VM Series and Azure GWLB

Deployment Guide

Version-0.1

Date: 09-21-2022

Author: Ravi Sankar Pegada

Table of Contents

About This Guide

Azure Gateway Load Balancer helps to easily deploy, scale, and manage VM-Series firewalls referred to as Network Virtual Appliances (NVAs) in Azure. Chaining a Gateway Load Balancer to your public endpoint requires a simple configuration.

Gateway Load Balancer provides the bump-in-the-wire technology to ensure all traffic to a public endpoint is first sent to the VM-Series before it is sent to an application.
In scenarios with Stateful Firewalls such as the VM-Series,it's especially important that flows are symmetrical. The Gateway Load Balancer maintains flow stickiness to a specific Firewall instance in the backend pool along with flow symmetry. As a result, a consistent route to your network virtual appliance is ensured – without additional NAT configuration. As a result, packets traverse the same network path in both directions and appliances that need this key capability are able to function seamlessly.

This guide explains how to configure and deploy the Azure GWLB using Palo Alto Networks VM-Series Firewall in the backend pool. The sections in the document provide details about the architecture, and configuration of the various components of this integration including Azure's Gateway and Standard Load balancers and VM Series firewall.

Challenges

VM-Series firewall on Azure brings unparalleled security features used to protect Azure workloads. The Palo Alto Networks next generation firewall uses a virtual machine that can be launched from Azure Marketplace or through templates. The VM-Series firewall provides a comprehensive set of security features to protect workloads housed in Azure.
This guide walks through the steps to secure Azure workloads using VM Series NGFW deployed alongside the Azure GWLB.

In non-GWLB environments, there are some additional challenges, requirements and configuration to secure workloads.

1. User defined routes(UDRs) to route traffic through VM Series firewall to secure their network traffic in Azure
1. Source IP address Conservation for the backend servers is difficult
2. SNAT is required to prevent asymmetric traffic flows
3. VNET Peering in a hub & spoke pattern of Application vnets with security vnet.

Solution

Gateway Load Balancer is part of Azure Load Balancer portfolio. The GWLB targets high performance and high availability scenarios with third-party Network Virtual Appliances (NVAs). With the capabilities of Gateway Load Balancer, you can easily deploy, scale, and manage NVAs.
Here are the list of advantages of integrating VM series firewall with Azure GWLB

- **Integrate VM Series Firewall transparently into the network path** to secure application traffic without modifying application network architecture.
- **Easily add or remove Firewall instances within the network path** to scale with ease to manage costs
- **Chaining of GWLB with Public Endpoints(Public Load Balancer or Public Virtual Machine)** will avoid administrators from taking care of VNET Peering and UDRs to route traffic through VM Series Firewall.
- **Avoid NATting of traffic in the transit** to provide complete visibility of Source's identity to the application and VM Series firewall.
- Gateway Load Balancer maintains flow stickiness to a specific instance in the backend pool along with flow symmetry. As a result, a consistent route to your network virtual appliance is ensured – without additional manual configuration.

Note-*This document focuses on the configuration using the Azure Portal and ARM templates. It is assumed that the reader is familiar with Palo Alto Networks NGFW concept, Azure components, and architecture. Please refer to the References section for more information.*

Azure GWLB Components

Gateway Load Balancer consists of the following components:

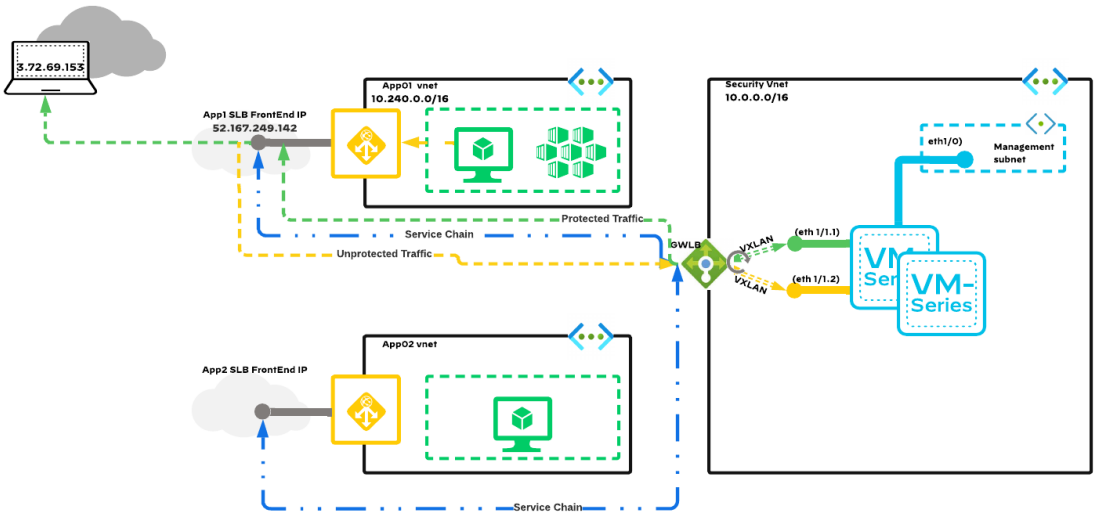
- **Frontend IP configuration** - The private IP address of your Gateway Load Balancer.
- **Load-balancing rules** - A load balancer rule is used to define how incoming traffic is distributed to all the instances within the backend pool. A load-balancing rule maps a given frontend IP configuration and port to multiple backend IP addresses and ports.
    - Gateway Load Balancer rules can only be HA port rules.

    - A Gateway Load Balancer rule can be associated with up to two backend pools.
- **Backend pool(s)** - The group of virtual machines or instances in a virtual machine scale set that is serving the incoming request. To scale cost-effectively to meet high volumes of incoming traffic, computing guidelines generally recommend adding more instances to the backend pool. Load Balancer instantly reconfigures itself via automatic reconfiguration when you scale instances up or down. Adding or removing VMs from the backend pool reconfigures the load balancer without extra operations. The scope of the backend pool is any virtual machine in a single virtual network.
- **Tunnel interfaces** - Gateway Load balancer backend pools have another component called the tunnel interfaces. The tunnel interfaces provide a communication channel between the GWLB and the NVAs in the backend pools through which network traffic is handled. Each backend pool can have up to 2 tunnel interfaces internal and external. The external and internal interfaces map to the VM-Series Security Zones; Internal to trust and external to untrust. Ingress traffic sent to the backend pool uses the external interface whereas return traffic is sent through the internal interface
- **Service Chain** - A Gateway Load Balancer can be referenced by a Standard Public Load Balancer frontend or a Standard Public IP configuration on a virtual machine. The addition of advanced networking capabilities in a specific sequence is known as service chaining. Traffic in the service chain is routed through using VXLAN thus allowing to conserve the original IP headers including the original source IP. The process of adding the VXLAN headers to traffic flows is called encapsulation and the removal of the VXLAN headers is called decapsulation. The encapsulation/decapsulation process is typically done by the device at the edge of the service chain which in our example is the public Standard Load Balancer. . The VM-Series integration is essentially both the support for VXLAN and the ability to read the original headers inside the VXLAN encapsulated flows.

Deployment Topology

The diagram below shows the Azure networking topology to show how to protect workloads in Azure using the VM-Series and the Azure GWLB.

The diagram below shows two application VNET and a security VNET where both the firewalls and GWLB are placed. The application VNETs have the application components including a public facing Standard load balancer and direct Internet to the application stack. This implies the design model uses a distributed ingress architecture in contrast with a more traditional centralized ingress architecture. Also noticeable in the design, the Application VNETs are not connected to the Security VNET. The path used to send traffic from the application to the security VNET is the service chain path. The service chain is however compatible with VNET peering and the two can be concurrently supported. The ingress path uses the service chain path whereas east-west traffic would leverage the VNET peering path when properly configured. The important consideration is still that when having multiple VNETs best practices for non-overlapping address space between VNET should be followed.



GWLB and VM Series deployment steps

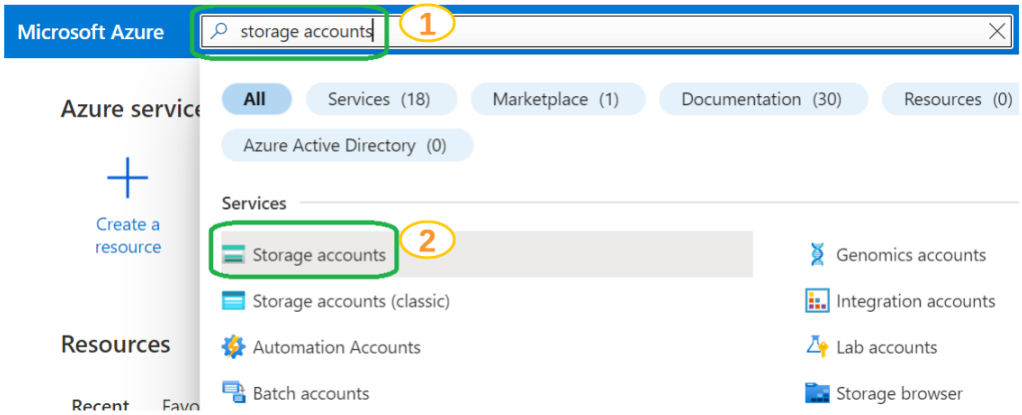
You can deploy Azure GWLB with VM Series and service chain with Public load balancer by following below mentioned steps.

- Create Storage account with bootstrap files on Azure that will be used to bootstrap VM Series firewall
- Create Azure GWLB and VM Series firewall using the Security stack ARM Template
- Launch Application behind a Public Load balancer and service chain with GWLB created using Application Stack ARM Template

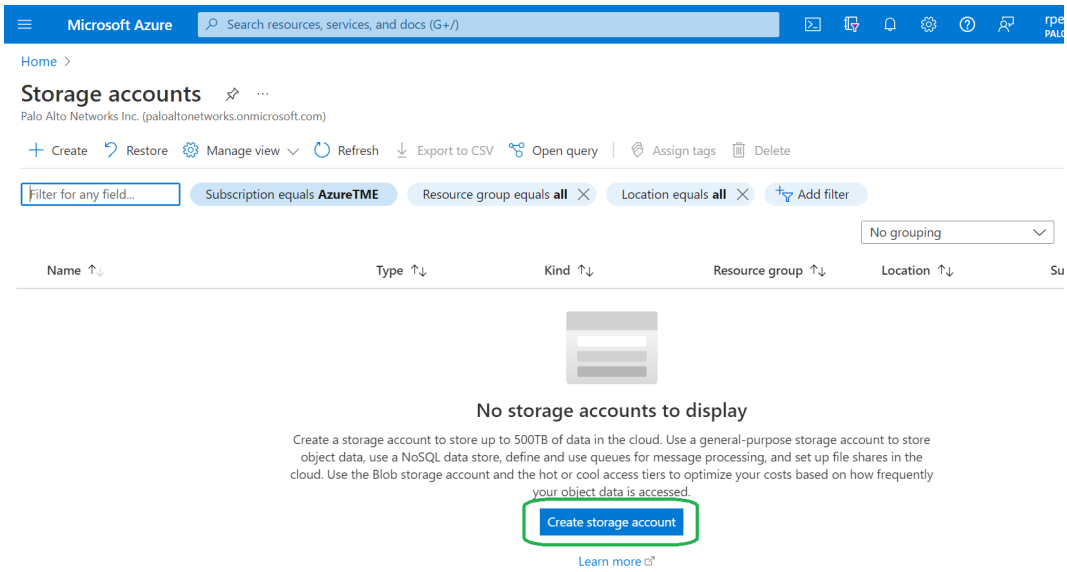
1. Create Storage account for the with bootstrap files used to bootstrap VM Series firewall

1.1. Create storage account

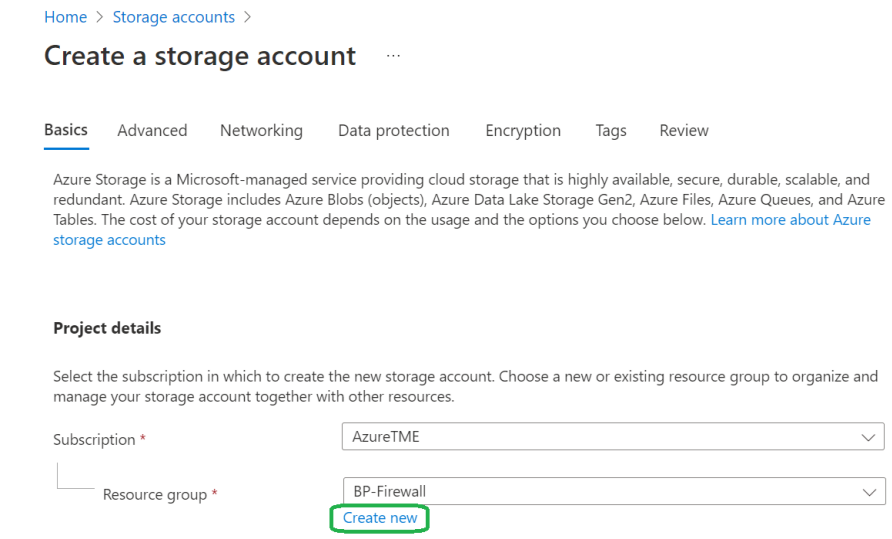
Login to Azure portal search for ‘storage accounts’ in global search and click on “Storage accounts” service as shown below



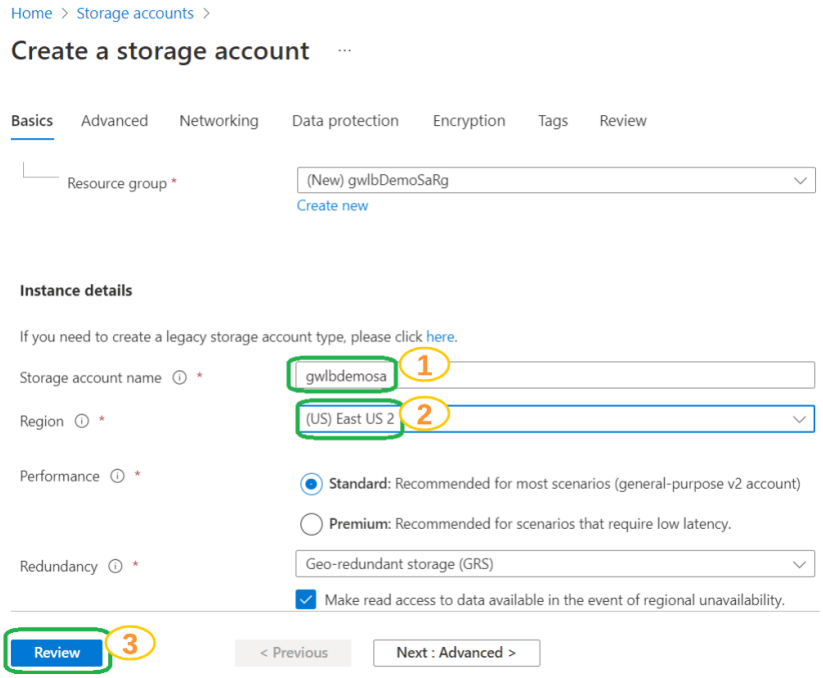
You will be presented with the following screen. Click on the “Create storage account” option available.



