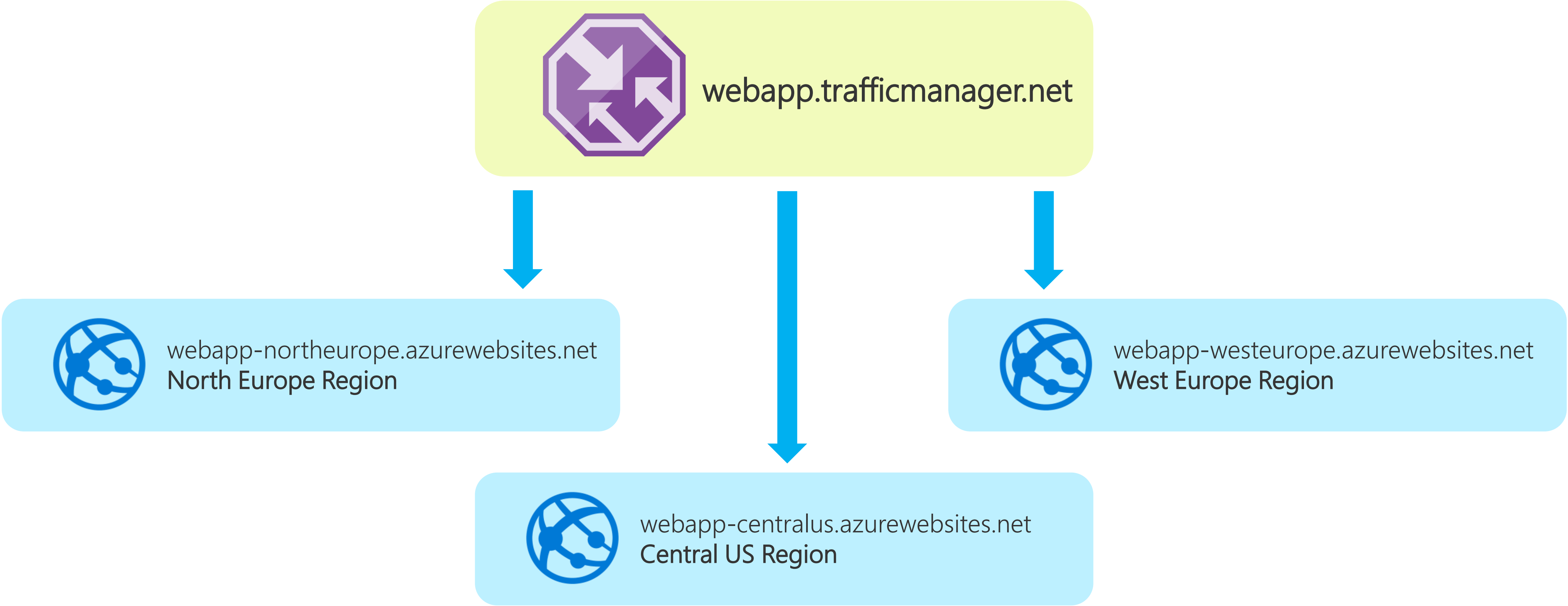
webapp.trafficmanager.net

webapp-onprem.contoso.com
On Premises

webapp-westeurope.azurewebsites.net
West Europe Region

webapp-centralus.azurewebsites.net
Central US Region

Traffic Manager has four routing methods.



webapp.trafficmanager.net

webapp-northeurope.azurewebsites.net
North Europe Region

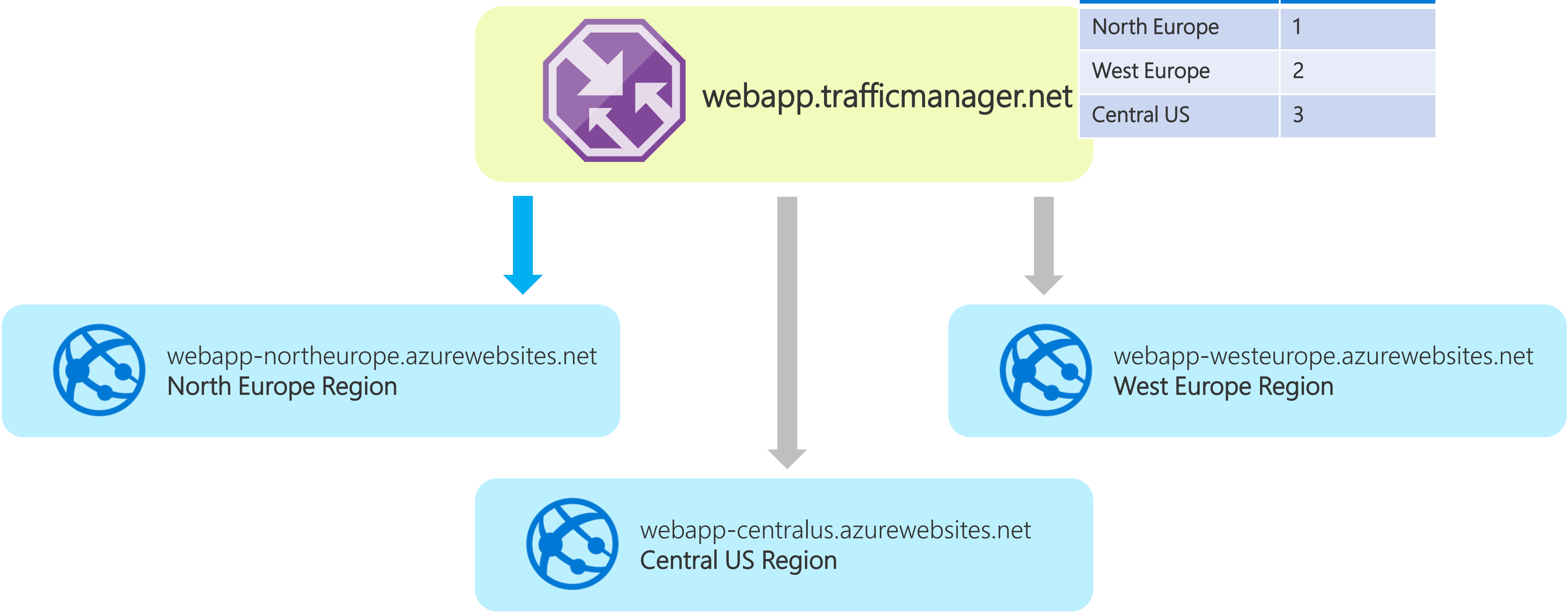
webapp-westeurope.azurewebsites.net
West Europe Region

