Deploy the Threat Defense Virtual Auto Scale Solution using GWLB on AWS to Inspect North-South Traffic

First Published: 2023-06-01

Introduction

This document explains how to deploy the Threat Defense Virtual auto scale solution using GWLB on AWS to inspect north-south traffic

How to Set Up the Threat Defense Virtual Auto Scale Solution using GWLB on AWS to Inspect North-South Traffic

The auto scale solution enables the deployment, scaling, and management of a group of Threat Defense Virtual instances that are hosted for traffic inspection. The traffic is distributed across single or multiple Threat Defense Virtual instances depending on performance or usage capacity.

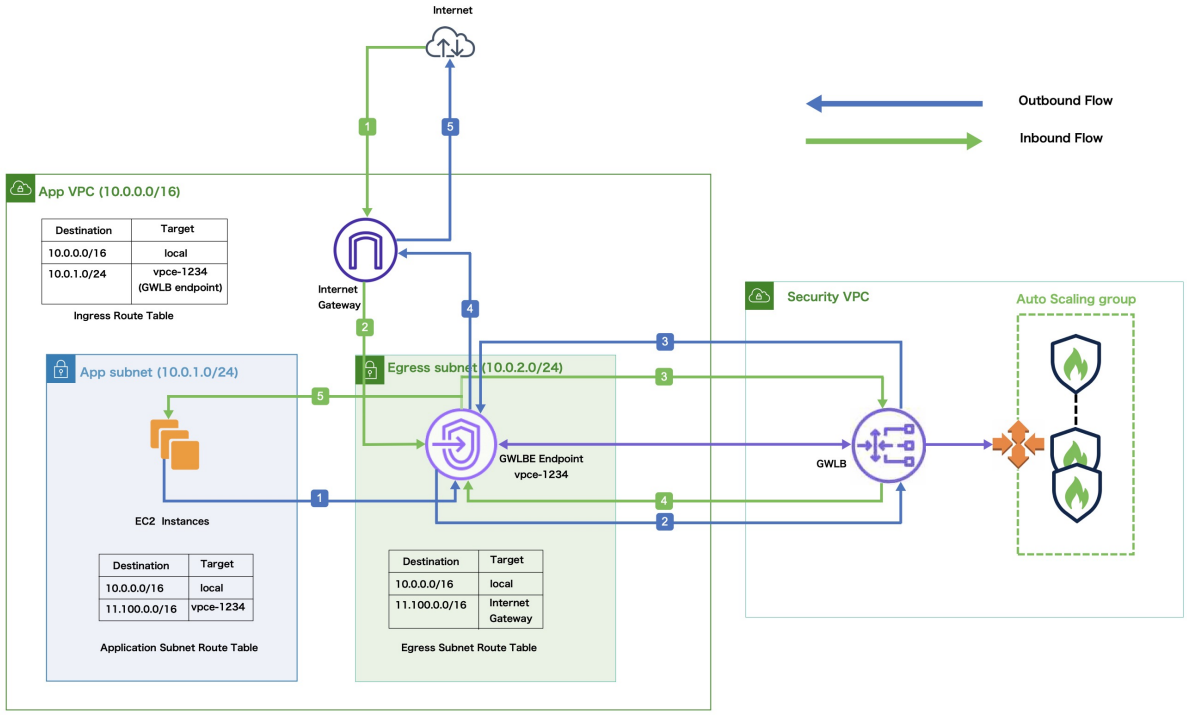
The GWLB acts as a single entry and exit point to manage internally and externally generated traffic and scales up or down the number of Threat Defense Virtual instances in real time based on traffic load.

Note The parameter values used in this use case are sample values. Change these values as per your requirement.

Sample Topology

This sample topology illustrates how inbound and outbound network traffic flow is distributed to Threat Defense Virtual instances through the GWLB and then routed to the application VPC and back.

