

Azure Cloud Framework



Azure Transit VNET Framework Deployment Guide

Date: June 20th, 2019

Revision: 1.0.a

Author: Dejan Stanic

This document will detail how to install the Azure Transit VNET Framework demo. The Transit VNET framework will use Panhandler as the deployment tool. The following prerequisites are required to be installed on your local computer prior to the installation of the Azure Transit VNET Framework.

Prerequisites:

Docker:

Docker will need to be installed to run Panhandler. Here are some links to help with the install:

For Mac:

<https://docs.docker.com/docker-for-mac/install/>

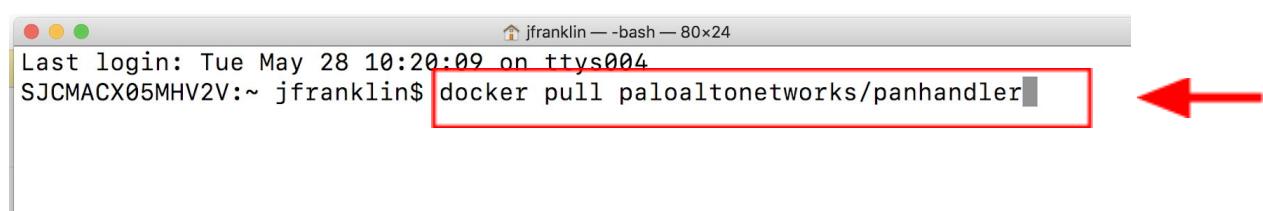
For Windows:

<https://docs.docker.com/docker-for-windows/install/>

Panhandler:

Once Docker is installed and running, you will need to pull the panhandler container image from docker hub and build a Panhandler container. From a Mac terminal window or a PC powershell window run the following commands:

```
docker pull paloaltonetworks/panhandler
```



```
jfranklin ~ bash 80x24
Last login: Tue May 28 10:20:09 on ttys004
SJCMACX05MHV2V:~ jfranklin$ docker pull paloaltonetworks/panhandler
```

After panhandler image is downloaded from Docker Hub, start Panhandler using the following commands:

Mac command:

```
docker run -t -p 8888:80 -v $HOME:/root/ paloaltonetworks/panhandler &
```

Windows command:

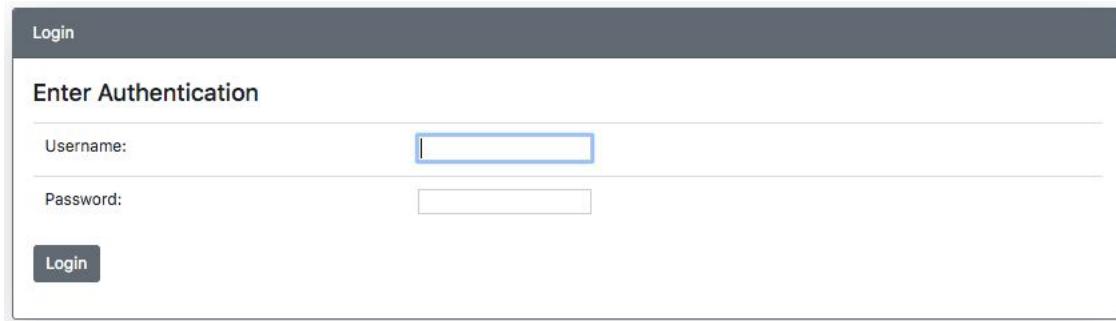
```
docker run -t -p 8888:80 -v /c/Users/%USERNAME%:/root/ paloaltonetworks/panhandler
```

```
[SJCMACX05MHV2V:~ jfranklin$ docker run -t -p 8888:80 -v $HOME:/root/ paloaltonetworks/panhandler &
[1] 81145
SJCMACX05MHV2V:~ jfranklin$ Adding package panhandler to celery
Adding package panhandler to celery
Adding app config for app: panhandler
```

At this point, you should have a container running on your machine with Panhandler. To access Panhandler, open a browser and navigate to the following URL:

<http://localhost:8888>

You should see the following login page.

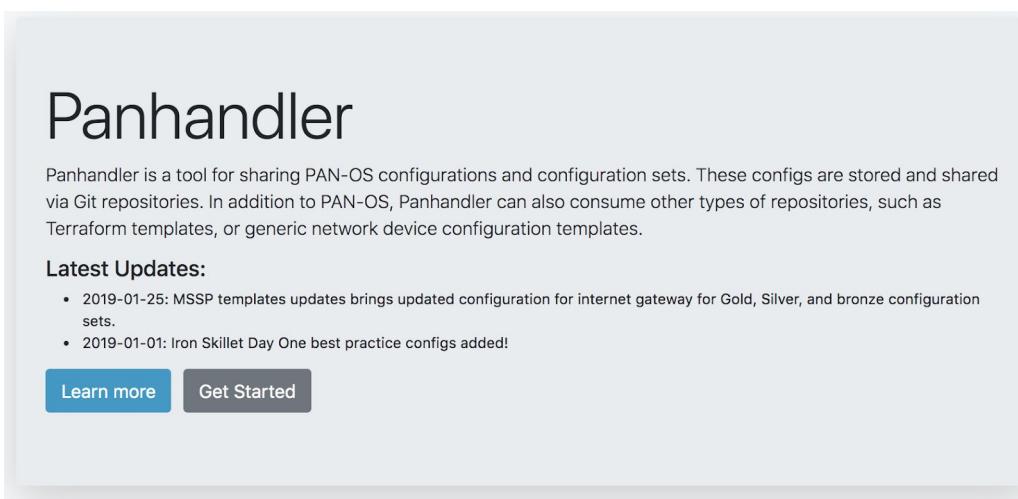


Default login username and password are:

Username: paloalto

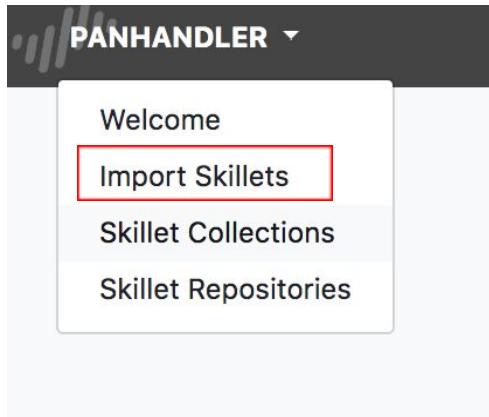
Password: panhandler

Once you are logged in, you will get a welcome screen



Deployment:

In the upper left hand corner, you will see the Panhandler drop down. Click on the drop down and navigate to the “Import Skillets” option.



To import the Azure Transit VNET Framework, you can scroll down to the Import Repository section, enter Repository Name and Git Repository URL and select Submit button.

Repository Name: Azure Transit VNET

GitHub URL: <https://github.com/dstanic-pan/azure-transit-vnet-v2>

A screenshot of a "Import Repository" form. At the top left is the title "Import Repository" and at the top right are two small icons. The main area has a light gray background. It contains the following fields:

- A label "Enter a valid git url and desired branch below"
- A "Repository Name:" label with a text input field containing "Azure Transit VNET".
- A "Git Repository HTTPS URL:" label with a text input field containing "https://github.com/dstanic-pan/azure-transit-vnet-v2.git". This field is highlighted with a red rectangular border.
- A "Branch:" label with a text input field containing "master".
- A blue "Submit" button at the bottom right, which is also highlighted with a red rectangular border.

Updates

Repositories are periodically updated to include bug fixes or additional functionality. After the initial import, periodically updating the skillets to the latest version is recommended. From the upper left-hand corner Panhandler drop-down list, select “Skillet Repositories”



Select Detail under “Azure Transit VNET”:

Azure Transit VNET

<https://github.com/dstanic-pan/azure-transit-vnet-v2.git> master

[Detail](#)

Select Update to Latest:

Repository Detail for Azure Transit VNET

Details

<https://github.com/dstanic-pan/azure-transit-vnet-v2.git> master
<https://github.com/dstanic-pan/azure-transit-vnet-v2.git>

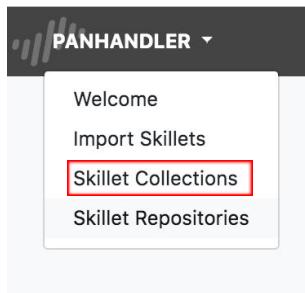
[Update To Latest](#) [Remove Repository](#) [Repositories](#)

Latest Updates

#	Message	Author	Date
1	update	dstanic-pan	2019-06-27 22:44:47+02:00
2	update	dstanic-pan	2019-06-27 22:42:55+02:00
3	update	dstanic-pan	2019-06-27 22:40:05+02:00

Commit History for branch: master

Once the repository has been added/updated, you are now ready to deploy the demo framework. From the upper left-hand corner Panhandler drop-down list, select “Skillet Collections”



You will see the screen below. Select the Azure Transit VNET

A screenshot of the "Skillet Collections" screen. At the top, there is a search bar and a "Search" button. Below that, there are several collection cards. The "Azure Transit VNET" card is highlighted with a red box around its "Go" button. The other cards include "AWS Jenkins Security Framework", "Azure Jenkins Security Framework", "Example Skillets", "Configure", "Deploy", and "2 Skillets in the Configure Collection". Each card has a "Go" button at the bottom right.

