FortiOS - AWS Administration Guide

Version 6.4

FORTINET DOCUMENT LIBRARY
https://docs.fortinet.com

FORTINET VIDEO GUIDE
https://video.fortinet.com

FORTINET BLOG
https://blog.fortinet.com

CUSTOMER SERVICE & SUPPORT
https://support.fortinet.com

FORTINET TRAINING & CERTIFICATION PROGRAM
https://www.fortinet.com/training-certification

NSE INSTITUTE
https://training.fortinet.com

FORTIGUARD CENTER
https://www.fortiguard.com

END USER LICENSE AGREEMENT
https://www.fortinet.com/doc/legal/EULA.pdf

FEEDBACK
Email: techdoc@fortinet.com

April 08, 2022
FortiOS 6.4 AWS Administration Guide
01-640-619425-20220408

TABLE OF CONTENTS

About FortiGate-VM for AWS

By combining stateful inspection with a comprehensive suite of powerful security features, FortiGate Next Generation Firewall (NGFW) technology delivers complete content and network protection. This solution is available for deployment on AWS.

In addition to advanced features such as an extreme threat database, vulnerability management, and flow-based inspection, features including application control, firewall, antivirus, IPS, web filter, and VPN work in concert to identify and mitigate the latest complex security threats.

The security-hardened FortiOS operating system is purpose-built for inspecting and identifying malware and supports direct Single Root I/O Virtualization (SR-IOV) for higher and more consistent performance.

FortiGate-VM for AWS supports active/passive high availability (HA) configuration with FortiGate-native unicast HA synchronization between the primary and secondary nodes. When the FortiGate-VM detects a failure, the passive firewall instance becomes active and uses AWS API calls to configure its interfaces/ports.

FortiGate-VM also supports active/active HA using elastic load balancing, as well as auto scaling.

Highlights of FortiGate-VM for AWS include the following:

- Delivers complete content and network protection by combining stateful inspection with a comprehensive suite of powerful security features.
- IPS technology protects against current and emerging network-level threats. In addition to signature-based threat detection, IPS performs anomaly-based detection, which alerts users to any traffic that matches attack behavior profiles.
- New Docker application control signatures protect your container environments from newly emerged security threats. See FortiGate-VM on a Docker environment.

Instance type support

FortiGate-VM supports the following instance types on AWS. Supported instances in the AWS marketplace listing may change without notice and vary between bring your own license (BYOL) and on-demand models. See Order types on page 11. As of May 2018, C3 and M-series instances no longer appear as recommended instances.

When you run FortiGate-native active-passive HA, each FortiGate-VM instance requires four network interfaces (port 1 to port 4). For details, see Deploying FortiGate-VM active-passive HA on AWS within one zone on page 123.

For up-to-date information on each instance type, see the following links:

- Amazon EC2 Instance Types
- Elastic Network Interfaces

Instance category	Instance type	vCPU	Max NIC (enabled by AWS)	FortiGate minimum order (BYOL) to consume all instance CPU
General purpose	T2.small	1	2	FG-VM01 or FG-VM01v

Instance category	Instance type	vCPU	Max NIC (enabled by AWS)	FortiGate minimum order (BYOL) to consume all instance CPU
Compute optimized	C4.large	2	3	FG-VM02 or FG-VM02v
	C4.xlarge	4	4	FG-VM04 or FG-VM04v
	C4.2xlarge	8	4	FG-VM08 or FG-VM08v
	C4.4xlarge	16	8	FG-VM16 or FG-VM16v
	C4.8xlarge	36	8	FG-VMUL or FG-VMULv
	C5.large (recommended by default)	2	3	FG-VM02 or FG-VM02v
	C5.xlarge	4	4	FG-VM04 or FG-VM04v
	C5.2xlarge	8	4	FG-VM08 or FG-VM08v
	C5.4xlarge	16	8	FG-VM16 or FG-VM16v
	C5.9xlarge	36	8	FG-VMUL or FG-VMULv
	C5.18xlarge	72	15	

You can apply a smaller FortiGate-VM license if you are OK with consuming less CPU than is present on your instance. See Models on page 9.

To change your instance type to the recommended C5 instance type, ensure that ENA is enabled. Otherwise the instance does not boot up properly.

In the following example, after changing the instance type to C5, ENA is not enabled. The example shows changing the ENA support attribute to `true`:

```
$ aws ec2 describe-instances --instance-ids i-xxxxxxx --query "Reservations[].Instances
     [].EnaSupport"
[]
$ aws ec2 modify-instance-attribute --instance-id i-xxxxxxx --ena-support
$ aws ec2 describe-instances --instance-ids i-xxxxxxx --query "Reservations[].Instances
     [].EnaSupport"
[
true
]
```

The instance can now boot up as a C5 instance type.