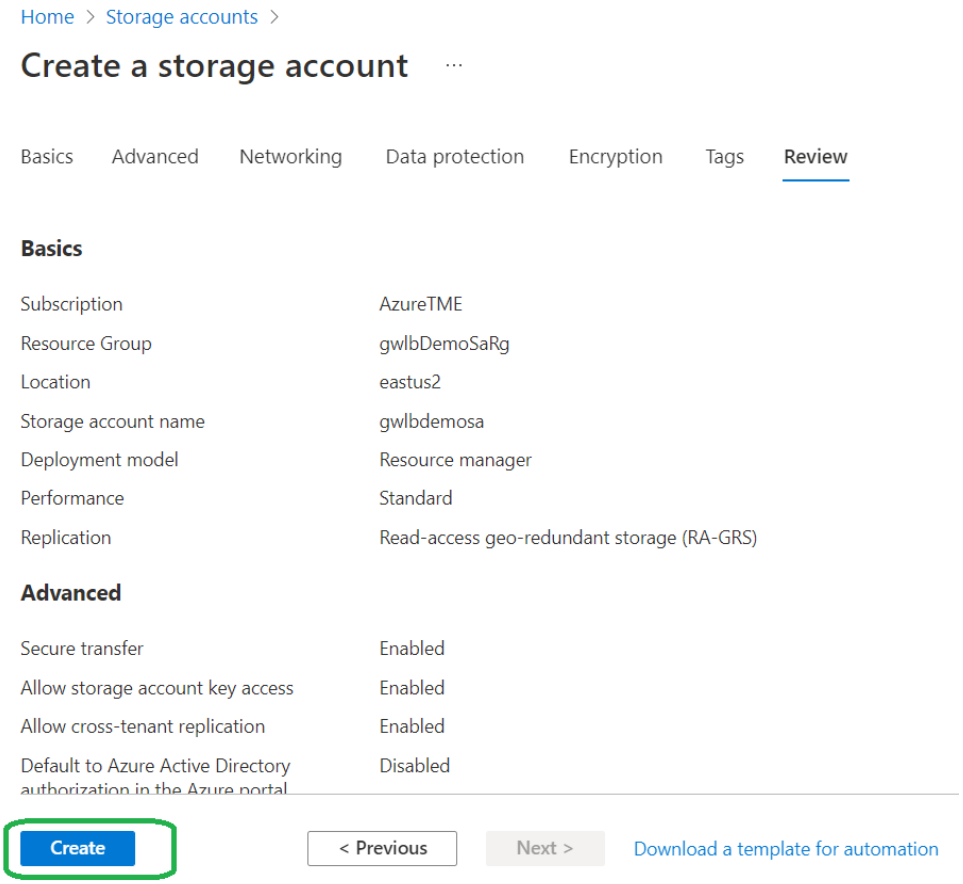
Create a new Resource group by clicking on ‘Create new’ option against Resource group as part of storage account creation and provide a name for new resource group creation. Ex: “gwlbDemoSaRg”



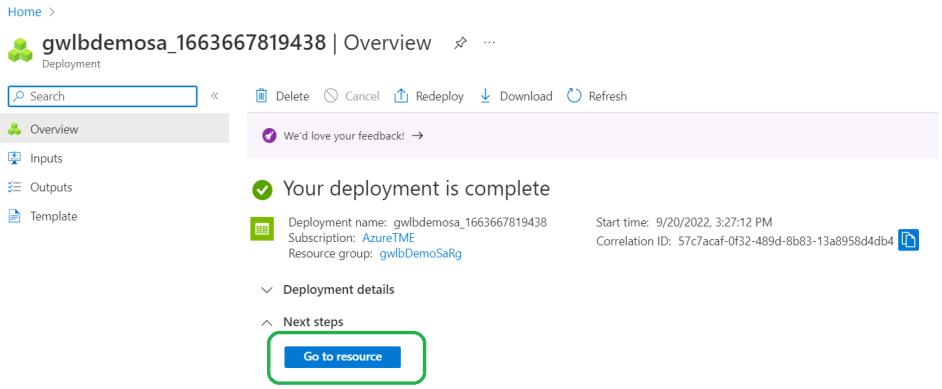
Key in Storage account name(Ex: gwlbdemosa) and Region(Ex: East US 2) in which you wish to create a storage account. And click on Review.



Validate the details provided and click on ‘Create’ to create a Storage account.

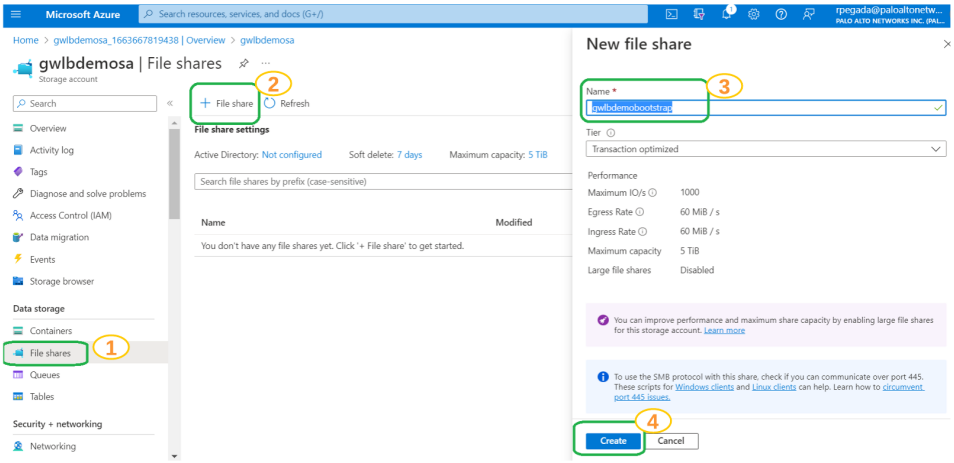


Once the storage account is created, click on 'Go to resource' as shown below to go to the storage account created.

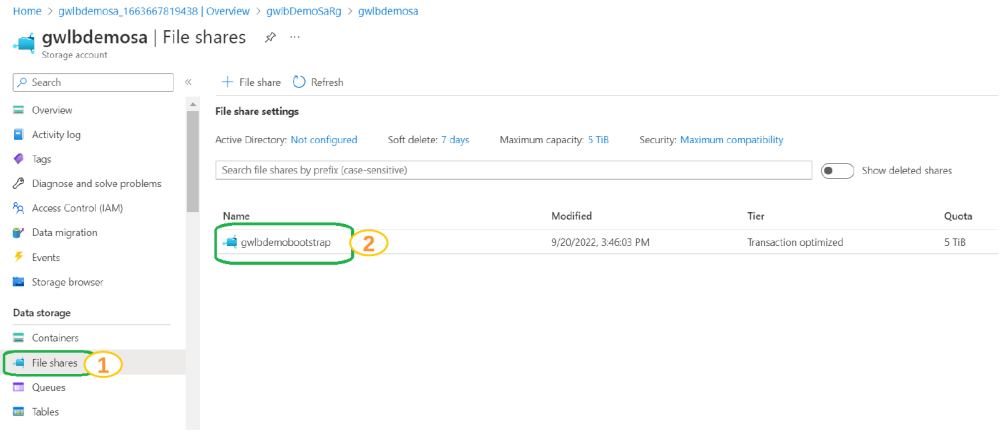


1.2. Add bootstrap file share

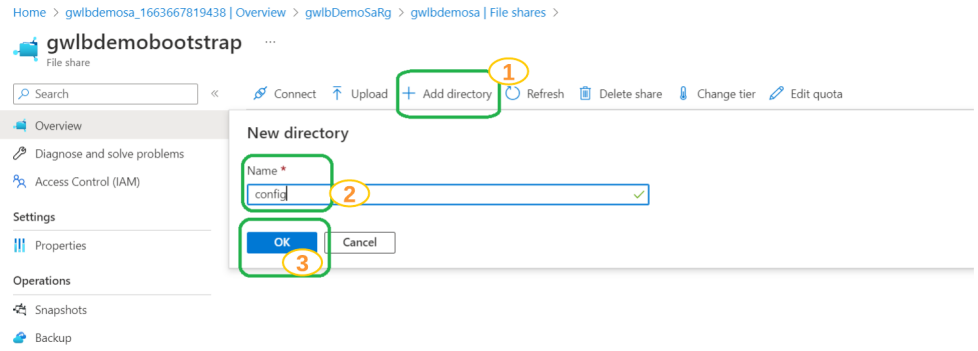
Add a bootstrap file share(Ex: gwlbdemobootstrap) within the storage account created.

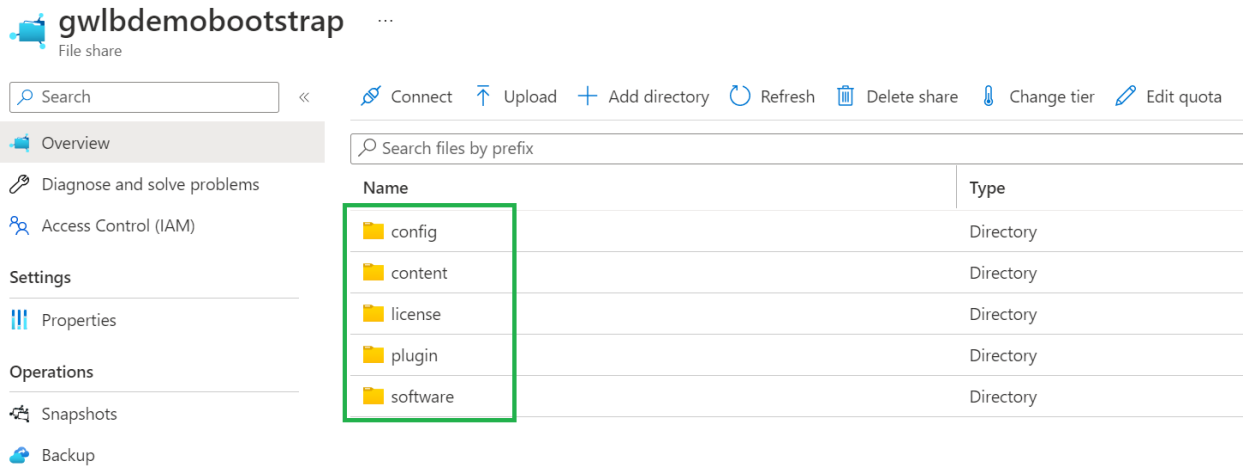


Open the bootstrap file share directory created to add folders used in the VM Series bootstrap process.



