



Azure Transit VNet

Deployment Guide

How to deploy a Transit VNet solution in Azure

<http://www.paloaltonetworks.com>

Table of Contents

Version History.....	3
1. About.....	4
2. Topology	4
3. Support Policy.....	7
4. Prerequisites	7
5. Launch the Transit VNet Hub Template.....	8
6. Launch the Transit VNet Spoke Template	17
7. VNet Peering Verification.....	25
8. Inbound and Outbound Traffic Tests	27
9. Cleanup	31

Version History

Version number	Comments
1.0	Initial GitHub check-in

1. About

This document will guideline how to deploy a Transit VNet solution in Azure. For more details about the advantages of the hub and spoke topology please refer to this link:

<https://docs.microsoft.com/en-us/azure/architecture/reference-architectures/hybrid-networking/hub-spoke>

Note: The Azure Transit VNet solution is considered advanced. It requires familiarity with Azure and Palo alto Networks. For a more entry level solution please refer to the following Azure Two Tier solution.

<https://github.com/PaloAltoNetworks/azure/tree/master/two-tier-sample>

Palo Alto Networks provides Microsoft Azure ARM Templates to deploy a Transit VNet solution of VM-Series firewalls used in conjunction with, Application Gateways, Standard Load Balancers, Basic Load Balancers, and User Defined Route Tables. This solution does not provide native bootstrapping, so you will be provided firewall configuration files for both the Hub and Spoke firewall pairs as well as detailed steps on how to apply the configuration files.

The Transit VNet solution allows you to provide centralized secured outbound internet access for all your Azure Virtual Networks. This secured outbound internet access is provided by two VM-Series firewall pairs positioned behind an Azure Standard any port load balancer in the Hub VNet. All outbound traffic originating from your Azure Virtual Networks will be provided with a secure single point of exit from your cloud architecture by way of the Hub Virtual Network. User Define Routes are used to route Spoke originating traffic to the Hub internal load balancer for packet forwarding to the Hub VM-Series Firewalls. With layer-7 security being provided by Palo Alto Networks Next Generation Firewalls, you will be able to utilize layer-7 application visibility, content identification, data exfiltration prevention, anti-spyware, anti-malware and many other security benefits.

2. Topology

The transit VNet solution deploys a classic hub-and-spoke. The Hub is deployed in a separate VNet, and each spoke is deployed in a separate VNet as well.

VNet Peering

For the different virtual networks to talk to each other, they must be peered in both the directions. VNet Peering works under the assumption that the peering networks **do not have overlapping subnets**. In this topology, when a VNet spoke is deployed, we will dynamically peer the spoke's VNet and the hub's VNet enabling traffic to flow between them.

Palo Alto Networks Azure Transit VNet Deployment Guide

For additional information on VNet Peering please reference the link below

<https://docs.microsoft.com/en-us/azure/virtual-network/virtual-network-peering-overview>

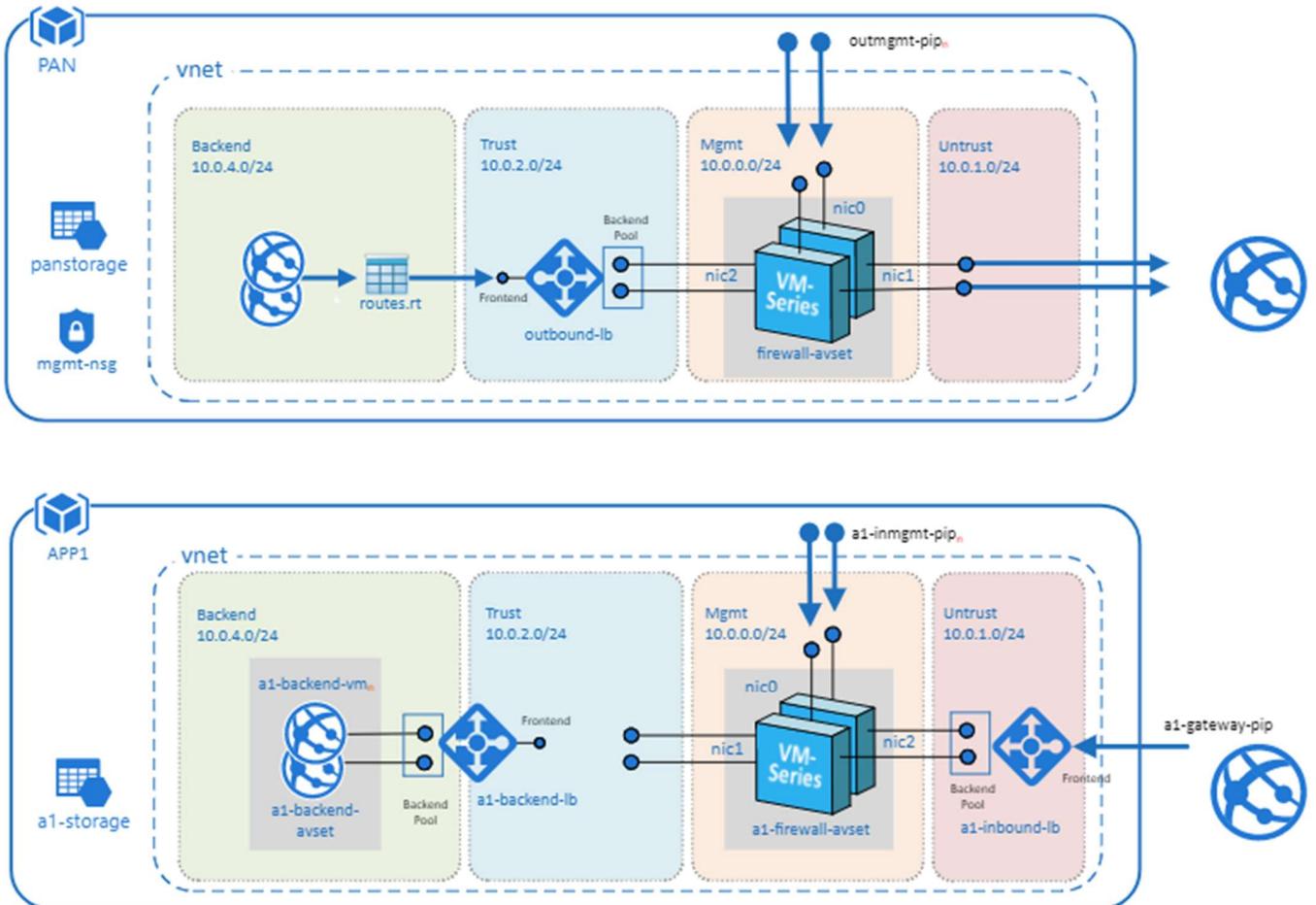


Figure 1

Hub Topology

In Figure 1 **PAN** represents the Hub VNet. The Hub VNet consists of Mgmt , Untrust and Trust subnets. An Azure internal LB[Outbound-LB] used for outbound traffic and a pair of VM Series FWs in an availability set. The Hub topology serves as the exit point of all non-return traffic for the Hub and Spoke topology.