FortiOS 6.4.3 and later versions support hot-adding vCPU and RAM. However, AWS may not support this. See Requirements for changing the instance type.

Region support

BYOL and on-demand deployments support the following regions. See Order types on page 11.

Instance support may vary depending on the regions.

For details about regions, see Regions and Availability Zones.

Region name	Region code
US East (N. Virginia)	us-east-1
US East (Ohio)	us-east-2
US West (N. California)	us-west-1
US West (Oregon)	us-west-2
Asia Pacific (Hong Kong)	ap-east-1
Asia Pacific (Mumbai)	ap-south-1
Asia Pacific (Seoul)	ap-northeast-2
Asia Pacific (Singapore)	ap-southeast-1
Asia Pacific (Sydney)	ap-southeast-2
Asia Pacific (Tokyo)	ap-northeast-1
Canada (Central)	ca-central-1
EU (Frankfurt)	eu-central-1
EU (Ireland)	eu-west-1
EU (London)	eu-west-2
EU (Paris)	eu-west-3
EU (Stockholm)	eu-north-1
Middle East (Bahrain)	me-south-1
South America (São Paulo)	sa-east-1
AWS GovCloud (US-East)	us-gov-east-1
AWS GovCloud (US-West)	us-gov-west-1

AWS China is supported but does not appear with these regions when you log into the AWS portal. To use AWS resources on AWS China, you must have an AWS China account separate from your global AWS account.

FortiGate-VM for AWS China only supports the BYOL licensing model. To activate it, you must obtain a license. See Deploying on AWS China on page 30.

Models

FortiGate-VM is available with different CPU and RAM sizes. You can deploy FortiGate-VM on various private and public cloud platforms. The following table shows the models conventionally available to order, also known as BYOL models. See Order types on page 11.

Model name	vCPU	
	Minimum	Maximum
FG-VM01/01v/01s	1	1
FG-VM02/02v/02s	1	2
FG-VM04/04v/04s	1	4
FG-VM08/08v/08s	1	8
FG-VM16/16v/16s	1	16
FG-VM32/32v/32s	1	32
FG-VMUL/ULv/ULs	1	Unlimited

The v-series and s-series do not support virtual domains (VDOMs) by default. To add VDOMs, you must separately purchase perpetual VDOM addition licenses. You can add and stack VDOMs up to the maximum supported number after initial deployment.

Generally there are RAM size restrictions to FortiGate-VM BYOL licenses. However, these restrictions are not applicable to AWS deployments. Any RAM size with certain CPU models are allowed. Licenses are based on the number of CPUs only.

Previously, platform-specific models such as FortiGate-VM for AWS with an AWS-specific orderable menu existed. However, the common model is now applicable to all supported platforms.

For information about each model's order information, capacity limits, and adding VDOMs, see the FortiGate-VM datasheet.

The primary requirement for the provisioning of a FortiGate-VM may be the number of interfaces it can accommodate rather than its processing capabilities. In some cloud environments, the options with a high number of interfaces tend to have high numbers of vCPUs.

The licensing for FortiGate-VM does not restrict whether the FortiGate can work on a VM instance in a public cloud that uses more vCPUs than the license allows. The number of vCPUs indicated by the license does not restrict the FortiGate-VM from working, regardless of how many vCPUs are included in the virtual instance. However, only the licensed number of vCPUs process traffic and management. The rest of the vCPUs are unused.

The following shows an example for a FGT-VM08 license:

License	1 vCPU	2 vCPU	4 vCPU	8 vCPU	16 vCPU	32 vCPU
FGT-VM08	OK	OK	OK	OK	FortiOS uses eight vCPUs for traffic and management. It does not use the rest.	FortiOS uses eight vCPUs for traffic and management. It does not use the rest.

You can provision a VM instance based on the number of interfaces you need and license the FortiGate-VM for only the processors you need.

Licensing

You must have a license to deploy FortiGate-VM for AWS.

Order types

On AWS, there are usually two order types: BYOL and on-demand.

BYOL offers perpetual (normal series and v-series) and annual subscription (s-series) licensing as opposed to on-demand, which is an hourly subscription available with marketplace-listed products. BYOL licenses are available for purchase from resellers or your distributors, and the publicly available price list, which is updated quarterly, lists prices. BYOL licensing provides the same ordering practice across all private and public clouds, no matter what the platform is. You must activate a license for the first time you access the instance from the GUI or CLI before you can start using various features.

With an on-demand subscription, the FortiGate-VM becomes available for use immediately after you create the instance. The marketplace product page mentions term-based prices (hourly or annual).