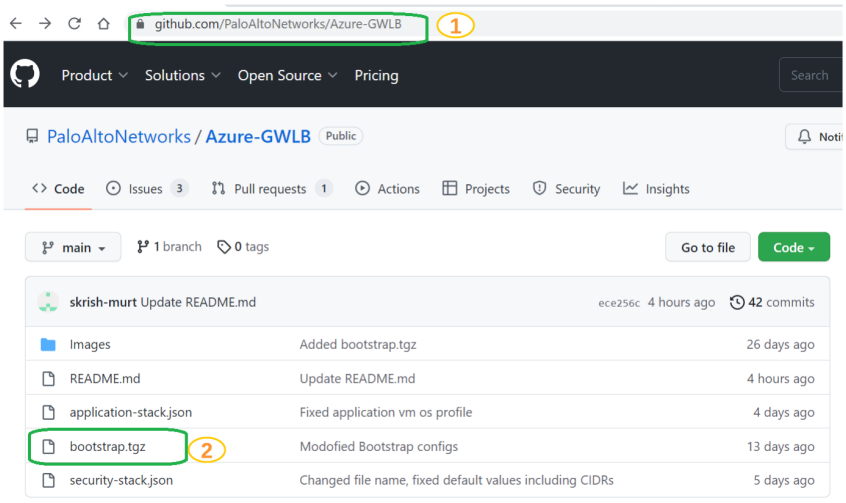
Add "**config, license, software, plugin and content**" folders with in the bootstrap directory(gwlbdemobootstrap) created as shown below





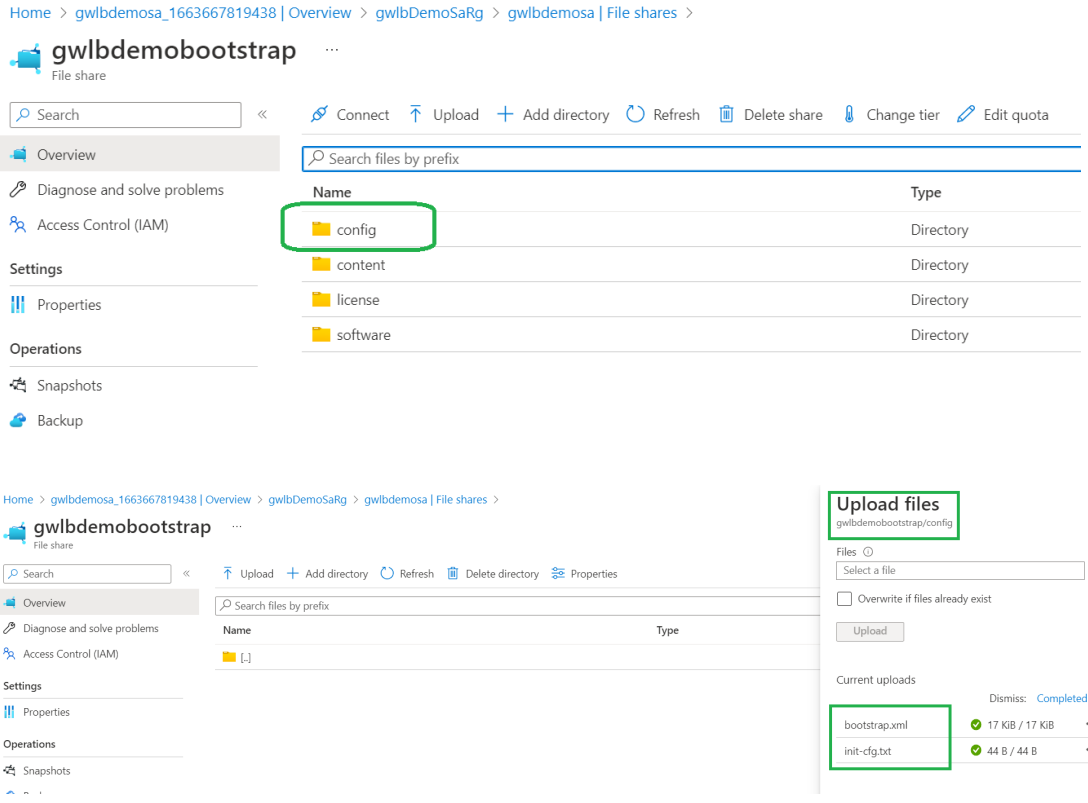
1.3. Update bootstrap file share with init and bootstrap configuration

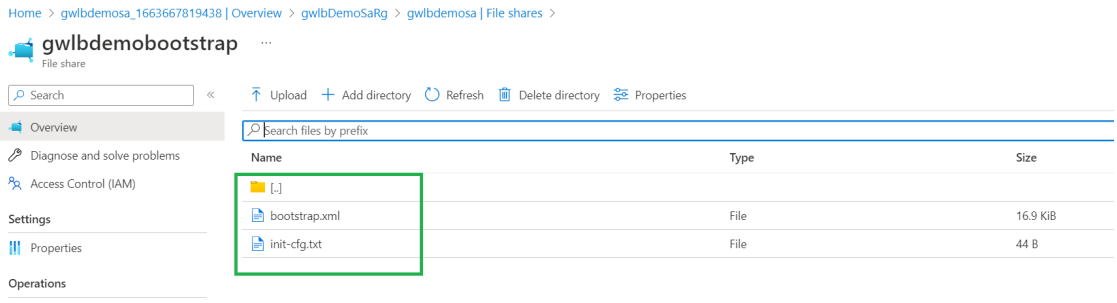
Copy init config and bootstrap configuration files from github repository into the “config” folder created in above step.
Go to https://github.com/PaloAltoNetworks/Azure-GWLB and click on ‘bootstrap.tgz’ to download the bootstrap files from github. Unzip the file after downloading.



Copy “init-cfg.txt” and “bootstrap.xml” files from bootstrap config folder downloaded from github to “config” folder of ‘gwlbdemobootstrap’ file share created in above step.

Click on the “config” folder as shown below and add “init-cfg.txt and bootstrap.xml” files.



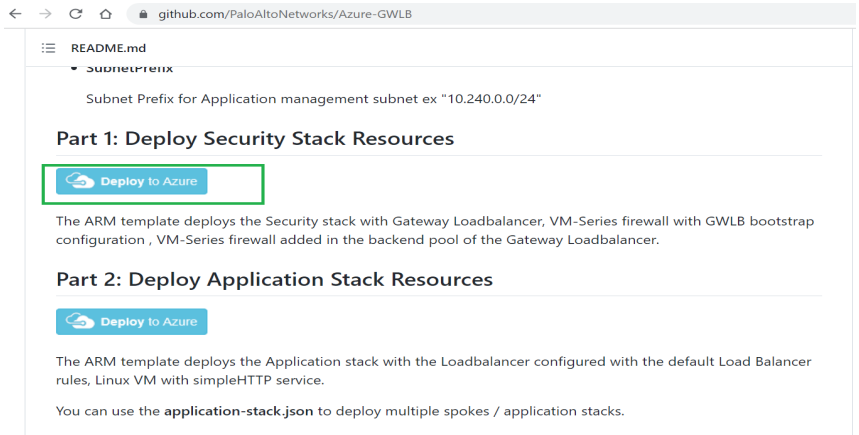


Info- *init-cfg.txt file has "plugin-op-commands=azure-gwlb-inspect:enable" to enable GWLB inspect functionality on VM Series firewall.*
And the bootstrap.xml file has configuration to configure interface eth1/1 and add sub interfaces eth1/1.1 and eth1/1.2 to accept vxlan traffic from GWLB. eth1/1.1 and eth1/1.2 corresponds to trust and untrust vxlan interfaces associated with GWLB

2. Create Azure GWLB and VM Series firewall using the Security stack ARM Template

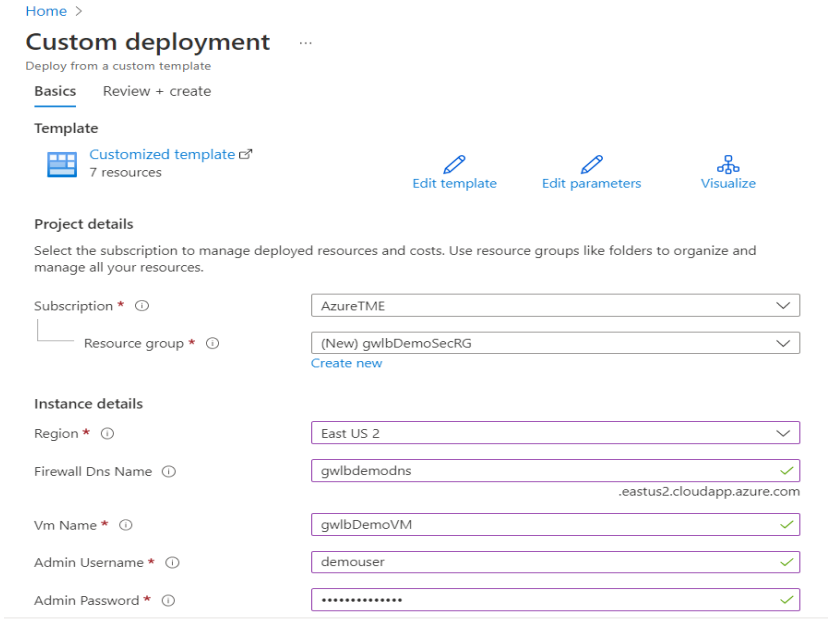
2.1. Deploy Security stack

To deploy GWLB and VM series as its backend pool go to https://github.com/PaloAltoNetworks/Azure-GWLB and scroll down to find 'Part 1: Deploy Security Stack Resources' and click on "Deploy to Azure" to deploy Azure GWLB and VM series as part of dedicated vnet(security vnet).



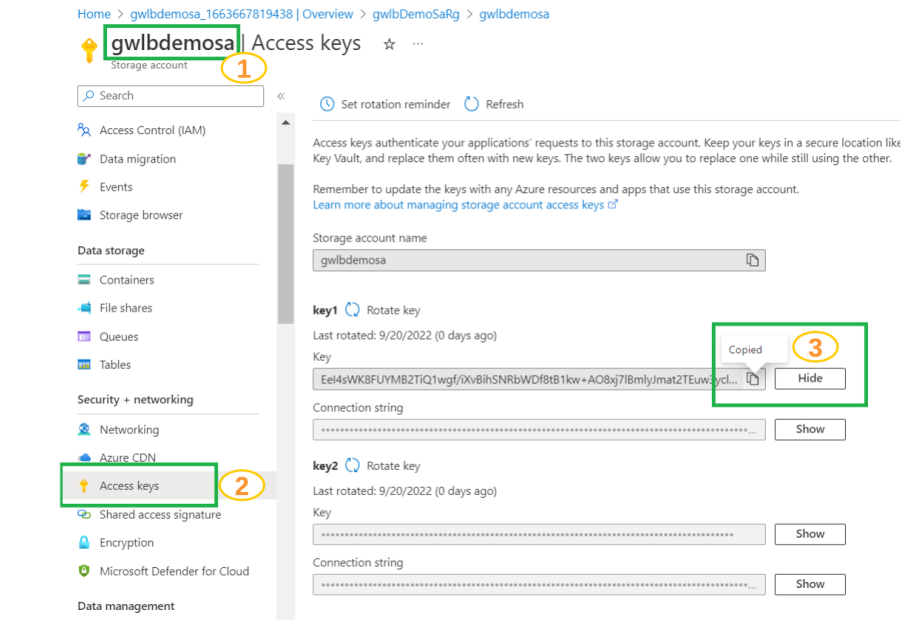
Fill in the below mentioned parameters and click on create to deploy the Security stack template.

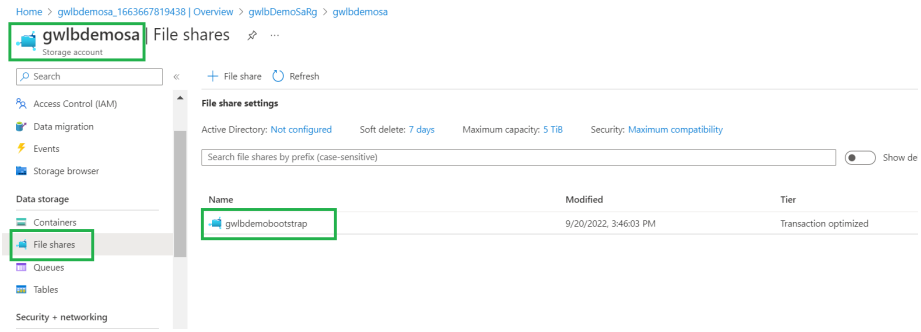
Resource Group : Click on “Create new” and create a new resource group Ex: gwlbDemoSecRG
Region : East US 2
Firewall Dns Name : gwlbdemodns
Vm Name : gwlbDemoVM
Admin Username : demouser
Admin Password : XXXXXXXXX



Fill in Storage account name, its Access key and Bootstrap Fire Share of the storage account that you have created in above steps.

Refer below screenshot for reference, copy these fields and fill in as part of template deployment