webapp-centralus.azurewebsites.net
Central US Region

1. Priority Routing

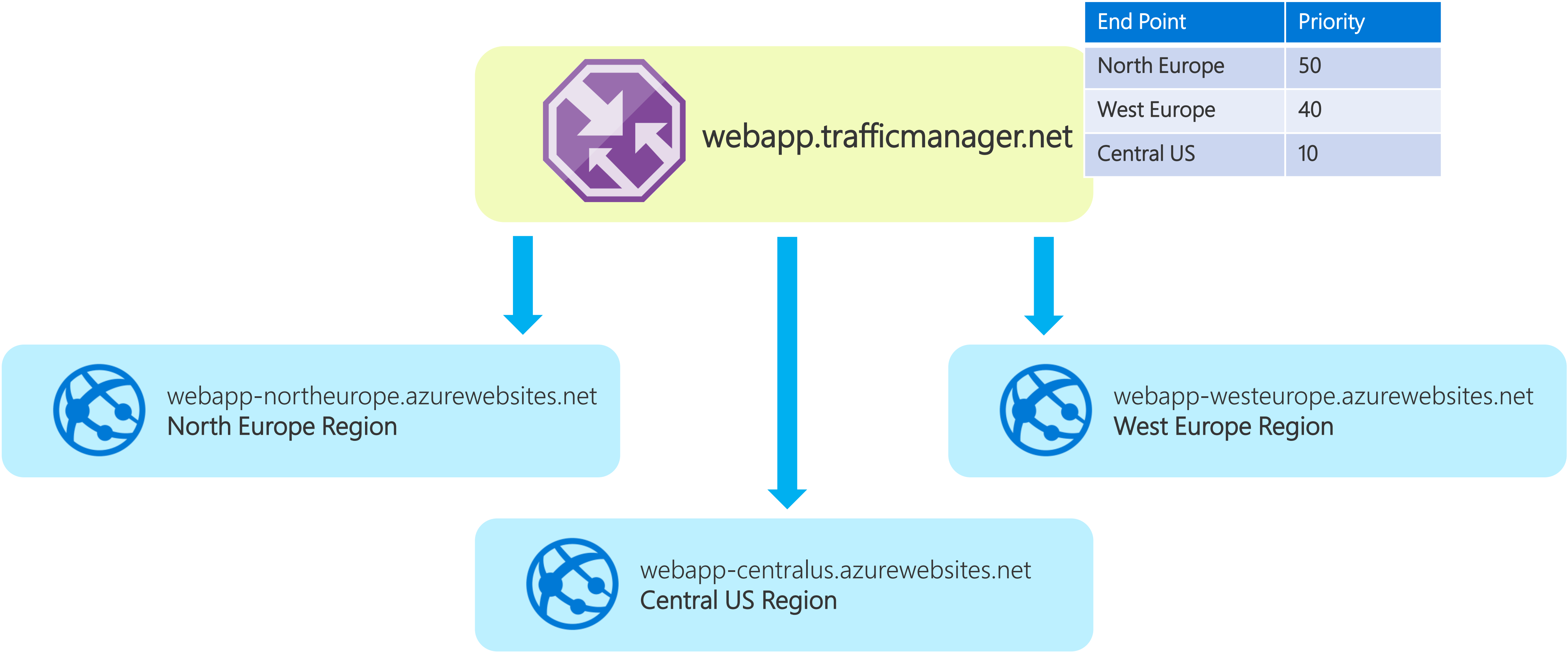
Use a primary endpoint for traffic; failover when primary is unavailable.

End Point	Priority
North Europe	1
West Europe	2
Central US	3


webapp.trafficmanager.net
webapp-northeurope.azurewebsites.net
North Europe Region
webapp-westeurope.azurewebsites.net
West Europe Region
webapp-centralus.azurewebsites.net
Central US Region

2. Weighted Routing

Distribute traffic according to weights.



End Point	Priority
North Europe	50
West Europe	40
Central US	10

webapp.trafficmanager.net

webapp-northeurope.azurewebsites.net
North Europe Region

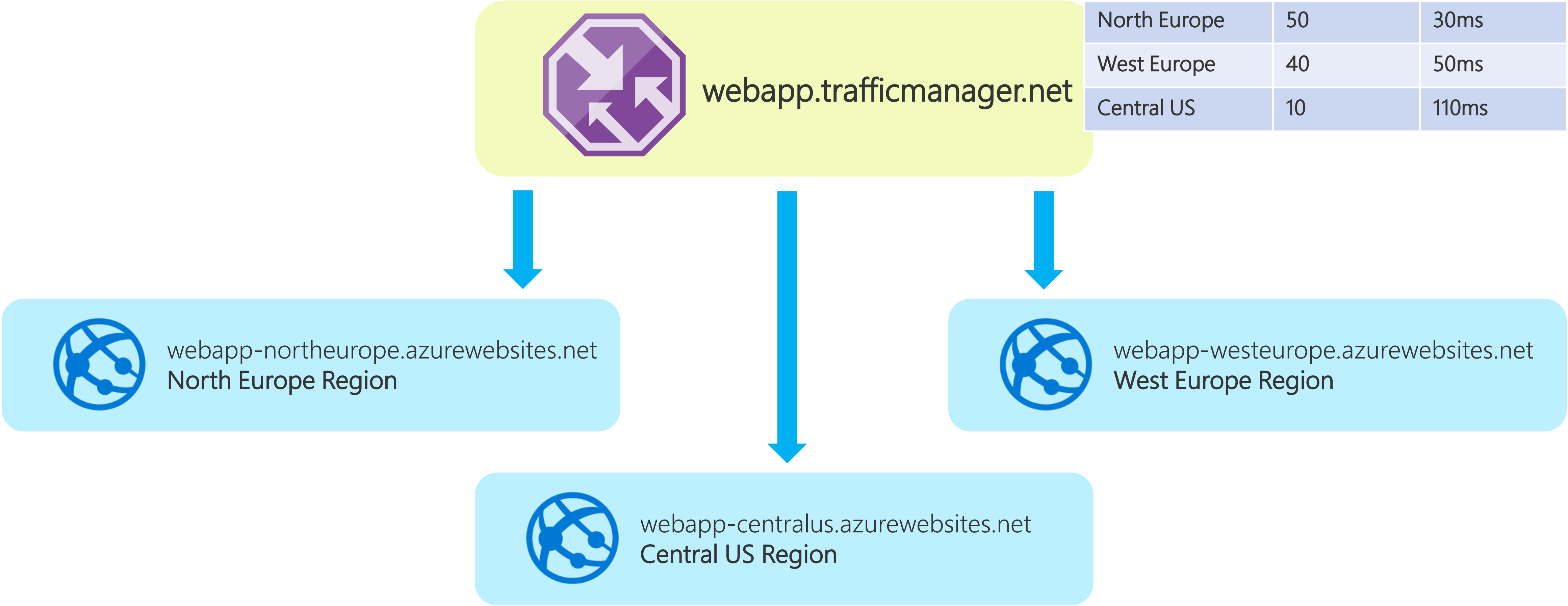
webapp-westeurope.azurewebsites.net
West Europe Region

webapp-centralus.azurewebsites.net
Central US Region

3. Performance Routing

Direct users to the 'closest' end point, based on latency.