The Hub topology consists of

- 2 VM-Series Firewalls
- 1 Standard internal Loadbalancer

Palo Alto Networks Azure Transit VNet Deployment Guide

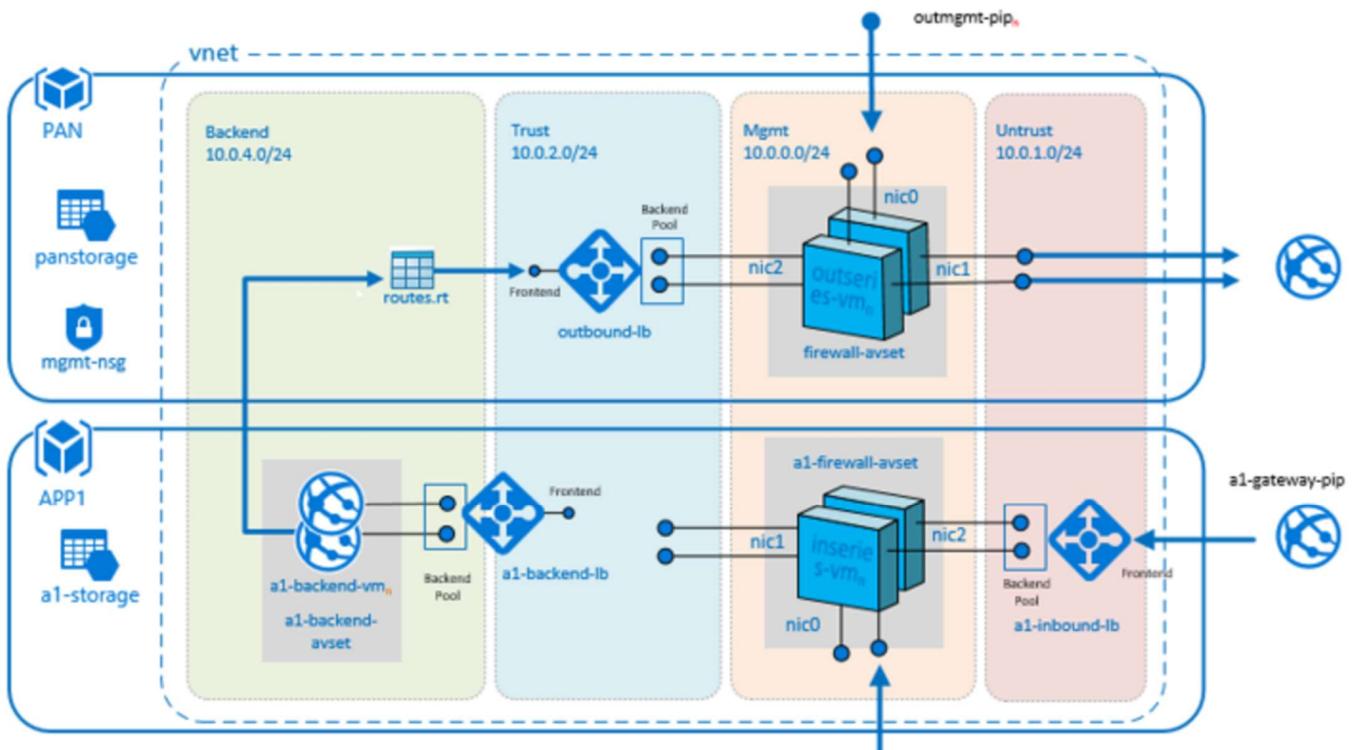
Spoke Topology

In Figure 1 APP1 represents the Spoke VNet. The spoke VNet allows an ingress point for all traffic destined to public facing resources. The subnets consist of Mgmt, Untrust, Trust and Backend Subnets for the application servers. An Application Gateway doubles as a public facing load balancer and sits on the front end. A pair of VM Series FWs in an availability set receive traffic from the public facing LB. An Internal LB sits behind the firewalls and sends traffic to the backend application servers. All return traffic egresses this same path. When a spoke subscribes to a hub, a UDR is also defined which has a default route to the Hub's Internal Load Balancer. This is so all packets that are not destined to the spoke's virtual network gets forwarded to the Hub Internal LB for routing.

The Spoke topology consists of

- 1 Application Gateway listening on port 80. The App Gateway also functions as a public facing external load balancer
- 2 VM-Series Firewalls
- 1 Internal Loadbalancer
- 2 Linux Web servers
- 1 UDR sending all default route traffic to the Hub vnet Standard Loadbalancer.

Hub & Spoke Topology



3. Support Policy

This solution is released under an as-is, best effort, support policy. These scripts should be seen as community supported and Palo Alto Networks will contribute our expertise as and when possible. We do not provide technical support or help in using or troubleshooting the components of the project through our normal support options such as Palo Alto Networks support teams, or ASC (Authorized Support Centers) partners and backline support options. The underlying product used (the VM-Series firewall) by the scripts or templates are still supported, but the support is only for the product functionality and not for help in deploying or using the template or script itself.

4. Prerequisites

Here are the prerequisites required to successfully launch this template:

1. AZURE account with appropriate permissions.
2. Clone or download the files from the following GitHub repository on to your local machine:
<https://github.com/PaloAltoNetworks/Azure-transit-VNet>
3. If the GitHub Repository has a deploy button you can deploy your templates using the button.



5. Launch the Transit VNet Hub Template

There are multiple ways to deploy your template. You can use Azure CLI, PowerShell, Deploy to Azure button or you can deploy the template manually. If the GitHub Repository has a **Deploy to Azure** button you can deploy your template by clicking the deploy button for each template. Below I will walk you through how to launch your ARM template manually.



In the Azure Resource Manager console you can launch the **azureDeployInfra.json** file directly from the Azure Portal. To do this click “**New**” then search “**Template Deployment**”, click the Template Deployment icon and select “**Create**”.

A screenshot of the Microsoft Azure portal interface. The top navigation bar shows "Microsoft Azure" and "New > Marketplace > Everything". On the left, there's a sidebar with "New" (highlighted), "Dashboard", "All resources", "Resource groups", "App Services", and "SQL databases". The main area is titled "Marketplace" with a search bar "Search resources, services and docs". A dropdown menu under "Marketplace" shows categories: "Everything" (selected), "Compute", "Networking", "Storage", and "Web + Mobile". Below this is a search bar "Filter" and a search input "Template Deployment". The results section is titled "Results" and shows one item: "Template deployment" by Microsoft, categorized under Compute. The table columns are NAME, PUBLISHER, and CATEGORY.

Palo Alto Networks Azure Transit VNet Deployment Guide

In the next screen click “Build your own template in the editor”

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes 'New > Marketplace > Everything > Template deployment > Custom deployment'. On the left sidebar, under the 'New' section, 'Virtual machines' is highlighted with a red box. The main content area is titled 'Custom deployment' with the sub-instruction 'Deploy from a custom template'. Below this, there's a 'Learn about template deployment' section with links to 'Read the docs' and 'Build your own template in the editor' (which has a red arrow pointing to it). The 'Common templates' section lists options like 'Create a Linux virtual machine', 'Create a Windows virtual machine', 'Create a web app', and 'Create a SQL database'.