Figure 1: Threat Defense Virtual Auto Scale Solution with GWLB



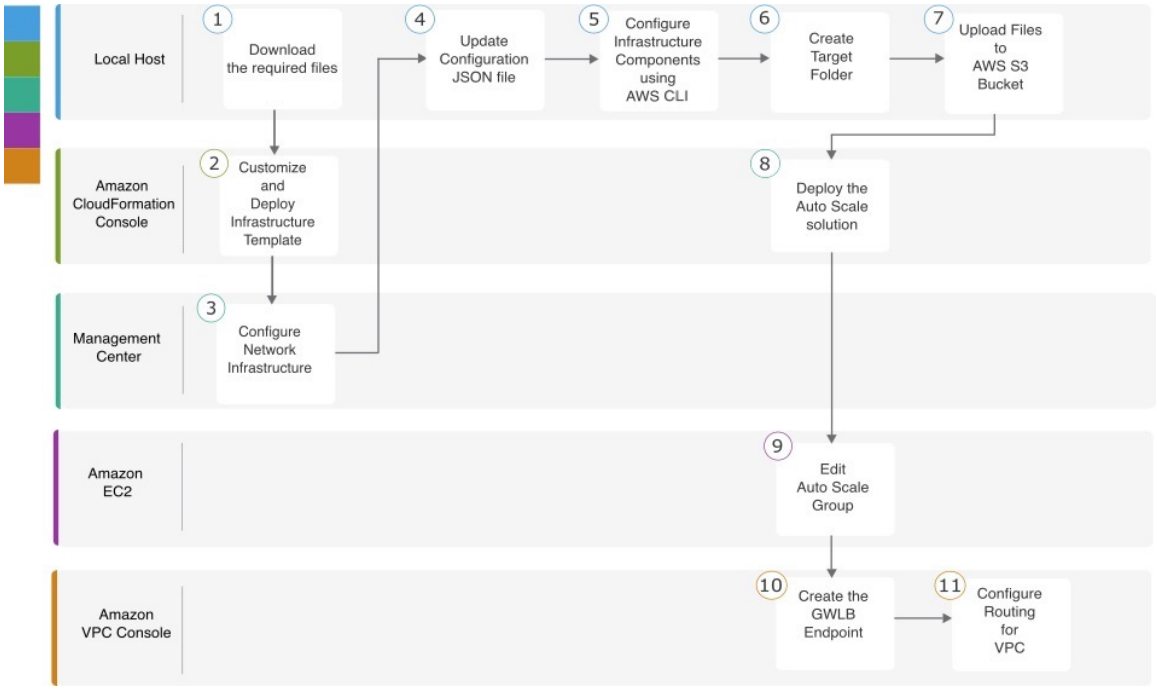
Inbound Traffic Inspection	
1	The Internet Gateway (IGW) receives traffic from the Internet.
2	Traffic is routed to the Gateway Load Balancer endpoint (GWLBe) as per the routes in the Ingress Route Table.
3	The GWLBe is attached to the endpoint service in the Security Virtual Private Cloud (VPC). The GWLB encapsulates the received traffic and forwards it to the Threat Defense Virtual auto scaling group for inspection.
4	The traffic inspected by the auto scaling group is returned to the GWLB and then to the GWLB endpoint.
5	The GWLB endpoint forwards the traffic to the Application VPC from where it is routed to the resources in the Application subnet.

Outbound Traffic Inspection	
1	Traffic from the Application subnet resources is routed to the GWLBe in the same VPC.
2	The GWLBe is attached to the endpoint service in the Security VPC. The GWLB encapsulates the received traffic and forwards the same to the auto scaling group for inspection.

Outbound Traffic Inspection	
3	The traffic inspected by the auto scaling group is returned to the GWLB and then to the GWLBe.
4	After the traffic arrives in the origin VPC, it is forwarded to the IGW as per the routes defined in the Egress Subnet Route Table.
5	The IGW sends the traffic to the Internet.

End-to-End Procedure

The following flowchart illustrates the workflow for deploying Threat Defense Virtual auto scale solution with GWLB on Amazon Web Services (AWS).



	Workspace	Steps
1	Local Host	Prerequisites
2	Amazon CloudFormation Console	Amazon CloudFormation console – Customize and Deploy the Infrastructure Template
3	Management Center	Management Center - Configure Network Infrastructure in Management Center for Threat Defense Virtual
4	Local Host	Local Host - Update the Configuration JSON File

	Workspace	Steps
5	Local Host	Local Host - Configure Infrastructure Components using AWS CLI on the Local Host
6	Local Host	Local Host – Create Target Folder
7	Local Host	Local Host - Upload AWS GWLB Auto Scale Solution Deployment Files to the Amazon S3 Bucket
8	Amazon CloudFormation Console	Amazon CloudFormation console - Deploy the Auto Scale Solution for the Threat Defense Virtual using GWLB
9	Amazon EC2 Console	Amazon EC2 console - Edit the Number of Instances in the Auto Scale Group
10	Amazon VPC Console	Create the GWLB Endpoint
11	Amazon VPC Console	Configure Routing for the Customer VPC

