Deploy the Threat Defense Virtual Auto Scale Solution on AWS

This document explains how to deploy the threat defense virtual auto scale solution on AWS.

About the Threat Defense Virtual Auto Scale Solution on AWS

The threat defense virtual instances deployed in public cloud environments such as AWS support applications that experience occasional spikes and dips in network traffic. A spike in traffic could lead to a scenario in which the number of deployed threat defense virtual instances is not enough to inspect the network traffic. A dip in traffic could lead to idle threat defense virtual instances leading to unnecessary operational costs.

The auto scale solution helps organisations to automatically scale up the number of threat defense virtual instances if there is a spike in traffic and also scale down the number of instances during a lull in traffic. This leads to efficient handling of network resources and reduces operational costs.

The threat defense virtual auto scale in AWS is a completely serverless implementation (no helper VMs involved in the automation of this feature) that adds auto scaling capability to threat defense virtual instances in the AWS environment.

Starting from version 6.4, Network Load Balancer (NLB)-based auto scale solution is supported on threat defense virtual managed by management center. Starting from version 7.2, Gateway Load Balancer (GWLB)-based auto scale solution is also supported.

Cisco provides CloudFormation templates and scripts for deploying an auto-scaling group of threat defense virtual firewalls using several AWS services, including Lambda, auto scaling groups, Elastic Load Balancing (ELB), Amazon S3 Buckets, SNS, and CloudWatch.

The threat defense virtual auto scale solution is a CloudFormation template-based deployment that provides:

- Completely automated threat defense virtual instance registration and de-registration with the management center.
- NAT policy, access control policy, and routes automatically applied to the scaled-out threat defense virtual instances.
- Support for load balancers and multi-availability zones.
- Works only with the management center; the device manager is not supported.

Enhancements to Auto Scale (Version 6.7)

- Custom metric publisher—A new lambda function polls the management center every second minute for memory consumption of all the threat defense virtual instances in the auto scale group, then publishes the value to the CloudWatch metric.
- A new scaling policy based on memory consumption is available.
- Threat Defense Virtual private IP connectivity for SSH and Secure Tunnel to the management center.
- Management Center configuration validation.
- Support for opening more listening ports on the ELB.
- Modified to single stack deployment. All lambda functions and AWS resources are deployed from a single stack for a streamlined deployment.

Auto Scale Solution with NLB

As the AWS Load Balancer allows only inbound-initiated connections, only externally generated traffic is allowed to pass inside via the Cisco threat defense virtual firewall.

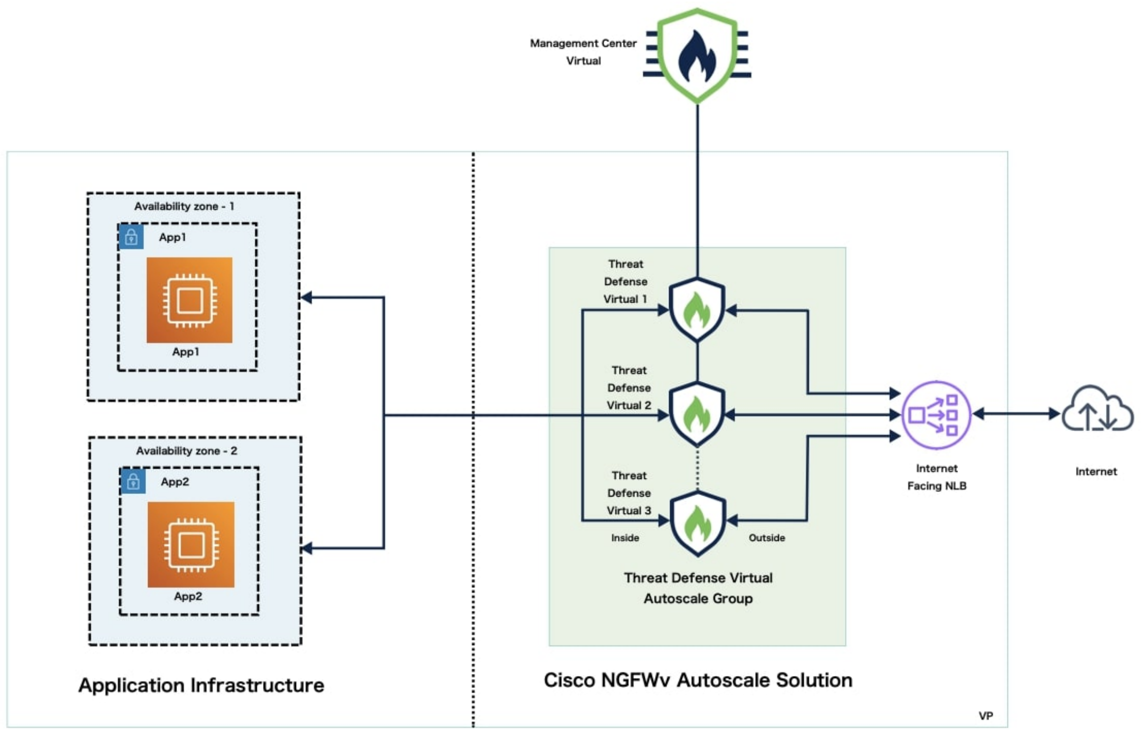
The internet-facing load balancer can be a Network Load Balancer or an Application Load Balancer. All the AWS requirements and conditions hold true for either case. As indicated in the sample topology given below, the right side of the dotted line is deployed via the threat defense virtual templates. The left side is user-defined.

Note Application-initiated outbound traffic will not go through the threat defense virtual.

Port-based bifurcation for traffic is possible. This can be achieved via NAT rules; see Create a Host object, Add Device Group, Auto Scale Solution with NLB - Configure and Deploy Network Address Translation (NAT) Policy, Create a Basic Access Control Policy, on page 30, Create a Basic Access Control Policy in Management Center. For example, traffic on Internet-facing LB DNS, Port: 80 can be routed to Application-1; Port: 88 traffic can be routed to Application-2.

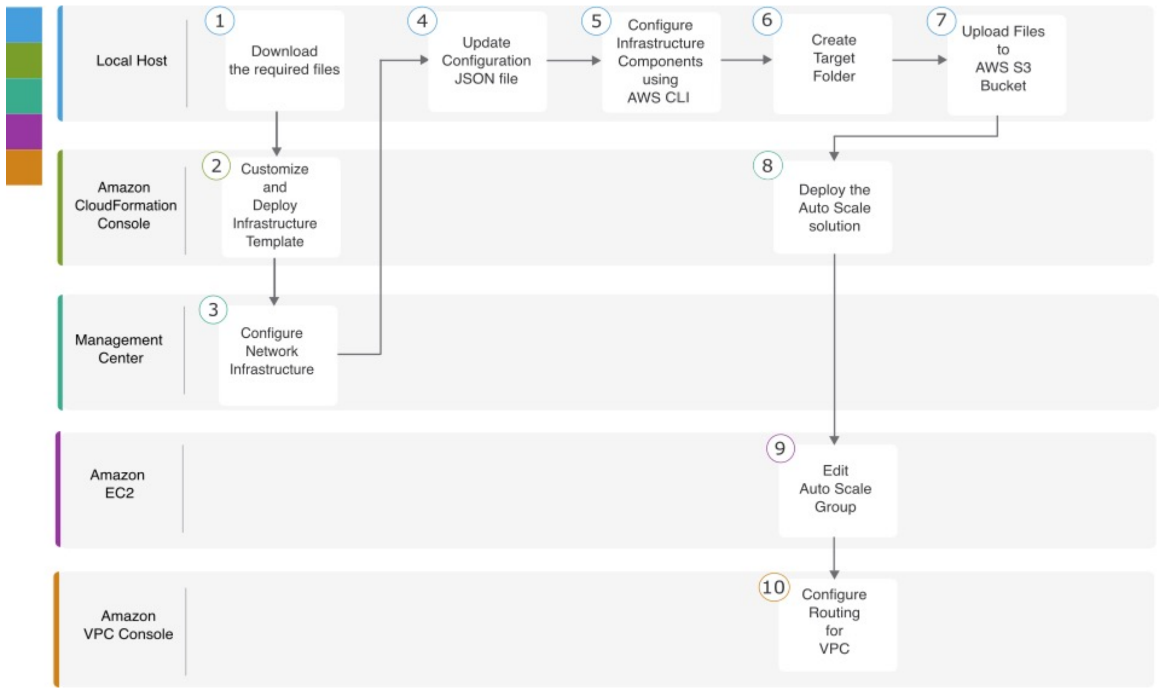
Sample Topology

Figure 1: Threat Defense Virtual Auto Scale Solution with NLB



End-to-End Process for Deploying Auto Scale Solution with NLB

The following flowchart illustrates the workflow for deploying threat defense virtual auto scale solution with NLB on Amazon Web Services (AWS).