This will take you to the template deployment screen. You will see six options.

Azure Login: In order to run the automation script, you will need to login to the Azure CLI. The Azure login script will launch the login process and provide a URL to connect to and a token to paste into the Azure Website.

Step 1: Infrastructure Deployment - This first step will deploy the Azure architecture, the template will deploy Azure VNets, Route Tables, Subnets, Availability Zones, Load Balancers, Network Security Groups, etc.. The Template will also deploy Palo Alto Networks Firewalls in LB-Sandwich architecture. For a more detailed

definition of the architecture, please see the “insert link and document name of ppt here”

Step 2: Outbound Traffic Inspection - The second step will configure outbound traffic inspection thru Hub NGFW's by configuring Routes, NAT and Security policies on VM-Series. It will also finalize initial configuration by updating default and spoke routes on FW's.

Step 3: Inbound Traffic Inspection - The third step will configure inbound traffic inspection to protected Web1 server by creating corresponding Routes, NAT and Security policies on VM-Series. Traffic will be initiated from internet thru Azure Public LB, NGFW's and finally web server located in Web subnet of Spoke1 VNET.

Step 4: East-West Traffic Segmentation - The fourth step will configure East-West traffic inspection and segmentation thru Hub NGFW's by configuring Routes, Address objects, and Security policies on VM-Series.

Step 5: Once the demo is complete and you are done with the infrastructure, this step will destroy the demo environment.

Skillet Collection: Azure Transit VNET

Sort	Filter	Search								
Name	Type	PAN-OS	Panorama	Panorama-GPCS	REST	Template	Terraform	Workflow	Python	
Azure Login (Pre-Deployment Step)										
This skillet will log into Azure. You will be prompted to follow a link and enter a device-code in your browser.										
Skillet type: python3										
Collections: Azure Transit VNET , All										
	Go									
Azure Transit VNET Framework Step 1 - Infrastructure deployment										
This VNET Framework Infrastructure. The template will deploy Azure VNets, Route Tables, Subnets, Availability Zones, Load Balancers, Network Security Groups, etc.. The Template will also deploy Palo Alto Networks Firewall's in LB-Sandwich architecture.										
Skillet type: python3										
Collections: Azure Transit VNET , All										
	Go									
Azure Transit VNET Framework Step 2 - Outbound Traffic Inspection										
This skillet demonstrates Outbound traffic inspection thru Hub NGFW's by configuring Routes, NAT and Security policies on VM-Series. It is very important to run this skillet after Step 1 as it also updates default and spoke routes on FW.										
Skillet type: panos										
Collections: Azure Transit VNET , All										
	Go									
Azure Transit VNET Framework Step 3 - Inbound Traffic Inspection										
This skillet demonstrates Inbound traffic inspection thru Hub NGFW's by configuring Routes, NAT and security policies on VM-Series.										
Skillet type: panos										
Collections: Azure Transit VNET , All										
	Go									
Azure Transit VNET Framework Step 4 - East West Traffic Segmentation										
This skillet demonstrates East-West traffic inspection and segmentation thru Hub NGFW's by configuring Routes, Address objects, and Security policies on VM-Series.										
Skillet type: panos										
Collections: Azure Transit VNET , All										
	Go									
Azure Transit VNET Framework Step 5 - Teardown										
This skillet will destroy the Azure Transit VNET Environment. Run this step once the demo is complete.										
Skillet type: python3										
Collections: Azure Transit VNET , All										
	Go									

Pre-deployment Login:

To complete the Azure Login, we are utilizing the Azure CLI login rather than a service principal. The CLI login process will create a token in your home directory that the Panhandler container will use to authenticate the automation templates running in the Panhandler container. To begin the login process, select “Go” on the Azure Login Template box.

Skillet Collection: Azure Transit VNET

Sort	Filter	Search							
Name	Type	PAN-OS	Panorama	Panorama-GPCS	REST	Template	Terraform	Workflow	Python
Azure Login (Pre-Deployment Step)		Azure Transit VNET Framework Step 1 - Infrastructure deployment							
This skillet will log into Azure. You will be prompted to follow a link and enter a device-code in your browser.		This skillet demonstrates Outbound traffic inspection thru Hub NGFW's by configuring Routes, NAT and Security policies on VM-Series. It is very important to run this skillet after Step 1 as it also updates default and spoke routes on FW.							
Skillet type: python3		Skillet type: panos							
Collections: Azure Transit VNET , All		Collections: Azure Transit VNET , All							
Go									
Azure Transit VNET Framework Step 3 - Inbound Traffic Inspection		Azure Transit VNET Framework Step 4 - East West Traffic Segmentation							
This skillet demonstrates Inbound traffic inspection thru Hub NGFW's by configuring Routes, NAT and security policies on VM-Series.		This skillet demonstrates East-West traffic inspection and segmentation thru Hub NGFW's by configuring Routes, Address objects, and Security policies on VM-Series.							
Skillet type: panos		Skillet type: panos							
Collections: Azure Transit VNET , All		Collections: Azure Transit VNET , All							
Go									
Azure Transit VNET Framework Step 5 - Teardown									
This skillet will destroy the Azure Transit VNET Environment. Run this step once the demo is complete.									
Skillet type: python3									
Collections: Azure Transit VNET , All									
Go									

You will see the window below:

Provision

Azure Login

No user input required. Click Submit to launch this Skillet

[Submit](#)

Click on the submit button. The first time the demo build is run, Panhandler will validate the correct version of libraries and tools are loaded into the environment. This will only run the first time you deploy the demo and depending on the speed of your internet connection it can take up to 10 minutes to complete. Below is the screen that you will see during the environment verification:

```

Preparing environment for: Azure Transit VNET Framework Step 1 - Infrastructure deployment
Using base prefix '/usr/local'
New python executable in /root/.pan_cnc/panhandler/repositories/Azure Transit VNET/.venv/bin/python3
Also creating executable in /root/.pan_cnc/panhandler/repositories/Azure Transit VNET/.venv/bin/python
Installing setuptools, pip, wheel...
done.
Installing requirements
Collecting adal==1.2.1 (from -r requirements.txt (line 1))
  Using cached https://files.pythonhosted.org/packages/00/72/53dce64f5d6c1aa57b8d408cb34dff1969ecbf10ab7e678f32c5e0e2397/adal-
Collecting amqp==2.4.2 (from -r requirements.txt (line 2))
  Using cached https://files.pythonhosted.org/packages/42/ec/cbbfaa8f75be8cbd019afb9d63258e2bdc95242f8c46a54b90db5fe03bd/amqp-
Collecting antlr4-python3-runtime==4.7.2 (from -r requirements.txt (line 3))
  Using cached https://files.pythonhosted.org/packages/03/c8/7848a0dd85158930b859eb8be1e38fc76a91f0a040d491723eb356d7358/applications-
Collecting applicationinsights==0.11.7 (from -r requirements.txt (line 4))
  Using cached https://files.pythonhosted.org/packages/43/c8/7848a0dd85158930b859eb8be1e38fc76a91f0a040d491723eb356d7358/applications-
Collecting arcomplete==1.9.5 (from -r requirements.txt (line 5))
  Using cached https://files.pythonhosted.org/packages/43/c8/7848a0dd85158930b859eb8be1e38fc76a91f0a040d491723eb356d7358/applications-

```

Once the environment verification is completed, (blue continue button will indicate the process is complete) you will see the following:

```

Using cached https://files.pythonhosted.org/packages/7e/91/526a6947247599b084ee5232e4f9190a38f398d7300d066af3ab571a5bfe/wcivic
Collecting websocket-client==0.56.0 (from -r requirements.txt (line 225))
  Using cached https://files.pythonhosted.org/packages/29/19/44753eab1fdb50770ac69605527e8859468f3c0fd7dc5a76dd9c4dbd7906/websocket-
Collecting xmltodict==0.12.0 (from -r requirements.txt (line 226))
  Using cached https://files.pythonhosted.org/packages/28/fd/30d5c1d3ac29cc229f6bdc40bbc20b28f716e8b363140c26eff19122d8a5/xmltodict-
Collecting zope.interface==4.6.0 (from -r requirements.txt (line 227))
Requirement already satisfied: plone in ./venv/lib/python3.6/site-packages (from azure-cli-core==2.0.63->-r requirements.txt (line 31))
Collecting wheel==0.38.0 (from azure-cli-core==2.0.63->-r requirements.txt (line 32))
  Using cached https://files.pythonhosted.org/packages/0c/80/16a865b47702a1f47a63c104c91abdd0a6704ee8a3b4ce4afc49bc39ff9d9/wheel-
Requirement already satisfied: setuptools in ./venv/lib/python3.6/site-packages (from PyHamcrest==1.9.0->-r requirements.txt (line 33))
  Installing collected packages: PyJWT, asnincrypto, six, pycparser, cffi, idna, cryptography, python-dateutil, chardet, certifi, wheel
    Found existing installation: wheel 0.33.4
      Uninstalling wheel-0.33.4:
        Successfully uninstalled wheel-0.33.4
Successfully installed Automat-0.7.0 Django-2.1.1 Jinja2-2.10.1 MarkupSafe-1.1.1 PyHamcrest-1.9.0 PyJWT-1.7.1 Py
Environment Created Successfully

```

[Continue](#)

Click on the Continue button.