End Point	Priority	Latency
North Europe	50	30ms
West Europe	40	50ms
Central US	10	110ms



webapp.trafficmanager.net

webapp-northeurope.azurewebsites.net
North Europe Region

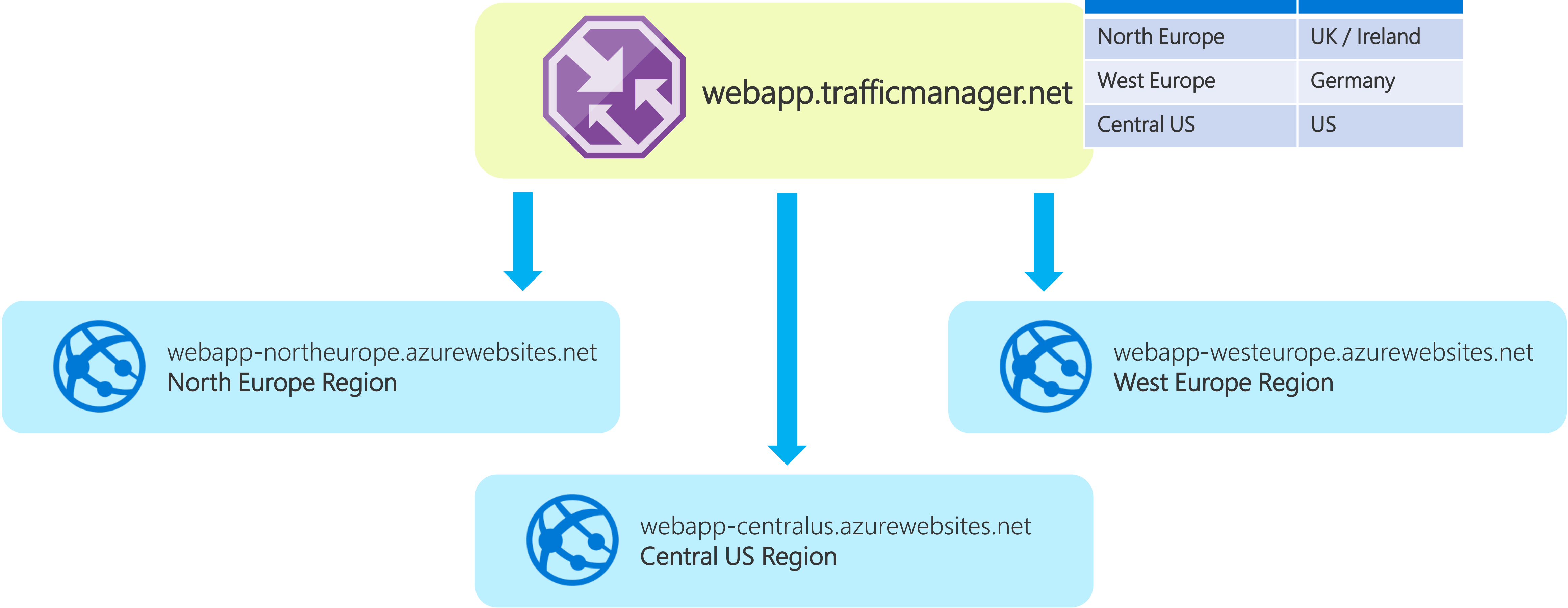
webapp-westeurope.azurewebsites.net
West Europe Region

webapp-centralus.azurewebsites.net
Central US Region

4. Geographic Routing

Direct users based on the geographic location the DNS query came from.

End Point	Assigned Geo
North Europe	UK / Ireland
West Europe	Germany
Central US	US