Prerequisites

- Download the **lambda-python-files** folder from GitHub. This folder contains the following files:
  - Python (.py) files that are used to create the lambda layer.
  - A configuration.json file that is used to add static routes and customize any network parameters, as required.
- Download the following CloudFormation templates from GitHub:
  - **infrastructure_gwlb.yaml –** Used to customize the components in the AWS environment.
  - **deploy_ngfw_autoscale_with_gwlb.yaml –** Used to deploy the AWS Auto Scale with GWLB solution.
- [Optional] Collect values for the template parameters, wherever possible. This will make it easier to enter the values quickly while deploying the templates on the AWS Management console.

Amazon CloudFormation console – Customize and Deploy the Infrastructure Template

Perform the steps given in this section to customize and deploy the infrastructure template.

Procedure

Step 1 On the AWS Management console, go to **Services** > **Management and Governance** > **CloudFormation**, and click **Create stack** > **With new resources(standard)**.

Step 2 Choose **Upload a template** file, click **Choose** file, and select **infrastructure_gwlb.yaml** from the folder in which you downloaded the files.

Step 3 Click **Next**

Step 4 On the **Specify stack details** page, enter a name for the stack.

Step 5 Provide values for the input parameters in the ***infrastructure_gwlb.yaml*** template.

Parameters	Values
Pod Configuration	
Pod Name	*infrastructure*
Pod Number	1
S3 Bucket Name	*demo-us-bkt*
VPC CIDR	*20.0.0.0/16*
Number Of Availability Zones	2
ListOfAzs (List of Availability Zones)	*us-west-1a,us-west-1b*
Name of the Management Subnets	*MgmtSubnet-1,MgmtSubnet-2*
MgmtSubnetCidrs	*20.1.250.0/24,20.1.251.0/24*
Name of the Inside Subnets	*InsideSubnet-1,InsideSubnet-2*
InsideSubnetCidrs	*20.1.100.0/24,20.1.101.0/24*
Name of the Outside Subnets	*OutsideSubnet-1,OutsideSubnet-2*
OutsideSubnetCidrs	*20.1.200.0/24,20.1.201.0/24*
Name of the Lambda Subnets	*LambdaSubnet-1,LambdaSubnet-2*
Lambda Subnet CIDR	*20.1.50.0/24,20.1.51.0/24*

Step 6 Click **Next**.

Step 7 Click **Next** on the **Configure Stack Options** window.

Step 8 On the **Review** page, review and confirm the settings.

Step 9 Click **Create Stack** to deploy the **infrastructure_gwlb.yaml** template and create the stack.

Step 10 After the deployment is complete, go to **Outputs** and note the **S3 Bucket Name**.

Management Center - Configure Network Infrastructure in Management Center for Threat Defense Virtual

Create and configure objects, device droups, health check port, and access policies, in the Management Center for the registered Threat Defense Virtual.

Create a Host object

Procedure

Step 1 Log in to the Management Center.

Step 2 Choose **Objects** > **Object Management**.

Step 3 Choose **Network** from the list of object types.

Step 4 Choose **Add Object** from the **Add Network** drop-down menu.

Step 5 Enter a **Name** - *aws-metadata-server*.

Step 6 Enter a Description.

Step 7 In the **Network** field, select the **Host** option and enter the IPv4 address - *169.254.169.254*.

Step 8 Click **Save**.

Create a Port object

Procedure

Step 1 Log in to the Management Center.

Step 2 Choose **Objects** > **Object Management**.

Step 3 Choose **Port** from the list of object types.

Step 4 Choose **Add Object** from the **Add Port** drop-down menu.

Step 5 Enter a **Name** - *test-port-object*.

Step 6 Choose a **Protocol**. You must choose the protocol that you have entered for the Host object type. Depending on the protocol you chose, constrain by **Port**.

Step 7 Enter *8080*. Note that you can customise the port number that you enter here as per your requirement.

Note You must constrain the object by port if you chose to match **All** protocols, using the **Other** drop-down list.

Step 8 Click **Save**.

Create Security Zone and Interface Group Objects

Procedure

Step 1 Choose **Objects** > **Object Management**.