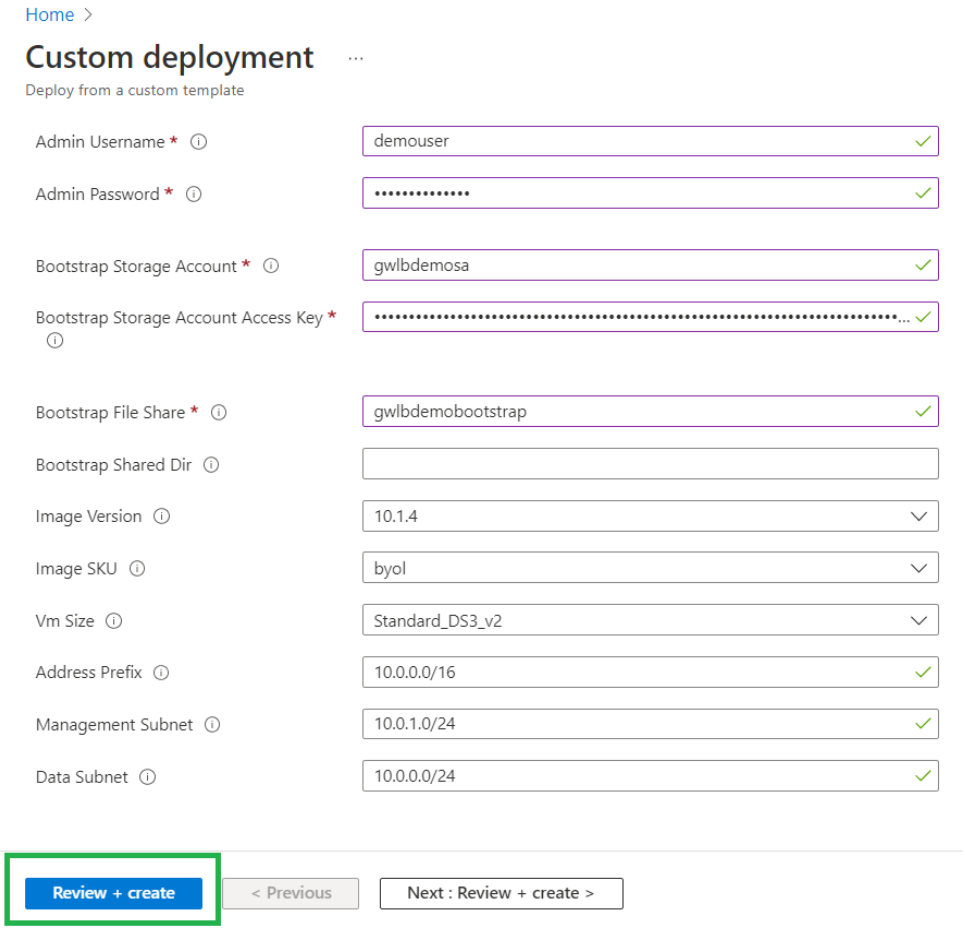


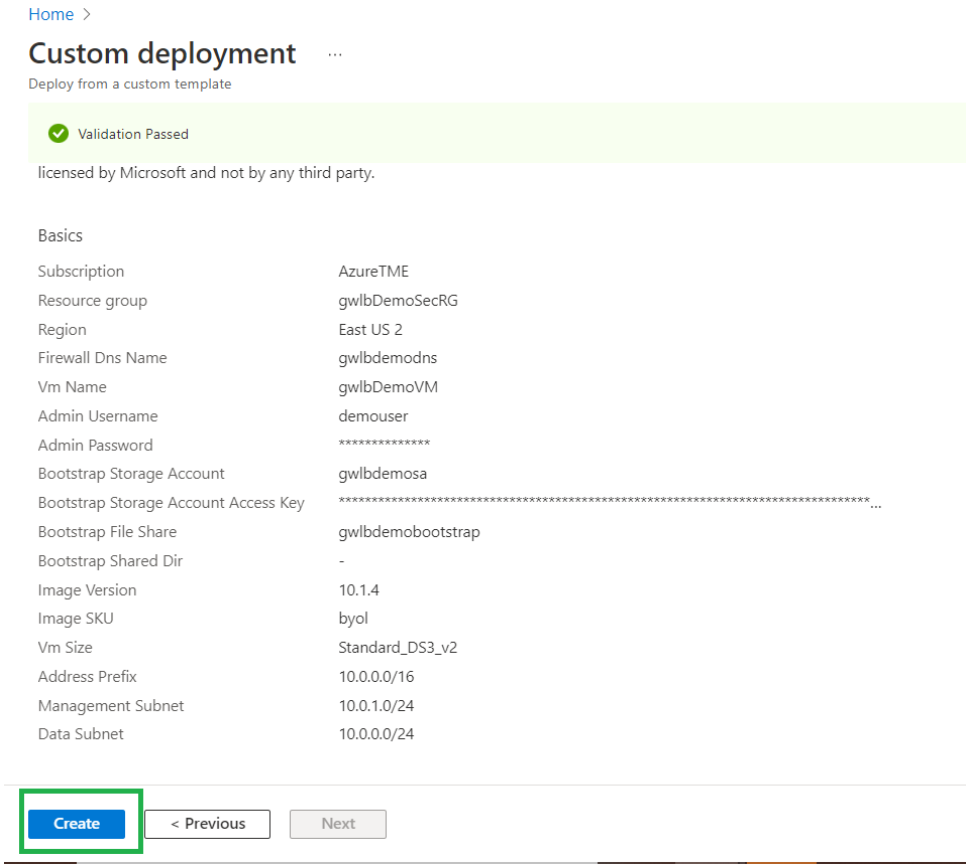
Bootstrap Storage Account : gwlbdemosa

Bootstrap Storage Account Access Key : xxxxxxxxxxxxxxxxxxxxx

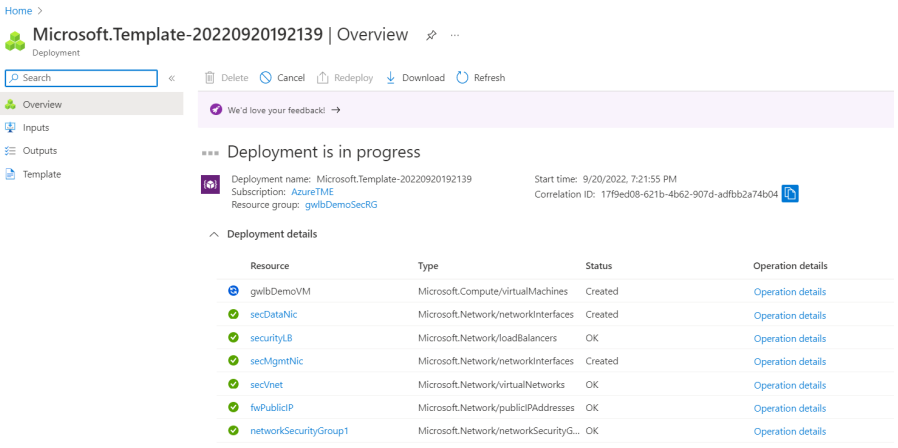
Bootstrap File Share : gwlbdemobootstrap

And leave remaining fields to defaults and click on Review + Create and Create after successful validation.

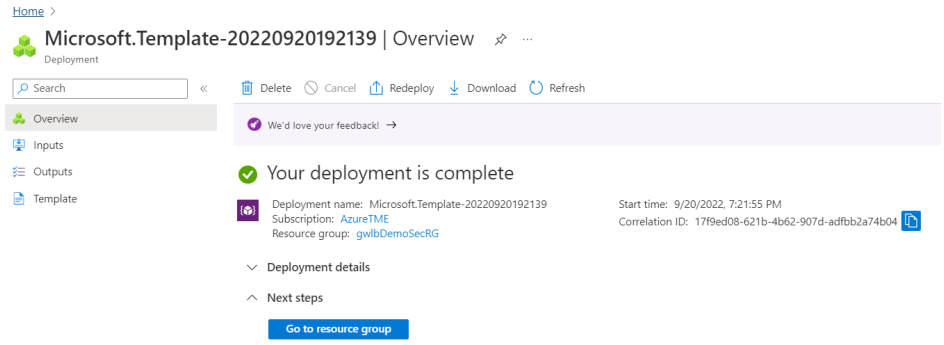




Now you will be entering into the “Deployment in progress” page as shown below. Wait for around 5-7 Min for this template to deploy all the resources.



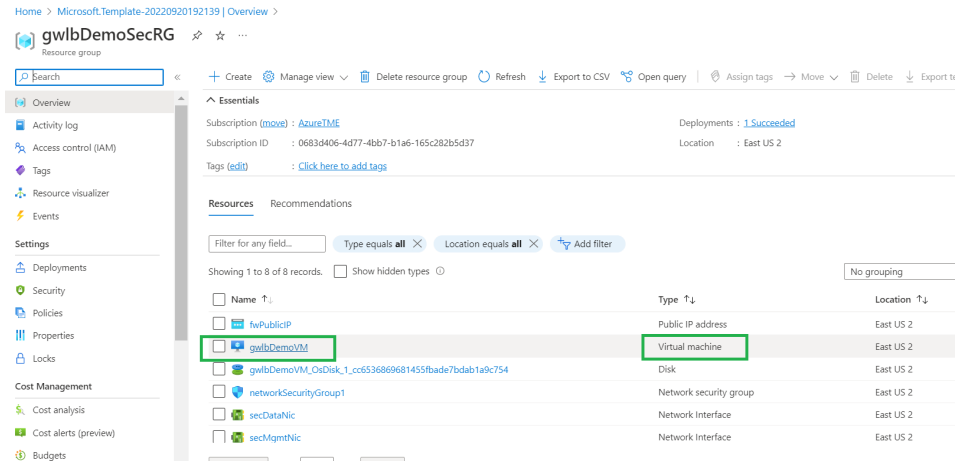
You should see below mentioned screen on successful deployment of Security stack template deployment



2.2. Validate bootstrap configuration on VM Series firewall

Now click on “Go to resource group” to check for the resources created

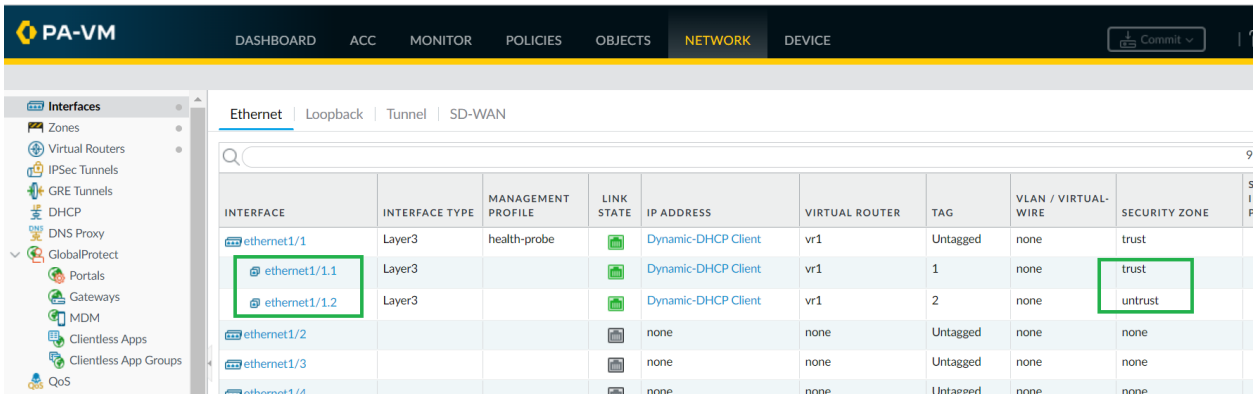
Login to the VM Series firewall by clicking on “gwlbDemoVM” virtual machine created as part of this deployment using its public IP address. And use username and password that was configured during deployment.



Check for the firewall to be configured with Sub-Interfaces ethernet1/1.1 and ethernet1/1.2. And a firewall policy to allow all traffic as per the bootstrap configuration.

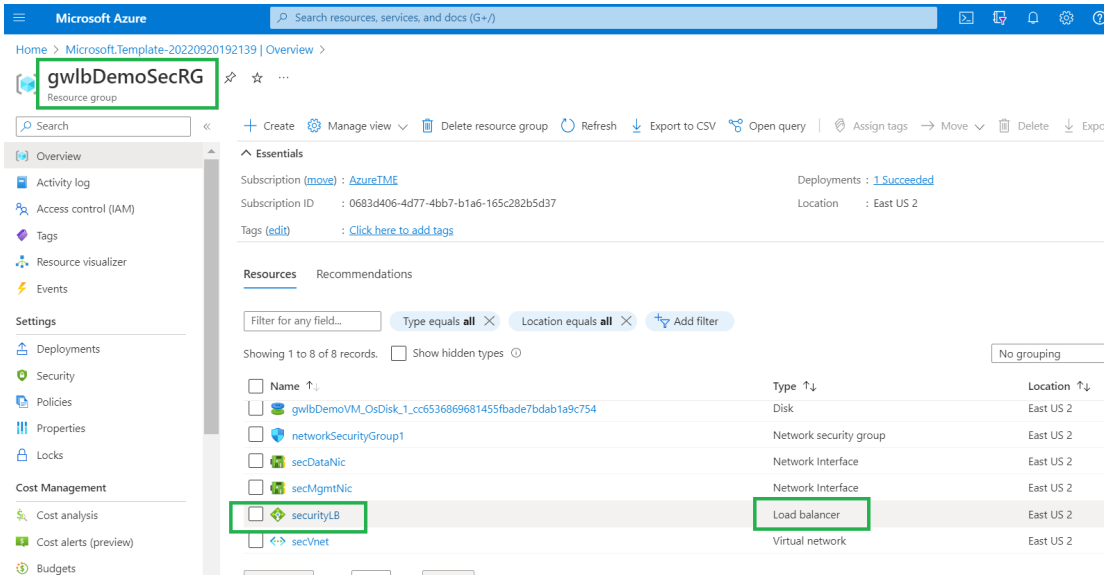
These sub interfaces correspond to the two vxlan tunnels that GWLB uses to forward traffic towards VM Series firewall.

Also make a note that the sub interfaces ethernet1/1.1 and ethernet1/1.2 are VLAN tagged with 1 & 2 respectively. If at all someone tries to deploy this manually, you need to take care of this vlan tags.

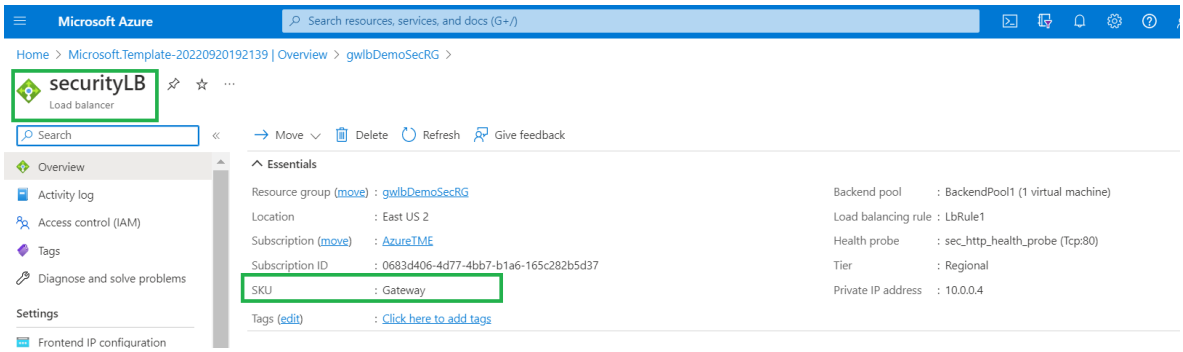


2.3. Validate Gateway Load Balancer configuration

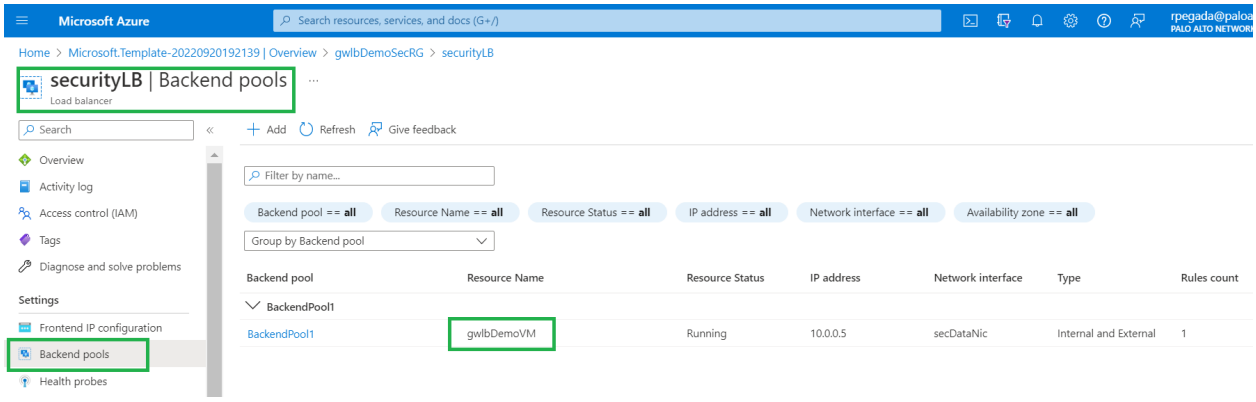
Validate Load Balancer configured by clicking on “securityLB” as shown in the below screenshot.