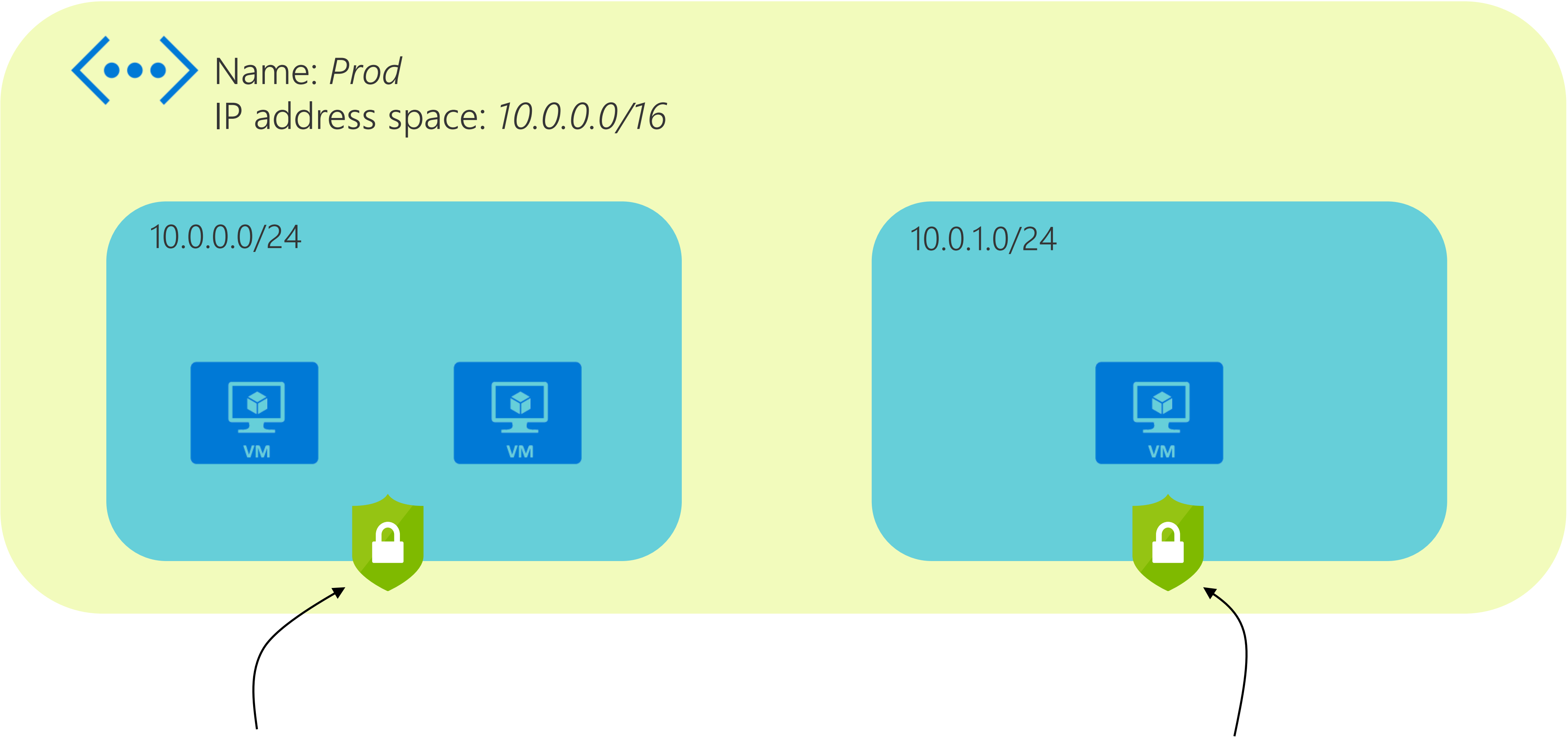
webapp.trafficmanager.net

webapp-northeurope.azurewebsites.net
North Europe Region

webapp-westeurope.azurewebsites.net
West Europe Region

webapp-centralus.azurewebsites.net
Central US Region

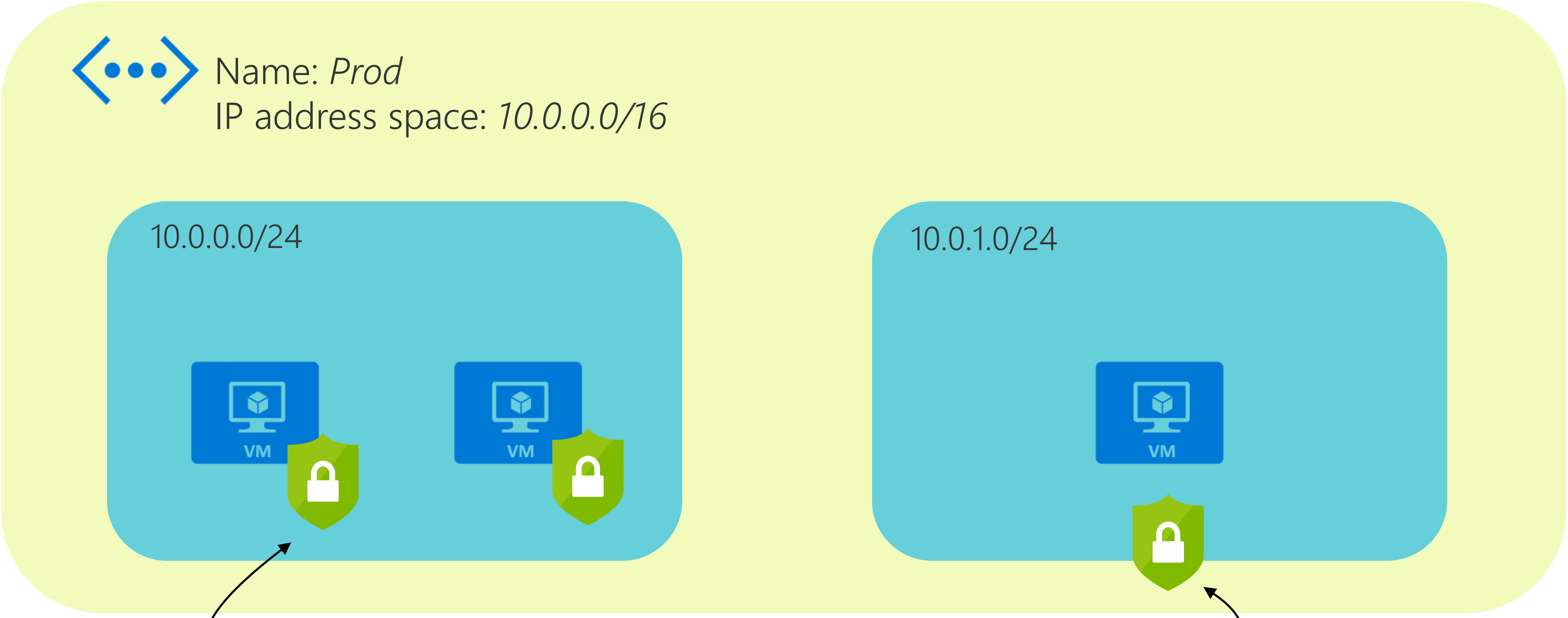
Network Security Groups



Network Security Groups (NSGs) lock down access to a subnet or VM.

An NSG is essentially a list of access control rules, permitting or denying traffic based on various criteria.

Source IP	Dest IP	Protocol	Port	Action
10.1.1.2	30.1.1.5	TCP	22	Allow
10.1.1.2	*	TCP	80	Allow
10.1.1.2	30.1.1.6	TCP	443	Allow
*	*	*	*	Deny



The NSG can be applied either at the VM (NIC) level, or at the subnet level.

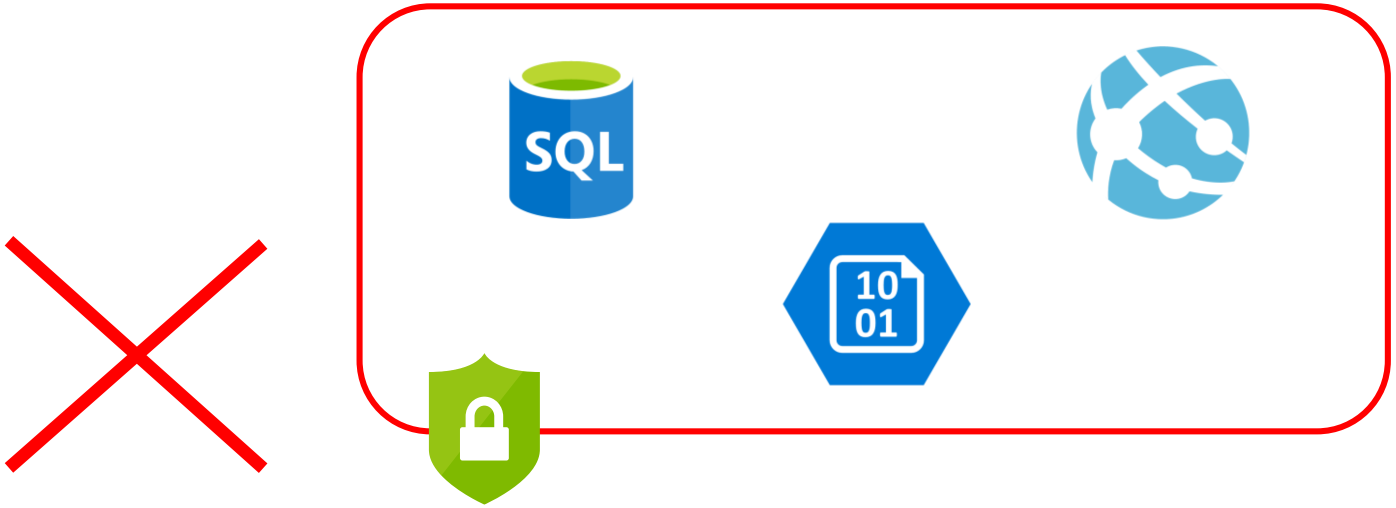
NSGs contain a number of default rules.