Local Host
Amazon CloudFormation Console
Management Center
Amazon EC2
Amazon VPC Console
1 Download the required files
2 Customize and Deploy Infrastructure Template
3 Configure Network Infrastructure
4 Update Configuration JSON file
5 Configure Infrastructure Components using AWS CLI
6 Create Target Folder
7 Upload Files to AWS S3 Bucket
8 Deploy the Auto Scale solution
9 Edit Auto Scale Group
10 Configure Routing for VPC

	Workspace	Steps
1	Local Host	Download the Required Files and CFTs from GitHub to your Local Host
2	Amazon CloudFormation Console	Auto Scale Solution with NLB - Customize and Deploy the NLB Infrastructure Template on the Amazon CloudFormation Console, on page 25
3	Management Center	Configure Network Infrastructure in the Management Center, on page 26
4	Local Host	Update the Configuration.json File, on page 30
5	Local Host	Configure Infrastructure Components using AWS CLI, on page 32
6	Local Host	Create Target Folder, on page 33
7	Local Host	Upload Files to Amazon S3 Bucket, on page 33

	Workspace	Steps
8	Amazon CloudFormation Console	Auto Scale Solution with NLB - Deploy the Auto Scale Solution with NLB, on page 34
9	Amazon EC2 Console	Edit the Auto Scale Group, on page 36
10	Amazon VPC Console	Configure Routing for VPC, on page 35

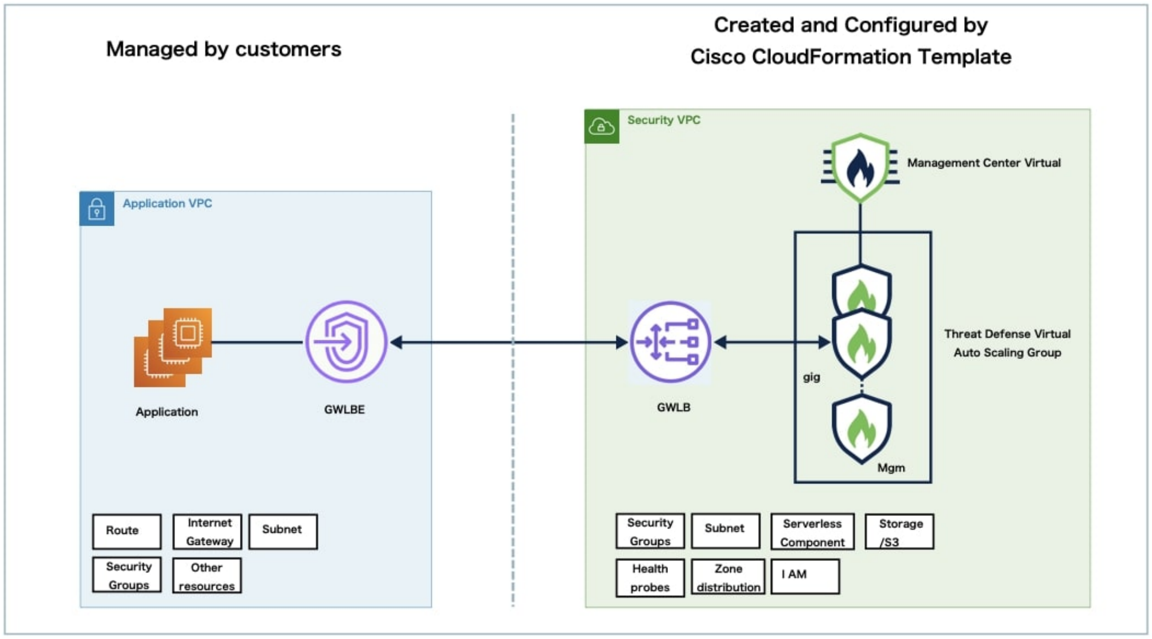
Auto Scale Solution with GWLB

The AWS Gateway Load Balancer (GWLB) allows both inbound and outbound connections, hence both internally and externally generated traffic is allowed to pass inside via the Cisco threat defense virtual firewall.

The GWLBe sends traffic to the GWLB, and then to the threat defense virtual for inspection. All of the AWS requirements and conditions hold true for either case. As indicated in the Use Case diagram, the right side of the dotted line is threat defense virtual GWLB Autoscale solution deployed via the threat defense virtual templates. The left side is completely user-defined.

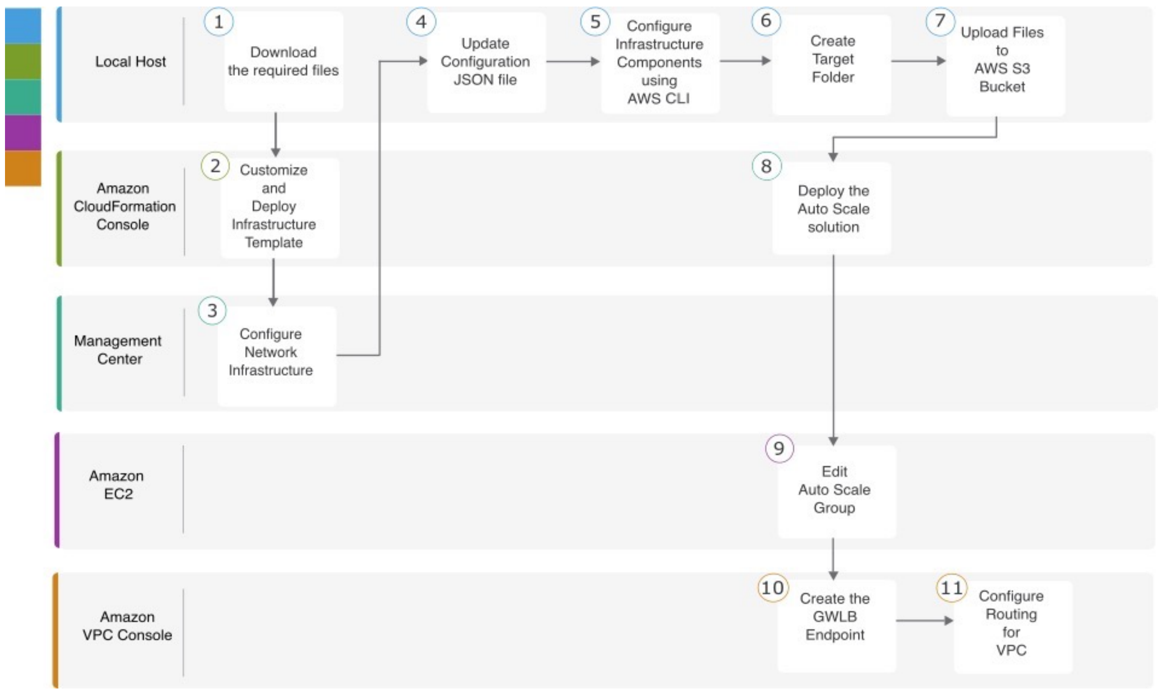
Sample Topology

Figure 2: Threat Defense Virtual Auto Scale Solution with GWLB



End-to-End Process for Deploying Auto scale Solution with GWLB

The following flowchart illustrates the workflow for deploying threat defense virtual auto scale solution with GWLB on Amazon Web Services (AWS).