For BYOL and on-demand deployments, cloud vendors charge separately for resource consumption on computing instances, storage, and so on, without use of software running on top of it (in this case the FortiGate-VM).

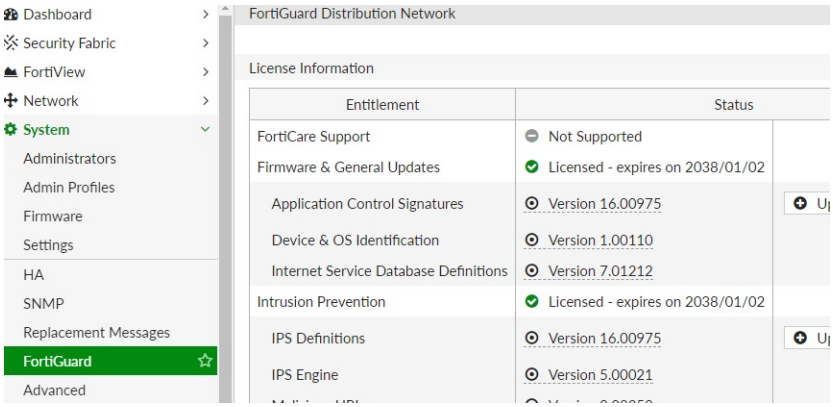
For BYOL, you typically order a combination of products and services including support entitlement. New s-series SKUs contain the VM base and service bundle entitlements for easier ordering. On-demand includes support, for which you must contact Fortinet Support with your customer information. See *Support Information* on the marketplace product page.

To purchase on-demand, all you need to do is subscribe to the product on the marketplace. However, you must contact Fortinet Support with your customer information to obtain support entitlement. See Creating a support account on page 12. For the latest on-demand pricing and support details, see the FortiGate-VM on-demand marketplace product page.

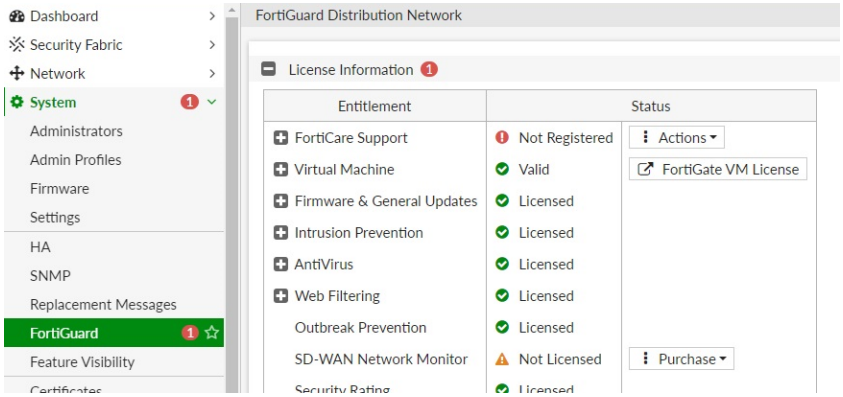
On-demand FortiGate-VM instances do not support the use of virtual domains (VDOMs). If you plan to use VDOMs, deploy BYOL instances instead.

On-demand and BYOL licensing and payment models are not interchangeable. For example, once you spin up a FortiGate-VM on-demand instance, you cannot inject a BYOL license on the same VM. Likewise, you cannot convert a FortiGate-VM BYOL instance to on-demand.

When using a FortiGate-VM on-demand instance prior to version 6.4.2, the FortiOS GUI may display expiry dates for FortiGuard services. However, these expiries are automatically extended for as long as the on-demand instance's lifespan. You do not need to be concerned about the expiry of FortiGuard services. For example, the following screenshot shows 2038/01/02.



FortiOS 6.4.2 and later versions do not display dates.



Creating a support account

FortiGate-VM for AWS supports on-demand and BYOL licensing models. See Order types on page 11.

To make use of Fortinet technical support and ensure products function properly, you must complete certain steps to activate your entitlement. The Fortinet support team can identify your registration in the system thereafter.

First, if you do not have a Fortinet account, create one at Customer Service & Support.

BYOL

You must obtain a license to activate the FortiGate-VM. If you have not activated the license, you will see the license upload screen when you log in to the FortiGate-VM and cannot proceed to configure the FortiGate-VM.

You can obtain licenses for the BYOL licensing model through any Fortinet partner. If you do not have a partner, contact awssales@fortinet.com for assistance in purchasing a license.

After you purchase a license or obtain an evaluation license (60-day term), you receive a PDF with an activation code.

To register a BYOL license:

1. Go to Customer Service & Support and create a new account or log in with an existing account.
2. Go to *Asset* > *Register/Activate* to start the registration process.