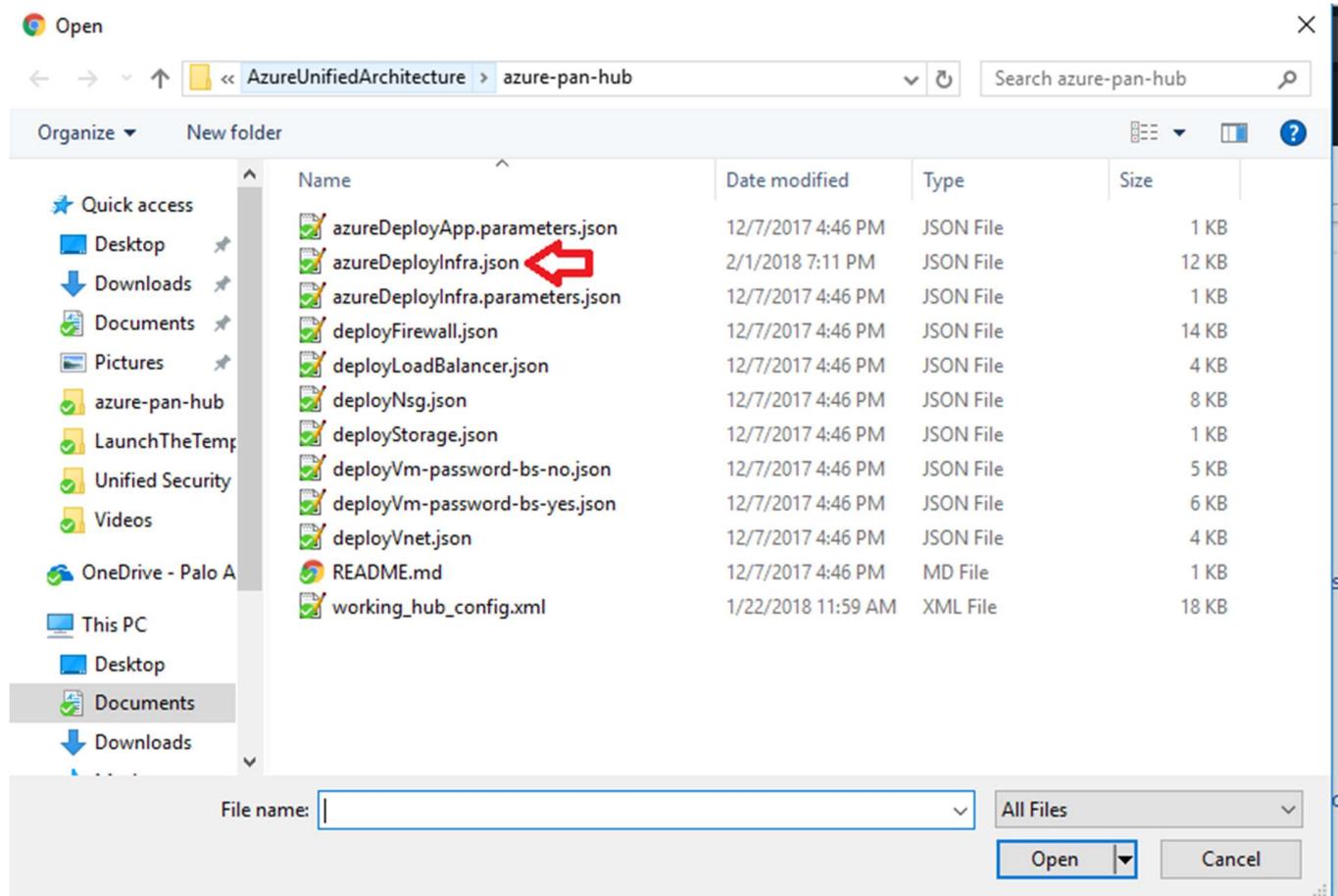
Select “Load File”

This screenshot shows the 'Edit template' page within the Azure portal. The URL in the address bar is 'Microsoft Azure New > Marketplace > Everything > Template deployment > Custom deployment > Edit template'. The left sidebar shows 'New' and 'Virtual machines'. The main area is titled 'Edit template' with the sub-instruction 'Edit your Azure Resource Manager template'. It features a toolbar with 'Add resource', 'Quickstart template', 'Load file' (which is highlighted with a red box), and 'Download'. Below the toolbar, there are sections for 'Parameters (0)', 'Variables (0)', and 'Resources (0)'. To the right, a code editor displays the following JSON template:

```
1 [ {  
2     "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",  
3     "contentVersion": "1.0.0.0",  
4     "parameters": {},  
5     "resources": []  
6 } ]
```

Palo Alto Networks Azure Transit VNet Deployment Guide

Select “**azureDeployInfra.json**” file from the Azure-Transit-VNet/azure-pan-hub directory that you cloned from GitHub, then click “**Save**” to bring up the parameters.



- a. Most of the **parameters** are self-explanatory and should be left at the defaults
- b. **Resource Group** – Always create a new resource Group. The hub template does not work in an existing resource group
- c. **Location** – Use a location where Standard Load Balancer Preview feature is enabled. List of regions is found here - <https://docs.microsoft.com/en-us/azure/load-balancer/load-balancer-standard-overview#region-availability>
- d. **Hub Load Balancer Sku** – Use the **Standard** SKU type which will load balance all outbound TCP traffic.
- e. **Network Security Group Name** is the NSG that is attached to the Firewall's Management subnet
- f. **Network Security Group Inbound Src IP** – This is the IP you will allow explicit access to the management interface of the virtual machines.
- g. **Virtual Network Name** – Use a unique name which will not be used in the spoke VNet deployment. Remember this name since it is an input parameter for the spoke template.
- h. For security purposes be sure to set **Security Group Inbound IP** for mgmt access to the firewall.
- i. **Virtual Network Address Prefix** – Use a network address which will not be used in the spoke deployment.
- j. **Virtual Network Name** – Use a unique name which will not be used in the spoke VNet deployment. Remember this name since it is an input parameter for the spoke template.
- k. **Virtual Network Address Prefix** – Use a network address which will not be used in the spoke deployment.
- l. **Load Balancer IP** – Use a static IP for Load Balancer in the Trust network. Remember this address since it is used as an input parameter for the spoke template.
- m. **Firewall Model** – The template default will work unless a larger size is required.
- n. It could take up to 5 minutes to complete the launch or It could take longer depending on Azure.
- o. **Username** and **password** that is entered by default for the devices is

user:pandemo password:Dem0pa\$\$w0rd

Palo Alto Networks Azure Transit VNet Deployment Guide

Custom deployment
Deploy from a custom template

TEMPLATE

Customized template
5 resources

Edit template Edit parameters Learn more

BASICS

* Subscription: AzureTME

* Resource group: Create new (radio button selected) Use existing
Create a resource group

* Location: Central US

SETTINGS

Hub Load Balancer Sku: Standard

Storage Name: Enter a globally unique name

Mgmt Public IP Dns: Enter a globally unique name

Network Security Group Name: nsg

Network Security Group Inbound IP: 0.0.0.0/0 

Av Set Name: outbound-avset

Storage Type: Standard_LRS

Virtual Network Name: vnet

Virtual Network Address Prefix: 10.0.0.0/16

Mgmt Subnet Name: Mgmt

Mgmt Subnet Prefix: 10.0.0.0/24

Palo Alto Networks Azure Transit VNet Deployment Guide

Acknowledge the terms and conditions and click “Purchase”

Untrusted Subnet Name <small> ⓘ</small>	Untrust
Untrusted Subnet Prefix <small> ⓘ</small>	10.0.1.0/24
Trusted Subnet Name <small> ⓘ</small>	Trust
Trusted Subnet Prefix <small> ⓘ</small>	10.0.2.0/24
Mgmt Public IP Name <small> ⓘ</small>	mgmt-pip
Load Balancer Name <small> ⓘ</small>	outbound-lb
Load Balancer IP <small> ⓘ</small>	10.0.2.4
Firewall Model <small> ⓘ</small>	byol
Firewall Vm Name <small> ⓘ</small>	outbound-vm-series
Firewall Vm Size <small> ⓘ</small>	Standard_D3_v2
Authentication Type <small> ⓘ</small>	password
Username <small> ⓘ</small>	pandemo
Password <small> ⓘ</small>	*****
Ssh Public Key <small> ⓘ</small>	

TERMS AND CONDITIONS

this template. Prices and associated legal terms for any Marketplace offerings can be found in the [Azure Marketplace](#); both are subject to change at any time prior to deployment.

Neither subscription credits nor monetary commitment funds may be used to purchase non-Microsoft offerings. These purchases are billed separately.

If any Microsoft products are included in a Marketplace offering (e.g. Windows Server or SQL Server), such products are licensed by Microsoft and not by any third party.

I agree to the terms and conditions stated above

Pin to dashboard

Purchase

Palo Alto Networks Azure Transit VNet Deployment Guide

Once the firewalls have launched, locate the **Management** interface public IP address in Azure.