Local Host
Amazon CloudFormation Console
Management Center
Amazon EC2
Amazon VPC Console
1 Download the required files
2 Customize and Deploy Infrastructure Template
3 Configure Network Infrastructure
4 Update Configuration JSON file
5 Configure Infrastructure Components using AWS CLI
6 Create Target Folder
7 Upload Files to AWS S3 Bucket
8 Deploy the Auto Scale solution
9 Edit Auto Scale Group
10 Create the GWLB Endpoint
11 Configure Routing for VPC

	Workspace	Steps
1	Local Host	Download the Required Files and CFTs from GitHub to your Local Host
2	Amazon CloudFormation Console	Auto Scale Solution with GWLB - Customize and Deploy the GWLB Infrastructure Template on the Amazon CloudFormation Console, on page 25
3	Management Center	Configure Network Infrastructure in the Management Center, on page 26
4	Local Host	Update the Configuration.json File, on page 30
5	Local Host	Configure Infrastructure Components using AWS CLI, on page 32
6	Local Host	Create Target Folder, on page 33
7	Local Host	Upload Files to Amazon S3 Bucket, on page 33

	Workspace	Steps
8	Amazon CloudFormation Console	Auto Scale Solution with GWLB - Deploy the Auto Scale Solution with GWLB, on page 34
9	Amazon EC2 Console	Edit the Auto Scale Group, on page 36
10	Amazon VPC Console	Auto Scale with GWLB solution - Create the GWLB Endpoint, on page 35
11	Amazon VPC Console	Configure Routing for VPC, on page 35

Guidelines and Limitations for the Threat Defense Virtual and AWS

Licensing

- BYOL (Bring Your Own License) using a Cisco Smart License Account is supported.
- PAYG (Pay As You Go) licensing, a usage-based billing model that allows customer to run the threat defense virtual without having to purchase Cisco Smart Licensing. All licensed features (Malware/Threat/URL Filtering/VPN, etc.) are enabled for a registered PAYG threat defense virtual device. Licensed features cannot be edited or modified from the management center. (Version 6.5+)

Note PAYG licensing is not supported on the threat defense virtual devices deployed in the device manager mode.

See the "Licenses" chapter in the Firepower Management Center Administration Guide for guidelines when licensing your threat defense virtual device.

Performance Tiers for Threat Defense Virtual Smart Licensing

Starting from Threat Defense Virtual version 7.0.0 release, the threat defense virtual supports performance-tiered licensing that provides different throughput levels and VPN connection limits based on deployment requirements.

Table 1: Threat Defense Virtual Licensed Feature Limits Based on Entitlement

Performance Tier	Device Specifications (Core/RAM)	Rate Limit	RA VPN Session Limit
FTDv5	4 core/8 GB	100 Mbps	50
FTDv10	4 core/8 GB	1 Gbps	250
FTDv20	4 core/8 GB	3 Gbps	250

Performance Tier	Device Specifications (Core/RAM)	Rate Limit	RA VPN Session Limit
FTDv30	8 core/16 GB	5 Gbps	250
FTDv50	12 core/24 GB	10 Gbps	750
FTDv100	16 core/34 GB	16 Gbps	10,000

Best Practices

- Ensure that you have configured the required components in the management center virtual. See Configure Network Infrastructure in the Management Center for more information.
- Ensure that you enter the required values for the parameters in the CloudFormation templates. See CloudFormation Templates on GitHub for more information.

Prerequisites

- An AWS account. You can create one at http://aws.amazon.com/.
- An SSH client (for example, PuTTY on Windows or Terminal on macOS) is required to access the threat defense virtual console.
- A Cisco Smart Account. You can create one at Cisco Software Central https://software.cisco.com/.
- A GitHub account to download the configuration files and templates.
- Threat Defense Virtual interface requirements:
  - Management interfaces (2)— One used to connect the threat defense virtual to the management center, second used for diagnostics; cannot be used for through traffic.
  - You can optionally configure a data interface for the management center management instead of the Management interface. The Management interface is a pre-requisite for data interface management, so you still need to configure it in your initial setup. Note that management center access from a data interface is not supported in High Availability deployments. For more information about configuring a data interface for the management center access, see the configure network management-data-interface command in the FTD command reference.
  - Traffic interfaces (2) — Used to connect the threat defense virtual to inside hosts and to the public network.
- Communication Paths — Public/elastic IPs for access to the threat defense virtual.

Components Required to Set Up the Auto Scale Solution with GWLB or NLB

The following components make up the auto scale solution.

CloudFormation Template

The CloudFormation template is used to deploy resources required to set up the auto scale solution in AWS. The template consists of:

- Auto Scale Group, Load Balancer, Security Groups, and other miscellaneous components.
- The template takes user input to customize the deployment.

Note The template has limitations in validating user input, hence it is the user's responsibility to validate input during deployment.

Lambda Functions

The auto scale solution is a set of Lambda functions developed in Python, which gets triggered from Lifecycle hooks, SNS, CloudWatch event/alarm events. The basic functionality includes:

- Add/Remove Diag, Gig0/0, and Gig 0/1 interfaces to instance.
- Register Gig0/1 interface to Load Balancer's Target Groups.
- Register a new threat defense virtual with the management center.
- Configure and deploy a new threat defense virtual via management center.
- Unregister (remove) a scaled-in threat defense virtual from the management center.
- Publish the memory metric from the management center.

Lambda Functions are delivered to customer in the form of a Python package.

Lifecycle Hooks

- Lifecycle hooks are used to get lifecycle change notification about an instance.
- In the case of instance launch, a Lifecycle hook is used to trigger a Lambda function which can add interfaces to the threat defense virtual instance, and register outside interface IPs to target groups.
- In the case of instance termination, a Lifecycle hook is used to trigger a Lambda function to deregister the threat defense virtual instance from the target group.

Simple Notification Service (SNS)

- Simple Notification Service (SNS) from AWS is used to generate events.
- Due to the limitation that there is no suitable orchestrator for Serverless Lambda functions in AWS, the solution uses SNS as a kind of function chaining to orchestrate Lambda functions based on events.

VPC

You should create the VPC as required for your application requirements. It is expected that the VPC have an internet gateway with at least one subnet attached with a route to the internet. Refer to the appropriate sections for the requirements for Security Groups, Subnets, etc.

Security Groups

All connections are allowed in the provided Auto Scale Group template. You only need the following connections for the auto scale solution to work.

Port	Usage	Subnet
8305	Management Center to Threat Defense Virtual Secured tunnel connection	Management subnets
Health Probe port (default: 8080)	Internet-facing Load Balancer health probes	Outside, Inside Subnets
Application ports	Application data traffic	Outside, Inside Subnets

Security Groups or ACLs for the Management Center Instance

These are needed to allow HTTPS connections between lambda functions and the management center. As lambda functions are to be kept in lambda subnets having a NAT gateway as the default route, the management center should be allowed to have inbound HTTPS connections from the NAT gateway IP address.

Subnets

Subnets can be created as needed for the requirements of the application. The threat defense virtual requires 3 subnets for operation.

Note If multiple availability zone support is needed, then subnets are needed for each zone as subnets are zonal properties within the AWS Cloud.

Outside Subnet

The Outside subnet should have a default route with '0.0.0.0/0' to the Internet gateway. This will contain the Outside interface of the threat defense virtual, and also the Internet-facing NLB will be in this subnet.

Inside Subnet

This can be similar to the application subnets, with or without NAT/Internet gateway. Please note that for the threat defense virtual health probes, it should be possible to reach the AWS Metadata Server (169.254.169.254) via port 80.

Note In this Auto Scale solution, Load Balancer health probes are redirected to the AWS Metadata Server via inside/Gig0/0 interface. However, you can change this with your own application serving the health probe connections sent to the threat defense virtual from the Load Balancer. In that case, you need to replace the AWS Metadata Server object with the application IP address to provide the health probe response.

Management Subnet

This subnet includes the threat defense virtual management interface. If you are using the management center on this subnet, then assigning an elastic IP address (EIP) to the threat defense virtual is optional. The diagnostic interface is also on this subnet.

Lambda Subnets

The AWS Lambda function requires two subnets having the NAT gateway as the default gateway. This makes the Lambda function private to the VPC. Lambda subnets do not need to be as wide as other subnets.

Application Subnets

There is no restriction imposed on this subnet from the auto scale solution, but if the application needs outbound connections outside the VPC, the respective routes should be configured on the subnet. This is because outbound-initiated traffic does not pass through load balancers.For more information, see AWS Elastic Load Balancing User Guide.

Serverless Components

S3 Bucket

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. You can place all the required files in the S3 bucket.

When templates are deployed, Lambda functions get created referencing zip files in the S3 bucket. Hence, the S3 bucket should be accessible to the user account.

CloudFormation Templates on GitHub

There are two sets of templates provided for the supported auto scale solutions – one set for setting up the auto scale solution using an NLB and another set for setting up the auto scale solution using a GWLB.

