Palo Alto Networks VM-Series Integration with AWS Gateway Load Balancer

This package will help you deploy VM-Series behind an AWS Gateway Load Balancer using the security_stack terraform template to secure your Inbound, Outbound and East-West traffic.

The app_stack is an optional deployment of a sample web application behind an Application Load Balancer to demonstrate the Traffic Flow.

You can deploy the security_stack and two sets of app_stack to bring up a topology as stated in the diagram below.

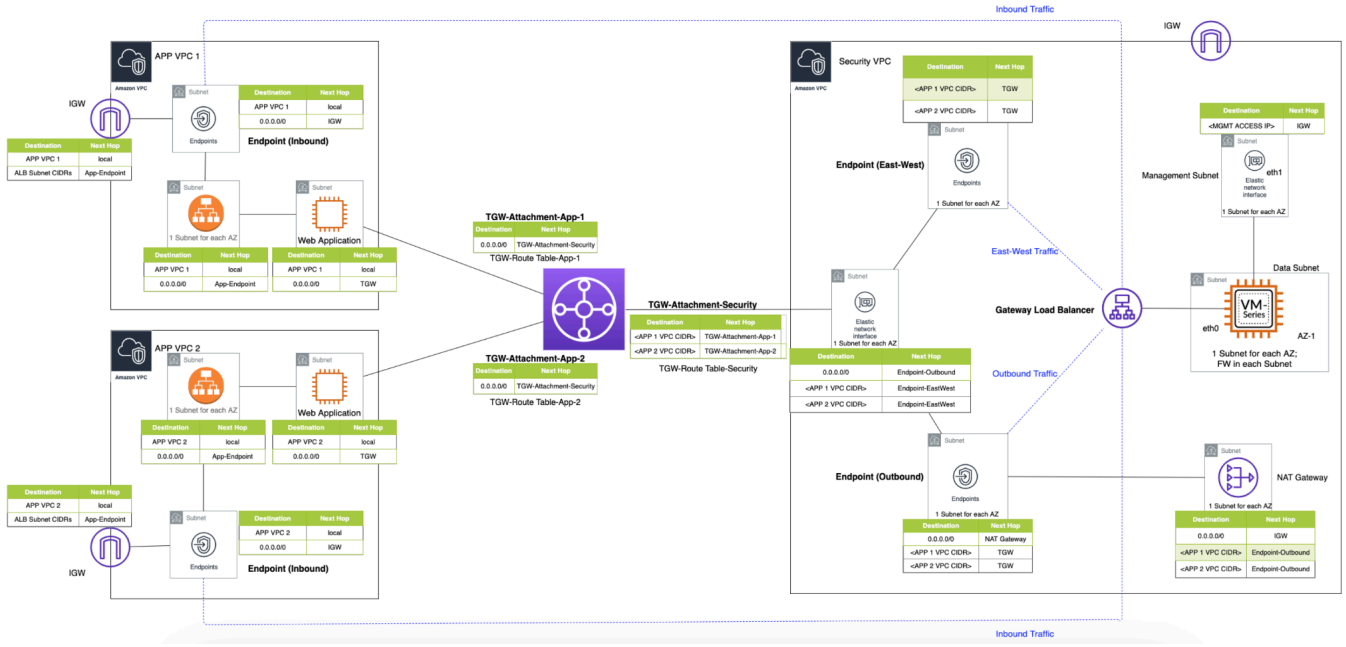


Table of Contents

1. Prerequisites
2. Deploy Security Stack
3. Deploy Application Stack (Optional)
4. Verify Traffic Inspection On VM-Series (Optional)
5. Support

Prerequisites

1. Install Python 3.6.10. You can use the following link for your reference: (https://www.python.org/downloads/release/python-3610/). Once the installation is complete, please use the following command to confirm the installation was successfully completed:

   python3 --version

   - This should point to Python 3.6

   npandey@M-C02FP0DZML87 ~ % python3 --version
   Python 3.6.10
   npandey@M-C02FP0DZML87 ~ %

2. Install Terraform 0.12 You use the following link for your reference: (https://developer.hashicorp.com/terraform/install)