3. In the *Registration* page, enter the registration code that was emailed to you, and select *Next* to access the registration form.
4. If you register the S-series subscription model, the site prompts you to select one of the following:
   a. Click *Register* to newly register the code to acquire a new serial number with a new license file.
   b. Click *Renew* to renew and extend the licensed period on top of the existing serial number, so that all features on the VM node continue working uninterrupted upon license renewal.
5. At the end of the registration process, download the license (.lic) file to your computer. You will upload this license later to activate the FortiGate-VM.
   After registering a license, Fortinet servers may take up to 30 minutes to fully recognize the new license. When you upload the license (.lic) file to activate the FortiGate-VM, if you get an error that the license is invalid, wait 30 minutes and try again.

On-demand

To create a support account for on-demand deployments:

1. Deploy and boot the FortiGate-VM on-demand Elastic Compute Cloud (EC2) instance and log into the FortiGate-VM GUI management console.
2. From the Dashboard, copy the FortiGate-VM serial number.
3. Go to Customer Service & Support and create a new account or log in with an existing account.
4. Go to *Asset* > *Register/Activate* to start the registration process.
5. In the *Registration* page, enter the serial number, and select *Next* to continue registering the product. Enter your details in the other fields.
6. After completing registration, contact Fortinet Customer Support and provide your FortiGate instance's serial number and the email address associated with your Fortinet account.

FortiCare-generated license adoption for AWS on-demand variant

FortiGate-VM AWS on-demand instances can obtain FortiCare-generated licenses and register to FortiCare.

The valid license allows you to register to FortiCare to use features including FortiToken with the FortiGate-VM instance.

The FortiGate-VM must be able to reach FortiCare to receive a valid on-demand license. Ensure connectivity to FortiCare (https://directregistration.fortinet.com/) by checking all related setup on security groups, access control lists, Internet gateways, route tables, public IP addresses, and so on.

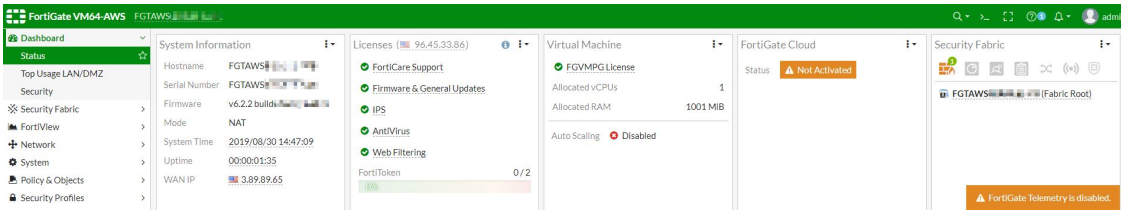
If you created the FortiGate-VM in a closed environment or it cannot reach FortiCare, the FortiGate-VM self-generates a local license as in previous versions of FortiOS. You can obtain a FortiCare license, ensure that the FortiGate-VM can connect to FortiCare, then run the `execute vm-license` command to obtain the license from FortiCare.

To deploy a FortiGate-VM 6.2.2 AWS on-demand instance:

When deploying a FortiGate-VM on-demand instance for AWS, you will use the FGT_VM64_AWS-v6-buildXXXX-FORTINET.out image. After deployment with this image, running `get system status` results in output that includes the following lines:

```
Version: FortiGate-VM64-AWS v6.2.2,buildXXXX,XXXXXX (GA)
Virus-DB: 71.00242(2019-08-30 08:19)
Extended DB: 1.00000(2018-04-09 18:07)
Extreme DB: 1.00000(2018-04-09 18:07)
IPS-DB: 6.00741(2015-12-01 02:30)
```

```
IPS-ETDB: 0.00000(2001-01-01 00:00)
APP-DB: 6.00741(2015-12-01 02:30)
INDUSTRIAL-DB: 6.00741(2015-12-01 02:30)
Serial-Number: FGTAWS12345678
```

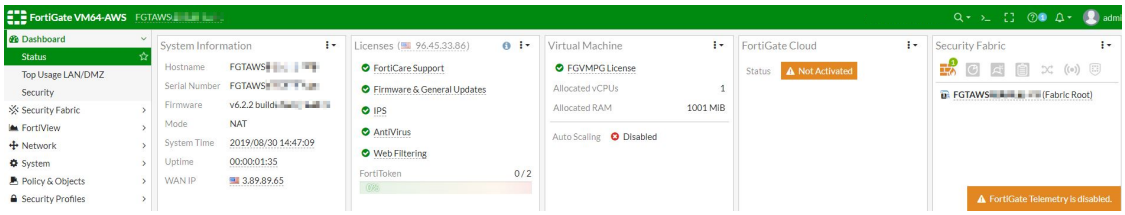


To upgrade a FortiGate-VM AWS on-demand instance from FortiOS 6.2.1 and earlier to 6.2.2:

Earlier versions used the FGT_VM64_AWSONDEMAND-v6-buildXXXX-FORTINET.out image to deploy a FortiGate-VM AWS on-demand instance. In 6.2.2, the FGT_VM64_AWS-v6-buildXXXX-FORTINET.out image is used to deploy a FortiGate-VM AWS on-demand instance.