The CLI login process will generate a security code that will need to be copied and pasted into the URL listed in the window. Each time the login process is run, new security code and token will be generated. Navigate to the URL in a browser on your computer:

<https://microsoft.com/devicelogin>

Provision Configuration

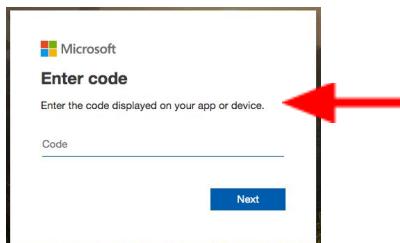
Executing Skillet: Azure Login

Logging in to Azure using device code

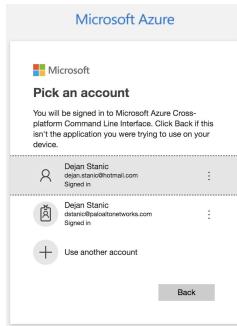
WARNING: To sign in, use a web browser to open the page <https://microsoft.com/devicelogin> and enter the code `BUDGNAHZA`.

Checking again in 2

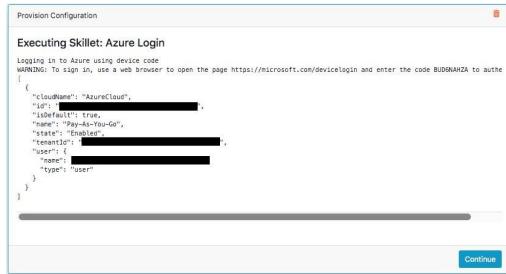
From the Microsoft website opened in your browser, enter your code into the site



You will then select the account you would like to login into Azure. You can use a personal account or the corporate account. Click on the account and the login process will be completed.



Once you log into Azure from the website, if you go back to Panhandler, you will see that Panhandler has picked up your login and will display the login as below:



Select Continue and you will be taken back to the main Azure Transit VNET Framework Window.

Launch Step 1: (Infrastructure deployment)

Click on the Go button on the step 1 deployment template.

Azure Transit VNET Framework Step 1 - Infrastructure deployment	
This skillet will log into Azure. You will be prompted to follow a link and enter a device-code in your browser.	Skillet type: python3
Collections: Azure Transit VNET, All	Go

Azure Transit VNET Framework Step 2 - Outbound Traffic inspection	
This skillet demonstrates Outbound traffic inspection thru Hub NGFW's by configuring Routes, NAT and Security policies on VM-Series. It is very important to run this skillet after Step 1 as it also updates default and spoke routes on FW.	Skillet type: panos
Collections: Azure Transit VNET, All	Go

Azure Transit VNET Framework Step 3 - Inbound Traffic inspection	
This skillet demonstrates Inbound traffic inspection thru Hub NGFW's by configuring Routes, NAT and security policies on VM-Series.	Skillet type: panos
Collections: Azure Transit VNET, All	Go

Azure Transit VNET Framework Step 4 - East West traffic Segmentation	
This skillet demonstrates East-West traffic inspection and segmentation thru Hub NGFW's by configuring Routes, Address objects, and Security policies on VM-Series.	Skillet type: panos
Collections: Azure Transit VNET, All	Go

Azure Transit VNET Framework Step 5 - Teardown	
This skillet will destroy the Azure Transit VNET Environment. Run this step once the demo is complete.	Skillet type: python3
Collections: Azure Transit VNET, All	Go

You should see the following screen:

Provision

Azure Transit VNET Framework Step 1 - Infrastructure deployment

FW Username:
paloalto Username - cannot use admin

FW Password:
***** Password - Make sure to comply with Azure password complexity requirements

Azure Resource Group:
dstanic-transit-vnet-demo-07 Create a unique resource group name

Azure Region:
North Europe Select your region from the drop down list

HUB VNET Space (i.e. transit-vnet 10.0.0.0/16):
10.0.0.0/16 Specify your HUB VNET CIDR

FW-Management Subnet Space (i.e. 10.0.0.0/24):
10.0.0.0/24 Specify your FW-Management Subnet CIDR.
Needs to be within the range of HUB VNET

UnTrust Subnet Space (i.e. 10.0.1.0/24):
10.0.1.0/24 Specify your UnTrust Subnet CIDR.
Needs to be within the range of HUB VNET

Trust Subnet Space (i.e. 10.0.2.0/24):
10.0.2.0/24 Specify your Trust Subnet CIDR.
Needs to be within the range of HUB VNET

LB-Subnet Space (i.e. 10.0.3.0/24):
10.0.3.0/24 Specify your Trust Subnet CIDR.
Needs to be within the range of HUB VNET

Spoke1 VNET Space (i.e. transit-vnet 10.1.0.0/16):
10.1.0.0/16 Specify your Spoke1 VNET CIDR

Web Subnet Space (i.e. 10.1.0.0/24):
10.1.0.0/24 Specify your Web Subnet CIDR.
Needs to be within the range of Spoke1 VNET

App Subnet Space (i.e. 10.0.1.0/24):
10.1.1.0/24 Specify your App Subnet CIDR.
Needs to be within the range of Spoke1 VNET

DB Subnet Space (i.e. 10.0.2.0/24):
10.1.2.0/24 Specify your DB Subnet CIDR.
Needs to be within the range of Spoke1 VNET

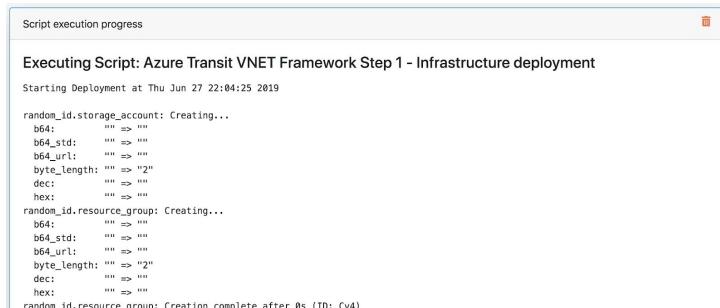
Dev Subnet Space (i.e. 10.0.3.0/24):
10.1.3.0/24 Specify your Dev Subnet CIDR.
Needs to be within the range of Spoke1 VNET

Internal Egress LB FrontEnd IP (IP must be part of LB-Subnet space i.e. 10.0.3.100):
10.0.3.100 Specify Egress Load Balancer IP Address.
Needs to be within the range of LB-Subnet of HUB VNET

Submit

Once the above information is added, select the Submit button to begin the installation.

Select the Continue button from the screen above and it will immediately start the build of the Azure Transit VNET Infrastructure, as shown below.



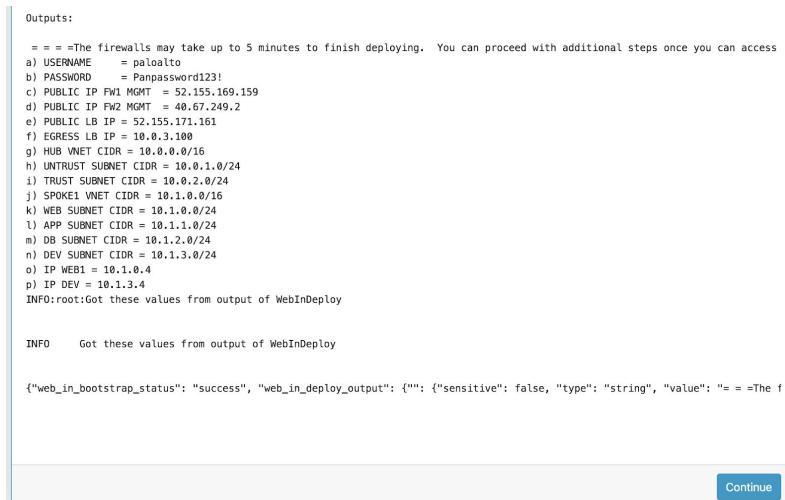
```
Script execution progress

Executing Script: Azure Transit VNET Framework Step 1 - Infrastructure deployment
Starting Deployment at Thu Jun 27 22:04:25 2019

random_id.storage_account: Creating...
b64:    "" => ""
b64_std: "" => ""
b64_url: "" => ""
byte_length: "" => "2"
dec:
hex:
random_id.resource_group: Creating...
b64:    "" => ""
b64_std: "" => ""
b64_url: "" => ""
byte_length: "" => "2"
dec:
hex:
random_id.resource_group: Creation complete after 8s (ID: fca4)
```

The build of the architecture will take approximately 8 - 10 minutes.