   Confirm the installation using the following command:

   terraform version

   - This should point to Terraform v0.12.x

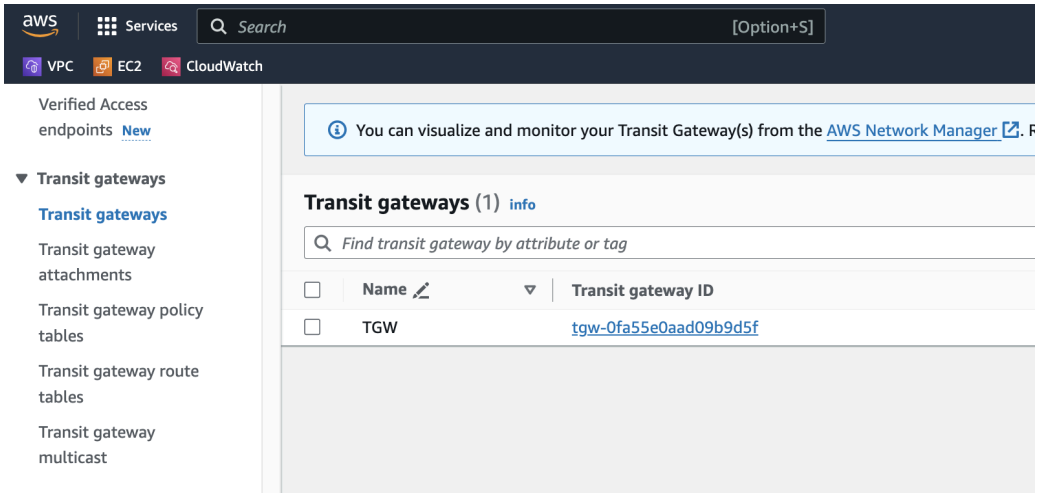
3. Install the python requirements using the following command:
   - pip3 install --upgrade -r requirements.txt

4. Create a local ssh keypair to access instances deployed by this terraform deployment.
    - Private Key: openssl genrsa -out private_key.pem 2048
    - Change Permissions: chmod 400 private_key.pem
    - Public Key: ssh-keygen -y -f private_key.pem > public_key.pub

5. To support East-West and Outbound Traffic, use an existing or create a new Transit Gateway on your AWS account. You can use the following link for your reference:

    https://docs.aws.amazon.com/vpc/latest/tgw/tgw-getting-started.html

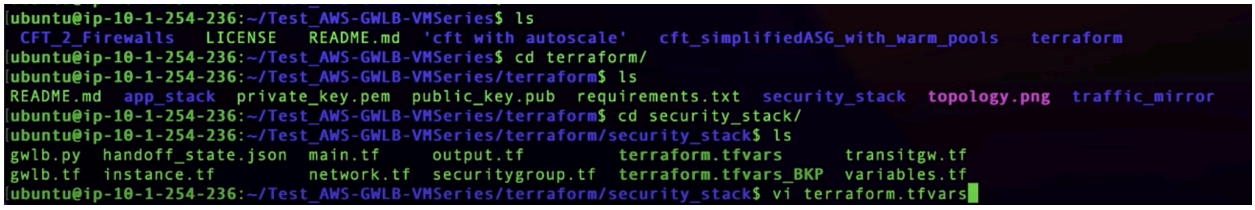
    - Transit Gateway should use the following settings:
        - Default route table association: disable
        - Default route table propagation: disable
    - Note the Transit Gateway ID



With this, your environment is now ready to deploy VM-Series in integration with AWS Gateway Load Balancer.

Deploy Security Stack

1. Setup the variables for the Security stack in security_stack/terraform.tfvars. Some of them are explained below:



   - Parameter(Mandatory) vpc_cidr:
     - Use a /23 or bigger CIDR block. Ex. 10.0.0.0/16
   - Parameter(Mandatory) tgw_id:
     - Use the Transit Gateway ID you noted in the Prerequisites Step 5.

     Ex. tgw_id = "tgw-0xxxxxxxxx"

   - Parameter(Mandatory) public_key:
     - Use the contents of the public key created in Prerequisites Step 4 as parameter value.

     Ex. public_key="ssh-rsa xxxxxxxxxx"

   - Parameter(Optional) user_data:
     - Option 1: Enter a Basic Configuration as User Data.

     Ex. user_data="type=dhcp-client\nhostname=PANW\nauthcodes=<Vm-Series Licensing Authcode>".