Auto Scale Solution with NLB

The following templates are available on GitHub:

- infrastructure.yaml
- deploy_ngfw_autoscale.yaml

Table 2: List of Template Parameters

Parameter	Allowed Values/Type	Description
PodNumber	String Allowed Pattern: '^\d{1,3}$'	This is the pod number. This will be suffixed to the Auto Scale Group name (threat defense virtual-Group-Name). For example, if this value is '1', then the group name will be *threat defense virtual-Group-Name-1*. It should be at least 1 numerical digit but not more than 3 digits. Default: 1.
AutoscaleGrpNamePrefix	String	This is the Auto Scale Group Name Prefix. The pod number will be added as a suffix. Maximum: 18 characters Example: Cisco-threat defense virtual-1.

Parameter	Allowed Values/Type	Description
NotifyEmailID	String	Auto Scale events will be sent to this email address. You need to accept a subscription email request. Example: admin@company.com.
VpcId	String	The VPC ID in which the device needs to be deployed. This should be configured as per AWS requirements. Type: AWS::EC2::VPC::Id If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.
LambdaSubnets	List	The subnets where Lambda functions will be deployed. Type: List<AWS::EC2::Subnet::Id> If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.
LambdaSG	List	The Security Groups for Lambda functions. Type: List<AWS::EC2::SecurityGroup::Id> If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.
S3BktName	String	The S3 bucket name for files. This should be configured in your account as per AWS requirements. If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.
LoadBalancerType	String	The type of Internet-facing Load Balancer, either “application” or “network”. Example: application
LoadBalancerSG	String	The Security Groups for the Load Balancer. In the case of a network load balancer, it won't be used. But you should provide a Security Group ID. Type: List<AWS::EC2::SecurityGroup::Id> If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.

Parameter	Allowed Values/Type	Description
LoadBalancerPort	Integer	The Load Balancer port. This port will be opened on LB with either HTTP/HTTPS or TCP/TLS as the protocol, based on the chosen Load Balancer type. Make sure the port is a valid TCP port, it will be used to create the Load Balancer listener. Default: 80
SSLcertificate	String	The ARN for the SSL certificate for secured port connections. If not specified, a port opened on the Load Balancer will be TCP/HTTP. If specified, a port opened on the Load Balancer will be TLS/HTTPS.
TgHealthPort	Integer	This port is used by the Target group for health probes. Health probes arriving at this port on the threat defense virtual will be routed to the AWS Metadata server and should not be used for traffic. It should be a valid TCP port. If you want your application itself to reply to health probes, then accordingly NAT rules can be changed for the threat defense virtual. In such a case, if the application does not respond, the threat defense virtual will be marked as unhealthy and deleted due to the Unhealthy instance threshold alarm. Example: 8080
AssignPublicIP	Boolean	If selected as "true" then a public IP will be assigned. In case of a BYOL-type threat defense virtual, this is required to connect to https://tools.cisco.com. Example: TRUE
InstanceType	String	The Amazon Machine Image (AMI) supports different instance types, which determine the size of the instance and the required amount of memory. Only AMI instance types that support the threat defense virtual should be used. Example: c4.2xlarge
LicenseType	String	The threat defense virtual license type, either BYOL or PAYG. Make sure the related AMI ID is of the same licensing type. Example: BYOL

Parameter	Allowed Values/Type	Description
AmiId	String	The threat defense virtual AMI ID (a valid Cisco threat defense virtual AMI ID). Type: AWS::EC2::Image::Id Please choose the correct AMI ID as per the region and desired version of the image. The Auto Scale feature supports version 6.4+, BYOL/PAYG images. In either case you should have accepted a License in the AWS marketplace. In the case of BYOL, please update 'licenseCaps' key in Configuration JSON with features such as 'BASE', 'MALWARE', 'THREAT', 'URLFilter' etc.
NoOfAZs	Integer	The number of availability zones that the threat defense virtual should span across, between 1 and 3. In the case of an ALB deployment, the minimum value is 2, as required by AWS. Example: 2
ListOfAzs	Comma separated string	A comma-separated list of zones in order. **Note** The order in which these are listed matters. Subnet lists should be given in the same order. If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value. Example: us-east-1a, us-east-1b, us-east-1c
MgmtInterfaceSG	String	The Security Group for the threat defense virtual Management interface. Type: List<AWS::EC2::SecurityGroup::Id> If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.
InsideInterfaceSG	String	The Security Group for the threat defense virtual inside interface. Type: AWS::EC2::SecurityGroup::Id If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.

Parameter	Allowed Values/Type	Description
OutsideInterfaceSG	String	The Security Group for the threat defense virtual outside interface. Type: AWS::EC2::SecurityGroup::Id If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value. Example: sg-0c190a824b22d52bb
MgmtSubnetId	Comma separated list	A comma-separated list of management subnet-ids. The list should be in the same order as the corresponding availability zones. Type: List<AWS::EC2::SecurityGroup::Id> If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.
InsideSubnetId	Comma separated list	A comma-separated list of inside/Gig0/0 subnet-ids. The list should be in the same order as the corresponding availability zones. Type: List<AWS::EC2::SecurityGroup::Id> If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.
OutsideSubnetId	Comma separated list	A comma-separated list of outside/Gig0/1 subnet-ids. The list should be in the same order as the corresponding availability zones. Type: List<AWS::EC2::SecurityGroup::Id> If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.
KmsArn	String	The ARN of an existing KMS (AWS KMS key to encrypt at rest). If specified, the management center and threat defense virtual passwords should be encrypted. The password encryption should be done using only the specified ARN. Generating Encrypted Password Example: " aws kms encrypt --key-id <KMS ARN> --plaintext <password> ". Please used such generated passwords as shown. Example: arn:aws:kms:us-east-1:[AWS Account]:key/7d586a25-5875-43b1-bb68-a452e2f6468e

Parameter	Allowed Values/Type	Description
ngfwPassword	String	All the threat defense virtual instances come up with a default password, which is entered in the *Userdata* field of the Launch Template (Autoscale Group). This input will change the password to new provided password once the threat defense virtual is accessible. Please use a plain text password if KMS ARN is not used. If KMS ARN is used, then an encrypted password should be used. Example: Cisco123789! or AQIAgcQFAGtz/hvaxMtJvY/x/rfHnI3lPpSXU
fmcServer	Numeric string	The IP address of managing the management center, which is reachable to both Lambda functions and the threat defense virtual management interface. Example: 10.10.17.21
fmcOperationsUsername	String	The Network-Admin or higher privileged user created in managing the management center. See the information about creating users and roles in the Cisco Secure Firewall Management Center Device Configuration Guide. Example: apiuser-1
fmcOperationsPassword	String	Please use a plain text password if KMS ARN is not mentioned. If mentioned, then an encrypted password should be used. Example: Cisco123@ or AQICAHgcQAtz/hvaxMtJvY/x/mKI3clFPpSXUHQRnCAajB
fmcDeviceGrpName	String	The management center device group name. Example: AWS-Cisco-NGFW-VMs-1
fmcPerformanceLicenseTier	String	The performance tier license used while registering the threat defense virtual device on the management center virtual. Allowed values: FTDv/FTDv5/FTDv10/FTDv20/FTDv30/FTDv50/FTDv100

Parameter	Allowed Values/Type	Description
fmcPublishMetrics	Boolean	If set to "TRUE", then a Lambda function will be created which runs once in every 2 minutes to fetch the memory consumption of registered threat defense virtual sensors in the provided device group. Allowed values: TRUE, FALSE Example: TRUE
fmcMetricsUsername	String	The unique management center user name for metric publication to AWS CloudWatch. See the information about creating users and roles in the Cisco Secure Firewall Management Center Device Configuration Guide. If the "fmcPublishMetrics' is set to "FALSE" then there is no need to provide this input. Example: publisher-1
fmcMetricsPassword	String	The management center password for metric publication to AWS CloudWatch. Please use a plain text password if KMS ARN is not mentioned. If mentioned, then an encrypted password should be used. If the "fmcPublishMetrics' is set to "FALSE" then there is no need to provide this input. Example: Cisco123789!
CpuThresholds	Comma separated integers	The lower CPU threshold and the upper CPU threshold. The minimum value is 0 and maximum value is 99. Defaults: 10, 70 Please note that the lower threshold should be less than the upper threshold. Example: 30,70
MemoryThresholds	Comma separated integers	The lower MEM threshold and the upper MEM threshold. The minimum value is 0 and maximum value is 99. Defaults: 40, 70 Please note that the lower threshold should be less than the upper threshold. If the "fmcPublishMetrics" parameter is "FALSE" then this has no effect. Example: 40,50