Once the Infrastructure build is complete, you will see the following:



```
Outputs:

= = =The firewalls may take up to 5 minutes to finish deploying. You can proceed with additional steps once you can access
a) USERNAME      = paloalto
b) PASSWORD      = Panpassword123!
c) PUBLIC IP FW1 MGMT = 52.155.169.159
d) PUBLIC IP FW2 MGMT = 40.67.249.2
e) PUBLIC LB IP = 52.155.171.161
f) EGRESS LB IP = 10.0.3.100
g) HUB VNET CIDR = 10.0.0.0/16
h) UNTRUST SUBNET CIDR = 10.0.1.0/24
i) TRUST SUBNET CIDR = 10.0.2.0/24
j) SPOKE1 VNET CIDR = 10.1.0.0/16
k) WEB SUBNET CIDR = 10.1.0.0/24
l) APP SUBNET CIDR = 10.1.1.0/24
m) DB SUBNET CIDR = 10.1.2.0/24
n) DEV SUBNET CIDR = 10.1.3.0/24
o) IP WEB1 = 10.1.0.4
p) IP DEV = 10.1.3.4
INFO:root:Got these values from output of WebInDeploy

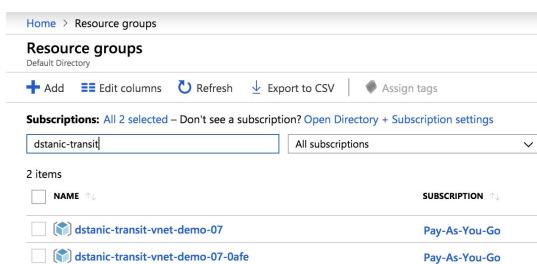
INFO      Got these values from output of WebInDeploy

{"web_in_bootstrap_status": "success", "web_in_deploy_output": {""": {"sensitive": false, "type": "string", "value": "= =The f
```

Continue

Copy deployment Output values into the text editor and select Continue from the screen above to take you back to the main Azure Transit VNET Framework page.

Now is a great time to login to your Azure environment and review deployed infrastructure.



Home > Resource groups

Resource groups

Default Directory

Add Edit columns Refresh Export to CSV Assign tags

Subscriptions: All 2 selected – Don't see a subscription? Open Directory + Subscription settings

NAME	SUBSCRIPTION
dstanic-transit-vnet-demo-07	Pay-As-You-Go
dstanic-transit-vnet-demo-07-0afe	Pay-As-You-Go

As you can see above, the deployment process created two Resource Groups:

dstanic-transit-vnet-demo-07-0afe - used for bootstrapping, hosts Storage account with Fileshare

dstanic-transit-vnet-demo-07 - all infrastructure resources are deployed in here

<input type="checkbox"/> NAME	<input type="checkbox"/> TYPE	<input type="checkbox"/> LOCATION
<input type="checkbox"/> fw1-mgmt-pip	Public IP address	North Europe
<input type="checkbox"/> fw1-untrust-pip	Public IP address	North Europe
<input type="checkbox"/> fw2-mgmt-pip	Public IP address	North Europe
<input type="checkbox"/> fw2-untrust-pip	Public IP address	North Europe
<input type="checkbox"/> Public-LB	Public IP address	North Europe
<input type="checkbox"/> RT-DevSubnet	Route table	North Europe
<input type="checkbox"/> RT-WebSubnet	Route table	North Europe
<input type="checkbox"/> Dev1	Virtual machine	North Europe
<input type="checkbox"/> VM-FW1	Virtual machine	North Europe
<input type="checkbox"/> VM-FW2	Virtual machine	North Europe
<input type="checkbox"/> Web1	Virtual machine	North Europe
<input type="checkbox"/> Hub-VNET	Virtual network	North Europe
<input type="checkbox"/> Spoke1-VNET	Virtual network	North Europe

The best way to ensure that FW's are ready for the next step is to login to the management console. Select PUBLIC IP FW1 MGMT, PUBLIC IP FW2 MGMT from output values and paste the IP's into your web browser. If needed, credentials can also be found in the Outputs. While reviewing Policies, Objects and Network tabs you will notice that only very basic configuration was loaded and that we are missing some Routes, Objects, Security and NAT policies. This is desired configuration as we will configure our FW's to enable different use-cases in Steps 2, 3 and 4.

The screenshot shows the Palo Alto Networks Management Console interface. At the top, there is a navigation bar with links to Apps, Google, LinkedIn, Palo Alto Networks, IP Calculator / IP..., Azure, AWS Console, Palo Alto Network..., and Google Calendar. Below the navigation bar, the main header says "paloalto NETWORKS".

The top navigation bar has tabs: Dashboard, ACC, Monitor, Policies (which is selected), Objects, Network, and Device.

The left sidebar contains a tree view of security features: Security, NAT, QoS, Policy Based Forwarding, Decryption, Tunnel Inspection, Application Override, Authentication, and DoS Protection.

The "Policies" tab displays a table for "Source" rules:

Name	Tags	Type	Zone	Address	User	HIP Profile	Zone
allow-azure-health-pr...	none	universal	any	168.63.129.16	any	any	any
intrazone-default	none	intrazone	any	any	any	any	(intrazone)
interzone-default	none	interzone	any	any	any	any	any

The "Network" tab displays a table for interfaces:

Interface	Interface Type	Management Profile	Link State	IP Address	Virtual Router	Tag
ethernet1/1	Layer3	allow-health-probe	Dynamic-DHCP Client	untrust-vr	Untagged	
ethernet1/2	Layer3	allow-health-probe	Dynamic-DHCP Client	trust-vr	Untagged	
ethernet1/3			none	none	Untagged	
ethernet1/4			none	none	Untagged	
ethernet1/5			none	none	Untagged	
ethernet1/6			none	none	Untagged	
ethernet1/7			none	none	Untagged	

You are now ready to run the first use-case and secure outbound traffic.

Launch Step 2: (Outbound traffic inspection)

During the deployment process, we have deployed two Linux test instances Web1 and Dev1 in corresponding subnets of Spoke1-VNET. Before launching the Step 2 login to your Dev1 instance and try to access public resources (i.e. wget www.google.com or ping 8.8.8.8).

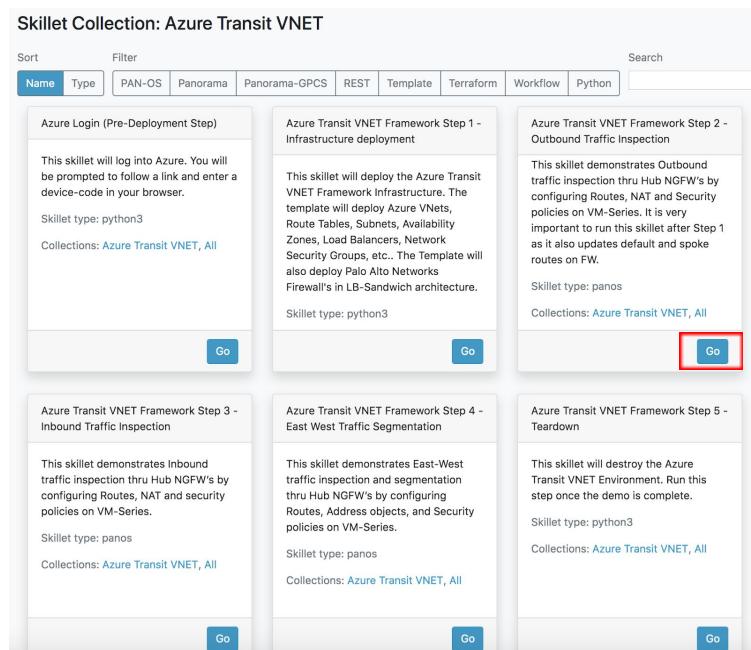


The screenshot shows the 'Dev1 - Serial console' interface. On the left, there's a sidebar with a search bar, configuration management, policies, run command, monitoring (insights, alerts), and a '...' button. The main area is a terminal window with the following output:

```
paloalto@Dev1:~$ wget www.linkedin.com
--2019-06-28 07:42:21-- http://www.linkedin.com/
Resolving www.linkedin.com (www.linkedin.com)...
^C
paloalto@Dev1:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
```

You will notice that at this stage it is not possible to reach them out.

To initiate Step 2 go back to the Panhandler, Click on the Go button on the step 2 Outbound Traffic Inspection deployment template.



The screenshot shows the 'Skillet Collection: Azure Transit VNET' interface. It lists five deployment steps:

- Azure Login (Pre-Deployment Step)**: Skillet type: python3, Go button.
- Azure Transit VNET Framework Step 1 - Infrastructure deployment**: Skillet type: python3, Go button.
- Azure Transit VNET Framework Step 2 - Outbound Traffic Inspection**: Skillet type: panos, Go button (highlighted with a red box).
- Azure Transit VNET Framework Step 3 - Inbound Traffic Inspection**: Skillet type: panos, Go button.
- Azure Transit VNET Framework Step 4 - East West Traffic Segmentation**: Skillet type: panos, Go button.
- Azure Transit VNET Framework Step 5 - Teardown**: Skillet type: python3, Go button.