Step 2 Choose **Interface** from the list of object types.

Step 3 Click **Add** > **Security Zone** or **Add** > **Interface Group**.

Step 4 Enter a **Name** - **`Inside-sz/Outside-sz`**.

Step 5 Choose an **Interface Type**.

Step 6 From the **Device** > **Interfaces** > drop-down list, choose a device that contains interfaces you want to add.

Step 7 When you create or edit a security zone, the **Device** > **Interfaces** > drop-down list displays the cluster names for high-availability devices. Choose the cluster that contains the interfaces you want to add.

Step 8 Choose one or more interfaces.

Step 9 Click **Add** to add the interfaces you chose, grouped by the device.

Step 10 Click **Save**.

Add Device Group

The Management Center allows you to group devices so you can easily deploy policies and install updates on multiple devices. You can expand and collapse the list of devices in the group.

Procedure

Step 1 Choose **Devices** > **Device Management**.

Step 2 From the **Add** drop-down menu, choose **Add Group**.

Step 3 To edit an existing group, click **Edit** (edit icon) for the group you want to edit.

Step 4 Enter a **Name** - *aws-ngfw-autoscale-dg*.

Step 5 Under **Available Devices**, choose one or more devices to add to the device group. Use **Ctrl** or **Shift** while clicking to choose multiple devices.

Step 6 Click **Add** to include the devices you chose in the device group.

Step 7 Click **OK** to add the device group.

Enable Port 443 (HTTP) for Health Check Probe

If you are using port 443 (HTTP) for the health check probe, perform the following procedure to enable the port for the health check probe.

Procedure

Step 1 Choose **Devices** > **Platform Settings** > **HTTP Access**.

Step 2 Select the **Enable HTTP Server** checkbox.

Step 3 Enter **443** in the **Port** field.

Step 4 Click **+ Add**.

Step 5 Select the relevant **IP Address** from the drop-down list.

Step 6 From the **Available Zones/Interfaces** window, select the outside interface that is connected to the GWLB or the outside subnet.

Step 7 Click **Add** to add that interface to the **Selected Zones/Interfaces** window.

Step 8 Click **OK**.

Step 9 Click **Save**.

Create a Basic Access Control Policy

When you create a new access control policy, it contains default actions and settings. After creating the policy, you are immediately placed in an edit session so that you can adjust the policy to suit your requirements.

Procedure

Step 1 Choose **Policies** > **Access Control**.

Step 2 Click **New Policy**.

Step 3 Enter a unique Name - *aws-access-policy* and Description.

Step 4 Specify the initial **Default Action - Block all traffic**.

Step 5 Click **Save**.

Step 6 Click the **Edit** icon for the new policy that you created.

Step 7 Click **Add Rule**.

Step 8 Set the following parameters:

- Name: *inside-to-outside*
- Insert: *into Mandatory*
- Action: *Allow*
- Add a source zone and destination zone.

Step 9 Click **Apply**.

Local Host - Update the Configuration JSON File

The **configuration.json** file is in the **lambda_python_files** folder that you downloaded from GitHub. Update the parameters in the **configuration.json** file with the parameters set up by you in the management center.

The scripts in the configuration.json file are as given below.

```
"licenseCaps": ["BASE", "MALWARE", "THREAT"],   // Management center virtual licenses
  "fmcIpforDeviceReg": "DONTRESOLVE",   // Management center virtual IP address
```

```
  "RegistrationId": "cisco",  // Registration ID used while configuring the manager in the
 Threat defense virtual
  "NatId": "cisco",  // NAT ID used while configuring the manager in the Threat defense
virtual
  "fmcAccessPolicyName": "aws-access-policy",  // Access policy name configured in the
Management center virtual
  "fmcInsideNicName": "inside", //Threat defense virtual inside interface name
  "fmcOutsideNicName": "outside", //Threat defense virtual outside interface name
  "fmcInsideNic": "GigabitEthernet0/0", // Threat defense virtual inside interface NIC Name
 - GigabitEthernet for c4 instance types, and TenGigabitEthernet for c5 instance types)
  "fmcOutsideNic": "GigabitEthernet0/1", // Threat defense virtual outside interface NIC
Name - GigabitEthernet for c4 instance types, and TenGigabitEthernet for c5 instance types
  "fmcOutsideZone": "Outside-sz", //Outside Interface security zone name that is set in the
 Management center virtual
  "fmcInsideZone": "Inside-sz", //Inside Interface security zone name that is set in the
Management center virtual
  "interfaceConfig": [
    {
      "managementOnly": "false",
      "MTU": "1500",
      "securityZone": {
        "name": "Inside-sz"
      },
      "mode": "NONE",
      "ifname": "inside",
      "name": "GigabitEthernet0/0"
    },
    {
      "managementOnly": "false",
      "MTU": "1500",
      "securityZone": {
        "name": "Outside-sz"
      },
      "mode": "NONE",
      "ifname": "outside",
      "name": "GigabitEthernet0/1"
    }
  ],  // Interface-related configuration
  "trafficRoutes": [
    {
      "interface": "inside",
      "network": "any-ipv4",
      "gateway": "",
      "metric": "1"
    }
  ] // This traffic route is used for the Threat defense virtual instance's health check
}
```