Check for the LB to be of SKU Gateway(Gateway load balancer)

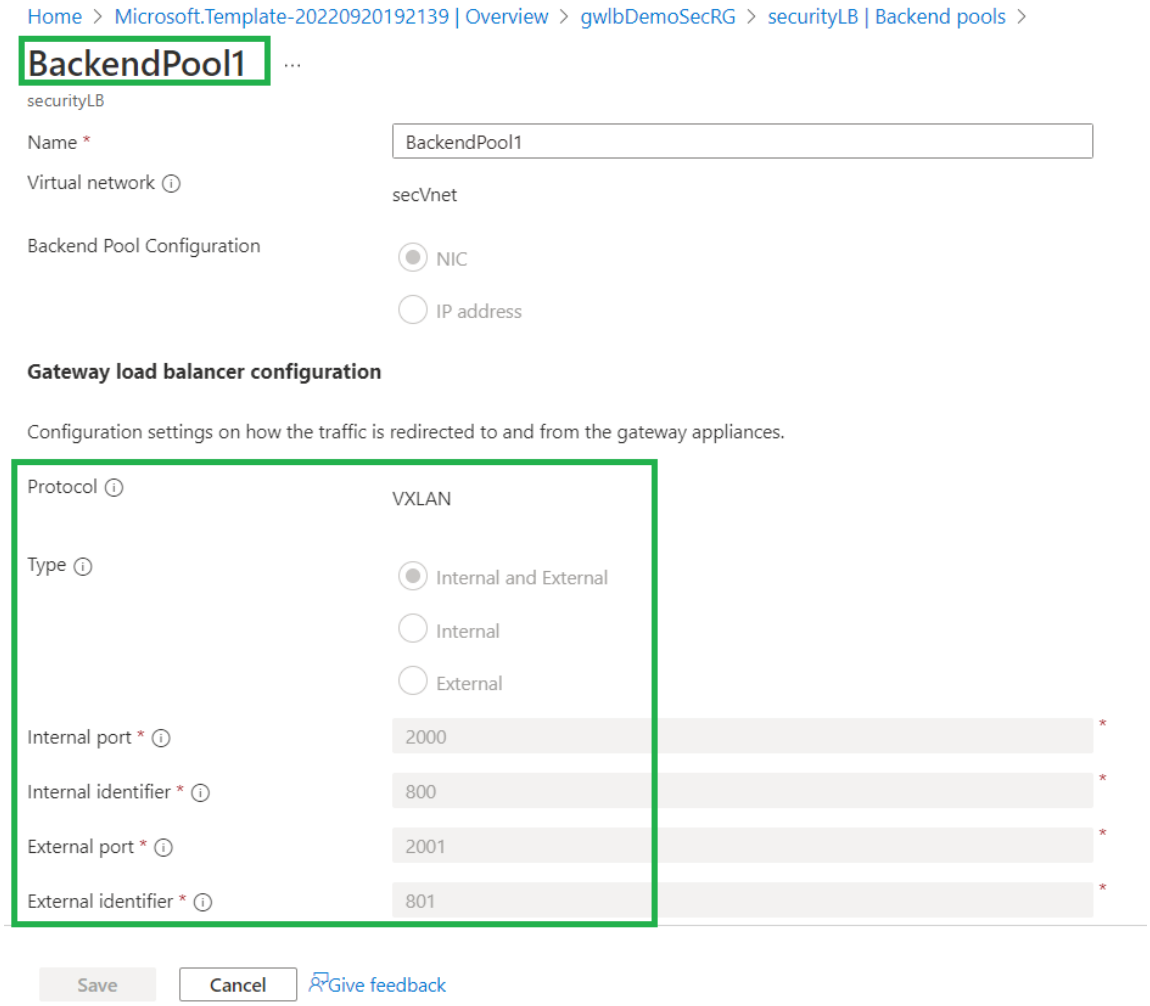


Check for VM Series to be configured as the backend pool for the GWLB.



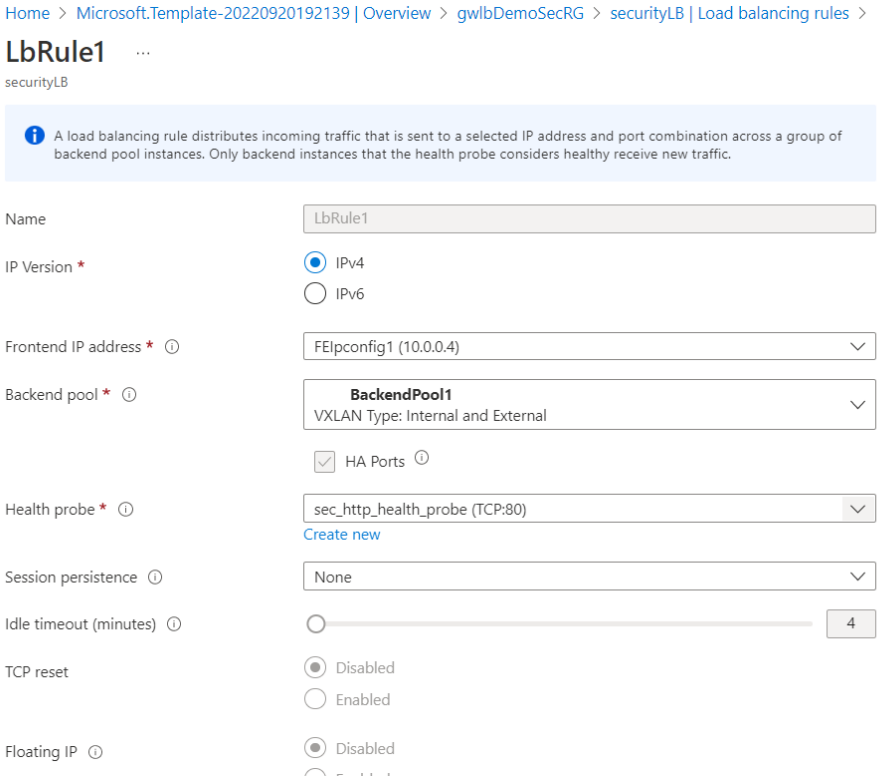
Click on the Backendpool to check for protocol used, internal and external ports and VNI configuration.

GWLB uses VXLAN protocol to encapsulate the data traffic and forward towards the backendpool(VM Series Firewall).



NOTE: Notice these are the default values that match the command that ran in the vm-series to enable the VXLAN inspection for the GWLB integration. If you selected a different set of values, you should update the template to reflect those values.

Also check for load balancing rules to be configured with the backendpool configured.



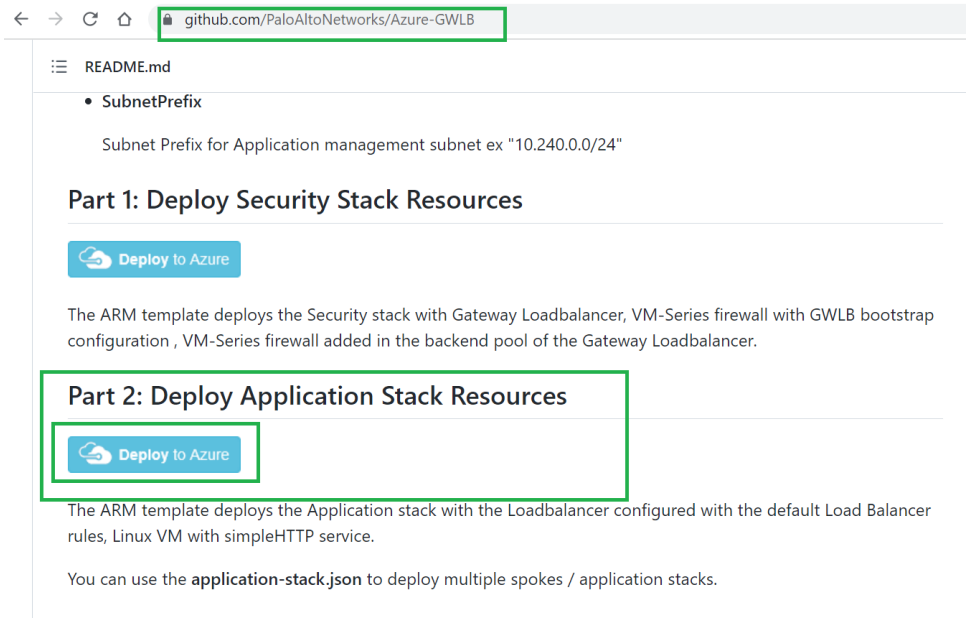
We are good with Security stack configuration now. We have GWLB and VM Series provisioned as part of security vnet.

3. Build Application Environment and extend the service chain from the local Public Load balancer to the GWLB

3.1. Deploy Application Stack

Now that we have deployed a security stack with GWLB. Lets go ahead and deploy Application stack that includes application server frontend with Public Load balancer.
Using this template we will configure service chaining between Public load balancer and GWLB so that all the traffic that is going through Public load balancer will be sent to GWLB for security inspection using VM series .

To deploy Application stack, go to https://github.com/PaloAltoNetworks/Azure-GWLB and scroll down to find 'Part 2: Deploy Application Stack Resources' and click on "Deploy to Azure" to deploy Application server frontend using Public load balancer in dedicated vnet(Application vnet).



Fill in the below mentioned parameters and click on create to deploy the Application stack template.

Resource group : gwlbDemoApp1RG(click on "Create new" and provide new resource group name)

Region : East US2

Security Resource Group : gwlbDemoSecRG(**This is the resource group in which we have deployed security stack in the above steps**)

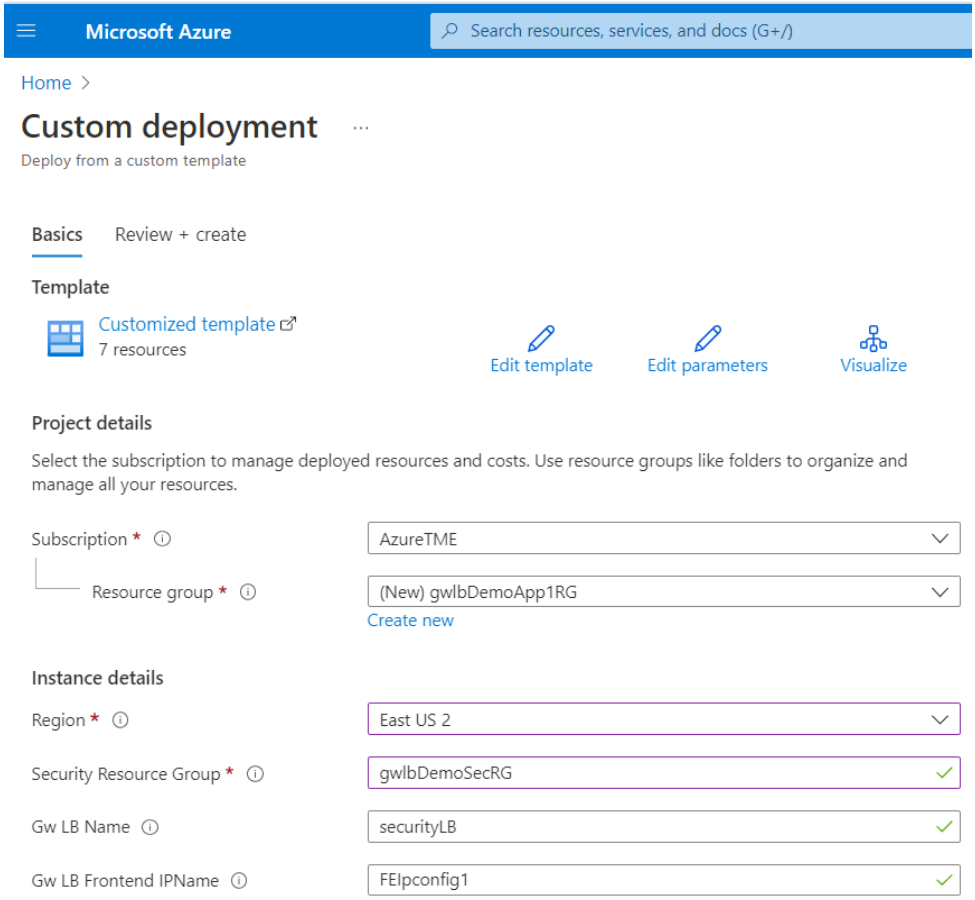
Gw LB Name : securityLB (**leave this to default, this is the GWLB that was added as part of security stack creation**)