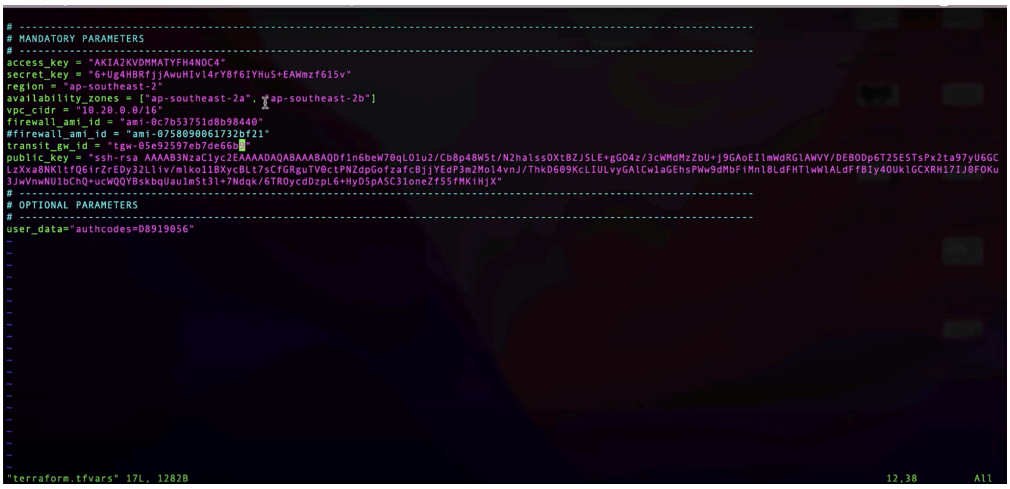
     - Option 2: Use an S3 Bootstrap bucket.

     Ex. user_data="vmseries-bootstrap-aws-s3bucket=<s3-bootstrap-bucket-name>".
     If Option 2 is used, please make sure you have the following line in init-cfg.txt file:
     plugin-op-commands=aws-gwlb-inspect:enable

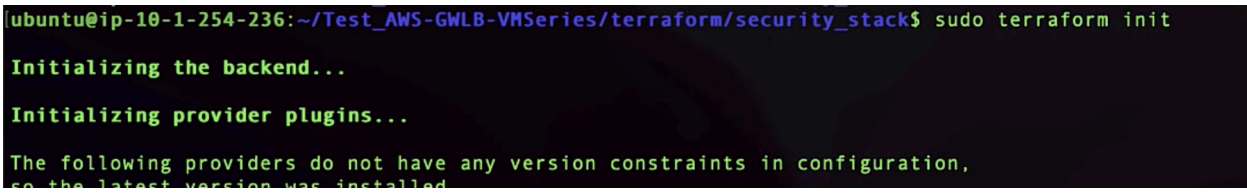
   - Parameter(Optional) prefix:
     - Use this parameter to prefix every resource with this string.

- Parameter(Optional) fw_mgmt_sg_list and app_mgmt_sg_list:
  - Use this parameter to input a list of IP addresses with CIDR from which you want Firewall/App Management to be accessible.

- Check out all optional terraform parameters for more functionality. The parameter definitions lie in the file variables.tf



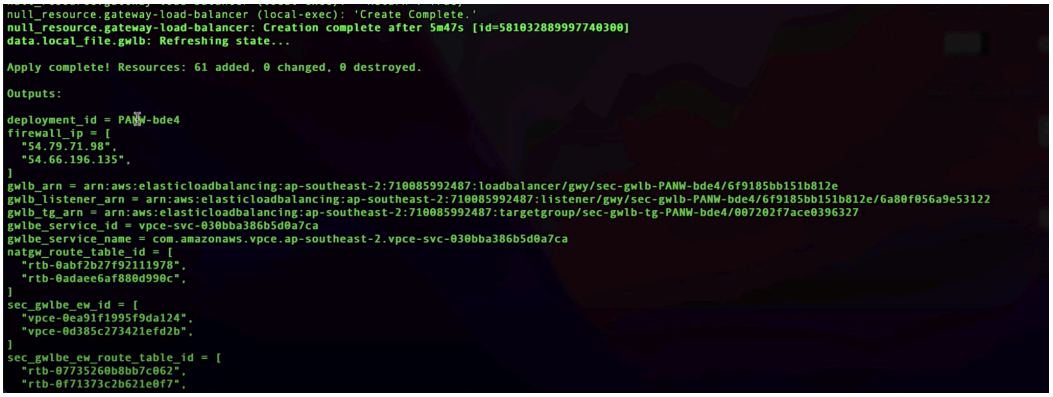
2. Deploy Security Stack using Terraform
   - Go to the security stack terraform directory cd /path/to/security_stack/
   - Initialize terraform terraform init



   - Apply the template terraform apply

   - This will start the deployment
   - You will see the output as shown in the screenshot once it is complete. It takes about 2-3 min to complete the deployment.

You will notice the resources getting deployed sequentially. Once the deployment completes, you will see the number of resources deployed and their identifiers as well.