PRIORITY	NAME	PORT	PROTOCOL	SOURCE
	Network security group nsg-mgmt (attached to subnet: Mgmt)			Impacts 1 subnets, 0 network interfaces

Log into the hub firewalls using **HTTPS**. Locate the **working_hub_config.xml** configuration snapshot and import this configuration into both firewalls. This is in the Hub directory that you exported from GitHub.

Name	Date modified	Type	Size
azur...parameters.json	12/7/2017 4:46 PM	JSON File	1 KB
azur...infra.json	2/1/2018 7:11 PM	JSON File	12 KB
azur...infra.parameters.json	12/7/2017 4:46 PM	JSON File	1 KB
deployFirewall.json	12/7/2017 4:46 PM	JSON File	14 KB
deployLoadBalancer.json	12/7/2017 4:46 PM	JSON File	4 KB
deployNsg.json	12/7/2017 4:46 PM	JSON File	8 KB
deployStorage.json	12/7/2017 4:46 PM	JSON File	1 KB
deployVm-password-bs-no.json	12/7/2017 4:46 PM	JSON File	5 KB
deployVm-password-bs-yes.json	12/7/2017 4:46 PM	JSON File	6 KB
deployVnet.json	12/7/2017 4:46 PM	JSON File	4 KB
README.md	12/7/2017 4:46 PM	MD File	1 KB
working_hub_config.xml	1/22/2018 11:59 AM	XML File	18 KB

Palo Alto Networks Azure Transit VNet Deployment Guide

Once you load the configuration and commit the changes make sure your ethernet1/1 and Ethernet1/2 interfaces now show green.

The screenshot shows a network interface configuration table. The columns are: Interface, Interface Type, Management Profile, Link State, IP Address, Virtual Router, Tag, VLAN / Virtual-Wire, Security Zone, Features, and Comment. The rows list seven interfaces: ethernet1/1, ethernet1/2, ethernet1/3, ethernet1/4, ethernet1/5, ethernet1/6, and ethernet1/7. Most interfaces have 'Dynamic-DHCP Client' as their management profile and 'Layer3' as their type. Their link state is 'Up'. IP addresses are listed as 'none' or 'Dynamic-DHCP Client'. Virtual routers are mostly 'default'. Tags are either 'Untagged' or 'none'. Security zones include 'untrust', 'trust', and 'none'. Features and comments are mostly blank.

Interface	Interface Type	Management Profile	Link State	IP Address	Virtual Router	Tag	VLAN / Virtual-Wire	Security Zone	Features	Comment
ethernet1/1	Layer3	ILBHealthCheck	Up	Dynamic-DHCP Client	default	Untagged	none	untrust		
ethernet1/2	Layer3	ILBHealthCheck	Up	Dynamic-DHCP Client	default	Untagged	none	trust		
ethernet1/3			Up	none	none	Untagged	none	none		
ethernet1/4			Up	none	none	Untagged	none	none		
ethernet1/5			Up	none	none	Untagged	none	none		
ethernet1/6			Up	none	none	Untagged	none	none		
ethernet1/7			Up	none	none	Untagged	none	none		

Verify the **virtual router** has the following configuration.

The screenshot shows the configuration of a virtual router named 'default'. In the 'Static Routes' tab, it lists three routes: 'defaultRoute' (0.0.0.0/0, interface ethernet1/1, ip-address 10.20.1.1), 'SpokeRoute' (192.168.0.0/24, interface ethernet1/2, ip-address 10.20.2.1), and 'HealthProbe' (168.63.129.128/29, interface ethernet1/2, ip-address 10.20.2.1). The 'IPv4' tab is selected. The 'Route Table' section shows the same three routes with their details: Admin Distance, Metric, and BFD status. Buttons for 'OK' and 'Cancel' are at the bottom right.

Name	Destination	Interface	Type	Value	Admin Distance	Metric	BFD	Route Table
defaultRoute	0.0.0.0/0	ethernet1/1	ip-address	10.20.1.1	default	10	None	unicast
SpokeRoute	192.168.0.0/24	ethernet1/2	ip-address	10.20.2.1	default	10	None	unicast
HealthProbe	168.63.129.128/29	ethernet1/2	ip-address	10.20.2.1	default	10	None	unicast

DefaultRoute: is to forward all outbound traffic to the untrust interface so that it egresses out of the Azure network.

Palo Alto Networks Azure Transit VNet Deployment Guide

SpokeRoute: is to forward all the inbound traffic and inter-spoke traffic back to the Trust interface so that it reaches the appropriate Spoke (application server). Note that the Network address of the all the spokes VNets should be part of this network address. If a new spoke is added whose network address is not part of this network address, then a new route needs to be added in the config to forward that traffic to the Trust interface.

HealthProbe: is to respond to the health probe packets generated by the Internal Load Balancer. For this lab the health check is configured to port 22 on the firewall Trust interface.

An **allow-all** security policy is created to forward all traffic. This should be modified to accommodate your policy preferences.

Name	Tags	Type	Source				Destination				Application	Service	Action	Profile	Options
1 allow_all	none	universal	any	any	any	any	any	any	any	any	application-d...	Allow	none		
2 intrazone-default	none	intrazone	any	any	any	any	(intrazone)	any	any	any	any	Allow	none		
3 interzone-default	none	interzone	any	any	any	any	any	any	any	any	any	Deny	none		

Verify that you have a **NAT rule** on the hub firewall for outbound traffic

Name	Tags	Original Packet						Translated Packet		
		Source Zone	Destination Zone	Destination Interface	Source Address	Destination Address	Service	Source Translation	Destination Translation	
1 hubNatRule	none			ethernet1/1	any	any	any	dynamic-ip-and-port ethernet1/1	none	

6. Launch the Transit VNet Spoke Template

Spoke Template Options

Azuredeploy.json – This launches the spoke template with VM-Series firewalls sandwiched between an external and internal load balancer. This provides secured external access to public facing workloads with return traffic egressing the spoke VNet. All internal originating traffic will be forwarded to the Hub VNet as the exit route to provide secure outbound access.

Azuredeploy-no-firewall.json – Launches the spoke template with no firewalls but still launches application servers. This scenario would NOT provide security using the VM-Series for public facing workloads. All internal originating traffic will be forwarded to the Hub VNet as the exit route to provide secure outbound access.

There are multiple ways to deploy your template. You can use Azure CLI, PowerShell, Deploy to Azure button or you can deploy the template manually. If the GitHub Repository has a **Deploy to Azure** button you can deploy your template by clicking the deploy button for each template. Below I will walk you through how to launch your ARM template manually.



From the Azure-Transit-VNet/azure-pan-spoke GitHub repository that you cloned, launch the **azuredeploy.json** file directly from the Azure Portal. You may need to bring up two azure portal browsers in order to locate information needed to fill out the parameters when launching this template. To do this click “**New**” then search “**Template Deployment**”, click the Template Deployment icon and select “**Create**”.

A screenshot of the Microsoft Azure portal interface. The top navigation bar shows "Microsoft Azure" and "New > Marketplace > Everything". On the left, there's a sidebar with "New", "Dashboard", "All resources", "Resource groups", "App Services", and "SQL databases". The main area is titled "Marketplace" with a search bar containing "Everything". A search result for "Template Deployment" is shown, with details: NAME: "Template deployment", PUBLISHER: "Microsoft", and CATEGORY: "Compute".

NAME	PUBLISHER	CATEGORY
Template deployment	Microsoft	Compute

Palo Alto Networks Azure Transit VNet Deployment Guide

In the next screen click “Build your own template in the editor”

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes 'New > Marketplace > Everything > Template deployment > Custom deployment'. On the left sidebar, under the 'New' section, there are links for 'Dashboard', 'All resources', 'Resource groups', 'App Services', 'SQL databases', 'SQL data warehouses', 'Azure Cosmos DB', and 'Virtual machines'. The main content area is titled 'Custom deployment' with the sub-instruction 'Deploy from a custom template'. Below this, a section titled 'Learn about template deployment' contains two links: 'Read the docs' and 'Build your own template in the editor'. A red arrow points to the 'Build your own template in the editor' link. Another section titled 'Common templates' lists four options: 'Create a Linux virtual machine', 'Create a Windows virtual machine', 'Create a web app', and 'Create a SQL database'.