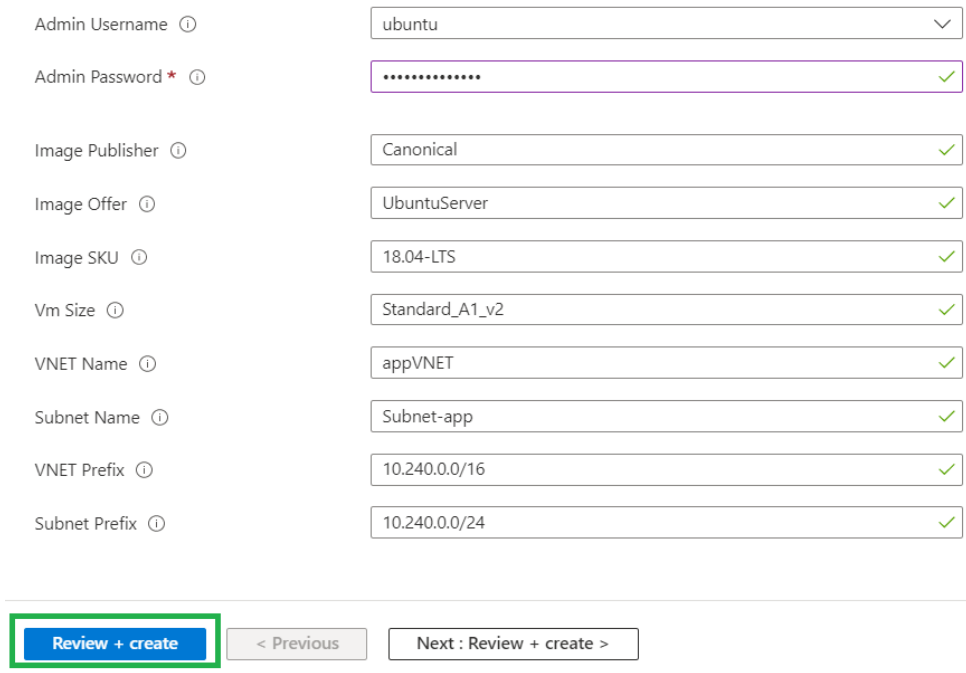
Gw LB Frontend IPName : FEIpconfig1(**leave this to default, this is the GWLB Frontend IP that was added as part of security stack creation**)

Admin Username : ubuntu(**leave this to default)**

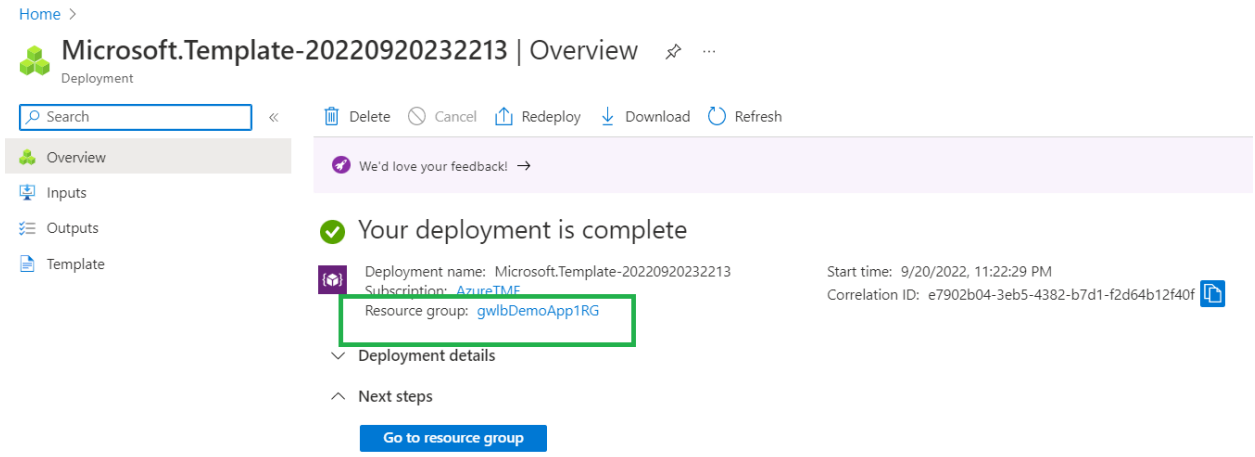
Admin Password : <Fill in the password>

Leave remaining fields to default values and click on “Review + Create” and then “Create” to deploy Application stack





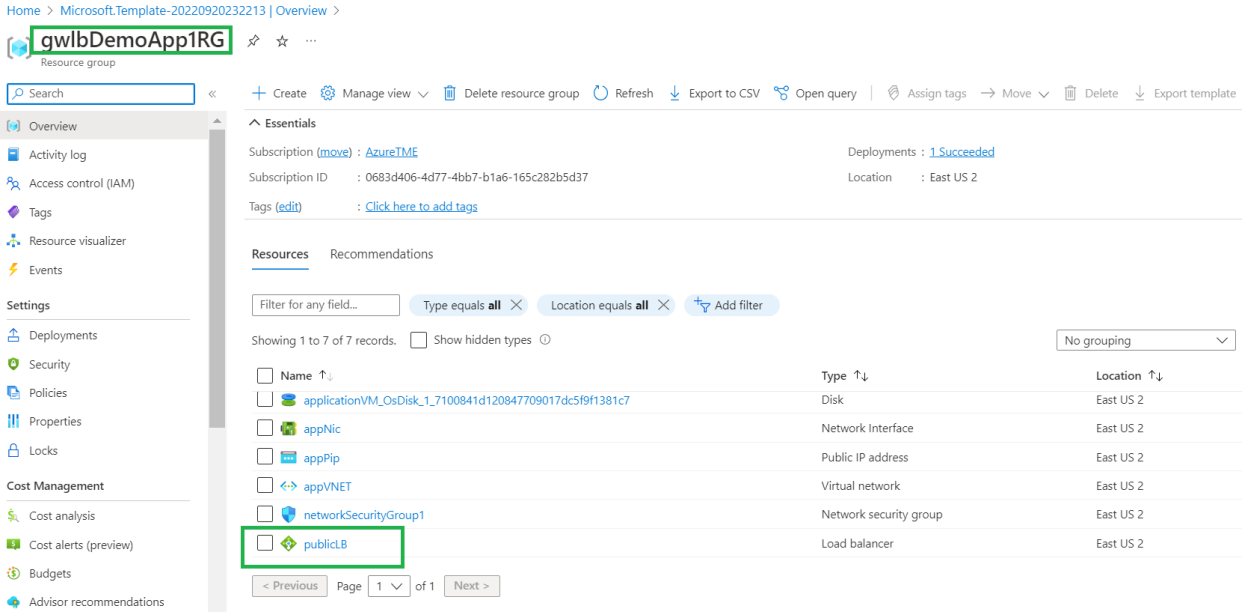
After clicking on “Create” you will see the below mentioned screen on successful deployment of Application stack.



3.2. Check for service chaining between GWLB and Public Load balancer

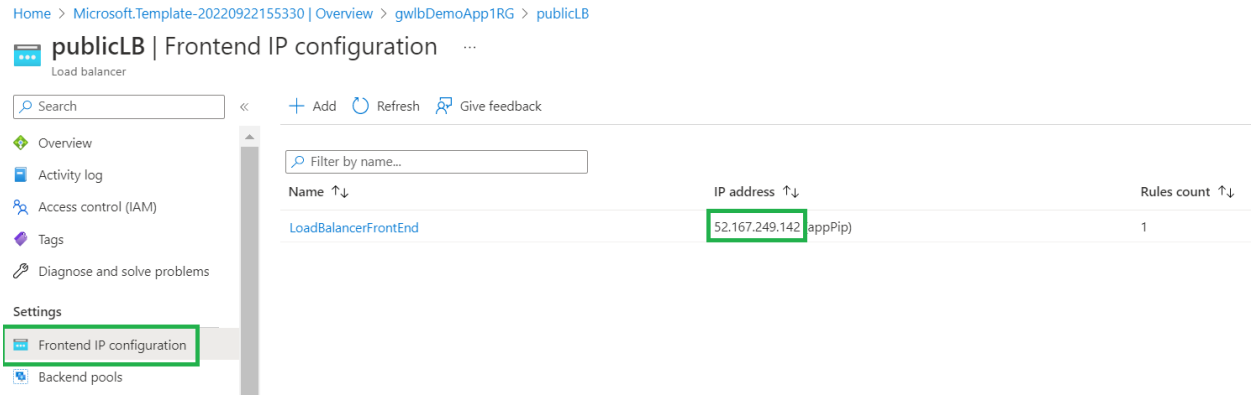
Click on “Go to resource group” to go to the resource group in which the application stack got deployed.

Within this resource group you can see that the Public load balancer got added by the name “publicLB”

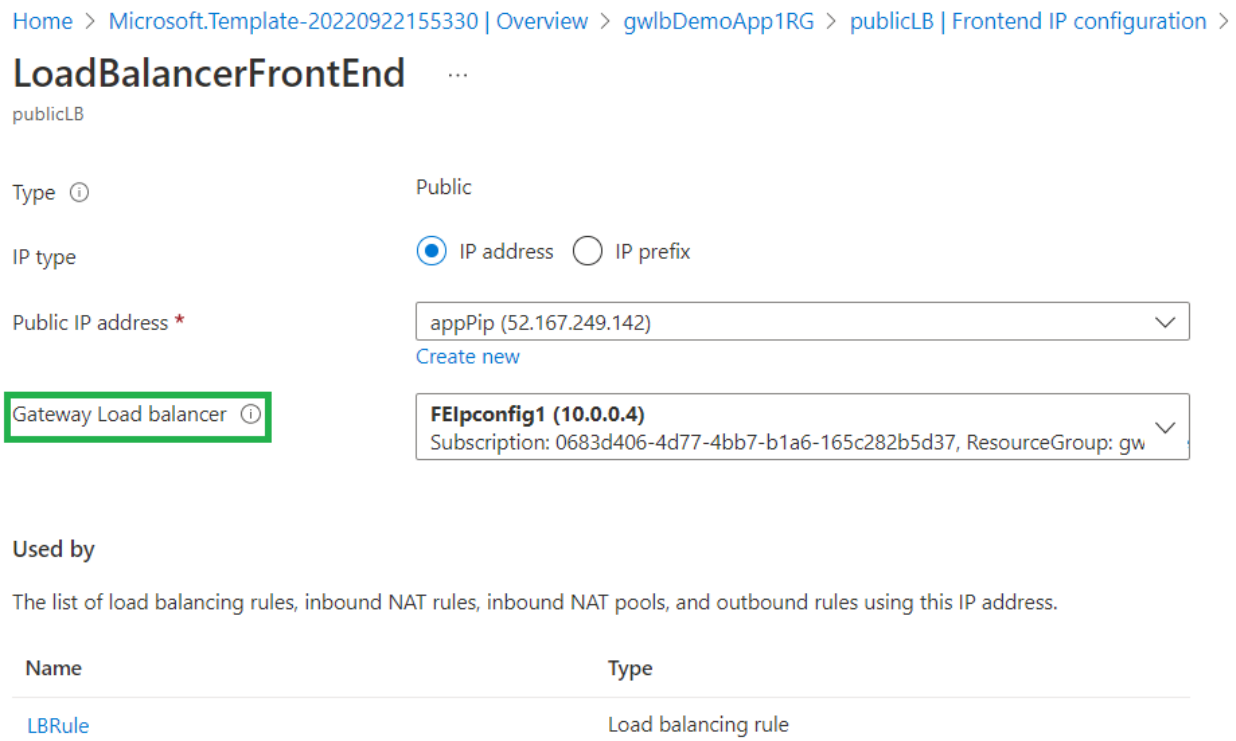


Click on this Public Load balancer to validate service chaining with GWLB created in earlier steps

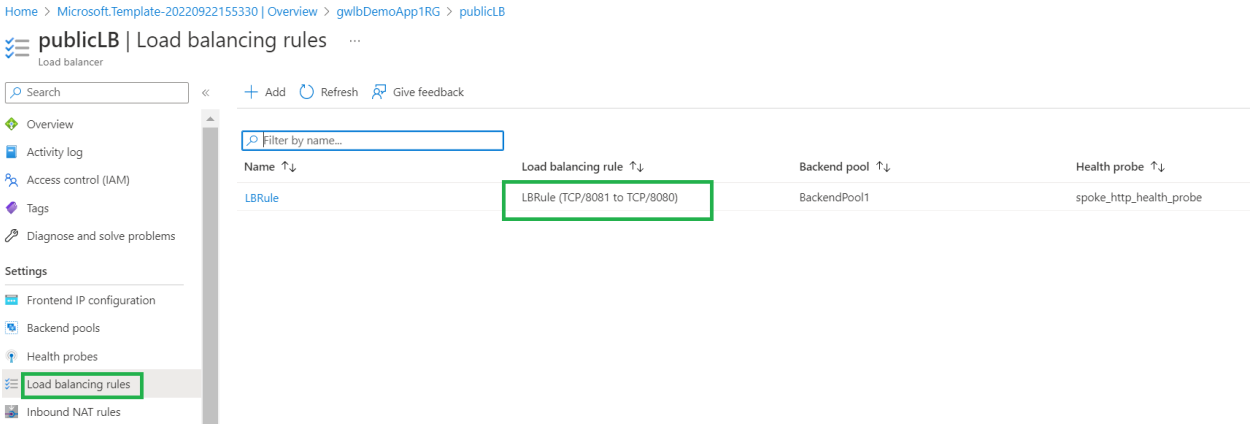
Under the Public load balancer go to Frontend IP configuration and click on “LoadBalancerFrontEnd” to check service chaining with GWLB.