Auto Scale Solution with GWLB

Templates available on GitHub

- infrastructure_gwlb.yaml
- deploy_ngfw_autoscale_with_gwlb.yaml

Table 3: List of Template Parameters

Parameter	Allowed Values/Type	Description
PodNumber	String Allowed Pattern: '^\d{1,3}$'	This is the pod number. This will be suffixed to the Auto Scale Group name (threat defense virtual-Group-Name). For example, if this value is '1', then the group name will be *threat defense virtual-Group-Name-1*. It should be at least 1 numerical digit but not more than 3 digits. Default: 1
AutoscaleGrpNamePrefix	String	This is the Auto Scale Group Name Prefix. The pod number will be added as a suffix. Maximum: 18 characters Example: Cisco-threat defense virtual-1
NotifyEmailID	String	Auto Scale events will be sent to this email address. You need to accept a subscription email request. Example: admin@company.com
VpcId	String	The VPC ID in which the device needs to be deployed. This should be configured as per AWS requirements. Type: AWS::EC2::VPC::Id If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.
LambdaSubnets	List	The subnets where Lambda functions will be deployed. Type: List<AWS::EC2::Subnet::Id> If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.
LambdaSG	List	The Security Groups for Lambda functions. Type: List<AWS::EC2::SecurityGroup::Id> If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.

Parameter	Allowed Values/Type	Description
S3BktName	String	The S3 bucket name for files. This should be configured in your account as per AWS requirements. If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.
LoadBalancerType	String	The type of Internet-facing Load Balancer, either “application” or “network”. Example: application
LoadBalancerSG	String	The Security Groups for the Load Balancer. In the case of a network load balancer, it won't be used. But you should provide a Security Group ID. Type: List<AWS::EC2::SecurityGroup::Id> If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.
LoadBalancerPort	Integer	The Load Balancer port. This port will be opened on LB with either HTTP/HTTPS or TCP/TLS as the protocol, based on the chosen Load Balancer type. Make sure the port is a valid TCP port, it will be used to create the Load Balancer listener. Default: 80
SSLcertificate	String	The ARN for the SSL certificate for secured port connections. If not specified, a port opened on the Load Balancer will be TCP/HTTP. If specified, a port opened on the Load Balancer will be TLS/HTTPS.
TgHealthPort	Integer	This port is used by the Target group for health probes. Health probes arriving at this port on the threat defense virtual will be routed to the AWS Metadata server and should not be used for traffic. It should be a valid TCP port. If you want your application itself to reply to health probes, then accordingly NAT rules can be changed for the threat defense virtual. In such a case, if the application does not respond, the threat defense virtual will be marked as unhealthy and deleted due to the Unhealthy instance threshold alarm. Example: 8080

Parameter	Allowed Values/Type	Description
AssignPublicIP	Boolean	If selected as "true" then a public IP will be assigned. In case of a BYOL-type threat defense virtual, this is required to connect to https://tools.cisco.com. Example: TRUE
InstanceType	String	The Amazon Machine Image (AMI) supports different instance types, which determine the size of the instance and the required amount of memory. Only AMI instance types that support the threat defense virtual should be used. Example: c4.2xlarge
LicenseType	String	The threat defense virtual license type, either BYOL or PAYG. Make sure the related AMI ID is of the same licensing type. Example: BYOL
AmiId	String	The threat defense virtual AMI ID (a valid Cisco threat defense virtual AMI ID). Type: AWS::EC2::Image::Id Please choose the correct AMI ID as per the region and desired version of the image. The Auto Scale feature supports version 6.4+, BYOL/PAYG images. In either case you should have accepted a License in the AWS marketplace. In the case of BYOL, please update 'licenseCaps' key in Configuration JSON with features such as 'BASE', 'MALWARE', 'THREAT', 'URLFilter' etc.
NoOfAZs	Integer	The number of availability zones that the threat defense virtual should span across, between 1 and 3. In the case of an ALB deployment, the minimum value is 2, as required by AWS. Example: 2
ListOfAzs	Comma separated string	A comma-separated list of zones in order. **Note** The order in which these are listed matters. Subnet lists should be given in the same order. If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value. Example: us-east-1a, us-east-1b, us-east-1c

Parameter	Allowed Values/Type	Description
MgmtInterfaceSG	String	The Security Group for the threat defense virtual Management interface. Type: List<AWS::EC2::SecurityGroup::Id> If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.
InsideInterfaceSG	String	The Security Group for the threat defense virtual inside interface. Type: AWS::EC2::SecurityGroup::Id If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.
OutsideInterfaceSG	String	The Security Group for the threat defense virtual outside interface. Type: AWS::EC2::SecurityGroup::Id If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value. Example: sg-0c190a824b22d52bb
MgmtSubnetId	Comma separated list	A comma-separated list of management subnet-ids. The list should be in the same order as the corresponding availability zones. Type: List<AWS::EC2::SecurityGroup::Id> If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.
InsideSubnetId	Comma separated list	A comma-separated list of inside/Gig0/0 subnet-ids. The list should be in the same order as the corresponding availability zones. Type: List<AWS::EC2::SecurityGroup::Id> If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.

Parameter	Allowed Values/Type	Description
OutsideSubnetId	Comma separated list	A comma-separated list of outside/Gig0/1 subnet-ids. The list should be in the same order as the corresponding availability zones. Type: List<AWS::EC2::SecurityGroup::Id> If the "*infrastructure.yaml*" file is used to deploy the infrastructure, the output section of the stack will have this value. Please use that value.
KmsArn	String	The ARN of an existing KMS (AWS KMS key to encrypt at rest). If specified, the management center and threat defense virtual passwords should be encrypted. The password encryption should be done using only the specified ARN. Generating Encrypted Password Example: " aws kms encrypt --key-id <KMS ARN> --plaintext <password> ". Please used such generated passwords as shown. Example: arn:aws:kms:us-east-1:[AWS Account]:key/7d586a25-5875-43b1-bb68-a452e2f6468e
ngfwPassword	String	All the threat defense virtual instances come up with a default password, which is entered in the *Userdata* field of the Launch Template (Autoscale Group). This input will change the password to new provided password once the threat defense virtual is accessible. Please use a plain text password if KMS ARN is not used. If KMS ARN is used, then an encrypted password should be used. Example: Cisco123789! or AQIAgcQFAGtz/hvaxMtJvY/x/rfHnI3lPpSXU
fmcServer	Numeric string	The IP address of managing the management center, which is reachable to both Lambda functions and the threat defense virtual management interface. Example: 10.10.17.21
fmcOperationsUsername	String	The Network-Admin or higher privileged user created in managing the management center. See the information about creating users and roles in the Cisco Secure Firewall Management Center Device Configuration Guide. Example: apiuser-1

Parameter	Allowed Values/Type	Description
fmcOperationsPassword	String	Please use a plain text password if KMS ARN is not mentioned. If mentioned, then an encrypted password should be used. Example: Cisco123@ or AQICAHgcQAtz/hvaxMtJvY/x/rnKI3clFPpSXUHQRnCAajB
fmcDeviceGrpName	String	The management center device group name. Example: AWS-Cisco-NGFW-VMs-1
fmcPerformanceLicenseTier	String	The performance tier license used while registering the threat defense virtual device on the management center virtual. Allowed values: FTDv/FTDv20/FTDv30/FTDv50/FTDv100 **Note** FTDv5 and FTDv10 performance tier licenses are not supported with AWS Gateway Load Balancer.
fmcPublishMetrics	Boolean	If set to "TRUE", then a Lambda function will be created which runs once in every 2 minutes to fetch the memory consumption of registered threat defense virtual sensors in the provided device group. Allowed values: TRUE, FALSE Example: TRUE
fmcMetricsUsername	String	The unique management center user name for metric publication to AWS CloudWatch. See the information about creating users and roles in the Cisco Secure Firewall Management Center Device Configuration Guide. If the "fmcPublishMetrics' is set to "FALSE" then there is no need to provide this input. Example: publisher-1
fmcMetricsPassword	String	The management center password for metric publication to AWS CloudWatch. Please use a plain text password if KMS ARN is not mentioned. If mentioned, then an encrypted password should be used. If the "fmcPublishMetrics' is set to "FALSE" then there is no need to provide this input. Example: Cisco123789!