This will launch the inputs page. You should see the following screen:

PAN-OS Configuration

Customize PAN-OS Skillet: Azure Transit VNET Framework Step 2 - Outbound Traffic Inspection

Security policy name:
allow-all-out Specify your Outbound Security Policy Name

Nat policy name:
outbound-nat Specify your Outbound NAT Policy Name

First IP from UnTrust Subnet CIDR (i.e. 10.0.1.1):
10.0.1.1 Important - Specify first IP of your UnTrust Subnet

First IP from Trust Subnet CIDR (i.e. 10.0.2.1):
10.0.2.1 Important - Specify first IP of your UnTrust Subnet

Spoke1 VNET Space (i.e. transit-vnet 10.1.0.0/16):
10.1.0.0/16 Important - Specify your Spoke1 VNET CIDR

Submit

Once the selection is complete, click “Submit” to initiate the configuration process.

PAN-OS Skillet

Configure Target information

PAN-OS IP:
52.155.169.159

PAN-OS Username:
paloalto

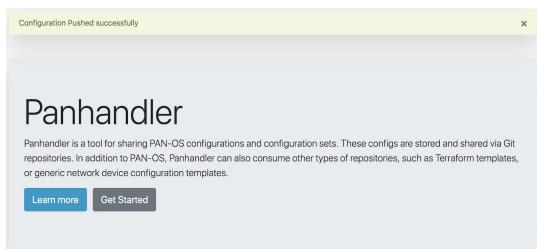
PAN-OS Password:

Commit Options:
Commit and wait to finish
 Perform Backup

Debug Submit

Provide the FW1 details (all details can be found in output values), click “Submit” to initiate API calls that will create Routes, Security and NAT Rules to enable Outbound traffic.

Once the process is complete, you will see the following:



Repeat the process to configure FW2. To simplify the process, select Back in your browser, change PAN-OS IP value to FW2, provide your password and click “Submit”.

Go back to your FW1 and FW2 management console and review Virtual Routers untrust-vr and trust-vr, Security and NAT policies.

Virtual Router:

Name	Destination	Interface	Type	Value	Admin Distance	Metric	BFD	Route Table
default	0.0.0.0/0	ethernet1/1	ip-address	10.0.1.1	default	10	None	unicast
spoke1-vnet	10.1.0.0/16	next-vr	trust-vr		default	10	None	unicast

Security Policies:

	Name	Tags	Type	Zone	Address	User	HIP Profile	Zone	Address	Application	Service	Action	Profile
1	allow-azure-health-nr	none	universal	any	168.63.129.16	any	any	any	any	esh	application-id	Allow	none
2	allow-all-out	none	universal	trust-zone	any	any	any	untrust-zone	any	any	any	Allow	any
3	intrazone-default	none	intrazone	any	any	any	any	(Intrazone)	any	any	any	Deny	none
4	interzone-default	none	interzone	any	any	any	any	any	any	any	any	Deny	none

NAT Rules:

	Name	Tags	Source Zone	Destination Zone	Destination Interface	Source Address	Destination Address	Service	Source Translation	Destination Translation	Hit Count
1	azure-probe-no-nat	none	untrust-zone	untrust-zone	any	168.63.129.16/...	any	any	none	none	2377
2	outbound-nat	none	trust-zone	untrust-zone	any	any	any	any	dynamic-ip-and-port	none	1330

Login to your Dev1 instance and try again to access public resources (i.e. wget www.google.com or ping 8.8.8.8). This time you should be able to access public resources.

Dev1 - Serial console

Virtual machine

Search (Ctrl+)

Feedback

Configuration management ...

Policies

Run command

Monitoring

Insights (preview)

Alerts

Metrics

Diagnostic settings

Advisor recommendations

Logs

Connection monitor

Support + troubleshooting

Resource health

Boot diagnostics

Reset password

Redeploy

Ubuntu Advantage support ...

```
paloalto@dev1:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=52 time=2.74 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=52 time=2.63 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=52 time=3.02 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=52 time=8.66 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=52 time=2.51 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=52 time=2.56 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=52 time=2.41 ms
64 bytes from 8.8.8.8: icmp_seq=8 ttl=52 time=2.52 ms
64 bytes from 8.8.8.8: icmp_seq=9 ttl=52 time=5.98 ms
```
--- 8.8.8.8 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8012ms
rtt min/avg/max/mdev = 2.41ms/6.73ms/8.660ms/2.056ms
paloalto@dev1:~$ curl https://www.linkedin.com/
--2019-06-28 08:28:33-- https://www.linkedin.com/
Resolving www.linkedin.com (www.linkedin.com)... 185.63.145.1, 2a05:f500:10:101::b93f:9101
Connecting to www.linkedin.com (www.linkedin.com)|185.63.145.1|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://www.linkedin.com/ [following]
--2019-06-28 08:28:33-- https://www.linkedin.com/
Connecting to www.linkedin.com (www.linkedin.com)|185.63.145.1|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 67253 (66K) [text/html]
Saving to: 'index.html.3'