As shown in the screenshot below, we can see that the Standard load balancer points to the Frontend IP of the Gateway load balancer thus ensuring the service chain exists between them.

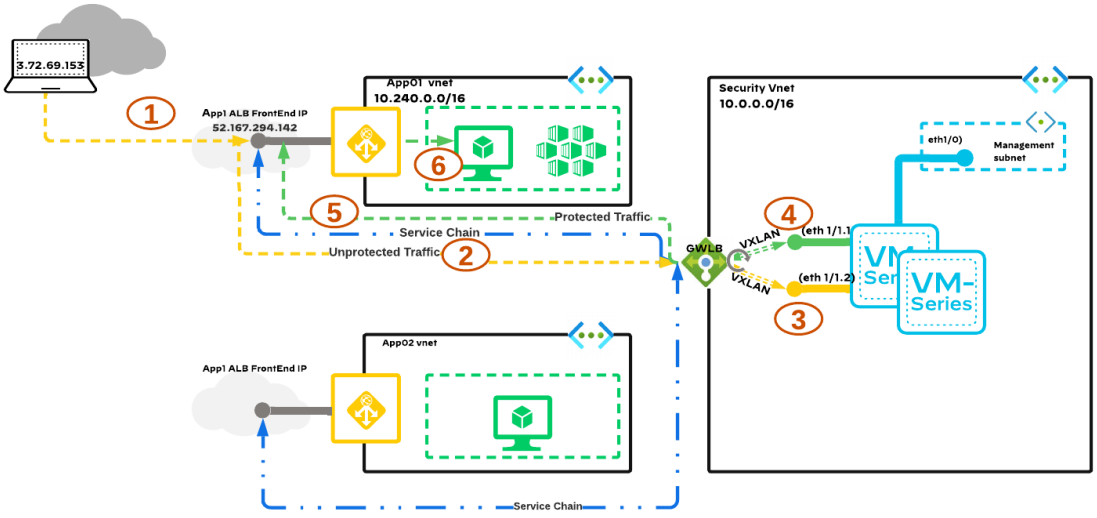


Check for the Load balancing rule. It is configured to redirect traffic destined to port 8081, to port 8080 in the backend pool



Use case validation

1. Verify inbound traffic security using VM Series behind GWLB



As per above topology, inbound traffic flow will be as mentioned below.

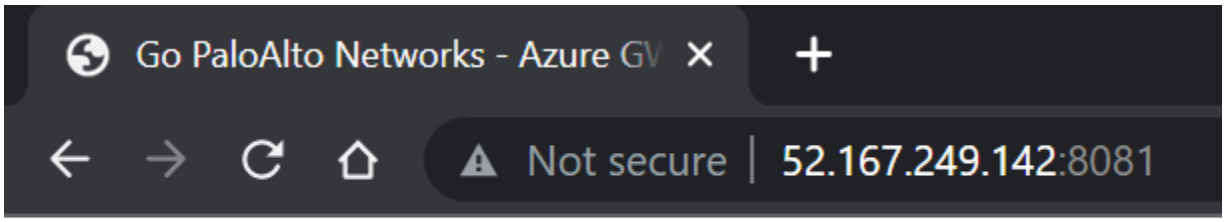
1. **User(3.72.69.153) tries to access application using App1 ALB Frontend IP address 52.167.294.142**
2. **Because of Service Chaining between ALB and GWLB, Unprotected traffic will be forwarded to GWLB**
3. **Unprotected traffic is then sent to VM series firewall by GWLB using VXLAN tunnel towards Untrust Sub interface ethernet1/1.2 of Palo Alto VM Series firewall for inspection.**
4. **After inspection VM Series firewall uses VXLAN tunnel from its Trust Sub interface ethernet1/1.1 towards GWLB to send Protected traffic**
5. **GWLB will now send Protected traffic towards Public Load Balancer Frontend IP address**
6. **App1 Public Load balancer will send Protected traffic to its backend pool application**

Response traffic will follow the same reverse path.

Application can be accessed by the user as shown below using “wget http://52.167.249.142:8081” and you will get the response as shown below.

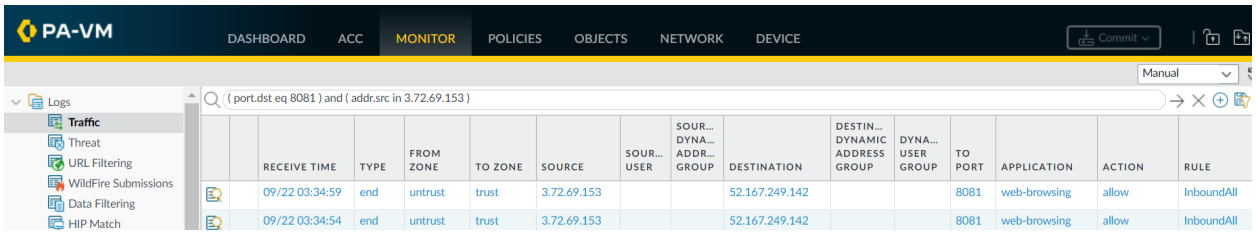
```
root@ip-10-1-254-236:/home/ubuntu# wget http://52.167.249.142:8081
--2022-09-22 10:27:35--  http://52.167.249.142:8081/
Connecting to 52.167.249.142:8081... connected.
HTTP request sent, awaiting response... 200 OK
Length: 242 [text/html]
Saving to: 'index.html.17'

index.html.17                       100%[=============================================================================>]     242  --.-KB/s    in 0s

2022-09-22 10:27:38 (16.4 MB/s) - 'index.html.17' saved [242/242]

root@ip-10-1-254-236:/home/ubuntu#
```

Same Application can be accessed from a web browser using “http://52.167.249.142:8081” and the output will look like below.



Welcome to PaloAlto Networks

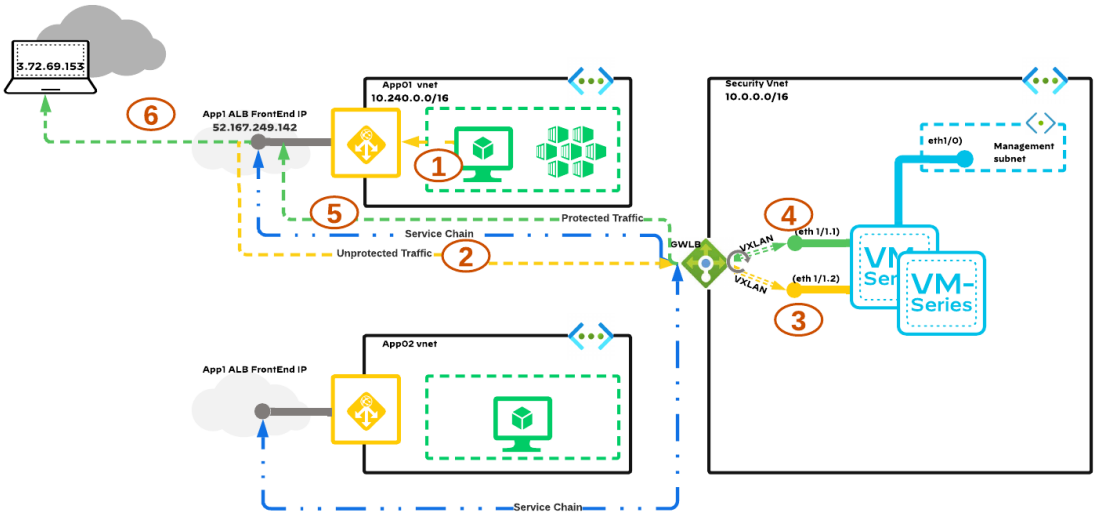
Now go to the firewall monitoring page and verify the logs to confirm that the user traffic is going through the VM Series firewall behind GWLB and appropriate security policy is being applied.



With this we can make sure that the inbound traffic is being protected by VM series firewall behind Azure GWLB

2. Verify outbound traffic security using VM Series behind GWLB

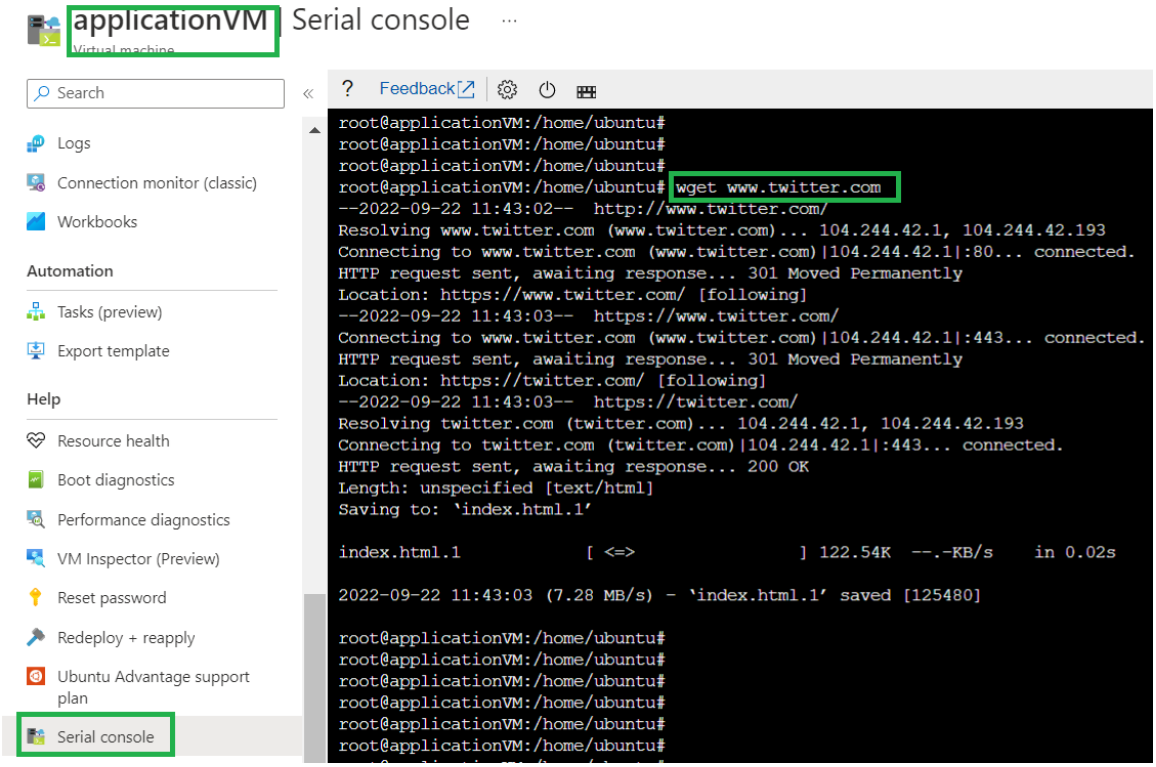
To send outbound traffic login to the Application server behind Public Load Balancer and try to access some internet applications.



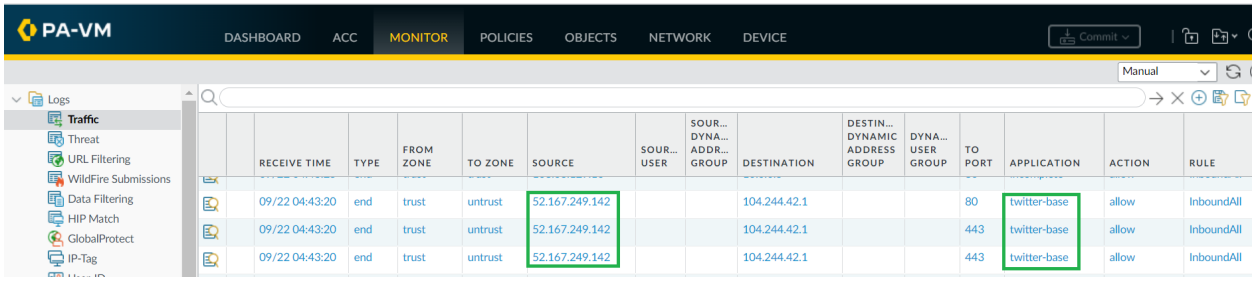
As per above topology, outbound traffic flow will be as shown below.

1. Try to send internet traffic(ex:twitter.com)
2. Because of Service Chaining between ALB and GWLB, Unprotected traffic will be forwarded to GWLB
3. Unprotected traffic is then sent to VM series firewall by GWLB using VXLAN tunnel towards Untrust Sub interface ethernet1/1.2 of Palo Alto VM Series firewall for inspection.
4. After inspection VM Series firewall uses VXLAN tunnel from its Trust Sub interface ethernet1/1.1 towards GWLB to send Protected traffic
5. GWLB will now send Protected traffic towards Public Load Balancer Frontend IP address
6. App1 Public Load balancer will send outbound internet traffic usign its Public IP address as source IP

As part of current topology let us login to the Linux server behind GWLB by using the serial console on azure portal or you can also login using SSH.
Try to access twitter website using “wget www.twitter.com” and the output will be as shown below.We can see that the twitter application was accessed successfully.(outbound traffic from application)



Now go to the VM Series firewall behind GWLB to check if this outbound traffic is being inspected by the firewall.



As per above screenshot we can see that the outbound traffic is being sent to Firewall for inspection.

Note: Over here we can also see that the source IP address of the traffic was reported as the public IP address of the Public Load balancer behind which we have the Application. As this traffic is internet outbound traffic, load balancer is first performing NAT before sending to GWLB.