Inbound

Name	Priority	Source IP	Destination IP
AllowvNetInBound	65000	VirtualNetwork	VirtualNetwork
AllowAzureLoadBalancerInBound	65001	AzureLoadBalancer	*
DenyAllInBound	65500	*	*

Outbound

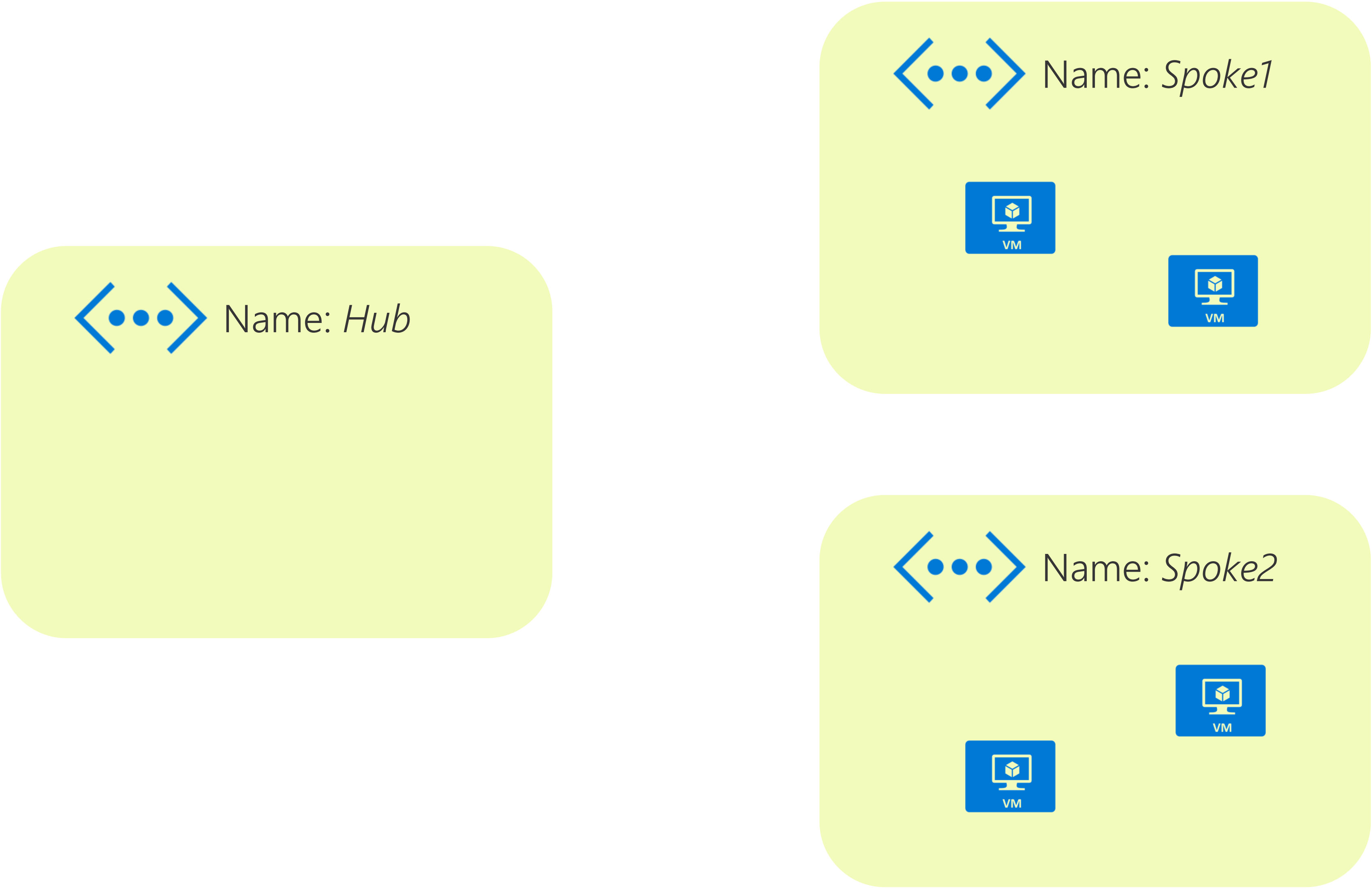
Name	Priority	Source IP	Destination IP
AllowvNetOutbound	65000	VirtualNetwork	VirtualNetwork
AllowInternetOutBound	65001	*	Internet
DenyAllInBound	65500	*	*



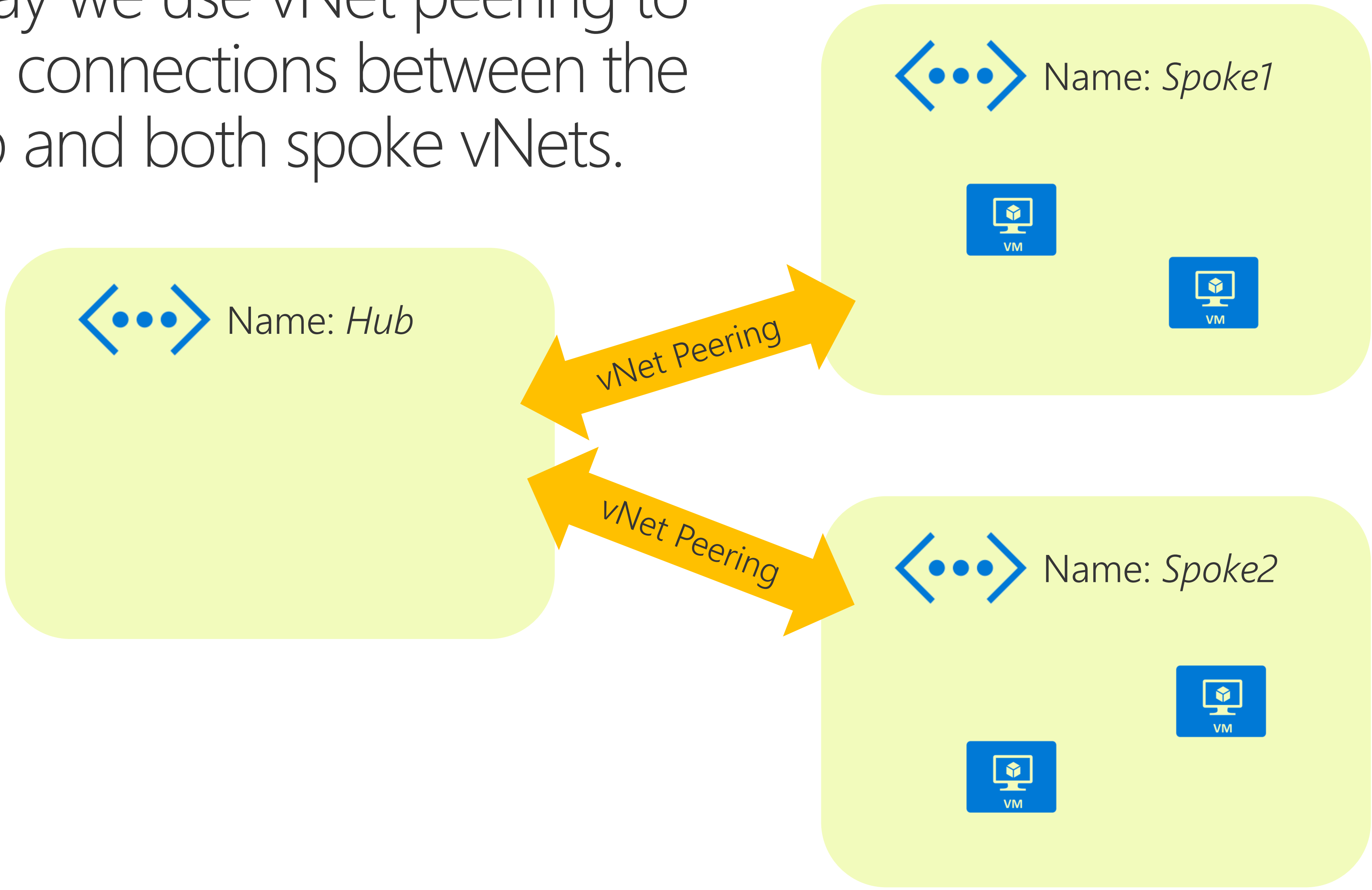
NSGs *only* work if a resource is connected to a vNet – they do not work for other resources (e.g. PaaS services).