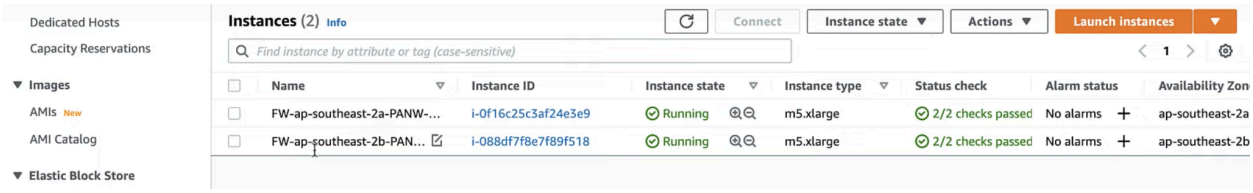
- The output will look like:

Unset

```
deployment_id = <prefix-id>
firewall_ip = [
   <Firewall Management IP>,
   <Firewall Management IP>,
]
gwlb_arn = <Gateway Load Balancer ARN>
gwlb_listener_arn = <Listener ARN>
gwlb_tg_arn = <Target Group ARN>
gwlbe_service_id = <VPC Endpoint Service ID>
gwlbe_service_name = <VPC Endpoint Service Name>
natgw_route_table_id = [
   <NAT Gateway Route Table ID>,
   <NAT Gateway Route Table ID>,
]
sec_gwlbe_ew_id = [
   <East-West VPC Endpoint ID>,
   <East-West VPC Endpoint ID>,
]
sec_gwlbe_ew_route_table_id = [
```

```
    <East-West Endpoint Route Table ID>,
    <East-West Endpoint Route Table ID>,
]
sec_gwlbe_ob_id = [
    <Outbound VPC Endpoint ID>,
    <Outbound VPC Endpoint ID>,
]
sec_gwlbe_ob_route_table_id = [
    <Outbound Endpoint Route Table ID>,
    <Outbound Endpoint Route Table ID>,
]
sec_tgwa_route_table_id = [
    <Transit Gateway Attachment Subnet Route Table ID>,
    <Transit Gateway Attachment Subnet Route Table ID>,
]
tgw_id = <Transit Gateway ID>
tgw_sec_attach_id = <Transit Gateway Security VPC Attachment ID>
tgw_sec_route_table_id = <Transit Gateway Security Route Table ID>
```

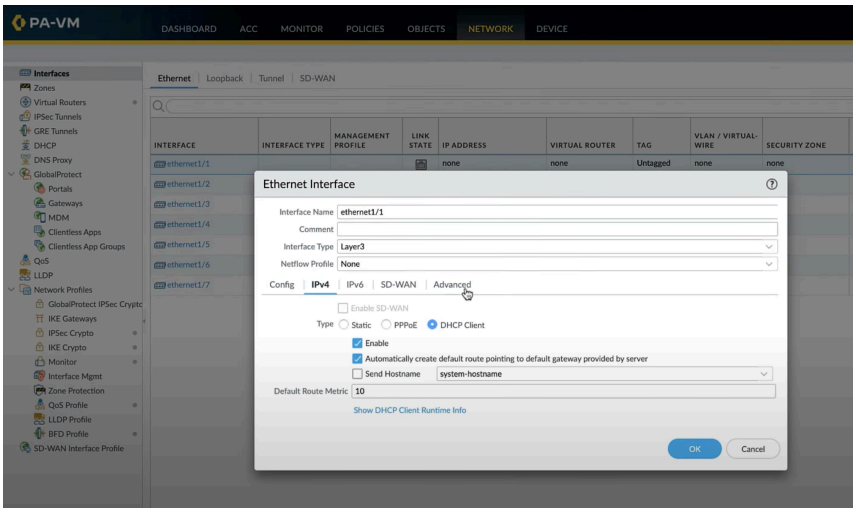
3. Wait for the VM Series Firewall to boot up. It will take a few minutes based on the user_data passed to the terraform. From the AWS console, you can observe the instance state and status of the firewall.