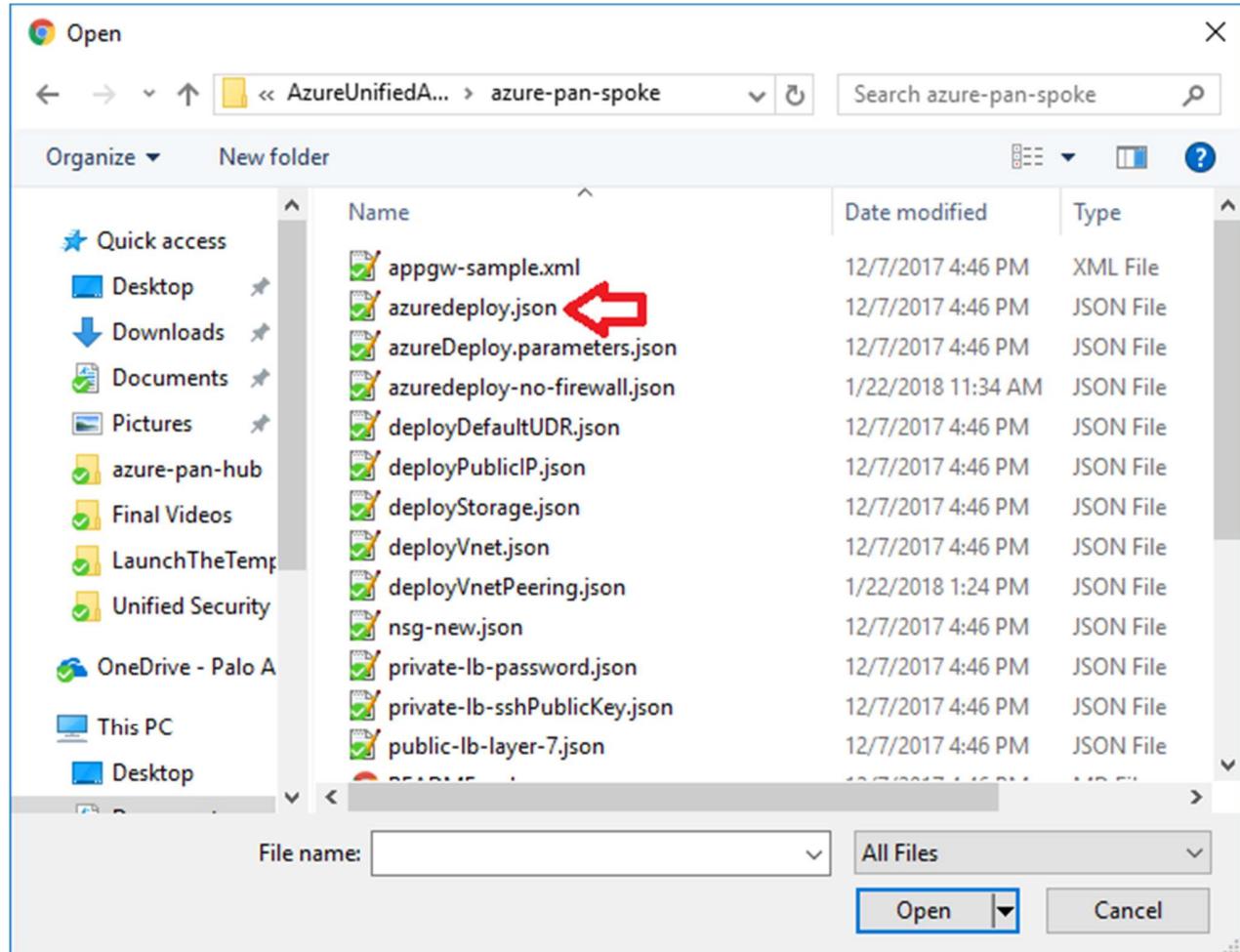
Select “Load File”

The screenshot shows the 'Edit template' page within the Azure portal. The top navigation bar includes 'New > Marketplace > Everything > Template deployment > Custom deployment > Edit template'. The left sidebar shows 'Dashboard', 'All resources', and 'Resource groups'. The main content area is titled 'Edit template' with the sub-instruction 'Edit your Azure Resource Manager template'. It features a toolbar with 'Add resource', 'Quickstart template', 'Load file' (which is highlighted with a red box), and 'Download'. Below the toolbar, there are sections for 'Parameters (0)', 'Variables (0)', and 'Resources (0)'. To the right, a code editor displays the following JSON template:

```
1 {  
2   "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",  
3   "contentVersion": "1.0.0.0",  
4   "parameters": {},  
5   "resources": []  
6 }
```

Palo Alto Networks Azure Transit VNet Deployment Guide

Select “**azuredeploy.json**” file from the Azure-Transit-VNet/azure-pan-spoke directory that you cloned from GitHub, then click “**Save**” to bring up the parameters.



Palo Alto Networks Azure Transit VNet Deployment Guide

- a. Most of the **parameters** are self-explanatory and should be left at the defaults
- b. **Resource Group** – Create a new Resource Group. This template does not work with existing resource groups.
- c. **Location** – It should be the same location as the hub since VNet peering does not work well across regions.
- d. **Hub Resource Group Name** – Give the Resource Group name of the hub created resource group.
- e. **Hub VNet Name** – Use the exact VNet name of the hub created earlier.
- f. **Hub Load Balancer IP** – Use the static IP given to the Load Balancer in the created in the hub template. You can find this information in the load balancer settings
- g. **Network Security Group Name** – The security group name for mgmt access
- h. **Network Security Group Inbound Src IP** – This is the IP you will allow explicit access to the management interface of the virtual machines.
- i. **Virtual Network Address Prefix** – This network address should be the subnet of the network address given in the “**SpokeRoute**” in the hub’s firewall configuration.
- j. **Mgmt, Trust and Untrust** subnets should be subnets of the VNet subnet created in the previous step.
- k. **Firewall VM Size** - Choose the Firewall Model and Size based on requirements. Use Standard D3 or D3 v2.
- l. **SSH Public Key** – If using a password then leave this section blank.

Palo Alto Networks Azure Transit VNet Deployment Guide

Custom deployment
Deploy from a custom template

BASICS

* Subscription: AzureTME

* Resource group: Create new (selected) / Use existing
Create a resource group

* Location: Central US

SETTINGS

* Hub Resource Group Name: hubrg ✓

Hub Vnet Name: hubvnet

* Hub Load Balancer IP: 10.0.2.4 ✓

Network Security Group Name: nsg-mgmt

Network Security Group Inbound Src IP: 0.0.0.0/0

Virtual Network Name: spoke-vnet

Virtual Network Address Prefix: 192.168.0.0/16

Mgmt Subnet Name: Mgmt

Mgmt Subnet Prefix: 192.168.0.0/24

Untrusted Subnet Name: Untrust

Untrusted Subnet Prefix: 192.168.1.0/24

Trusted Subnet Name: Trust

Trusted Subnet Prefix: 192.168.2.0/24

App Gateway Name: myAppGw

* App Gateway Dns Name:

Palo Alto Networks Azure Transit VNet Deployment Guide

App Gateway Subnet Name <small> ⓘ</small>	AppGWSUBNET
App Gateway Subnet Prefix <small> ⓘ</small>	192.168.3.0/24
Internal Load Balancer Name <small> ⓘ</small>	myPrivateLB
Backend Subnet Name <small> ⓘ</small>	backendSubnet
Backend Subnet Prefix <small> ⓘ</small>	192.168.4.0/24
Backend Vm Size <small> ⓘ</small>	Standard_D1
Firewall Model <small> ⓘ</small>	byol
Firewall Vm Name <small> ⓘ</small>	VM-Series
Firewall Vm Size <small> ⓘ</small>	Standard_D3
* Mgmt Public IP Address Name <small> ⓘ</small>	
* Storage Account Name <small> ⓘ</small>	
Storage Account Type <small> ⓘ</small>	Standard_LRS
* Username <small> ⓘ</small>	
Authentication Type <small> ⓘ</small>	password
Password <small> ⓘ</small>	
Ssh Public Key <small> ⓘ</small>	

TERMS AND CONDITIONS

[Azure Marketplace Terms](#) | [Azure Marketplace](#)

By clicking "Purchase," I (a) agree to the applicable legal terms associated with the offering; (b) authorize Microsoft to charge or bill my current payment method for the fees associated the offering(s), including applicable taxes, with the same billing frequency as my Azure subscription, until I discontinue use of the offering(s); and (c) agree that, if the deployment involves 3rd party offerings, Microsoft may share my contact information and other details of such deployment with the publisher of that offering.

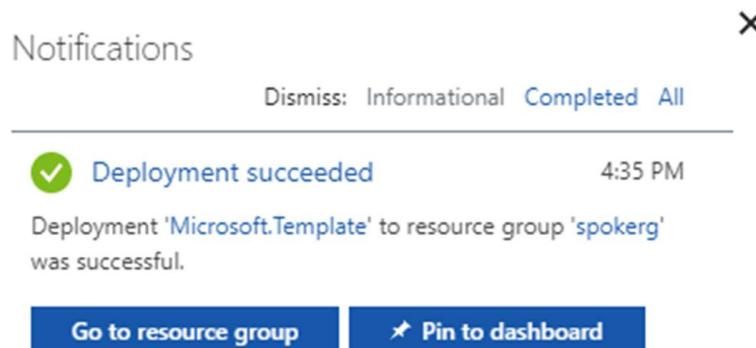
I agree to the terms and conditions stated above

Pin to dashboard

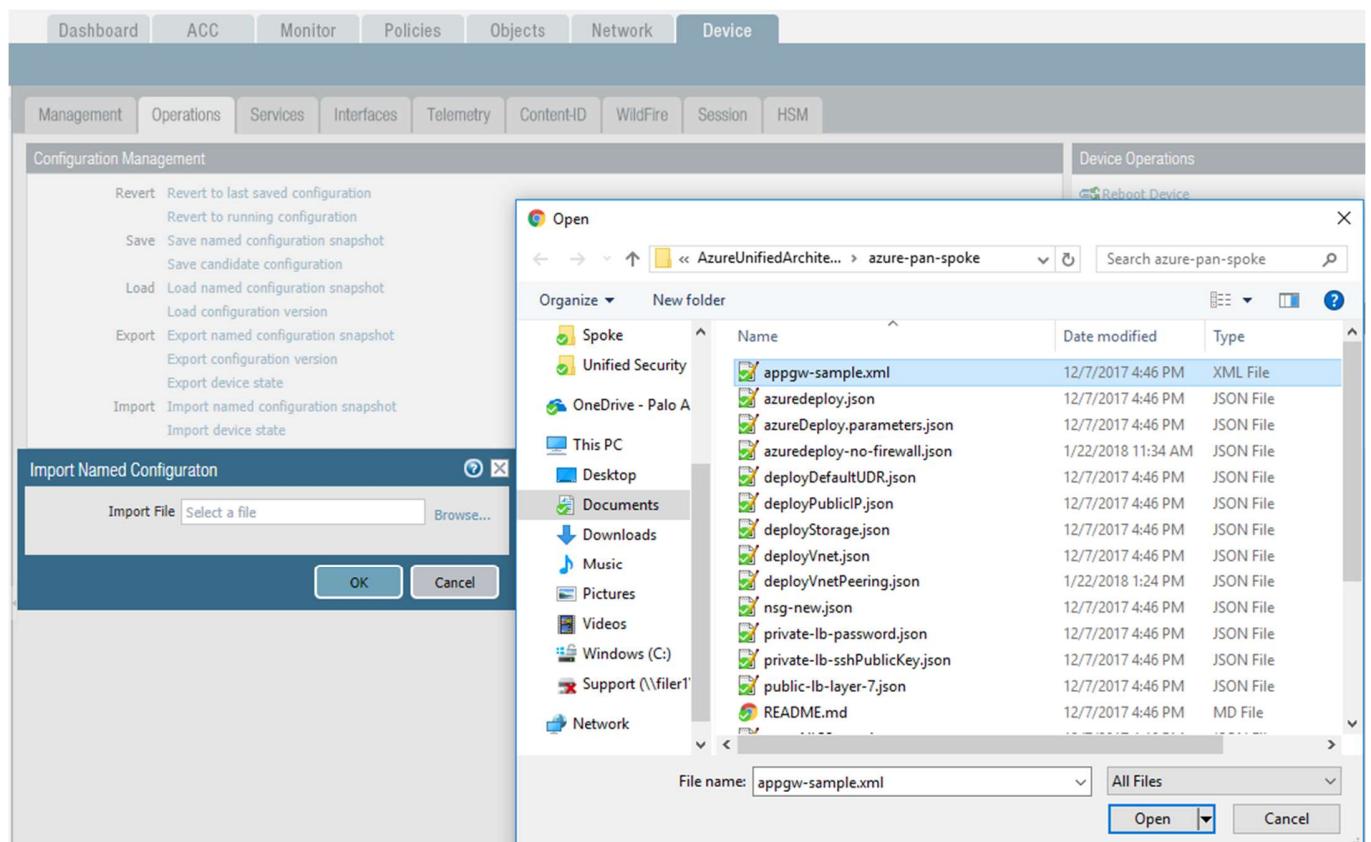
Purchase

Palo Alto Networks Azure Transit VNet Deployment Guide

Once the Spoke template has successfully launched you will see Deployment Succeeded.



Log into the spoke firewalls using **HTTPS**. Locate the **appgw-sample.xml** configuration snapshot and import this configuration into both firewalls. This is in the Spoke directory that you exported from GitHub.



Palo Alto Networks Azure Transit VNet Deployment Guide

Once you load the configuration and commit the changes. Once you have committed the changes make sure your ethernet1/1 and Ethernet1/2 interfaces now show green

Interface	Interface Type	Management Profile	Link State	IP Address	Virtual Router	Tag	VLAN / Virtual-Wire	Security Zone	Features	Comment
ethernet1/1	Layer3		Up	Dynamic-DHCP Client	default	Untagged	none	untrust		
ethernet1/2	Layer3		Up	Dynamic-DHCP Client	default	Untagged	none	trust		
ethernet1/3			Up	none	none	Untagged	none	none		
ethernet1/4			Up	none	none	Untagged	none	none		
ethernet1/5			Up	none	none	Untagged	none	none		
ethernet1/6			Up	none	none	Untagged	none	none		
ethernet1/7			Up	none	none	Untagged	none	none		

Verify the spoke firewall **virtual router** has the following configuration.

Name	Destination	Interface	Type	Value	Admin Distance	Metric	BFD	Route Table
appgw	0.0.0.0/0	ethernet1/1	ip-address	192.168.1.1	default	10	None	unicast

appgw: is to forward all traffic originating from the firewall to the untrust interface. Traffic originating from spoke resources behind the firewall will egress through the Hub VNet.

Palo Alto Networks Azure Transit VNet Deployment Guide

An **allow-all** security policy on the firewall is created to receive all traffic although the application gateway load balancer only listens for port 80. This should be modified to accommodate your policy preferences.