When upgrading from an earlier FortiOS version, you must first upgrade using the FGT_VM64_AWSONDEMAND image, then use the FGT_VM64_AWS image.

1. In FortiOS, perform an upgrade using the FGT_VM64_AWSONDEMAND-v6-buildXXXX-FORTINET.out image.
2. Perform another upgrade, this time using the FGT_VM64_AWS-v6-buildXXXX-FORTINET.out image. This process is irreversible.



3. Run `get system status` results in output that includes the following lines:
   ```
   Version: FortiGate-VM64-AWS v6.2.2,buildXXXX,XXXXXX (GA)
   Virus-DB: 71.00246(2019-08-30 12:19)
   Extended DB: 1.00000(2018-04-09 18:07)
   Extreme DB: 1.00000(2018-04-09 18:07)
   IPS-DB: 14.00680(2019-08-30 02:29)
   IPS-ETDB: 0.00000(2001-01-01 00:00)
   APP-DB: 14.00680(2019-08-30 02:29)
   INDUSTRIAL-DB: 14.00680(2019-08-30 02:29)
   Serial-Number: FGTAWS1234567890
   ```
4. For future upgrades, use the FGT_VM64_AWS-v6-buildXXXX-FORTINET.out image to retain on-demand status. You cannot directly upgrade a FortiGate-VM AWS on-demand instance from 6.2.1 or earlier to 6.2.3 and later versions. You must first follow the procedure detailed above.

Migrating a FortiGate-VM instance between license types

When deploying a FortiGate-VM on public cloud, you determine the license type (on-demand or BYOL) during deployment. The license type is fixed for the VM's lifetime. The image that you use to deploy the FortiGate-VM on the public cloud marketplace predetermines the license type.

Migrating a FortiGate-VM instance from one license type to another requires a new deployment. You cannot simply switch license types on the same VM instance. However, you can migrate the configuration between two VMs running as

different license types. There are also FortiOS feature differences between on-demand and BYOL license types. For example, a FortiGate-VM on-demand instance is packaged with Unified Threat Management protection and does not support VDOMs, whereas a FortiGate-VM BYOL instance supports greater protection levels and features depending on its contract.

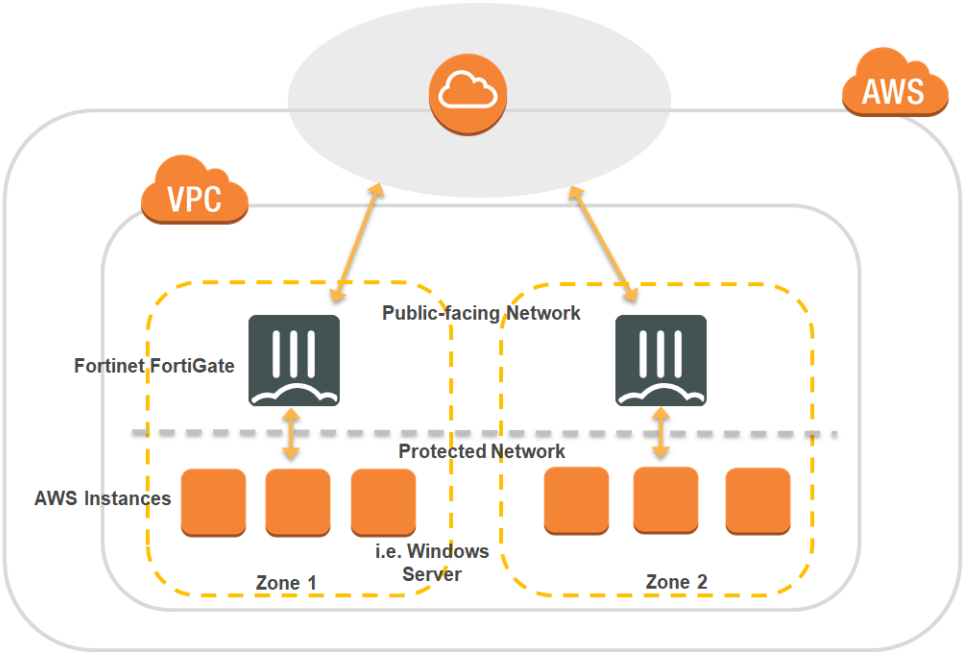
To migrate FortiOS configuration to a FortiGate-VM of another license type:

1. Connect to the FortiOS GUI or CLI and back up the configuration. See Configuration backups.
2. Deploy a new FortiGate-VM instance with the desired license type. If deploying a BYOL instance, you must purchase a new license from a Fortinet reseller. You can apply the license after deployment via the FortiOS GUI or bootstrap the license and configuration during initial bootup using custom data as described in Bootstrapping the FortiGate-VM at initial bootup using user data on page 21.
3. Restore the configuration on the FortiGate-VM instance that you deployed in step 2. As with the license, you can inject the configuration during initial bootup. Alternatively, you can restore the configuration in the FortiOS GUI as described in Configuration backups.
4. If you deployed an on-demand instance in step 2, register the license. To receive support for an on-demand license, you must register the license as described in Creating a support account on page 12.

Deploying FortiGate-VM on AWS

Launching FortiGate-VM on AWS

See Single FortiGate-VM deployment on page 96.



The most basic deployment consists of one FortiGate-VM with two elastic network interfaces (ENIs) facing a public subnet and private subnet, with the FortiGate-VM deployed inline between the two subnets. A single FortiGate-VM protects a single virtual private cloud (VPC) with a single availability zone (AZ). The public subnet's default gateway is an AWS Internet gateway, and the FortiGate-VM's private subnet-facing ENI is the private subnet's default gateway. Protected EC2 instances such as web servers, database servers, or other endpoints are assumed to exist in the private subnet. One elastic/public IP address or IPv4 DNS name must be allocated to the FortiGate-VM in the public subnet for you to access the FortiGate-VM remotely via HTTPS or SSH over the Internet for initial configuration.

Security best practices

General AWS security best practices can be found at AWS Security Best Practices.

In addition to following the general AWS guidelines, there are best practices to follow when deploying FortiGate-VM for AWS.

Opening ports in the security group

By default, when you deploy FortiGate-VM, there is a predefined security group that you can select based on Fortinet's recommendation. The following ports are allowed in the predefined security group assuming immediate and near-future needs.

	Protocol/ports	Purpose
Incoming	TCP 22	SSH
	TCP 80	HTTP
	TCP 443	HTTPS, management GUI access to the FortiGate-VM
	TCP 541	Management by FortiManager located outside AWS
	TCP 3000	Not immediately required, but typically used for incoming access to web servers, and so on
	TCP 8080	
Outgoing	Any	

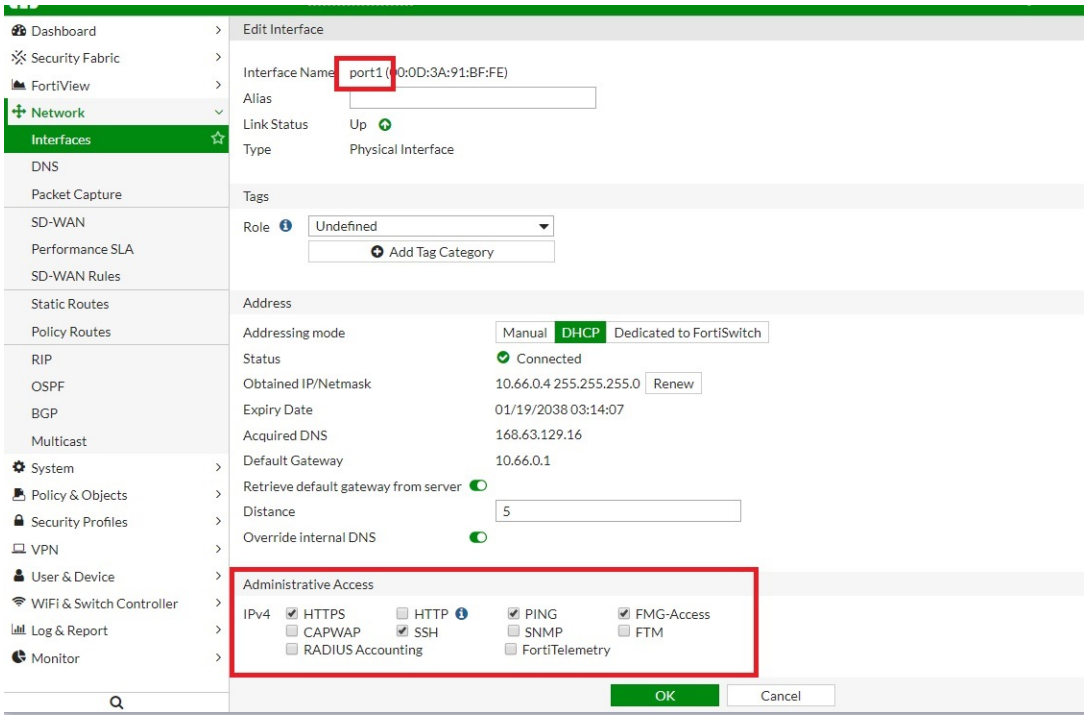
FortiGate-specific open ports are explained in Fortinet Communication Ports and Protocols.