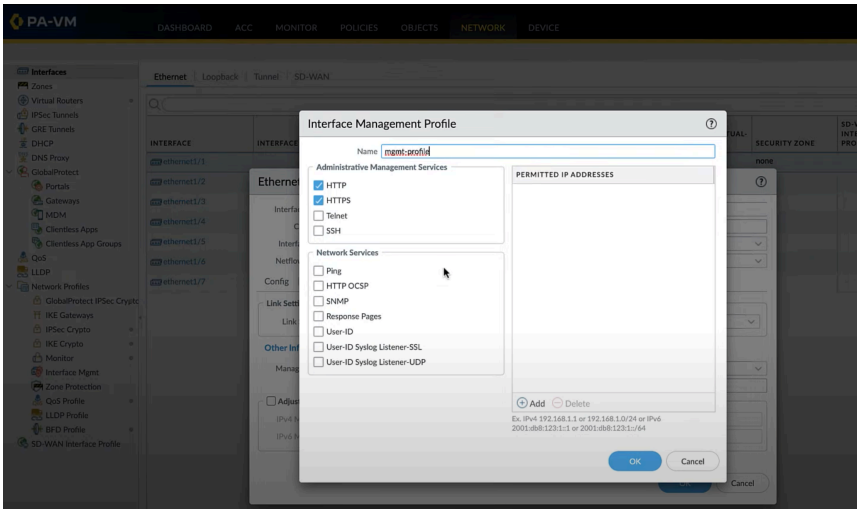
You can log in to the firewall using ssh as shown in the next step to verify if the firewall is ready. It usually takes a few minutes before you can move to the next step.

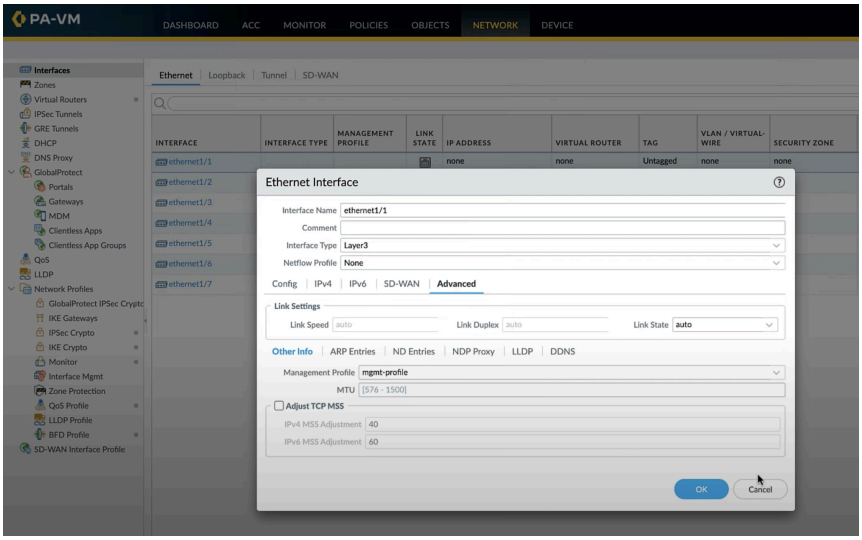
4. Once the VM-Series is up and running, log in to your firewall:
    - ssh -i private_key.pem admin@<Firewall IP>
    - firewall_ip can be found in the Security Stack deployment output

- Once you have logged in, you will need to configure your firewall to allow traffic:
  - Configure interface 'ethernet1/1' as layer 3 with DHCP
  - Configure interface 'ethernet1/1' with a virtual router, zone and interface management profile.

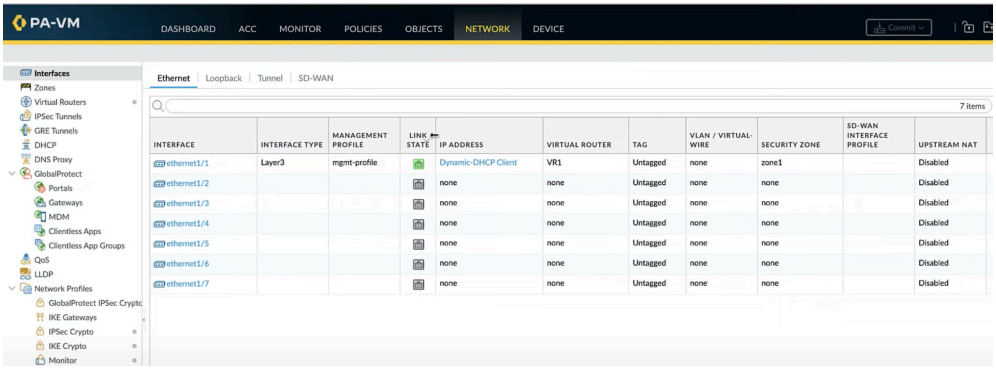


  - This template configures a health check (TCP over port 80) on the Target Group so configure the interface management profile(for ethernet1/1) on the firewall to accept this traffic.





- With this, 'ethernet1/1' behaves as both an ingress and egress interface.
- This firewall configuration will allow traffic to flow based on Interzone default policy.