Local Host - Configure Infrastructure Components using AWS CLI on the Local Host

The templates do not create the Lambda layer and encrypted passwords for the threat defense virtual and management center. Configure these components using the procedures given below. See AWS Command Line Interface for more information on the AWS CLI.

Procedure

Step 1 Create Lambda Layer Zip File.

Create a python folder on your Linux host and then create the Lambda layer.

a) Create a python folder in your Linux host, such as Ubuntu 22.04.
b) Install Python 3.9 on your Linux host. A sample script to install Python 3.9 is given below.

```
$ sudo apt update
$ sudo apt install software-properties-common
$ sudo add-apt-repository ppa:deadsnakes/ppa
$ sudo apt install python3.9
$ sudo apt install python3-virtualenv
$ sudo apt install zip
$ sudo apt-get install python3.9-distutils
$ sudo apt-get install python3.9-dev
$ sudo apt-get install libffi-dev
```

c) Create a lambda layer zip file, *autoscale_layer.zip*, in your Linux environment. This file provides essential Python libraries for Lambda functions.

Run the following scripts to create the *autoscale_layer.zip* file.

```
#!/bin/bash
mkdir -p layer
mkdir -p python
virtualenv -p /usr/bin/python3.9 ./layer/
source ./layer/bin/activate
pip3 install attrs==23.1.0
pip3 install bcrypt==3.2.2
pip3 install certifi==2022.12.7
pip3 install cffi==1.15.1
pip3 install chardet==3.0.4
pip3 install cryptography==2.9.1
pip3 install idna==2.10
pip3 install jsonschema==3.2.0
pip3 install paramiko==2.7.1
pip3 install pycparser==2.21
pip3 install pycryptodome==3.15.0
pip3 install PyNaCl==1.5.0
pip3 install pyrsistent==0.19.3
pip3 install requests==2.23.0
pip3 install scp==0.13.2
pip3 install six==1.16.0
pip3 install urllib3==1.25.11
echo "Copy from ./layer directory to ./python\n"
cp -r ./layer/lib/python3.9/site-packages/* ./python/
zip -r autoscale_layer.zip ./python
```

d) After creating the **autoscale_layer.zip** file, copy the **autoscale_layer.zip** file to the **lambda-python-files** folder that is downloaded from GitHub.

Step 2 (Optional) Create Encrypted Passwords for the Threat Defense Virtual and Management Center.

If a KMS ARN value has been entered in the infrastructure_gwlb.yaml template file, the passwords that you set up in the threat defense virtual and management centre have to be encrypted. See Finding the key ID and key ARN to identify the key ARN using the AWS KMS console. On your local host, encrypt the password by running the following AWS CLI command.

```
$ aws kms encrypt --key-id <KMS-ARN> --plaintext 'MyC0mplIc@tedProtect1oN'
{
    "KeyId": "KMS-ARN",
    "CiphertextBlob":
"AQICAHgcQFAGtz/hvaxMtJvY/x/rfHnKI3clFPpSXUU7HQRnCAFwfXhXHJAHL8tcVmDqurALAAAAajBoBgkqhki
G9w0BBwagWzBZAgEAMFQGCSqGSIb3DQEHATAeBglghkgBZQMEAS4wEQQM45AIkTqjSekX2mniAgEQgCcOav6Hhol
+wxpWKtXY4y1Z1d0z1P4fx0jTdosfCbPnUExmNJ4zdx8="
```

```
}
$
```

The value of `CiphertextBlob` is the encrypted password. Use this password as the value of the **NGFWv Password** (threat defense virtual password) or the FMC Password for AutoScale Automation (management center password) parameter in the`infrastructure_gwlb.yaml` file. You can also use this password as the value of the **FMC Password for Publishing Metrics to CloudWatch**.