To configure bare-minimum access that gives the most strict incoming access, allow only TCP 443 to access the FortiGate-VM GUI console as mentioned in Connecting to the FortiGate-VM on page 107 and close all other ports. You may want to allow ICMP for pinging, and so on, as needed.

Administrative access

This is rather an ordinary consideration than AWS-specific to secure the FortiGate-VM and protect it by configuring allowed and restricted protocols and ports in corporate security scenes.

One example is to configure the local admin access to one of the FortiGate-VM's local network interfaces. Log into the GUI, go to *Network* > *Interfaces*, then chose the desired port to configure under *Administrative Access*.



To configure general firewall policies to protect VMs in the networks, refer to Setting up a Windows Server in the protected network on page 110 or the FortiOS documentation for details.

IAM roles

To deploy FortiGate-VM on the marketplace, you must log into the AWS portal as an AWS user. Your organization's administrator may have granted permissions via certain IAM roles. AWS security best practices explain when and in what use cases you need IAM roles. How you manage IAM users and roles is up to your organization.

When deploying FortiGate-VM on marketplace web or EC2 console, your AWS account must have appropriate permissions, including being able to subscribe to AWS resources through the marketplace, access EC2 resources, browse AWS resource groups, and so on.

Login credentials

By default, you can log into the FortiGate-VM through HTTPS or SSH using the username "admin" and the FortiGate-VM's instance ID as the initial password. SSH also requires your AWS key.

The instance ID is relatively secure as it is visible only within the AWS portal or by running the AWS CLI. However, it may be viewable to those who have access to AWS resources but should not have access to the FortiGate-VM within the same organization. It is strongly recommended to change the initial password the first time you log in or activate the license. You can also create other administrative users using more complex character strings than "admin" in a manner difficult to guess, or add two-factor authentication or other methods to secure login.

AWS services and components

FortiGate-VM for AWS is an Elastic Compute Cloud (EC2) instance with an Elastic Block Store (EBS) volume attached. The following lists AWS services and components that you must understand when deploying FortiGate-VM for different purposes:

Ordinary FortiGate-VM single instance deployment or FortiGate-native active-passive high availability

Service/component	Description
Virtual private cloud (VPC)	This is where the FortiGate-VM and protected VMs are situated and users control the network. The public-facing interface is routed to the Internet gateway, which is created within the VPC.
EC2	FortiGate-VM for AWS is an EC2 VM instance. Every instance has a unique instance ID.
Subnets, route tables	You must appropriately configure FortiGate-VM with subnets and route tables to handle traffic.
Internet gateways	The AWS gateway as a VPC component that allows communication between instances in your VPC and the Internet.
Elastic IP address (EIP)	At least one public IP address must be allocated to the FortiGate-VM to access and manage it over the Internet.
Security groups	AWS public-facing protection. Allow only necessary ports and protocols.
AMI	A special type of deployable image used on AWS. You can launch FortiGate-VM (BYOL) directly from the publicly available FortiGate AMI instead of using the marketplace. See Deploying from BYOL AMI on page 26. The on-demand AMI is launchable but does not allow you to properly boot up as it is not intended to be deployed from AMI.
CloudFormation Templates (CFT)	FortiGate instances can be deployed using CFTs where tailor-made resource instantiation is defined. Fortinet provides CFTs for the following use cases: • Deploying FortiGate-native A-P HA • Customer-required scenarios with particular topologies CFTs are available on GitHub. Fortinet-provided CFTs are not supported within the regular Fortinet technical support scope. Contact awssales@fortinet.com with questions.

Additional or alternative HA using AWS mechanisms

Service/component	Description
Auto Scaling	Auto scaling can automatically scale out by instantiating additional FortiGate-VM instances at times of high workloads. See Deploying auto scaling on AWS on page 39. To run auto scaling, you must enable/subscribe to coexisting AWS services: • Route 53 • API gateway • Load Balancer • CloudWatch • Lambda • SNS • DynamoDB • Simple Storage Services (S3) (BYOL only) These services are not always required for AWS auto scaling in general, but are predefined in Fortinet-provided Lambda scripts.
Load Balancer	Also called Elastic Load Balancer (ELB). A network load balancer automatically distributes traffic across multiple FortiGate-VM instances when configured properly. Topologies will be different depending on how you distribute incoming and outgoing traffic and cover AZs. There are two use cases to use LB with FortiGate-VM: • Deploying and configuring ELB-based HA/load balancing on page 113 • Used in conjunction with auto scaling. See Deploying auto scaling on AWS on page 39.