Congratulations! You have successfully deployed the security stack.

Deploy Application Stack (Optional)

1. You can now deploy a Sample Application stack to view the traffic flow.
2. Set up the variables for the App Stack in app_stack/terraform.tfvars.

```
ubuntu@ip-10-1-254-236:~/Test_AWS-GWLB-VMSeries/terraform/app_stack$ pwd
/home/ubuntu/Test_AWS-GWLB-VMSeries/terraform/app_stack
ubuntu@ip-10-1-254-236:~/Test_AWS-GWLB-VMSeries/terraform/app_stack$ ls
alb.tf    gwlbe.tf           instance.tf  network.tf  securitygroup.tf          terraform.tfvars  web_server.sh
gwlbe.py  handoff_state.json  main.tf      output.tf   terraform.tfstate.backup  variables.tf
ubuntu@ip-10-1-254-236:~/Test_AWS-GWLB-VMSeries/terraform/app_stack$
```

- Use the Security Stack Terraform output and setup for the variables.
- Check out all optional terraform parameters for more functionality. The parameter definitions lie in the file variables.tf

```
# ------------------------------------------------------------------------------------------------------------------------
# MANDATORY PARAMETERS
# ------------------------------------------------------------------------------------------------------------------------
access_key = "AKIA2KVDMMATYFH4NOC4"
secret_key = "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"
region = "ap-southeast-2"
availability_zone = "ap-southeast-2a"
vpc_cidr = "10.30.0.0/16"
public_key = "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDf1n6beW70qLO1u2/Cb8p48W5t/N2halssOXtBZJSLE+gGO4z/3cWMdMzZbU+j9GAoEIlmWdR
GlAWVY/DEBODp6T25ESTsPx2ta97yU6GCLzXxa8NKltfQ6irZrEDy32Lliv/mlko11BXycBLt7sCfGRguTV0ctPNZdpGofzafcBjjYEdP3m2Mol4vnJ/ThkD609Kc
LIULvyGAlCw1aGEhsPWw9dMbFiMnl8LdFHTlwWlALdFfBIy4OUklGCXRH17IJ0FOKu3JwVnwNU1bChQ+ucWQQYBskbqUau1mSt3l+7Ndqk/6TROycdDzpL6+HyD5p
ASC31oneZf55fMKiHjX"

# Parameters to be filled in from Security Stack Deployment Output
gwlbe_service_id = "vpce-svc-030bba386b5d0a7ca"
gwlbe_service_name = "com.amazonaws.vpce.ap-southeast-2.vpce-svc-030bba386b5d0a7ca"
natgw_route_table_id = ["rtb-0abf2b27f92111978","rtb-0adaee6af880d990c"]
sec_gwlbe_ew_id = ["vpce-0d385c273421efd2b","vpce-0ea91f1995f9da124"]
sec_gwlbe_ew_route_table_id = ["rtb-0f71373c2b621e0f7","rtb-07735260b8bb7c062"]
sec_gwlbe_ob_id = ["vpce-0f240d2865745638c","vpce-04db4fe1bc5a6eadb"]
sec_gwlbe_ob_route_table_id = ["rtb-040026fadb02d577a","rtb-0a9db781a54786ce4"]
sec_tgwa_route_table_id = ["rtb-0ff0eb6d1470b1b25","rtb-024db770211c8fb95"]
tgw_id = "tgw-05e92597eb7de66b9"
tgw_sec_attach_id = "tgw-attach-09cc478995252b7c8"
tgw_sec_route_table_id = "tgw-rtb-0a17e36aec8b46206"

# ------------------------------------------------------------------------------------------------------------------------
# OPTIONAL PARAMETERS
# ------------------------------------------------------------------------------------------------------------------------
~
~
~
~
"terraform.tfvars" 27L, 1891B                                                                    13,27         All
```

3. Deploy App Stack using Terraform
   - Go to the app stack terraform directory cd /path/to/app_stack/
   - Initialize terraform terraform init
   - Apply the template terraform apply
   - This will start the deployment

```
aws_route.sec-agwe-ob-to-tgw[0]: Creation complete after 3s [id=r-rtb-040026fadb02d577a805491791]

Apply complete! Resources: 39 added, 0 changed, 0 destroyed.

Outputs:

app_fqdn = app-alb-PANW-1b2c-1715037188.ap-southeast-2.elb.amazonaws.com
app_gwlbe_id = vpce-0bbc3ebd59f486daf
app_mgmt_ip = 13.211.73.89
deployment_id = PANW-1b2c
ubuntu@ip-10-1-254-236:~/Test_AWS-GWLB-VMSeries/terraform/app_stack$
```

You will see this output once the deployment is complete.