Name	Tags	Type	Source				Destination			Application	Service	Action	Profile	Options
			Zone	Address	User	HIP Profile	Zone	Address						
1 allow_all	none	universal	any	any	any	any	any	any	any	application-d...	Allow	none	none	none
2 intrazone-default	none	intrazone	any	any	any	any	(intrazone)	any	any	any	Allow	none	none	none
3 interzone-default	none	interzone	any	any	any	any	any	any	any	any	Deny	none	none	none

Verify that you have a **NAT rule** on the spoke firewall for inbound traffic

Dashboard	ACC	Monitor	Policies	Objects	Network	Device
Name	Tags	Source Zone	Destination Zone	Destination Interface	Source Address	Destination Address

Original Packet

Name	Tags	Source Zone	Destination Zone	Destination Interface	Source Address	Destination Address	Service	Source Translation	Translated Packet	Destination Translation
1 ilb	none	any	p2p untrust	any	any	firewall-untrust...	any	dynamic-ip-and-port ethernet1/2		address: internal-load-balancer-IP

7. VNet Peering Verification

Within Azure Portal verify that **VNet Peering** has been configured automatically between the Hub VNet and Spoke VNet. To check this in Azure navigate to Virtual Networks > select the virtual network **name**.

Virtual networks	
Palo Alto Networks	
	Add
	Columns
... More	
Filter by name...	
2 items	
NAME	↑↓
	spoke-vnet
	vnet

Palo Alto Networks Azure Transit VNet Deployment Guide

Then select **Peerings**

The screenshot shows the 'vnet - Peerings' blade in the Azure portal. At the top, there's a search bar labeled 'Search (Ctrl+ /)'. Below it is a navigation menu with the following items:

- Overview
- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems

Below the navigation menu is a section titled 'SETTINGS' with the following options:

- Address space
- Connected devices
- Subnets
- DNS servers
- Peerings

The 'Peerings' option is highlighted with a light blue background.

Here you should see the name of the peer **VNet** with a status of **connected**. **Gateway Transit** should be disabled. Check this on both the hub and spoke VNet.

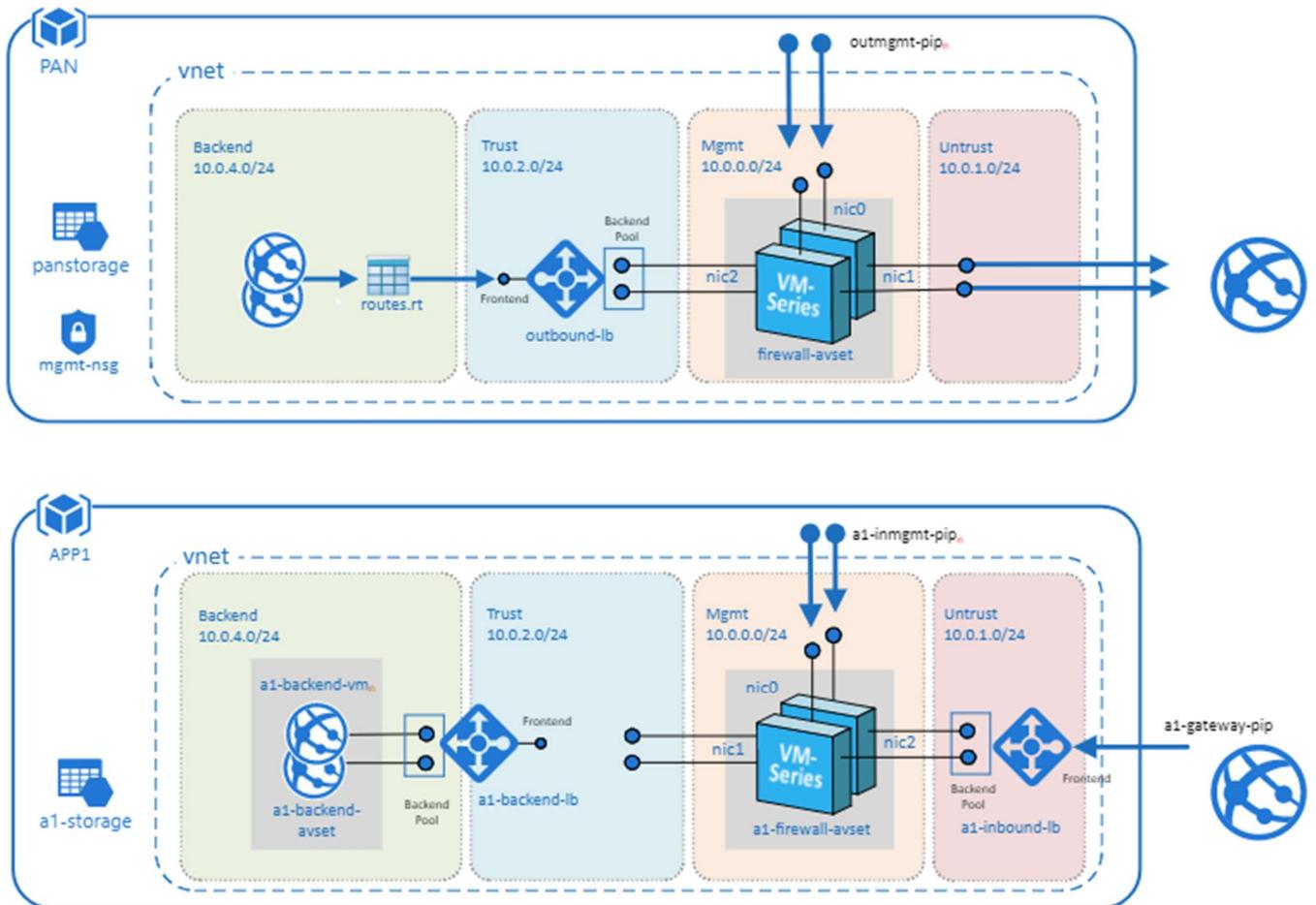
NAME	PEERING STATUS	PEER	GATEWAY TRANSIT	...
vnet-spoke-vnetvnet-peering	Connected	spoke-vnet	Disabled	...

8. Inbound and Outbound Traffic Tests

Once you have confirmed that both the Hub and Spoke templates were successfully deployed, you have imported and loaded the firewall configuration and confirmed VNet Peering, you will want to test your proof of concept with live traffic.

Outbound Traffic Test

As per the diagram all traffic originating from within the Azure virtual networks will exit through the Hub VNet.



One way to test this setup is to originate traffic from a backend Linux VM deployed in the spoke to www.google.com by using wget www.google.com. From there check the traffic logs of the Hub firewalls

Palo Alto Networks Azure Transit VNet Deployment Guide

for www.google.com traffic or web-browsing traffic if using another port 80 based website for wget tests. You will need a license to see logs in the traffic logs or you can edit the template to use PAYG1 or PAYG2.

By default you will not be able to access the Linux servers in the spoke. To access the Linux devices you will need to add a public IP address to one of the Spoke backend Linux servers. Then add a route on the UDR named “**defaultBackendUDR**” for mgmt traffic, that will allow your public IP address with a next hop of “**Internet**”

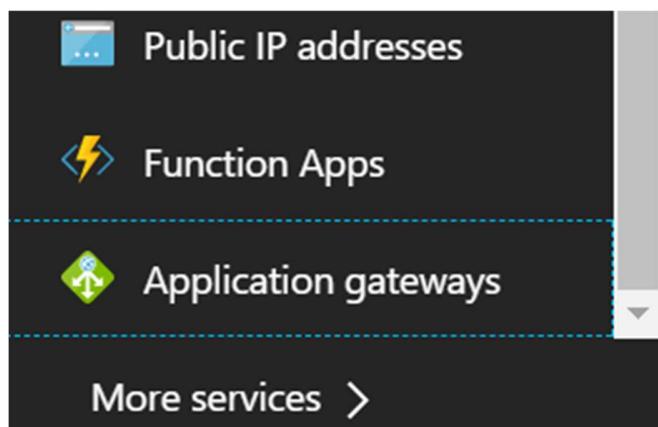
+ Add			
Search routes			
NAME	ADDRESS PREFIX	NEXT HOP	...
defaultRoute	0.0.0.0/0	10.0.2.4	...
mgmt-traffic	.0.0/16	Internet 	...