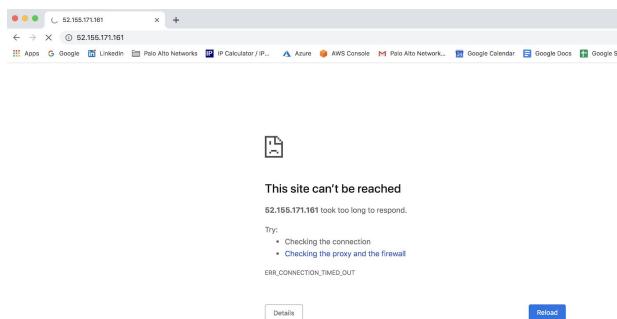
index.html.3 100%[=====] 65.68K 413KB/s in 0.28
2019-06-28 08:28:34 (413 KB/s) - 'index.html.3' saved [67253/67253]
paloalto@dev1:~$
```

Select the Monitor tab on FW1 and FW2 and review the traffic log. You should see similar to the picture below.

|   | Receive Time   | Type | From Zone  | To Zone      | Source   | Source User | Destination  | To Port | Application   | Action | Rule          | Session End Reason | Bytes | HTTP/2 Connection Session ID |
|---|----------------|------|------------|--------------|----------|-------------|--------------|---------|---------------|--------|---------------|--------------------|-------|------------------------------|
| 🔗 | 06/28 01:29:00 | end  | trust-zone | untrust-zone | 10.1.3.4 |             | 8.8.8.8      | 53      | dns           | allow  | allow-all-out | aged-out           | 526   | 0                            |
| 🔗 | 06/28 01:28:47 | end  | trust-zone | untrust-zone | 10.1.3.4 |             | 185.63.145.1 | 80      | linkedin-base | allow  | allow-all-out | tcp-fin            | 2.1k  | 0                            |
| 🔗 | 06/28 01:28:47 | end  | trust-zone | untrust-zone | 10.1.3.4 |             | 185.63.145.1 | 443     | linkedin-base | allow  | allow-all-out | tcp-fin            | 79.8k | 0                            |
| 🔗 | 06/28 01:28:34 | end  | trust-zone | untrust-zone | 10.1.3.4 |             | 8.8.8.8      | 0       | ping          | allow  | allow-all-out | aged-out           | 588   | 0                            |
| 🔗 | 06/28 01:28:28 | end  | trust-zone | untrust-zone | 10.1.3.4 |             | 8.8.8.8      | 0       | ping          | allow  | allow-all-out | aged-out           | 1.2k  | 0                            |
| 🔗 | 06/28 01:28:22 | end  | trust-zone | untrust-zone | 10.1.3.4 |             | 8.8.8.8      | 0       | ping          | allow  | allow-all-out | aged-out           | 392   | 0                            |
| 🔗 | 06/28 01:28:16 | end  | trust-zone | untrust-zone | 10.1.3.4 |             | 8.8.8.8      | 0       | ping          | allow  | allow-all-out | aged-out           | 1.2k  | 0                            |
| 🔗 | 06/28 01:28:01 | end  | trust-zone | untrust-zone | 10.1.3.4 |             | 8.8.8.8      | 0       | ping          | allow  | allow-all-out | aged-out           | 1.2k  | 0                            |
| 🔗 | 06/28 01:27:55 | end  | trust-zone | untrust-zone | 10.1.3.4 |             | 8.8.8.8      | 0       | ping          | allow  | allow-all-out | aged-out           | 1.2k  | 0                            |
| 🔗 | 06/28 01:27:49 | end  | trust-zone | untrust-zone | 10.1.3.4 |             | 8.8.8.8      | 0       | ping          | allow  | allow-all-out | aged-out           | 1.2k  | 0                            |

### Launch Step 3: (Inbound traffic inspection)

During the deployment process, we have deployed Web1 with Apache web server in Spoke1-VNET. Before launching the Step 3 select PUBLIC LB IP from output values and paste the IP's into your web browser.



You will notice that at this stage it is not possible to reach out your Web1 server.

To initiate Step 3 go back to the Panhandler, Click on the Go button on Step 3 Inbound Traffic Inspection deployment template.

Skillet Collection: Azure Transit VNET

Sort Filter Search

Name Type PAN-OS Panorama Panorama-GPCS REST Template Terraform Workflow Python

|                                                                                                                                                                                                    |                                                                                                                                                                                                                                                                                                                             |                                                                                                                                                                                                                                                                                                              |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Azure Login (Pre-Deployment Step)                                                                                                                                                                  | Azure Transit VNET Framework Step 1 - Infrastructure deployment                                                                                                                                                                                                                                                             | Azure Transit VNET Framework Step 2 - Outbound Traffic Inspection                                                                                                                                                                                                                                            |
| This skillet will log into Azure. You will be prompted to follow a link and enter a device-code in your browser.<br>Skillet type: python3<br>Collections: Azure Transit VNET, All                  | This skillet will deploy the Azure Transit VNET Framework Infrastructure. The template will deploy Azure VNets, Route Tables, Subnets, Availability Zones, Load Balancers, Network Security Groups, etc.. The Template will also deploy Palo Alto Networks Firewall's in LB-Sandwich architecture.<br>Skillet type: python3 | This skillet demonstrates Outbound traffic inspection thru Hub NGFW's by configuring Routes, NAT and Security policies on VM-Series. It is very important to run this skillet after Step 1 as it also updates default and spoke routes on FW.<br>Skillet type: panos<br>Collections: Azure Transit VNET, All |
| <button>Go</button>                                                                                                                                                                                | <button>Go</button>                                                                                                                                                                                                                                                                                                         | <button>Go</button>                                                                                                                                                                                                                                                                                          |
| Azure Transit VNET Framework Step 3 - Inbound Traffic Inspection                                                                                                                                   | Azure Transit VNET Framework Step 4 - East West Traffic Segmentation                                                                                                                                                                                                                                                        | Azure Transit VNET Framework Step 5 - Teardown                                                                                                                                                                                                                                                               |
| This skillet demonstrates Inbound traffic inspection thru Hub NGFW's by configuring Routes, NAT and security policies on VM-Series.<br>Skillet type: panos<br>Collections: Azure Transit VNET, All | This skillet demonstrates East-West traffic inspection and segmentation thru Hub NGFW's by configuring Routes, Address objects, and Security policies on VM-Series.<br>Skillet type: panos<br>Collections: Azure Transit VNET, All                                                                                          | This skillet will destroy the Azure Transit VNET Environment. Run this step once the demo is complete.<br>Skillet type: python3<br>Collections: Azure Transit VNET, All                                                                                                                                      |