Local Host – Create Target Folder

Use the command given below to create a target folder containing the files that have to be uploaded to the Amazon S3 bucket.

python3 make.py build

This creates a folder named 'target' on your local host. The target folder contains the *zip* files and *yaml* files required for the deployment of the auto scale solution.

Local Host - Upload AWS GWLB Auto Scale Solution Deployment Files to the Amazon S3 Bucket

Use the command given below to upload all the files in the target directory to the Amazon S3 bucket.

$ cd ./target

$ aws s3 cp . s3://demo-us-bkt --recursive

Amazon CloudFormation console - Deploy the Auto Scale Solution for the Threat Defense Virtual using GWLB

Procedure

Step 1 On the AWS Management console, go to **Services** > **Management and Governance** > **CloudFormation** > **Stacks,** and click the stack that was created by the template.

Step 2 Click **Create stack** > **With new resources(standard)**.

Step 3 Select **Upload a template** file, click **Choose** File, and select ***deploy_ngfw_autoscale_with_gwlb.yaml*** from the target folder.

Step 4 Click **Next**.

Step 5 On the **Specify stack details** page, enter a name for the stack.

Step 6 Provide values for the input parameters in the ***deploy_ngfw_autoscale_with_gwlb.yaml*** template.

Stack Name: Threat-Defense-Virtual

Parameter	Values
Pod Configuration	
Autoscale Group Name Prefix	*NGFWv-AutoScale*
Pod Number	*1*
Autoscale Email Notification	*username@cisco.com*
Infrastructure Details	
VPC ID	*vpc-05277f76370396df4*
S3 Bucket Name	*demo-us-bkt*
Subnets for Lambda Functions	*subnet-0f6bbd4de47d50c6b,subnet-0672f4c24156ac443*
Security Groups for Lambda Functions	*sg-023dfadb1e7d4b87e*
Number of Availability Zones	*2*
Availability Zones	*us-west-1a, us-west-1b*
Subnets List for NGFWv Management Interface	*subnet-0e0bc4961de87b170*
Subnets List for NGFWv Inside Interface	*subnet-0f6acf3b548d9e95b*
Subnets List for NGFWv Outside Interface	*subnet-0cc7ac70df7144b7e*
GWLB Configuration	
Enter a port for NGFWv instance health check	*22*
Cisco NGFWv Instance Configuration	
NGFWv Instance type	*C4.xlarge*
NGFWv Instance License type	*BYOL*
Assign Public IP for NGFWv from AWS IP Pool	*true*
Security Groups for NGFWv Instance	*sg-088ae4bc1093f5833*
Security Group for NGFWv Instance inside	*sg-0e0ce5dedcd9cd4f3*
Security Group for NGFWv Instance outside	*sg-07dc50ff47d0c8126*
NGFWv AMI-ID	*ami-00faf58c7ee8d11e1*
KMS Master Key ARN (conditional)	
NGFWv Password	*W1nch3sterBr0s*
FMC Automation Configuration	
FMC host IP address	*3.38.137.49*

Parameter	Values
FMC Username for AutoScale Automation	*autoscaleuser*
FMC Password for AutoScale Automation	*W1nch3sterBr0s*
FMC Device Group Name	*aws-ngfw-autoscale-dg*
Performance Tier value for FMCv licensing	*FTDv20*
FMC Device Group Metrics Publish Configuration	
Publish Custom Metrics from FMC	*TRUE*
FMC Username for Publishing Metrics to CloudWatch	*metricuser*
FMC Password for Publishing Metrics to CloudWatch	*W1nch3sterBr0s*
Scaling Configuration	
Lower,Upper CPU Thresholds	*10,70*
Lower,Upper Memory Thresholds	*40,70*

Step 7 Click **Next** on the **Configure Stack Options** window.

Step 8 On the **Review** page, review and confirm the settings.

Step 9 Click **Create Stack** to deploy the ***deploy_ngfw_autoscale_with_gwlb.yaml*** template and create the stack.

This completes deployment of both the templates that are required to set up the auto scale solution for threat defense virtual using GWLB.