Another way to accomplish this would be to install a **Bastion Host** or **Jump Box** into the Backend Subnet and SSH from that device.

Inbound Traffic Test

When launching the spoke template with firewalls, the spoke VNet will have an Application Gateway (External LB), A set of firewalls and an internal Load balancer. This allows the spoke to host its own public facing workloads. Once you have launched the Spoke template with firewalls you can test access to the public facing workload by

Navigating to “**Application gateways**” within the Azure Portal



Palo Alto Networks Azure Transit VNet Deployment Guide

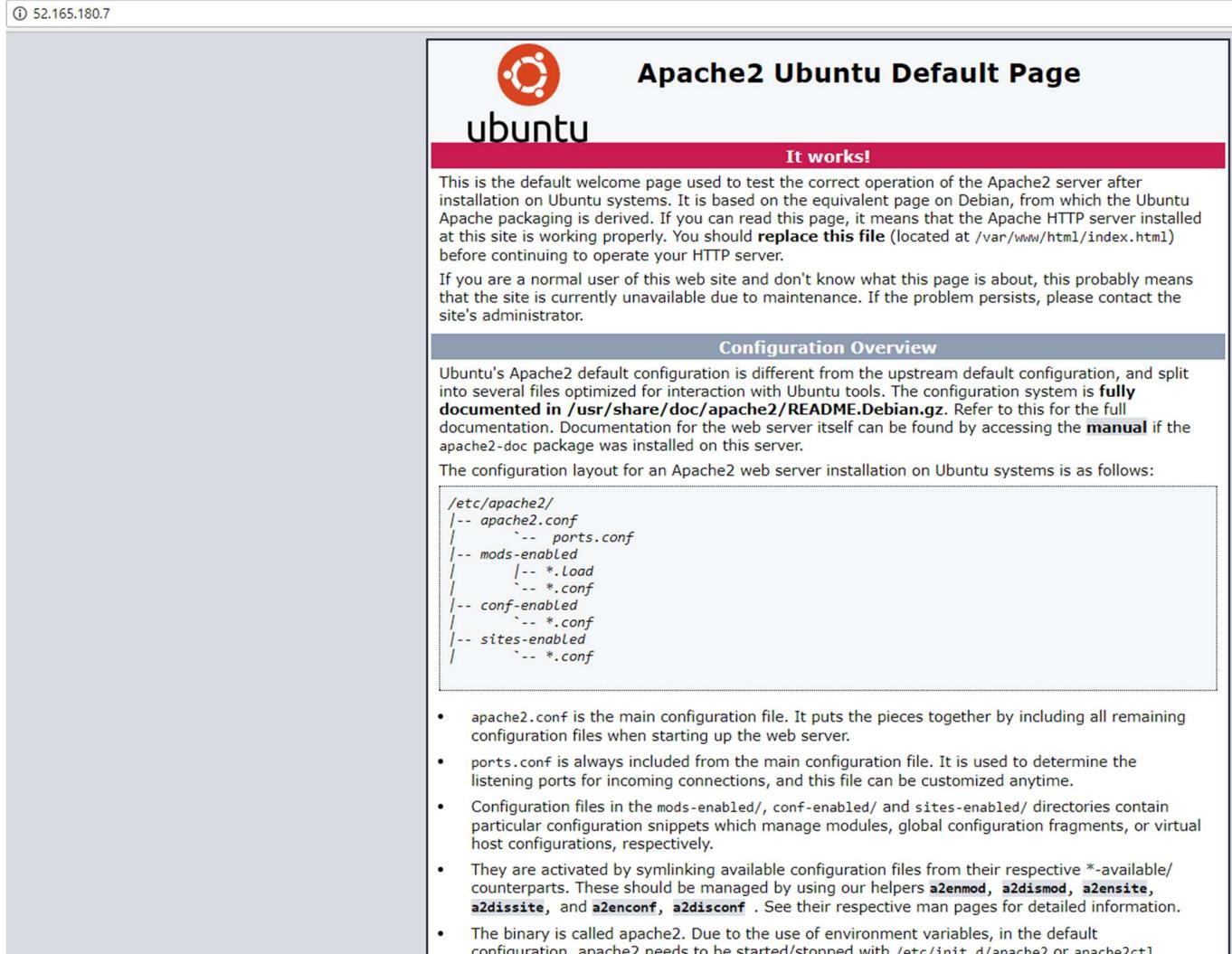
Selecting the name of your **Application Gateway** that was created when you launched the Transit VNet Spoke template. You can find the name of your **Resource Group** to help you differentiate from any other Application Gateways.

<input type="checkbox"/>	NAME	PUBLIC IP ADDR...	PRIVATE IP ADD...	RESOURCE GROUP
<input type="checkbox"/>	js-waf-appgw1	13.93.203.139	-	js-waf-appgw1
<input type="checkbox"/>	myAppGw	52.165.180.7	-	spokerg
<input type="checkbox"/>	myAppGw-jstestuuid1	104.45.230.21	-	jstestuuid1

Locate the **Public IP address** for your Application Gateway.

<input type="checkbox"/>	NAME	PUBLIC IP ADDR...	PRIVATE IP ADD...	RESOURCE GROUP
<input type="checkbox"/>	js-waf-appgw1	13.93.203.139	-	js-waf-appgw1
<input type="checkbox"/>	myAppGw	52.165.180.7	-	spokerg
<input type="checkbox"/>	myAppGw-jstestuuid1	104.45.230.21	-	jstestuuid1

Place the **Public IP address** in your web browser. This IP address is the public facing IP of the Application Gateway Load Balancer. You will see the default Ubuntu Page.



The screenshot shows a web browser window displaying the Apache2 Ubuntu Default Page. The URL in the address bar is 52.165.180.7. The page features the Ubuntu logo and the title "Apache2 Ubuntu Default Page". A red banner at the top right says "It works!". Below the banner, text explains that this is the default welcome page for testing the Apache2 server. It mentions that the configuration is based on Debian and includes a note about replacing the index file. A "Configuration Overview" section details the Apache2 configuration layout, showing a tree structure of files under /etc/apache2/. At the bottom, a list of bullet points describes the components of the configuration layout.

① 52.165.180.7

Apache2 Ubuntu Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in `/usr/share/doc/apache2/README.Debian.gz`**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|   '-- ports.conf
|-- mods-enabled
|   '-- *.Load
|   '-- *.conf
|-- conf-enabled
|   '-- *.conf
|-- sites-enabled
|   '-- *.conf
```

- `apache2.conf` is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
- `ports.conf` is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
- Configuration files in the `mods-enabled/`, `conf-enabled/` and `sites-enabled/` directories contain particular configuration snippets which manage modules, global configuration fragments, or virtual host configurations, respectively.
- They are activated by symlinking available configuration files from their respective `*-available/` counterparts. These should be managed by using our helpers `a2enmod`, `a2dismod`, `a2ensite`, `a2dissite`, and `a2enconf`, `a2disconf`. See their respective man pages for detailed information.
- The binary is called `apache2`. Due to the use of environment variables, in the default configuration, `apache2` needs to be started/stopped with `/etc/init.d/apache2` or `apache2ctl`.

9. Cleanup

You can clean up the setup by deleting the **resource groups** for both the hub and spoke deployments. Once you have deleted the resource groups for both the hub and spoke you will have successfully deleted all resources created in this deployment.