- The output will look like:

```
Unset
app_gwlbe_id = <Inbound VPC Endpoint ID>
app_fqdn = <Application Load Balancer FQDN>
app_mgmt_ip = <Application Instance Management IP>
deployment_id = <Prefix-ID>
```

- Once deployed the Application will be ready in a few minutes.

Congratulations! You have successfully deployed an Application Stack.

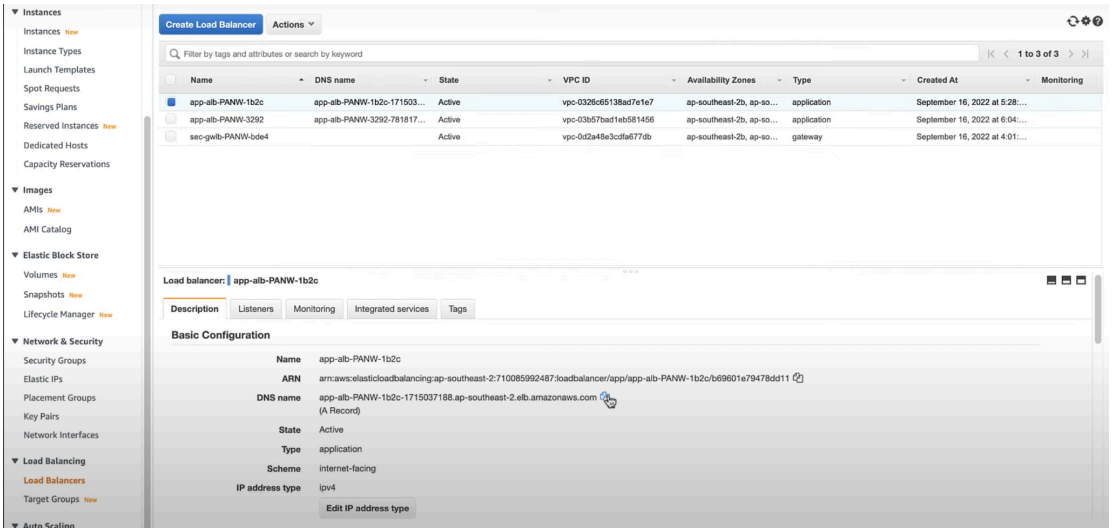
Inspect Traffic On VM-Series (Optional)

You can now inspect the traffic on VM-Series Firewall:

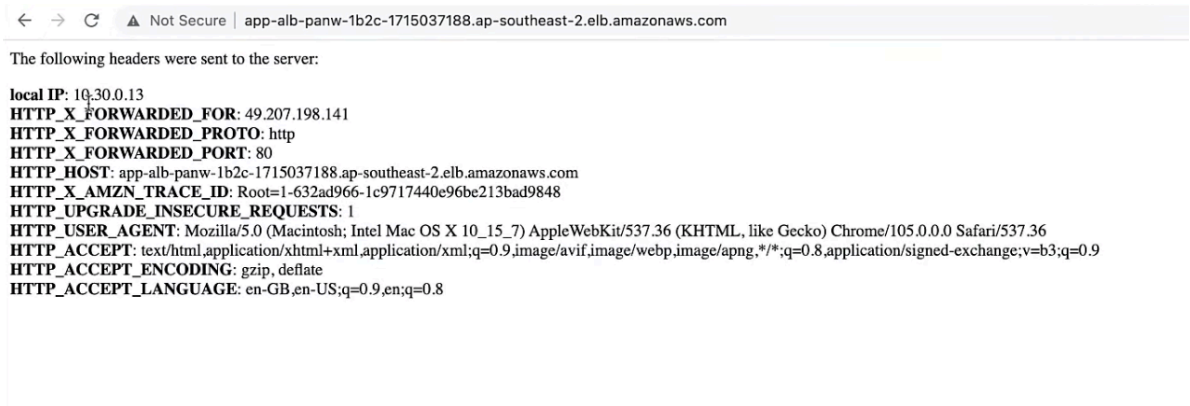
1. Inbound Traffic Inspection: This refers to the traffic flow originating from the internet destined towards applications/workloads hosted within the AWS environment.
    - Access the Application Web Page from your browser
    - Go to a browser and execute http://<fqdn>

      app_fqdn can be found in the App Stack deployment output

    - This inbound traffic destined for the application is inspected by the Firewall and can be seen in it.