Advanced Network Topologies

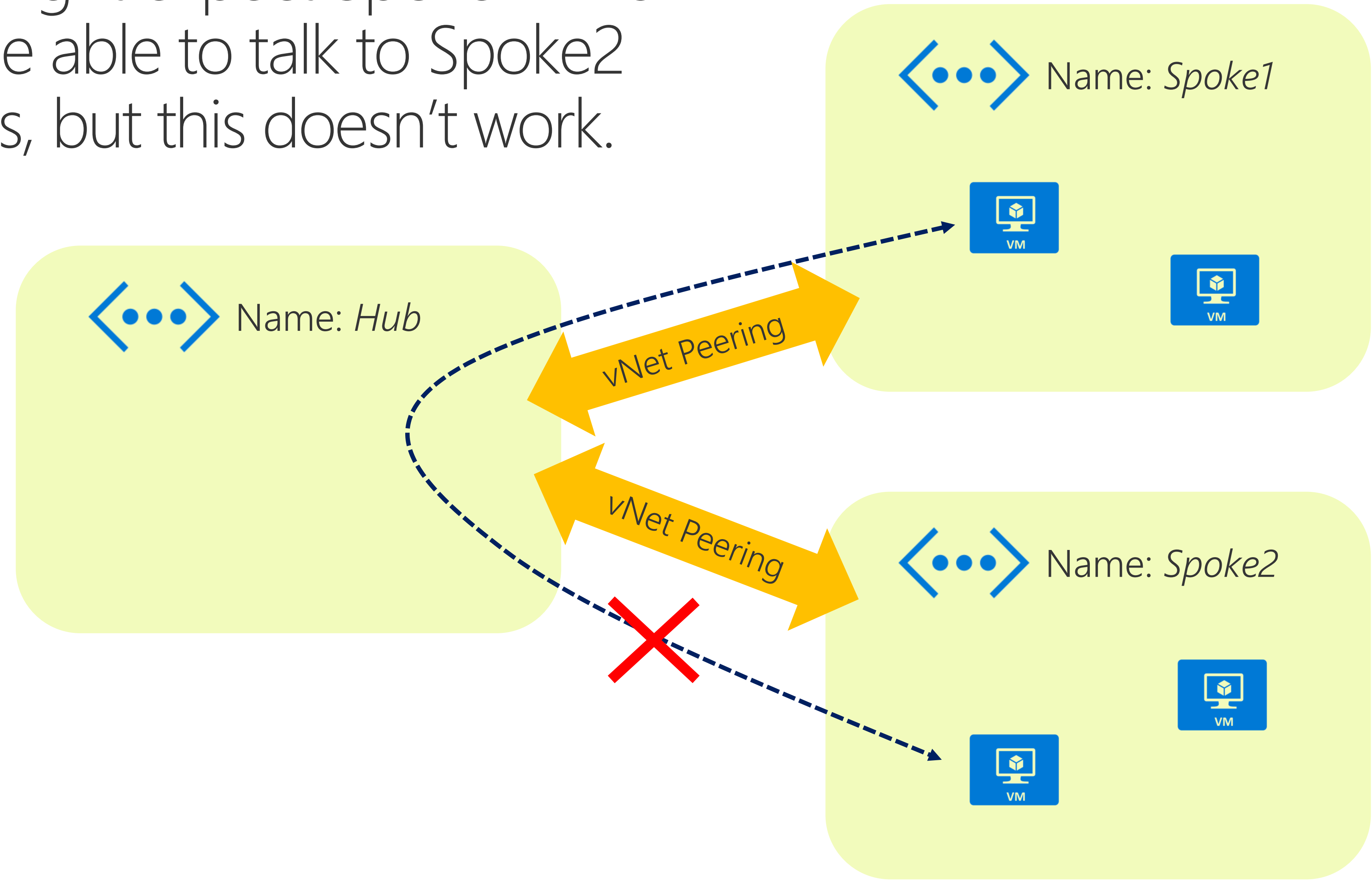
How do we build multiple vNet topologies in Azure?