Amazon EC2 console - Edit the Number of Instances in the Auto Scale Group

By default, the Auto Scale group has the minimum and maximum number of threat defense virtual instances set to 0 and 2 respectively. Change these values as per your requirement.

Procedure

Step 1 On the AWS Management console, go to **Services** > **Compute** > **EC2**, and click **Auto Scaling Groups**.

Step 2 Select the auto scaling group created by you and click **Edit** to modify the values in the **Desired capacity**, **Minimum capacity**, **Maximum capacity** fields as per your requirement. These values correspond to the number of threat defense virtual instances that you want to bring up for the auto scaling functionality. Set the **Desired capacity** to a value that is within the minimum and maximum capacity values.

Step 3 Click **Update**.

Note We recommend that you launch only one threat defense virtual instance and verify that the behaviour of this instance is as expected. You can then launch more instances as per your requirement.

Amazon VPC dashboard console - Create the GWLB Endpoint and Configure Routing for the Customer VPC

You have to create the GWLB endpoint and configure routing for the customer VPC after deploying both the CloudFormation templates.

Create the GWLB Endpoint

Procedure

Step 1 On the AWS Management console, go to **Services** > **Networking & Content Delivery** > **VPC** > **Endpoint Services**.

Step 2 Click **Create Endpoint Service**

Step 3 Under **Load balancer** type, choose **Gateway**.

Step 4 Under **Available load balancers**, choose the Gateway Load balancer that was created as part of the Auto scale deployment.

Step 5 Click **Create**.

Step 6 Copy the Service name of the newly created endpoint service.

Step 7 Go to **Services** > **Networking & Content Delivery** > **VPC** > **Endpoints**.

Step 8 Click **Create endpoint**.

Step 9 Under **Service category**, choose **Other endpoint services**.

Step 10 For **Service name**, enter the name of the service, and then choose **Verify service**.

Step 11 In the **VPC** field, select the VPC, *App VPC*, in which to create the endpoint.

Step 12 Under **Subnets**, select the subnet, *Egress subnet*, in which to create the endpoint.

Step 13 For IP address type, choose the IPv4 option to assign IPv4 addresses to the endpoint network interfaces.

Step 14 Click **Create endpoint**.

Step 15 Go to **Services** > **Networking & Content Delivery** > **VPC** > **Endpoint services**, click the **Endpoint Connections** tab, choose the **Endpoint ID** that you created earlier, and click **Actions > Accept endpoint connection request**.

Configure Routing for the Customer VPC

Procedure

Step 1 On the AWS Management console, go to **Services** > **Networking & Content** > **Virtual Private Cloud** > **Route tables**.

Step 2 Create the Ingress Route Table and perform the following steps:

a. Click **Actions** > **Edit routes**.

b. For IPv4, click **Add route**. For **Destination**, enter the IPv4 CIDR block 10.0.1.0/24 of the subnet for the application servers. For **Target**, select the VPC endpoint.

c. Click **Save changes**.

d. In the **Edge Associations** tab, click **Edit edge associations**, and choose **Internet gateway**.

e. Click **Save changes**.

Step 3 Select the route table for the subnet with the application servers and perform the following steps:

a. Click **Actions** > **Edit routes**.

b. For IPv4, click **Add route**. For **Destination**, enter **0.0.0.0/0**. For **Target**, select the VPC endpoint.

c. Click **Save changes**.

Step 4 Select the route table for the subnet with the Gateway Load Balancer endpoint, and perform the following steps:

a. Click **Actions** > **Edit routes**.

b. For IPv4, click **Add route**. For **Destination**, enter **0.0.0.0/0**. For **Target**, select the internet gateway.

c. Click **Save changes**.

Amazon CloudWatch - Validate Deployment

Once the template deployment is successful, go to the Amazon CloudWatch console to ensure that logs are being collected and the required alarms have been created.

Logs

Check the log files to troubleshoot any issues with Management Center connectivity.

Procedure

Step 1 On the **AWS Management** console, go to **Services** > **Management & Governance** > **CloudWatch**.

Step 2 Click **Log groups** and click any log group displayed here to view the logs.

Alarms

Ensure that the required alarms have been created on the Amazon CloudWatch console.

Procedure

Step 1 On the **AWS Management** console, go to **Services** > **Management & Governance** > **CloudWatch**.

Step 2 Click **Alarms** > **All Alarms** to display the list of alarms along with the conditions which will trigger the scale-out and scale-in functions.