The web page response should look like this-



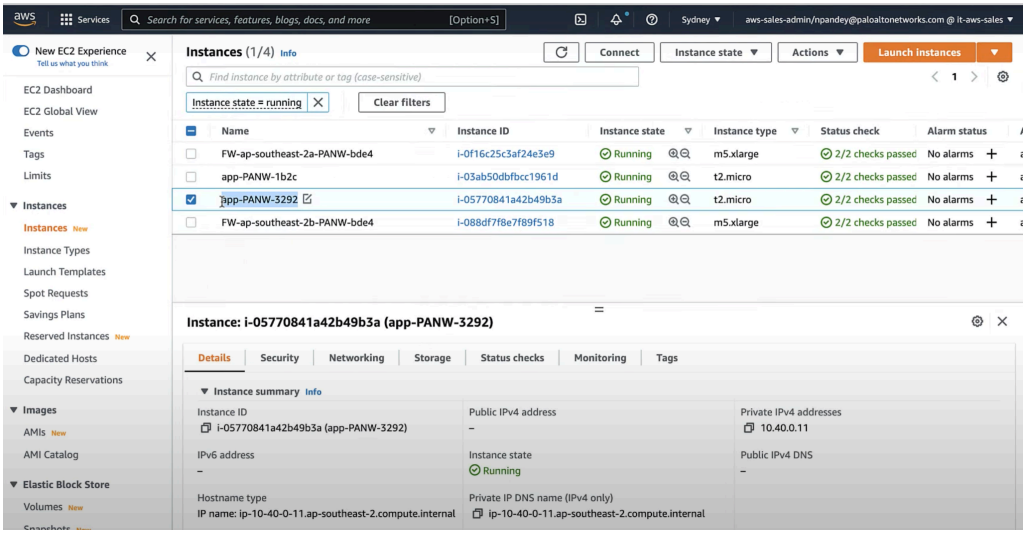
2. Outbound Traffic Inspection: This is the traffic towards the internet originating from within the AWS environment.
    - Log in to your Application Instance ssh -i private_key.pem ubuntu@<Application IP>
    - app_mgmt_ip can be found in the App Stack deployment output
    - Execute the following command: curl https://www.paloaltonetworks.com
    - This outbound traffic coming from the application(via Transit Gateway) is inspected by the Firewall and can be seen in it.
3. East-West Traffic Inspection: This is the traffic between the workloads within the AWS environment.
    - To achieve this, we will have to deploy another app_stack to send traffic between two applications
    - Create a copy of the directory app_stack

```
Unset
mkdir app_stack_2
cp app_stack/* app_stack_2/
```

- Go to the app stack 2 directory and confirm you are not carrying any terraform state files from the previous application

```
Unset
cd app_stack_2
rm -rf *.tfstate
```

- Setup the variables and deploy another application stack(Follow [Application Stack Steps](#Deploy Application Stack))
- Make sure App 1 and App2 deployments have different vpc_cidr.
- Once deployed, wait for the application instance to boot up. This should take 2-3 min for the deployment to complete.



- Login to your new Application Instance (APP 2) ssh -i private_key.pem ubuntu@<Application IP>

  app_mgmt_ip can be found in the App Stack deployment output

- Execute the following command: curl http://<APP 1 Private IP>
- This East-West traffic coming from App 2 and destined to App 1 goes via the Transit gateway to be inspected by the Firewall.

4. Inbound, Outbound and East-West Traffic Isolation (Advanced):
    - To isolate your inbound, outbound and east-west traffic, you can associate specific endpoints to sub-interfaces.
    - Follow the document to learn more about this feature: Associate a VPC-Endpoint with a VM-Series Sub-interface

With this, you inspected Inbound, Outbound and East-West traffic on the Application with VM-Series Firewall.

Video Link - https://www.youtube.com/watch?v=jPQeSJaUavg

Support

These templates are released under an as-is, best-effort, support policy. These scripts should be seen as community supported and Palo Alto Networks will contribute our expertise as and when possible. We do not provide technical support or help in using or troubleshooting the components of the project through our normal support options such as Palo Alto Networks support teams, or ASC (Authorized Support Centers) partners and backline support options. If additional support is needed then please reach out to Palo Alto Networks Professional Services. The underlying product used (the VM-Series firewall) by the scripts or templates are still supported, but the support is only for the product functionality and not for help in deploying or using the template or script itself. Unless explicitly tagged, all projects or work posted in our GitHub repository (at https://github.com/PaloAltoNetworks) or sites other than our official Downloads page on https://support.paloaltonetworks.com are provided under the best effort policy.