Parameter	Allowed Values/Type	Description
CpuThresholds	Comma separated integers	The lower CPU threshold and the upper CPU threshold. The minimum value is 0 and maximum value is 99. Defaults: 10, 70 Please note that the lower threshold should be less than the upper threshold. Example: 30,70
MemoryThresholds	Comma separated integers	The lower MEM threshold and the upper MEM threshold. The minimum value is 0 and maximum value is 99. Defaults: 40, 70 Please note that the lower threshold should be less than the upper threshold. If the "fmcPublishMetrics" parameter is "FALSE" then this has no effect. Example: 40,50

Download the Required Files and CFTs from GitHub to your Local Host

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:

- Templates for the Auto Scale solution with NLB-
  - **infrastructure.yaml –** Used to customize the components in the AWS environment.
  - **deploy_ngfw_autoscale.yaml –** Used to deploy the AWS Auto Scale with NLB solution.
- Templates for the Auto Scale solution with GWLB-
  - **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.

Note 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.

Auto Scale Solution with NLB - Customize and Deploy the NLB Infrastructure Template on the Amazon CloudFormation Console

Perform the steps given in this section if you are deploying the auto scale solution with NLB.

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.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.yaml*** template.

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.yaml** template and create the stack.

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

Auto Scale Solution with GWLB - Customize and Deploy the GWLB Infrastructure Template on the Amazon CloudFormation Console

Perform the steps given in this section if you are deploying the auto scale solution using GWLB.

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.

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**.

Configure Network Infrastructure in the Management Center

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

You can manage the threat defense virtual using the management center, a full-featured, multidevice manager on a separate server. The threat defense virtual registers and communicates with the management center on the Management interface that you allocated to the threat defense virtual virtual machine.

See About Secure Firewall Threat Defense Virtual with the Secure Firewall Management Center for more information.

All the objects used for the threat defense virtual configuration should be created by user.

Important A device group should be created and rules should be applied on it. All the configurations applied on the device group will be pushed to the threat defense virtual instances.

Add Device Group

The management centre 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.

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**.

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.

Create a Host object

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**.

Step 6 Enter a Description.

Step 7 In the **Network** field, select the **Host** option and enter the following values.

a) Name of the object type as **aws-metadata-server**.
b) Depending on the type of host protocol, enter the following IP address for IPv4 - **169.254.169.254**.

Step 8 Click **Save**.

Create a Port object

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**.

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**, or choose an ICMP **Type** and **Code**.

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

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** - *Routed*.

Step 6 Click **Save**.

Enable Port for Health Check Probe

You can enable port 22 (SSH) or port 443 (HTTP) for the health check probe.

Enable Port 22 (SSH) for Health Check Probe

If you are using port 22 (SSH) for the health check probe, perform the following procedure to enable the port for the health check probe.

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

Step 2 Click **+ Add**.

Step 3 Select the relevant **IP Address** from the drop-down list.

Step 4 From the **Available Zones/Interfaces** window, select the outside interface that is connected to the GWLB or the outside subnet.

Step 5 Click **Add** to add that interface to the **Selected Zones/Interfaces** window.

Step 6 Click **OK**.

Step 7 Click **Save**.

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.

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**.

Auto Scale Solution with NLB - Configure and Deploy Network Address Translation (NAT) Policy

A typical NAT rule converts internal addresses to a port on the outside interface IP address. This type of NAT rule is called interface Port Address Translation (PAT). See Configure NAT in Managing the Secure Firewall Threat Defense Virtual with the Secure Firewall Management Center for information about the NAT policy.

One mandatory rule is required in your NAT policy. An example of a NAT rule is given below:

- Source Zone: Outside Zone
- Dest Zone: Inside Zone
- Original-sources: any-ipv4
- Original source port: Original/default
- Original Destinations: Interface

- Original-destination-port: 8080/or any health port that user configures
- Translated-sources: any-ipv4
- Translated source port: Original/default
- Translated-destination: aws-metadata-server
- Translated-destination-port: 80/HTTP

Similarly, any data-traffic NAT rules can be added, so that this configuration will be pushed to the threat defense virtual devices.

Important NAT Policy created should be applied on the device group. The management center validation from the Lambda function verifies this.

Step 1 Log in to Secure Firewall Management Center.

Step 2 On the **Devices** menu, click **NAT**.

Step 3 Click **New Policy** > **Threat Defense NAT** to create a new policy.

Step 4 Enter the Name and Description for the NAT policy.

Step 5 Click **Save**.

You can see a new policy is added and listed on the NAT page.

Step 6 Click **Add Rule**.

Step 7 Select the **Manual NAT Rule** from the **NAT Rule** drop-down list.

Step 8 Select **In Category** and **NAT Rule** Before from the Insert drop-down list.

Step 9 Select **Static** from the **Type** drop-down menu.

Step 10 Enter the description.

Step 11 In the **Interface Objects** menu, add the source and destination objects.

Step 12 In the **Translations** menu, add the following values for each parameter.

Parameter	Values
Original Source	any-ipv4
Original Destination	Address
Original Source Port	HTTP
Original Destination Port	8080
Translated Source	any-ipv4
Translated Source Port	Original/default
Translated Destination	aws-metadata-server
Translated Destination Port	80/HTTP

Step 13 Click **Save** to save and add the Rule.

Step 14 Select the new rule you have created to deploy on the threat defense virtual.

Step 15 Click **Deploy** > **Deployment** to deploy the policy to assigned devices. The changes are not active until you deploy them.

Create a Basic Access Control Policy

Configure access control to allow traffic from inside to outside. An Access Policy with all required policies can be created, health port object should be allowed such that traffic on this port is allowed to reach. See Configure Access Control in Managing the Secure Firewall Threat Defense Virtual with the Secure Firewall Management Center for information about the Access 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.

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

Step 2 Click **New Policy**.

Step 3 Enter a unique Name 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**.

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. Please note that the JSON key should not be changed.

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-asg-policy",  //Access policy name configured in the Management
 center virtual
  "fmcNatPolicyName": "AWS-Cisco-NGFW-VMs",  //NAT Policy name configured in the Management
 center virtual (Not required for GWLB-based deployment)
  "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
  "MetadataServerObjectName": "aws-metadata-server", //Host object name created for the IP
 169.254.169.254 in the Management center virtual  (Not required for GWLB-based deployment)

  "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
}
```

You can configure static routes for the threat defense virtual by modifying the **trafficRoutes** parameter in this file. An example of a static route configuration is given below.

```
{
      "interface": "inside",
      "network": "any-ipv4",
      "gateway": "",
      "metric": "1"
 }
```

Configure Infrastructure Components using AWS CLI

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.

Create the Lambda Layer Zip File to Manage Compute Resources

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

Step 1 Create a python folder in your Linux host, such as Ubuntu 22.04.

Step 2 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
```