| <button>Go</button>                                                                                                                                                                                | <button>Go</button>                                                                                                                                                                                                                                                                                                         | <button>Go</button>                                                                                                                                                                                                                                                                                          |

This will launch the inputs page. You should see the following screen:

PAN-OS Configuration

Customize PAN-OS Skillet: Azure Transit VNET Framework Step 3 - Inbound Traffic Inspection

Public Load Balancer Public Frontend IP:  
52.155.171.161

Web server Web1 IP:  
10.1.0.4

Inbound security policy name:  
allow-web-in

Inbound NAT policy name:  
inbound-nat

Important - Specify Public LB IP. It can be found in Outputs

Important - Specify Web1 IP. It can be found in Outputs

Specify your Inbound Security Policy Name

Specify your Inbound NAT Policy Name

Submit

Once the selection is complete, click "Submit" to initiate the configuration process.

PAN-OS Skillet

Configure Target information

PAN-OS IP:  
52.155.169.159

PAN-OS Username:  
paloalto

PAN-OS Password:  
\*\*\*\*\*

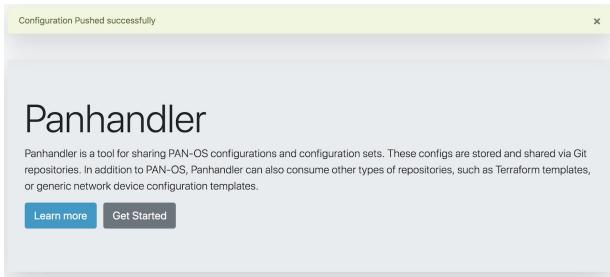
Commit Options:

- Commit and wait to finish
- Perform Backup

Debug Submit

Provide the FW1 details (all details can be found in output values), click “Submit” to initiate API calls that will create Objects, Security and NAT Rules to enable Inbound traffic.

Once the process is complete, you will see the following:



Repeat the process to configure FW2. To simplify the process, select Back in your browser, change PAN-OS IP value to FW2, provide your password and click “Submit”.

Go back to your FW1 and FW2 management console and review Address objects, Security and NAT policies.

Address objects:

| Name          | Location | Type       | Address        |
|---------------|----------|------------|----------------|
| public-lb-pip |          | IP Netmask | 52.155.171.161 |
| web1          |          | IP Netmask | 10.1.0.4       |

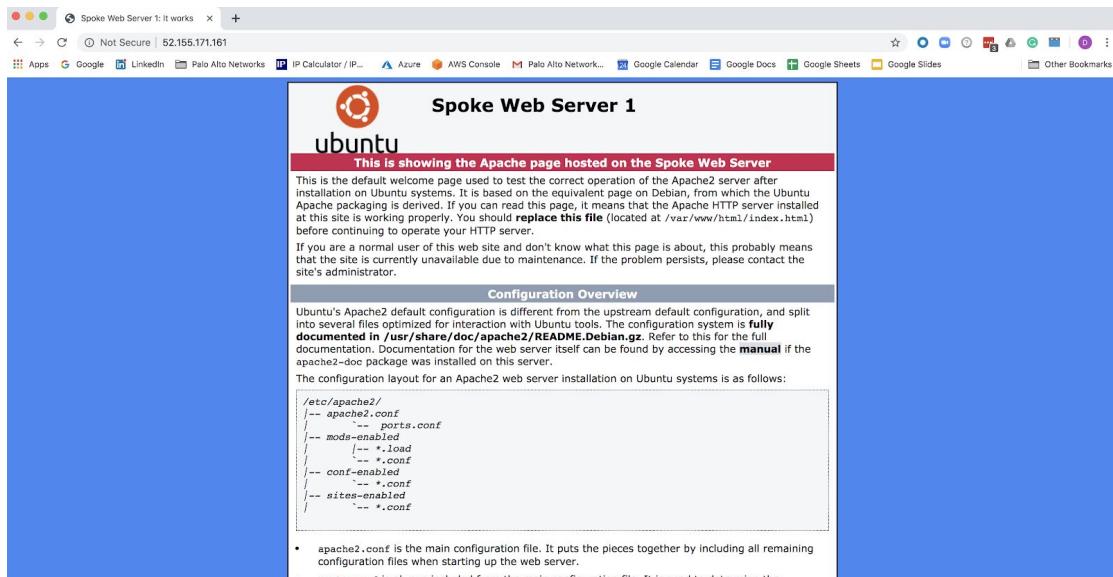
## Security Policies:

|   | Name                     | Tags | Type      | Zone         | Address       | User | HIP Profile | Zone         | Address | Application  | Service          | Action | Profile |
|---|--------------------------|------|-----------|--------------|---------------|------|-------------|--------------|---------|--------------|------------------|--------|---------|
| 1 | allow-azure-health-pr... | none | universal | any          | 168.63.129.16 | any  | any         | any          | any     | ssh          | application-d... | Allow  | none    |
| 2 | allow-all-out            | none | universal | trust-zone   | any           | any  | any         | untrust-zone | any     | any          | any              | Allow  | any     |
| 3 | allow-web-in             | none | universal | untrust-zone | any           | any  | any         | trust-zone   | any     | web-browsing | application-d... | Allow  | none    |

## NAT Rules:

|   | Name               | Tags | Original Packet |                  |                       |                   |                     |         | Translated Packet   |                         |               | Hit Count |
|---|--------------------|------|-----------------|------------------|-----------------------|-------------------|---------------------|---------|---------------------|-------------------------|---------------|-----------|
|   |                    |      | Source Zone     | Destination Zone | Destination Interface | Source Address    | Destination Address | Service | Source Translation  | Destination Translation |               |           |
| 1 | azure-probe-no-nat | none | untrust-zone    | untrust-zone     | any                   | 168.63.129.16/... | any                 | any     | none                | none                    |               | 5380      |
| 2 | outbound-nat       | none | trust-zone      | untrust-zone     | any                   | any               | any                 | any     | dynamic-ip-and-port | none                    |               | 1642      |
| 3 | inbound-nat        | none | untrust-zone    | untrust-zone     | any                   | any               | public-lb-pip       | any     | none                | destination-translation | address: web1 | 6         |
|   |                    |      |                 |                  |                       |                   |                     |         |                     |                         | port: 80      |           |

Refresh your browser with PUBLIC LB IP from output values. This time you should see the following:

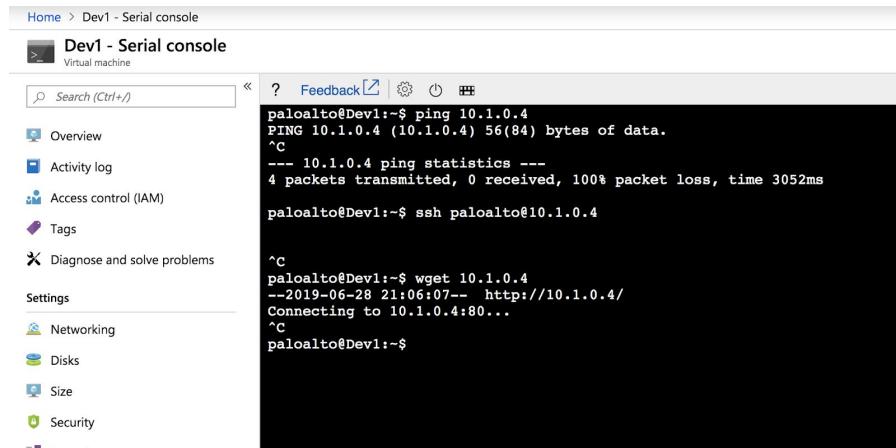


Select the Monitor tab on FW1 and FW2 and review the traffic log. You should see similar to the picture below.

| Receive Time   | Type | From Zone    | To Zone      | Source        | Source User | Destination    | To Port | Application    | Action | Rule              | Session End Reason | Bytes | HTTP/2 Connection Session ID |
|----------------|------|--------------|--------------|---------------|-------------|----------------|---------|----------------|--------|-------------------|--------------------|-------|------------------------------|
| 06/28 14:00:01 | end  | untrust-zone | trust-zone   | 84.207.227.14 |             | 52.155.171.161 | 80      | incomplete     | allow  | allow-web-in      | tcp-fin            | 416   | 0                            |
| 06/28 13:59:56 | end  | untrust-zone | trust-zone   | 84.207.227.14 |             | 52.155.171.161 | 80      | web-browsing   | allow  | allow-web-in      | tcp-fin            | 5.0k  | 0                            |
| 06/28 13:53:59 | end  | untrust-zone | trust-zone   | 84.207.227.14 |             | 52.155.171.161 | 80      | incomplete     | allow  | allow-web-in      | tcp-fin            | 556   | 0                            |
| 06/28 13:53:59 | end  | untrust-zone | trust-zone   | 84.207.227.14 |             | 52.155.171.161 | 80      | web-browsing   | allow  | allow-web-in      | tcp-fin            | 18.3k | 0                            |
| 06/28 13:51:29 | drop | untrust-zone | untrust-zone | 84.207.227.14 |             | 52.155.171.161 | 80      | not-applicable | deny   | intrazone-default | policy-deny        | 62    | 0                            |
| 06/28 13:51:28 | drop | untrust-zone | untrust-zone | 84.207.227.14 |             | 52.155.171.161 | 80      | not-applicable | deny   | intrazone-default | policy-deny        | 62    | 0                            |
| 06/28 13:51:28 | drop | untrust-zone | untrust-zone | 84.207.227.14 |             | 52.155.171.161 | 80      | not-applicable | deny   | intrazone-default | policy-deny        | 62    | 0                            |
| 06/28 13:50:57 | drop | untrust-zone | untrust-zone | 84.207.227.14 |             | 52.155.171.161 | 80      | not-applicable | deny   | intrazone-default | policy-deny        | 78    | 0                            |
| 06/28 13:50:56 | drop | untrust-zone | untrust-zone | 84.207.227.14 |             | 52.155.171.161 | 80      | not-applicable | deny   | intrazone-default | policy-deny        | 78    | 0                            |
| 06/28 13:50:56 | drop | untrust-zone | untrust-zone | 84.207.227.14 |             | 52.155.171.161 | 80      | not-applicable | deny   | intrazone-default | policy-deny        | 78    | 0                            |
| 06/28 13:50:41 | drop | untrust-zone | untrust-zone | 84.207.227.14 |             | 52.155.171.161 | 80      | not-applicable | deny   | intrazone-default | policy-deny        | 78    | 0                            |
| 06/28 13:50:40 | drop | untrust-zone | untrust-zone | 84.207.227.14 |             | 52.155.171.161 | 80      | not-applicable | deny   | intrazone-default | policy-deny        | 78    | 0                            |

## Launch Step 4: (East-West traffic inspection and segmentation)

During the deployment process, we have deployed two Linux test instances in Spoke1-VNET Web1 in Web Subnet and Dev1 in Dev Subnet. Before launching the Step 4 login to your Dev1 instance and try to ping, ssh and wget your Web1 instance.



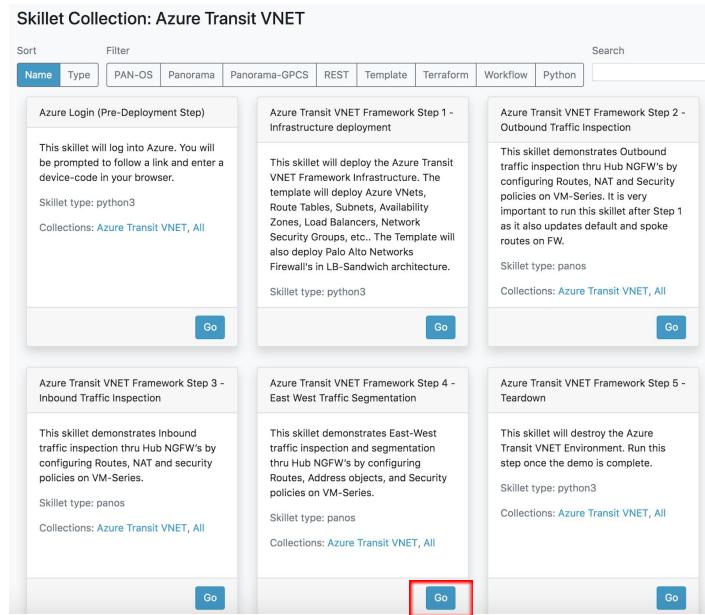
The screenshot shows a terminal window titled "Dev1 - Serial console". The terminal output is as follows:

```
paloalto@Dev1:~$ ping 10.1.0.4
PING 10.1.0.4 (10.1.0.4) 56(84) bytes of data.
^C
--- 10.1.0.4 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3052ms
paloalto@Dev1:~$ ssh paloalto@10.1.0.4

^C
paloalto@Dev1:~$ wget 10.1.0.4
--2019-06-28 21:06:07-- http://10.1.0.4/
Connecting to 10.1.0.4:80...
^C
paloalto@Dev1:~$
```

You will notice that at this stage it is not possible to reach Web1.

To initiate the Step 4 go back to the Panhandler, Click on the Go button on Step 4 - East-West Traffic Segmentation deployment template. This will launch the inputs page.



The screenshot shows the "Skillet Collection: Azure Transit VNET" page. It lists five deployment steps:

- Azure Login (Pre-Deployment Step): Skillet type: python3, Collections: Azure Transit VNET, All. Go button.
- Azure Transit VNET Framework Step 1 - Infrastructure deployment: Skillet type: python, Collections: Azure Transit VNET, All. Go button.
- Azure Transit VNET Framework Step 2 - Outbound Traffic Inspection: Skillet type: panos, Collections: Azure Transit VNET, All. Go button.
- Azure Transit VNET Framework Step 3 - Inbound Traffic Inspection: Skillet type: panos, Collections: Azure Transit VNET, All. Go button.
- Azure Transit VNET Framework Step 4 - East West Traffic Segmentation: Skillet type: panos, Collections: Azure Transit VNET, All. Go button (highlighted with a red box).
- Azure Transit VNET Framework Step 5 - Teardown: Skillet type: python3, Collections: Azure Transit VNET, All. Go button.

You should see the following screen:

PAN-OS Configuration

Customize PAN-OS Skillet: Azure Transit VNET Framework Step 4 - East West Traffic Segmentation

WebSubnet cidr:  
10.1.0.0/24 Specify Web Subnet CIDR

AppSubnet cidr:  
10.1.1.0/24 Specify App Subnet CIDR

DBSubnet cidr:  
10.1.2.0/24 Specify DB Subnet CIDR

DevSubnet cidr:  
10.1.3.0/24 Specify Dev Subnet CIDR

East West Security policy name:  
allow-all-web-dev Specify your East-West Security Policy Name

Source Subnet1 for example dev-subnet:  
web-subnet Select Web-Subnet from the drop down list

Source Subnet2 for example dev-subnet:  
dev-subnet Select Dev-Subnet from the drop down list

Destination Subnet1 for example web-subnet:  
dev-subnet Select Dev-Subnet from the drop down list

Destination Subnet2 for example dev-subnet:  
web-subnet Select Web-Subnet from the drop down list

Application type1 for example any, web-browsing, ssh:  
web Select first application type web

Application type2 for example any, web-browsing, ssh:  
ssh Select second application type ssh

Action type allow or deny:  
allow Select Action Type Allow

Submit

Once the selection is complete, click “Submit” to initiate the configuration process.

PAN-OS Skillet

Configure Target information

PAN-OS IP:  
52.155.169.159

PAN-OS Username:  
paloalto

PAN-OS Password:  
\*\*\*\*\*

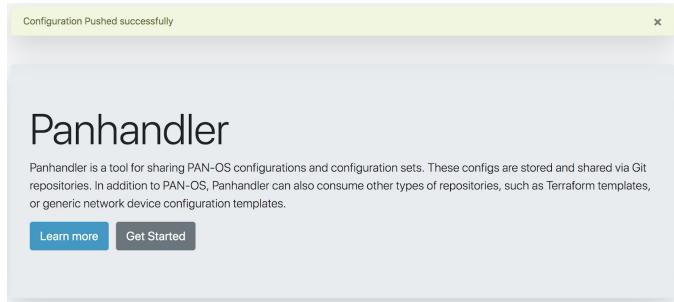
Commit Options:  
Fast Commit. Do not wait on commit to finish

Perform Backup

Debug Submit

Provide the FW1 details (all details can be found in output values), click “Submit” to initiate API calls that will create Objects and Security policies to enable East-West ssh and web traffic between Web and Dev subnets.

Once the process is complete, you will see the following:



Repeat the process to configure FW2. To simplify the process, select Back in your browser, change PAN-OS IP value to FW2, provide your password and click “Submit”.

Go back to your FW1 and FW2 management console and review Address objects and Security policies.

### Address Objects:

A screenshot of the Palo Alto Networks management console. The top navigation bar includes Dashboard, ACC, Monitor, Policies, Objects (which is selected), Network, and Device. On the left, there's a sidebar with icons for Address Groups, Regions, Applications, Application Groups, Application Filters, Services, Service Groups, Tags, GlobalProtect, HIP Objects, and HIP Profiles. The main pane shows a table of Address objects. A red box highlights the first five rows: app-subnet, db-subnet, dev-subnet, public-lb-pip, and web1. The columns include Name, Location, Type, and Address. The last row, web1, is also highlighted with a red box.

### Security Policies:

| Name                     | Tags | Type      | Source       |               |      |             | Destination  |            |            |              | Application      | Service | Action | Profile | Options |
|--------------------------|------|-----------|--------------|---------------|------|-------------|--------------|------------|------------|--------------|------------------|---------|--------|---------|---------|
|                          |      |           | Zone         | Address       | User | HIP Profile | Zone         | Address    |            |              |                  |         |        |         |         |
| allow-azure-health-pr... | none | universal | any          | 168.63.129.16 | any  | any         | any          | any        | any        | ssh          | application-d... | Allow   | none   |         |         |
| allow-all-out            | none | universal | trust-zone   | any           | any  | any         | untrust-zone | any        | any        | any          | any              | any     | Allow  |         |         |
| allow-web-In             | none | universal | untrust-zone | any           | any  | any         | trust-zone   | any        | any        | web-browsing | application-d... | Allow   | none   |         |         |
| allow-all-web-dev        | none | universal | trust-zone   | web-subnet    | any  | any         | trust-zone   | dev-subnet | web-subnet | web-browsing | application-d... | Allow   | none   |         |         |
| intrazone-default        | none | intrazone | any          | any           | any  | any         | (intrazone)  | any        | any        | any          | any              | Deny    | none   |         |         |
| interzone-default        | none | interzone | any          | any           | any  | any         | any          | any        | any        | any          | any              | Deny    | none   |         |         |

Login to your Dev1 instance and try again to ping, ssh and wget your Web1 instance. You should be able to wget and ssh but not to ping. This is because we created a security policy that only allows ssh and web-browsing between Dev and Web Subnets.

Home > Dev1 - Serial console

### Dev1 - Serial console

Virtual machine

Search (Ctrl+ /)

- Overview
- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems
- Settings
  - Networking
  - Disks
  - Size
  - Security
  - Extensions
  - Continuous delivery (Preview)
  - Availability set

```

? Feedback | ⚙️ ⏹
palaloalto@Dev1:~$ wget 10.1.0.4
--2019-06-28 21:28:30-- http://10.1.0.4/
Connecting to 10.1.0.4:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 11356 (11K) [text/html]
Saving to: 'index.html.5'

index.html.5 100%[=====] 11.09K ---KB/s in 0s

2019-06-28 21:28:30 (224 MB/s) - 'index.html.5' saved [11356/11356]

palaloalto@Dev1:~$ ssh palaloalto@10.1.0.4
palaloalto@10.1.0.4's password:
Permission denied, please try again.
palaloalto@10.1.0.4's password:

palaloalto@Dev1:~$ ping 10.1.0.4
PING 10.1.0.4 (10.1.0.4) 56(84) bytes of data.
^C
--- 10.1.0.4 ping statistics ---
8 packets transmitted, 0 received, 100% packet loss, time 7172ms

palaloalto@Dev1:~$
```

Select the Monitor tab on FW1 and FW2 and review the traffic log. You should see similar to the picture below.

|   | ( addr.src in 10.1.3.4 ) |      |            |            |          |             |             |         |              |        |                   |                    |       |                              |
|---|--------------------------|------|------------|------------|----------|-------------|-------------|---------|--------------|--------|-------------------|--------------------|-------|------------------------------|
|   | Receive Time             | Type | From Zone  | To Zone    | Source   | Source User | Destination | To Port | Application  | Action | Rule              | Session End Reason | Bytes | HTTP/2 Connection Session ID |
| 🔗 | 06/28 14:28:56           | drop | trust-zone | trust-zone | 10.1.3.4 |             | 10.1.0.4    | 0       | ping         | deny   | intrazone-default | policy-deny        | 196   | 0                            |
| 🔗 | 06/28 14:28:55           | end  | trust-zone | trust-zone | 10.1.3.4 |             | 10.1.0.4    | 22      | ssh          | allow  | allow-all-web-dev | tcp-fin            | 5.0k  | 0                            |
| 🔗 | 06/28 14:28:50           | drop | trust-zone | trust-zone | 10.1.3.4 |             | 10.1.0.4    | 0       | ping         | deny   | intrazone-default | policy-deny        | 588   | 0                            |
| 🔗 | 06/28 14:28:43           | end  | trust-zone | trust-zone | 10.1.3.4 |             | 10.1.0.4    | 80      | web-browsing | allow  | allow-all-web-dev | tcp-fin            | 13.0k | 0                            |
| 🔗 | 06/28 14:28:36           | end  | trust-zone | trust-zone | 10.1.3.4 |             | 10.1.0.4    | 22      | ssh          | allow  | allow-all-web-dev | tcp-fin            | 4.8k  | 0                            |
| 🔗 | 06/28 14:27:32           | end  | trust-zone | trust-zone | 10.1.3.4 |             | 10.1.0.4    | 80      | web-browsing | allow  | allow-all-web-dev | tcp-fin            | 13.1k | 0                            |

## Launch Step 5:

Click on the Go button on the Step 5 deployment template. This will clean up the environment, destroying everything thus far created.

The screenshot shows a grid of deployment templates. The Step 5 template, titled "Azure Transit VNET Framework Step 5 - Teardown", is highlighted with a red border around its "Go" button. The other templates are: "Azure Login (Pre-Deployment Step)", "Azure Transit VNET Framework Step 1 - Infrastructure deployment", "Azure Transit VNET Framework Step 2 - Outbound Traffic Inspection", "Azure Transit VNET Framework Step 3 - Inbound Traffic Inspection", "Azure Transit VNET Framework Step 4 - East West Traffic Segmentation", and "Azure Transit VNET Framework Step 5 - Teardown". Each template has a brief description, the type (Skillet type: python or panos), and a "Go" button.

You should see the following screen:

This is a configuration form for the Step 5 Teardown skillet. It includes fields for FW Username (set to "palalto") and FW Password (redacted). At the bottom is a "Submit" button.

Select Submit. Note that this process will take several minutes to complete. The process is complete when you get the “Continue” button at the bottom of the page.

```
DEBUG:srest.http_logger: 'Pragma': 'no-cache'
DEBUG:srest.http_logger: 'Expires': '-1'
DEBUG:srest.http_logger: 'x-ms-rateLimit-remaining-subscription-reads': '11998'
DEBUG:srest.http_logger: 'x-ms-request-id': 'a01a1f84-bda2-4fb5-9aa2-5695403cc75a'
DEBUG:srest.http_logger: 'x-ms-correlation-request-id': 'a01a1f84-bda2-4fb5-9aa2-5695403cc75a'
DEBUG:srest.http_logger: 'x-ms-routing-request-id': 'WESTCENTRALUS:20190610T210023:a01a1f84-bda2-4fb5-9aa2-5695403cc75a'
DEBUG:srest.http_logger: 'Strict-Transport-Security': 'max-age=31536000; includeSubDomains'
DEBUG:srest.http_logger: 'X-Content-Type-Options': 'nosniff'
DEBUG:srest.http_logger: 'Date': 'Mon, 10 Jun 2019 21:00:23 GMT'
DEBUG:srest.http_logger: 'Content-Length': '0'
DEBUG:srest.http_logger:Response content:
DEBUG:srest.http_logger:
Removed state file ./WebInBootstrap/terraform.tfstate
```