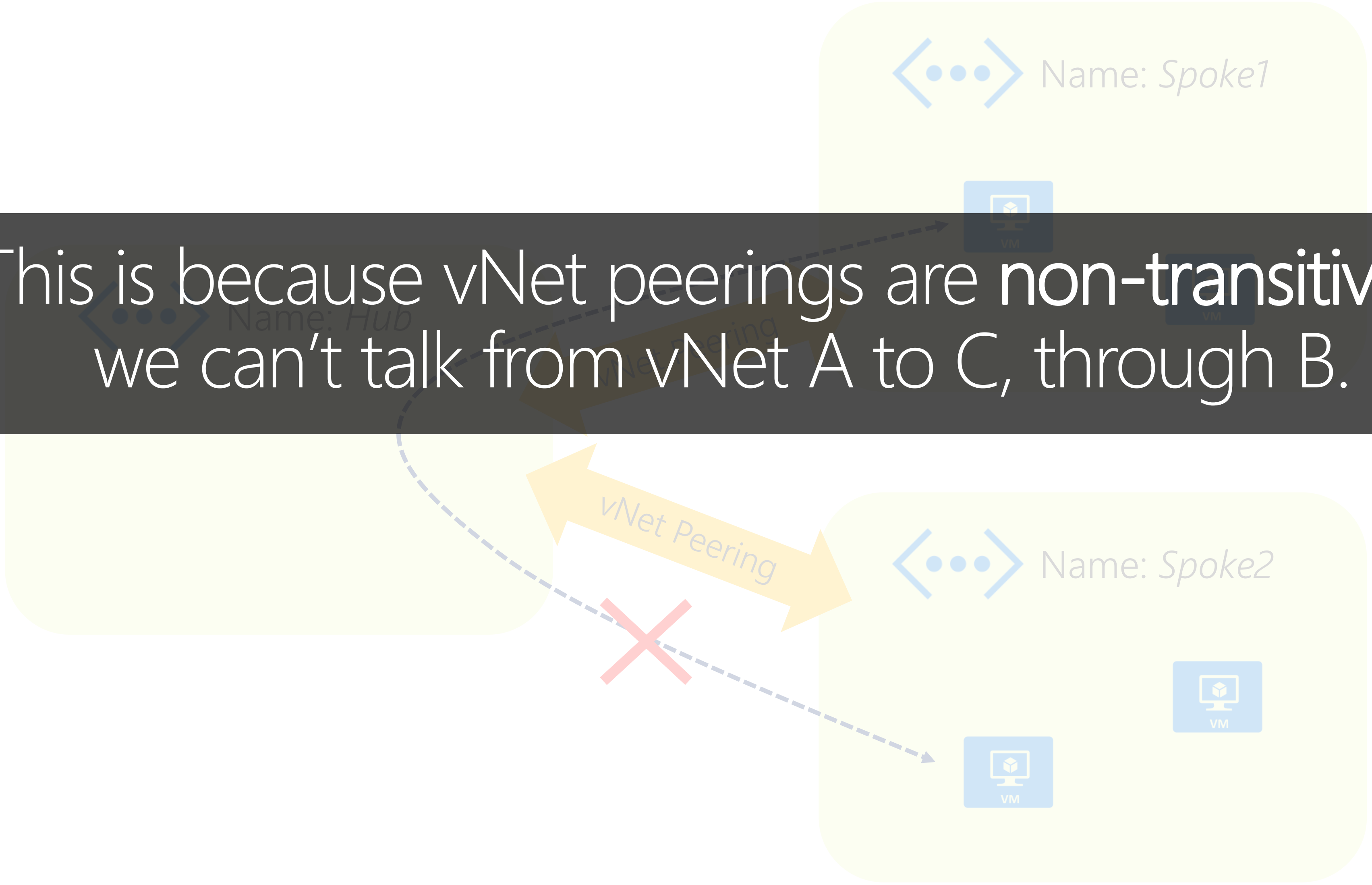


Let’s say we use vNet peering to create connections between the hub and both spoke vNets.



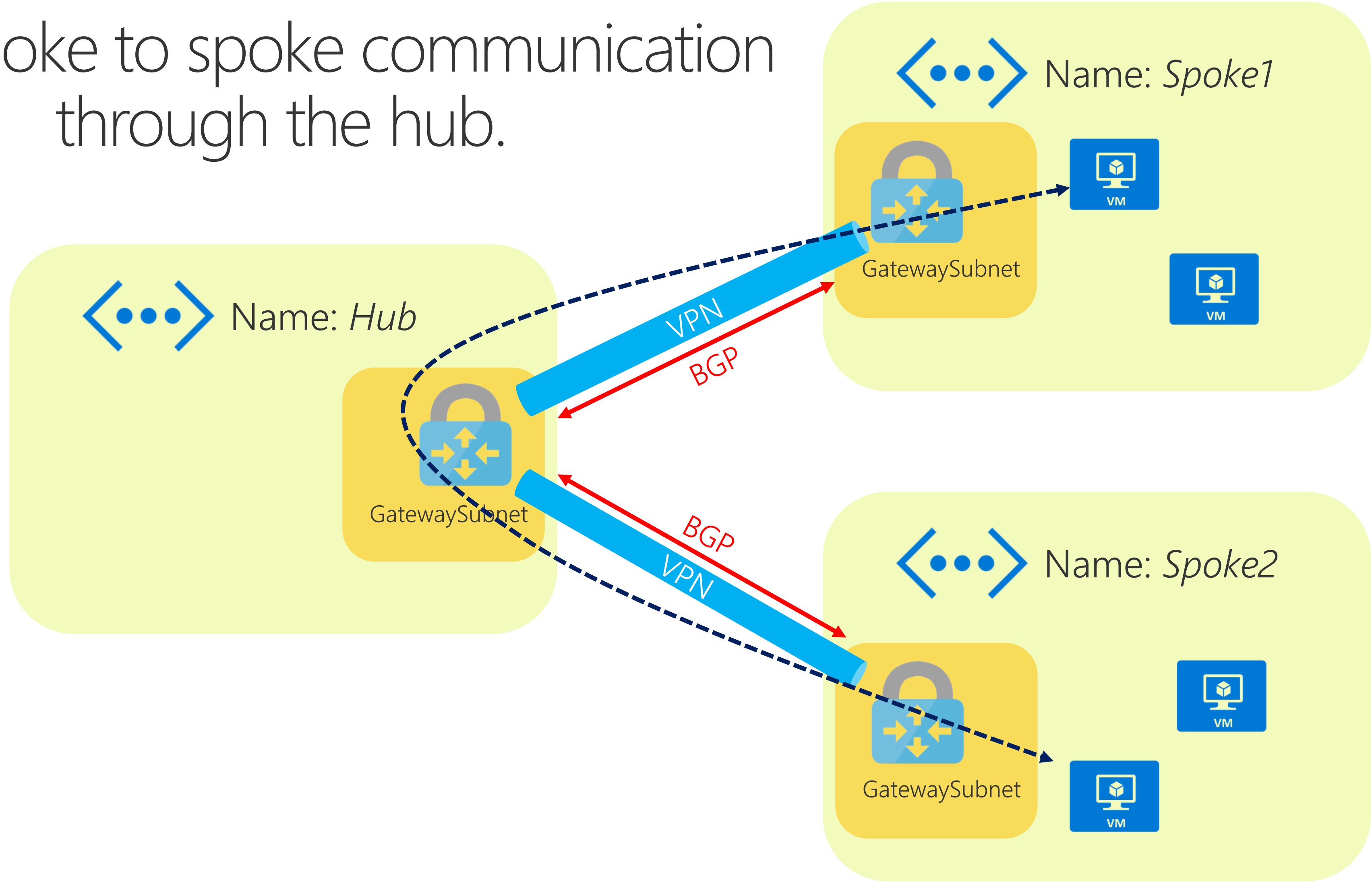
We might expect Spoke1 VMs to be able to talk to Spoke2 VMs, but this doesn't work.