Step 3 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
```

Step 4 After creating the **autoscale_layer.zip** file, copy the **autoscale_layer.zip** file to the **lambda-python-files** folder that is downloaded from GitHub.

(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/rfHnKI3clFPpSXUU7HQRnCAFwfXhXH
JAHL8tcVmDqurALAAAAajBoBgkqhki
G9w0BBwagWzBZAgEAMFQGCSqGSIb3DQEHATAeBglghkgBZQMEAS4wEQQM45
AIkTqjSekX2mniAgEQgCcOav6Hhol
+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**.

Create Target Folder

On the local host, 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 deployment of the auto scale solution.

Upload Files to Amazon S3 Bucket

On the local host, use the commands given below to upload all the files in the target directory to the Amazon S3 bucket.

$ cd ./target

$ aws s3 cp . s3://*<bucket-name>* **--recursive**

Auto Scale Solution with NLB - Deploy the Auto Scale Solution with NLB

Perform the steps given in this section if you are deploying the auto scale solution with NLB.

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.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.yaml*** template.

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.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 with NLB.

Auto Scale Solution with GWLB - Deploy the Auto Scale Solution with GWLB

Perform the steps given in this section if you are deploying the auto scale solution with GWLB.

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.

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.

Auto Scale with GWLB solution - Create the GWLB Endpoint

Perform the steps given in this section if you are deploying the auto scale solution using GWLB.

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 Under **Require acceptance for endpoint,** choose **Acceptance required**. This ensures that you have to manually accept any endpoint service connection requests.

Step 6 Under **Supported IP address types**, choose **IPv4**.

Step 7 Click **Create**.

Step 8 Copy the Service name of the newly created endpoint service.

Step 9 Go to **Services** > **Networking & Content Delivery** > **VPC** > **Endpoints**.

Step 10 Click **Create endpoint**.

Step 11 Under **Service category**, choose **Other endpoint services**.

Step 12 For **Service name**, enter the name of the service, and then choose **Verify service**.

Step 13 In the **VPC** field, select the VPC in which to create the endpoint.

Step 14 Under **Subnets**, select the subnet in which to create the endpoint.

Step 15 For IP address type, choose the IPv4 option to assign IPv4 addresses to the endpoint network interfaces.

Step 16 Click **Create endpoint**.

Configure Routing for VPC

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

Step 2 Select the route table for the internet gateway and perform the following steps:

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

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

c. 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**.

Edit 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.

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.

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.

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.

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.

Maintenance Tasks

Scaling Processes

This topic explains how to suspend and then resume one or more of the scaling processes for your Auto Scale group.

Start and Stop Scale Actions

To start and stop scale out/in actions, follow these steps.

- For AWS Dynamic Scaling—Refer to the following link for information to enable or disable scale out actions:

  Suspending and Resuming Scaling Processes

Health Monitor

Every 60 minutes, a CloudWatch Cron job triggers the Auto Scale Manager Lambda for the Health Doctor module:

- If there are unhealthy IPs which belong to a valid threat defense virtual VM, that instance gets deleted if the threat defense virtual is more than an hour old.
- If those IPs are not from a valid threat defense virtual machine, then only IPs are removed from the Target Group.

The health monitor also validates the management center configuration for device group, access policy, and NAT rules. In case of an unhealthy IP/instance, or if the management center validation fails, the health monitor sends an email to the user.

Disable Health Monitor

To disable a health monitor, in ***constant.py*** make the constant as "True".

Enable Health Monitor

To enable a health monitor, in ***constant.py*** make the constant as "False".

Disable Lifecycle Hooks

In the unlikely event that Lifecycle hook needs to be disabled, if disabled it won't add additional interfaces to Instances. It can also cause a series of failed deployment of the threat defense virtual instances.

Disable Auto Scale Manager

To disable Auto Scale Manager, respective CloudWatch Events "notify-instance-launch" and "notify-instance-terminate" should be disabled. Disabling this won't trigger Lambda for any new events. But already executing Lambda actions will continue. There is no abrupt stop of Auto Scale Manager. Trying abrupt stopping by stack deletion or deleting resources can cause an indefinite state.

Load Balancer Targets

Because the AWS Load Balancer does not allow instance-type targets for instances having more than one network interface, the Gigabit0/1 interface IP is configured as a target on Target Groups. As of now however, the AWS Auto Scale health checks work only for instance-type targets, not IPs. Also, these IPs are not automatically added or removed from target groups. Hence our Auto Scale solution programmatically handles both of these tasks. But in the case of maintenance or troubleshooting, there could be a situation demanding manual effort to do so.

Register a Target to a Target Group

To register the threat defense virtual instance to the Load Balancer, its Gigabit0/1 instance IP (outside subnet) should be added as a target in Target Group(s). See Register or Deregister Targets by IP Address.

Deregister a Target from a Target Group

To deregister the threat defense virtual instance to the Load Balancer, its Gigabit0/1 instance IP (outside subnet) should be deleted as a target in Target Group(s). See Register or Deregister Targets by IP Address.

Instance Stand-by

AWS does not allow instance reboot in the Auto Scale group, but it does allow a user to put an instance in Stand-by and perform such actions. However, this works best when the Load Balancer targets are instance-type. However, the threat defense virtual machines cannot be configured as instance-type targets, because of multiple network interfaces.

Put an Instance in Stand-by

If an instance is put into stand-by, its IP in Target Groups will still continue to be in the same state until the health probes fail. Because of this, it is recommended to deregister respective IPs from the Target Group before putting the instance into stand-by state; see Load Balancer Targets, on page 38 for more information.

Once the IPs are removed, see Temporarily Removing Instances from Your Auto Scaling Group.

Remove an Instance from Stand-by

Similarly you can move an instance from stand-by to running state. After removal from stand-by state, the instance's IP should be registered to Target Group targets. See Load Balancer Targets, on page 38.

For more information about how to put instances into stand-by state for troubleshooting or maintenance, see the AWS News Blog.

Remove/Detach Instance from Auto Scale Group

To remove an instance from the Auto Scale group, first it should be moved to stand-by state. See "Put Instances on Stand-by". Once the instance is in the stand-by state it can be removed or detached. See Detach EC2 Instances from Your Auto Scaling Group.

There won't be any changes on the management center side. Any changes required should be performed manually.

Terminate an Instance

To terminate an instance it should be put into stand-by state; see Instance Stand-by, on page 38. Once the instance is in stand-by, you can proceed to terminate.

Instance Scale-In Protection

To avoid an accidental removal of any particular instance from the Auto Scale group, it can be made as Scale-In protected. If an instance is Scale-In protected, it won't be terminated due to a Scale-In event.

Please refer to the following link to put an instance into Scale-In protected state.

https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-instance-termination.html

Important It is recommended to make the minimum number of instances which are healthy (the target IP should be healthy, not just the EC2 instance) as Scale-In protected.

Changes to Configuration