Add additional VM Series Firewall behind GWLB

Using the ARM template located at https://github.com/PaloAltoNetworks/Azure-GWLB we were able to deploy a VM series behind GWLB as part of the above steps.

Now to add an additional VM series firewall behind the already provisioned GWLB, follow the steps mentioned below.

1. Create a VM using the Microsoft Azure CLI.

   Provide the input parameters for the command below.

   *az vm create *
   *--resource-group <myResourceGroup> *
   *--name <myPA-VM> *
   *--vnet-name secVnet *
   *--subnet Subnet-mgmt *
   *--public-ip-sku Standard *
   *--size Standard_DS3_V2 *
   *--nsg networkSecurityGroup1 *
   *--admin-username <username> *
   *--admin-password <password> *
   *--image paloaltonetworks:vmseries-flex:bundle1:10.1.4 *
   *--plan-name bundle1 *
   *--plan-product vmseries-flex *
   *--plan-publisher paloaltonetworks *

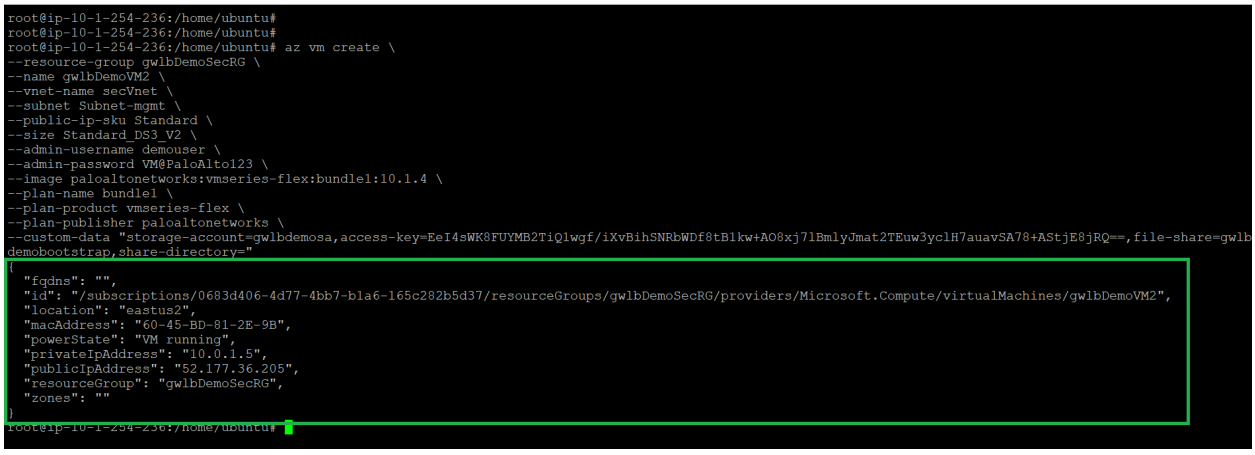
--custom-data "storage-account=<myStorageAccountName>,access-key=<myAccessKey>,file-share=<FileName>,share-directory=<SharedDirectoryName>"

As part of this command specify the below mentioned parameters while keeping remaining parameters to default

resource-group	: Specify the resource group in which we have deployed a security stack. It will be **gwlbDemoSecRG**
name	: Name of the 2nd VM series firewall that you plan to deploy behind the already deployed GWLB. It will be **gwlbDemoVM2**
admin-username	: Username of your choice (Ex: demouser)
admin-password	: Password of your choice
custom-data	: Specify the storage account name (it will be **gwlbdemosa**), its associated access key and bootstrap file share(it will be **gwlbdemobootstrap**) that you have created as part of Azure GWLB Deployment Guide

On executing this command you will first see “Running” status as shown below

After successful completion of this command execution you will see the output as mentioned below. This will take around 3-4 min for the VM to get provisioned



2. Create NIC in the data subnet.

   az network nic create -g <myResourceGroup> --vnet-name secVnet

   --subnet Subnet-data -n <myDataNIC> --accelerated-networking

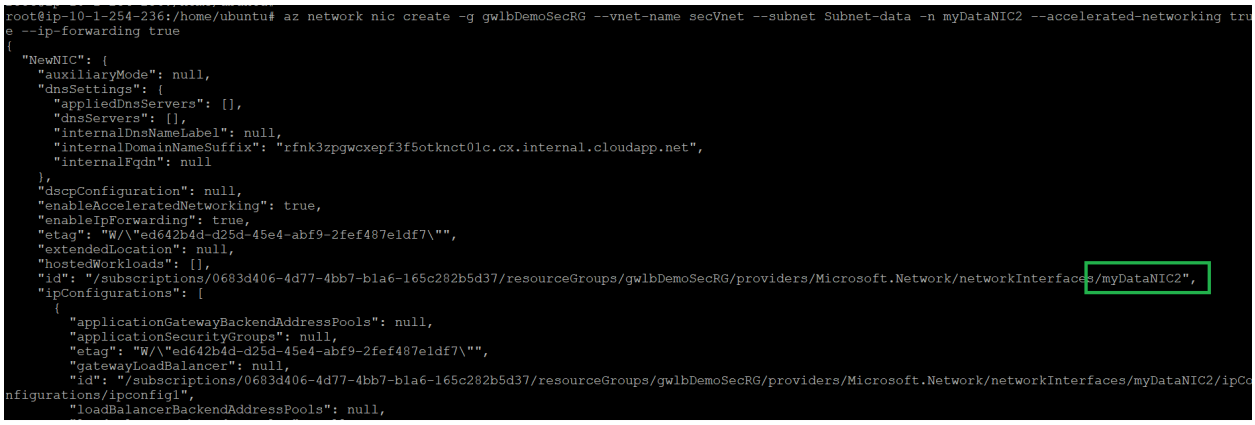
   true --ip-forwarding true

   As part of this command specify the below mentioned parameters while keeping remaining parameters to default

   resource-group : Specify the resource group in which we have deployed a security stack. It will be **gwlbDemoSecRG**

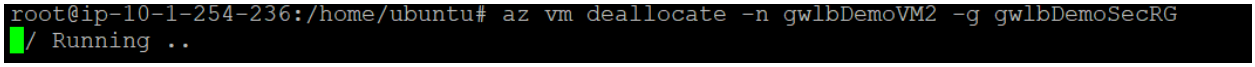
   Name of the data nic(*myDataNIC*) : **myDataNIC2**

   Output of this command execution will look as shown below.



3. Stop the VM created in Step 1.

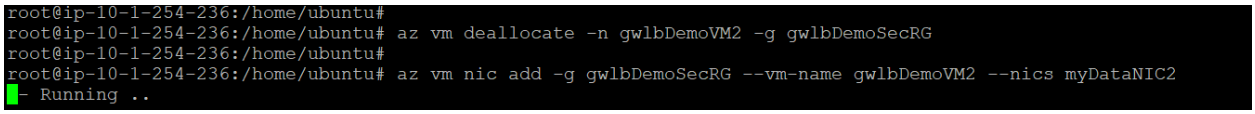
   az vm deallocate -n <myPA-VM> -g <myResourceGroup>



   This will take couple of minutes

4. Add the NIC created in Step 2 to the VM.

   az vm nic add -g <myResourceGroup> --vm-name <myPA-VM> --nics <myDataNIC>



   On Successful creation of the NIC, the output will look like this.

```
root@ip-10-1-254-236:/home/ubuntu# az vm nic add -g gwlbDemoSecRG --vm-name gwlbDemoVM2 --nics myDataNIC2
[
  {
    "deleteOption": null,
    "id": "/subscriptions/0683d406-4d77-4bb7-b1a6-165c282b5d37/resourceGroups/gwlbDemoSecRG/providers/Microsoft.Network/networkInterfaces/gwlbDemoVM2VMNic",
    "primary": true,
    "resourceGroup": "gwlbDemoSecRG"
  },
  {
    "deleteOption": null,
    "id": "/subscriptions/0683d406-4d77-4bb7-b1a6-165c282b5d37/resourceGroups/gwlbDemoSecRG/providers/Microsoft.Network/networkInterfaces/myDataNIC2",
    "primary": false,
    "resourceGroup": "gwlbDemoSecRG"
  }
]
root@ip-10-1-254-236:/home/ubuntu#
```

5. Add the VM to the backend address pool of the GWLB.

   az network nic ip-config address-pool add --address-pool BackendPool1 --ip-config-name ipconfig1 --nic-name <myDataNIC> --resource-group <myResourceGroup> --lb-name securityLB

   Successful command execution will look like this

```
root@ip-10-1-254-236:/home/ubuntu#
root@ip-10-1-254-236:/home/ubuntu# az network nic ip-config address-pool add --address-pool BackendPool1 --ip-config-name ipconfig1 --nic-name myDataNIC2 --r
esource-group gwlbDemoSecRG --lb-name securityLB
{
  "applicationGatewayBackendAddressPools": null,
  "applicationSecurityGroups": null,
  "etag": "W/\"daf4a710-1f96-49b2-a95e-912609b9cc30\"",
  "gatewayLoadBalancer": null,
  "id": "/subscriptions/0683d406-4d77-4bb7-b1a6-165c282b5d37/resourceGroups/gwlbDemoSecRG/providers/Microsoft.Network/networkInterfaces/myDataNIC2/ipConfigur
ations/ipconfig1",
  "loadBalancerBackendAddressPools": [
    {
      "backendIpConfigurations": null,
      "drainPeriodInSeconds": null,
      "etag": null,
      "id": "/subscriptions/0683d406-4d77-4bb7-b1a6-165c282b5d37/resourceGroups/gwlbDemoSecRG/providers/Microsoft.Network/loadBalancers/securityLB/backendAdd
ressPools/BackendPool1",
      "inboundNatRules": null,
      "loadBalancerBackendAddresses": null,
      "loadBalancingRules": null,
      "location": null,
      "name": null,
      "outboundRule": null,
      "outboundRules": null,
      "provisioningState": null,
      "resourceGroup": "gwlbDemoSecRG",
      "tunnelInterfaces": null,
      "type": null
    }
  ],
```

6. Start the VM.

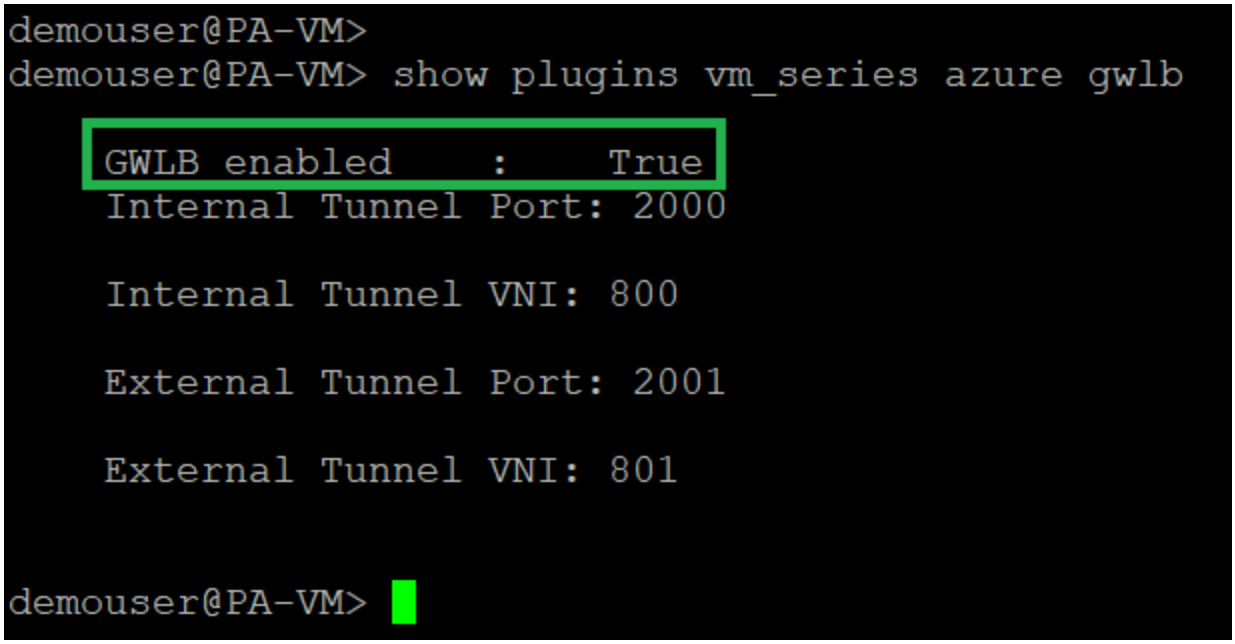
   az vm start -n <myPA-VM> -g <myResourceGroup>

```
root@ip-10-1-254-236:/home/ubuntu# az vm start -n gwlbDemoVM2 -g gwlbDemoSecRG
| Running ..
```

7. Connect to the firewall using SSH. Enter the following in the firewall CLI to verify if the GWLB is enabled.

   show plugins vm_series azure gwlb

   After starting the VM, we can use the above command to check if the newly added VM series firewall enabled with GWLB functionality and it was configured with appropriate configuration to connect with GWLB and send traffic over VXLAN tunnel towards GWLB.



We can see that the GWLB was enabled and also we have internal and external VXLAN tunnel ports and VNIs configured which will be used to send and receive traffic towards GWLB.

Summary

Integrating VM Series with Azure GWLB helps us seamlessly integrate security vnet into customer infrastructure without modifying any of their network components.

And with this integration we are going to get complete source's identity to actual application

Also we can avoid vnet peering and UDR creation to route traffic through VM Series firewall.

References

Microsoft Azure Gateway Load Balancer Overview - https://learn.microsoft.com/en-us/azure/load-balancer/gateway-overview