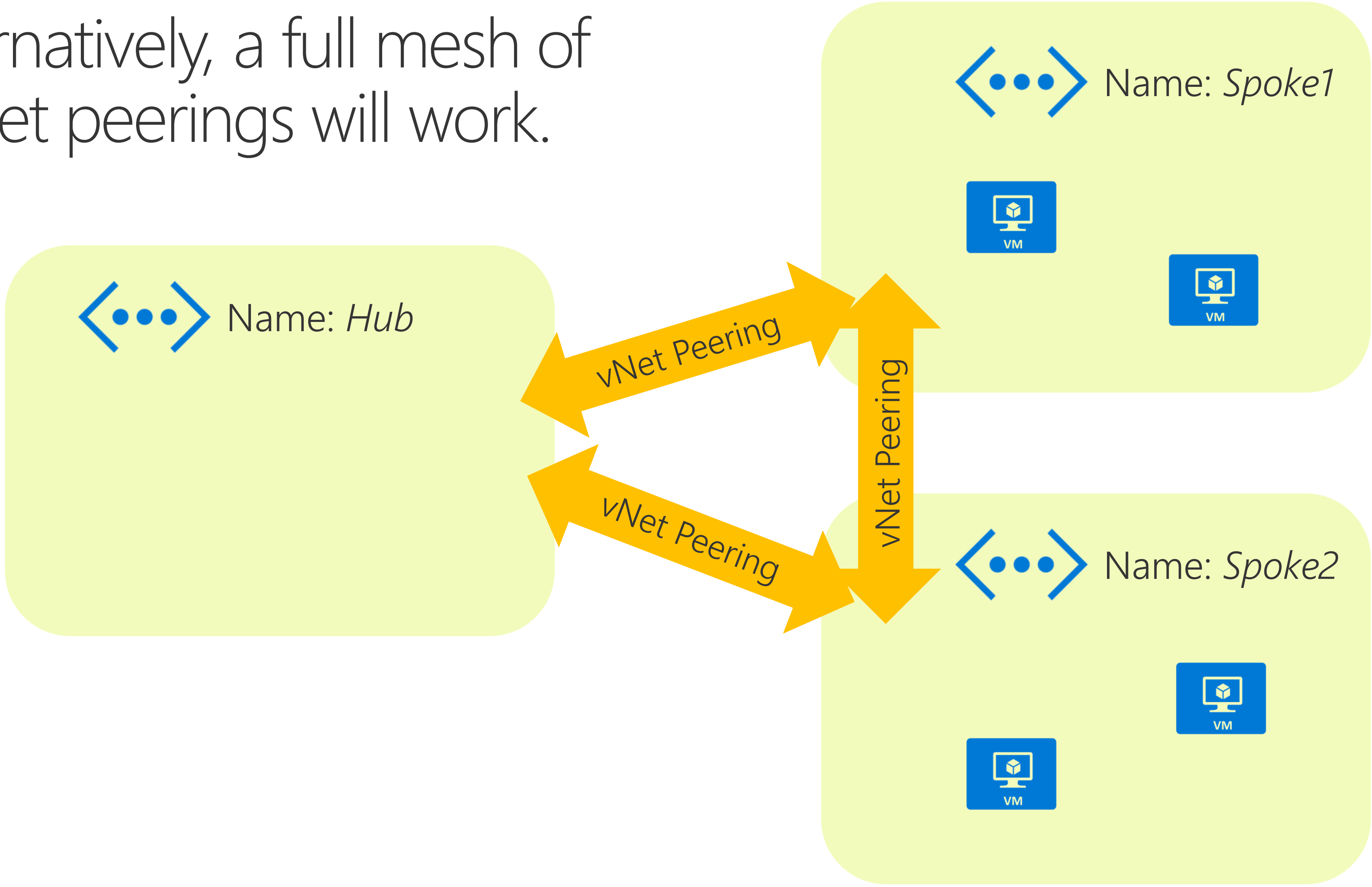


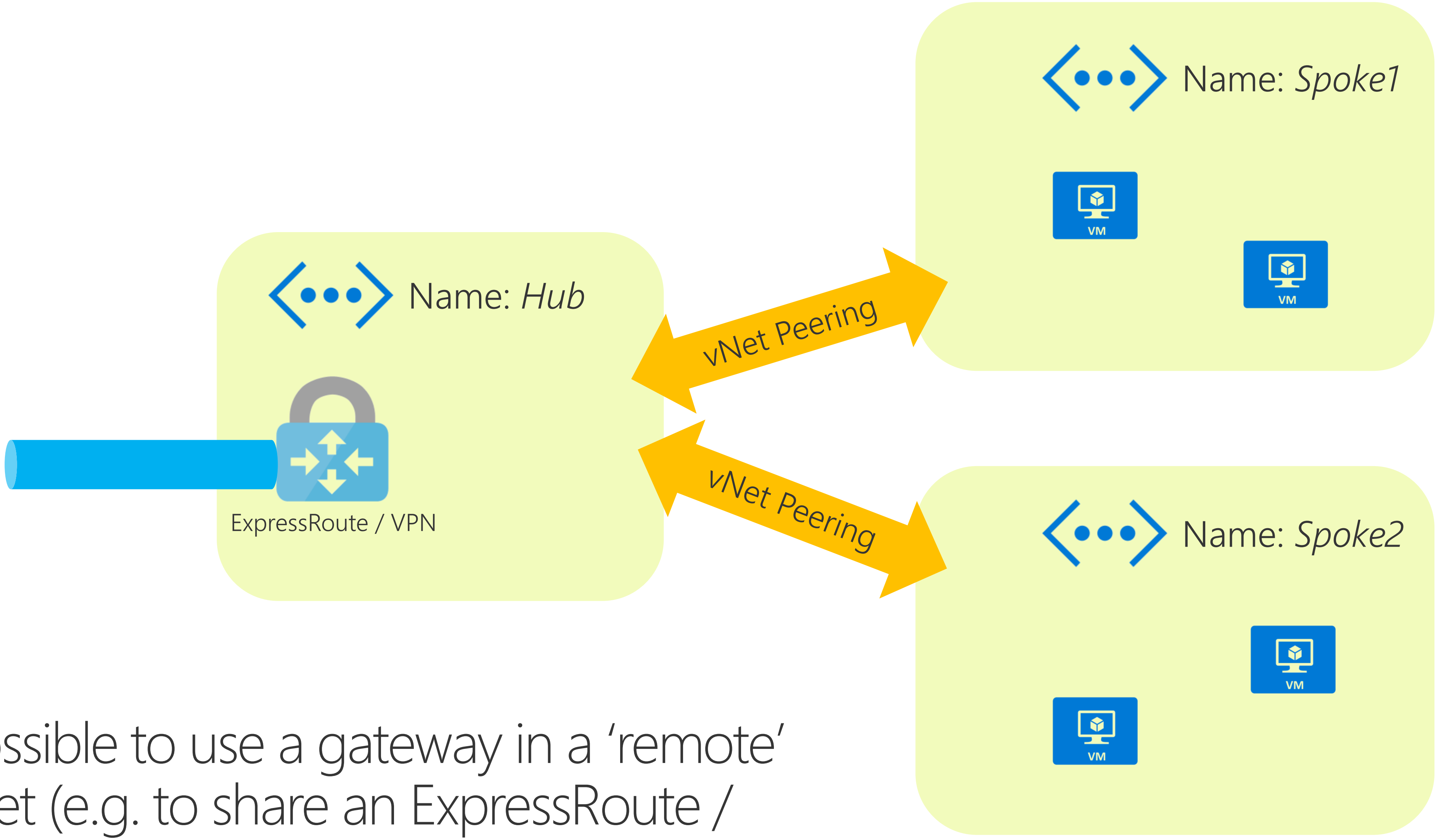
This is because vNet peerings are **non-transitive** –
we can’t talk from vNet A to C, through B.

Using a site-2-site VPN with BGP will allow spoke to spoke communication through the hub.



Alternatively, a full mesh of vNet peerings will work.





It's possible to use a gateway in a 'remote' vNet (e.g. to share an ExpressRoute / VPN connection among spoke vNets).

Microsoft