Any changes in configuration won't be automatically reflected on already running instances. Changes will be reflected on upcoming devices only. Any such changes should be manually pushed to already existing devices.

If you are facing issues while manually updating the configuration on existing instances, we recommend removing these instances from the Scaling Group and replacing them with new instances.

Change the Management Center User Name and Password

In the case of changes to the management center IP, username, or password—the respective changes should be performed on Auto Scale Manager Lambda function and custom metric publisher Lambda function environment variables. See Using AWS Lambda Environment Variables.

When Lambda runs next time, it will reference the changed environment variables.

Note Environment variables are directly fed to Lambda functions. There is no password complexity check here.

Change the Threat Defense Virtual Admin Password

A change to the threat defense virtual password requires the user to change it on each device manually for running instances. For new threat defense virtual devices to be onboarded, the threat defense virtual password will be taken from the Lambda environment variables. See Using AWS Lambda Environment Variables.

Change Registration and NAT IDs

For new threat defense virtual devices to be onboarded with different registration and NAT IDs, for the management center registration this information should be changed in Configuration.json file. The Configuration.json file can be located in Lambda resource page.

Changes to Access Policy and NAT Policy

Any changes to Access policies or NAT policies are automatically applied to upcoming instances with the help of the Device Group assignment. However, to update existing threat defense virtual instances you need to manually push configuration changes and deploy them from the management center.

Changes to AWS Resources

You can change many things in AWS post deployment, such as the Auto Scale Group, Launch Configuration, CloudWatch events, Scaling Policies etc. You can import your resources into a CloudFormation stack or create a new stack from your existing resources.

See Bringing Existing Resources Into CloudFormation Management for more information about how to manage changes performed on AWS resources.

Collect and Analyze CloudWatch Logs

In order to export CloudWatch logs please refer to Export Log Data to Amazon S3 Using the AWS CLI.

Troubleshooting

AWS CloudFormation Console

You can verify the input parameters to your CloudFormation stack in the AWS CloudFormation Console, which allows you to create, monitor, update and delete stacks directly from your web browser.

Navigate to the required stack and check the parameter tab. You can also check inputs to Lambda Functions on the Lambda Functions environment variables tab. The *configuration.json* file can also be viewed on the Auto Scale Manager Lambda function itself.

To learn more about the AWS CloudFormation console, see the *AWS CloudFormation User Guide*.

Amazon CloudWatch Logs

You can view logs of individual Lambda functions. AWS Lambda automatically monitors Lambda functions on your behalf, reporting metrics through Amazon CloudWatch. To help you troubleshoot failures in a function, Lambda logs all requests handled by your function and also automatically stores logs generated by your code through Amazon CloudWatch Logs.

You can view logs for Lambda by using the Lambda console, the CloudWatch console, the AWS CLI, or the CloudWatch API. To learn more about log groups and accessing them through the CloudWatch console, see the Monitoring system, application, and custom log files in the *Amazon CloudWatch User Guide*.

Load Balancer Health Check Failure

The load balancer health check contains information such as the protocol, ping port, ping path, response timeout, and health check interval. An instance is considered healthy if it returns a 200 response code within the health check interval.

If the current state of some or all your instances is `OutOfService` and the description field displays the message that the `Instance has failed at least the Unhealthy Threshold number of health checks consecutively`, the instances have failed the load balancer health check.

You should check the health probe NAT rule in the management center configuration. For more information, see Troubleshoot a Classic Load Balancer: Health checks.

Traffic Issues

To troubleshoot traffic issues with your threat defense virtual instances, you should check the Load Balancer rules, the NAT rules, and the static routes configured in the threat defense virtual instances.

You should also check the AWS virtual network/subnets/gateway details provided in the deployment template, including security group rules, etc. You can also refer to AWS documentation, for example, Troubleshooting EC2 instances.

Connection to the Management Center Failed

If the management connection is disrupted, you should check the configuration and credentials. See "Requirements and Prerequisites for Device Management" in *Firepower Management Center Configuration Guide*.

Device Failed to Register with the Management Center

If the device fails to register with the management center fails, you need to determine if the management center configuration is faulty/unreachable, or if the management center has the capacity to accommodate a new device. See "Add a Device to the " in *Firepower Management Center Configuration Guide*.

Unable to SSH into the Threat Defense Virtual

If you are unable to SSH into the threat defense virtual, check to see if the complex password was passed to the threat defense virtual via the template.

Use Case - Auto Scale Solution for Threat Defense Virtual using GWLB on AWS to Inspect North-South Traffic

This is a use case document that explains how to set up auto scaling of Threat Defense Virtual instances using a Gateway Load Balancer (GWLB) in the AWS environment 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.

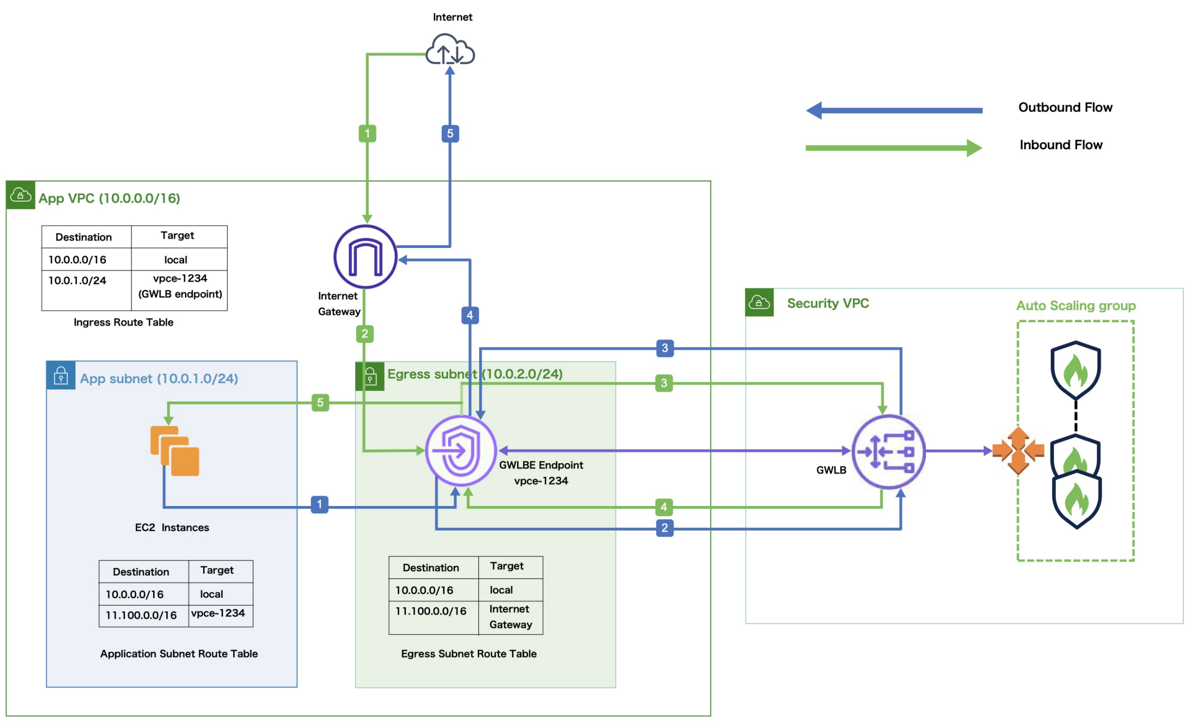
The GWLB acts as a single entry and exit point to manage internally and externally generated traffic and scales up or down the number of Threat Defense Virtual instances in real time based on traffic load.

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 3: Threat Defense Virtual Auto Scale Solution with GWLB



Destination	Target
10.0.0.0/16	local
10.0.1.0/24	vpce-1234 (GWLB endpoint)

Ingress Route Table

Destination	Target
10.0.0.0/16	local
11.100.0.0/16	vpce-1234

Application Subnet Route Table

Destination	Target
10.0.0.0/16	local
11.100.0.0/16	Internet Gateway

Egress Subnet Route Table

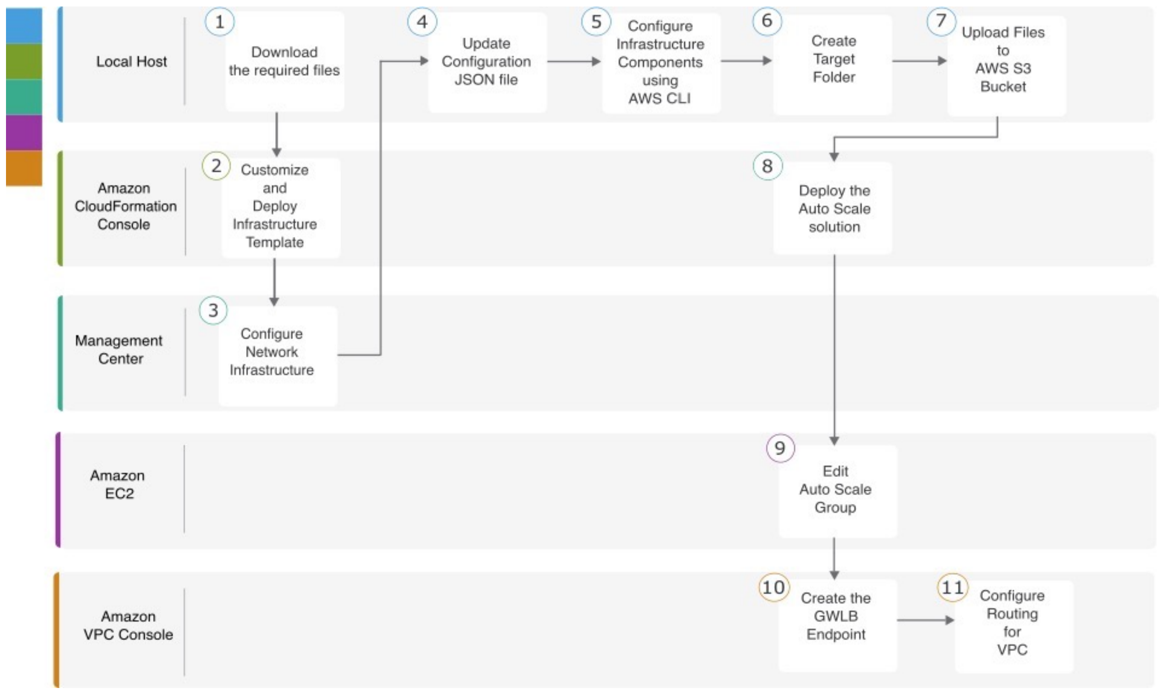
Inbound Traffic Inspection	
1	The Internet Gateway (IGW) receives traffic from the Internet.

Inbound Traffic Inspection	
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.
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).


Local Host
Amazon CloudFormation Console
Management Center
Amazon EC2
Amazon VPC Console
1 Download the required files
2 Customize and Deploy Infrastructure Template
3 Configure Network Infrastructure
4 Update Configuration JSON file
5 Configure Infrastructure Components using AWS CLI
6 Create Target Folder
7 Upload Files to AWS S3 Bucket
8 Deploy the Auto Scale solution
9 Edit Auto Scale Group
10 Create the GWLB Endpoint
11 Configure Routing for VPC

	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
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

	Workspace	Steps
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.

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

Parameters	Values
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

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

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

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** - *Routed*.

Step 6 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.

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.

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.

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.

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

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)	

Parameter	Values
NGFWv Password	*W1nch3sterBr0s*
FMC Automation Configuration	
FMC host IP address	*3.38.137.49*
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.

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

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

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.

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.

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.