Monitoring

Service/component	Description
CloudWatch	Monitoring service for various AWS resources. You can use CloudWatch in three scenarios with FortiGate-VM: • Monitor FortiGate-VM instance health and alert when needed. • Define auto scaling scale-out triggers to fire alarms • Monitor GuardDuty events You must subscribe to CloudWatch to use corresponding features.

Related AWS services used as prerequisites for additional HA or extra features

Service/component	Description
Lambda	AWS Lambda lets you run certain scripts and codes without provisioning servers. Fortinet provides Lambda scripts for: • Running auto scaling

Service/component	Description
	• GuardDuty integration To use the scripts, you must subscribe to Lambda. Fortinet-provided Lambda scripts are not supported within the regular Fortinet technical support scope. Contact awssales@fortinet.com with questions.
API Gateway	It acts as a front door by providing a callback URL for the FortiGate-VM to send its API calls and process FortiGate-VM config-sync tasks to synchronize OS configuration across multiple FortiGate-VM instances at the time of auto scaling scale-out. It is required if the config-sync feature needs to be incorporated into auto scaling.
DynamoDB	A handy flexible database. Fortinet-provided scripts use DynamoDB to store information about varying states of auto scaling conditions.
SNS	Managed message service used to communicate between AWS components. Fortinet-provided scripts use SNS to deliver subscription notifications from CFTs to Lambda for auto scaling.
GuardDuty	Managed threat detection service that monitors unwanted behaviors/activities related to AWS resources. Fortinet can leverage externally available lists of malicious IP addresses stored at certain locations. GuardDuty can be used to populate such a list. See Populating threat feeds with GuardDuty on page 161. To use this feature, you must subscribe to GuardDuty.
S3	AWS storage. You can use S3 in four scenarios with FortiGate-VM: • As the location where the list of blocklisted IP addresses is stored which is pointed by the FortiGate-VM in integrating with GuardDuty. See Populating threat feeds with GuardDuty on page 161. You must allow the FortiGate-VM access to the S3 bucket/directory on S3 configuration. • To store license keys which are parsed when provisioning additional FortiGate-VM instances in the event of auto scaling scale-out. • To store a license key and the FortiGate-VM config file to bootstrap the FortiGate-VM at initial boot-up. See Bootstrapping the FortiGate-VM at initial bootup using user data on page 21. • To store license keys which are parsed when provisioning A-P HA.

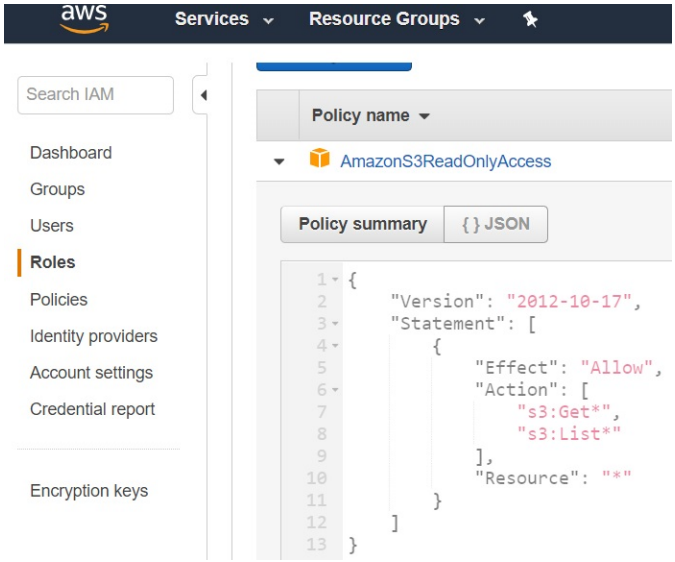
Bootstrapping the FortiGate-VM at initial bootup using user data

If you are installing and configuring your applications on Amazon EC2 dynamically at instance launch time, you will typically need to pull and install packages, deploy files, and ensure services are started. The following bootstrapping instructions help simplify, automate, and centralize FortiGate-VM NGFW deployment directly from the configuration scripts stored in AWS S3. This is also called "cloud-init".

Setting up IAM roles

IAM roles need S3 bucket read access. This example applies the existing AmazonS3ReadOnlyAccess policy to the role by adding the following code or selecting S3ReadOnlyAccess from the policy list in adding to the role:

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "s3:Get*",
            "s3:List*"
         ],
         "Resource": "*"
      }
   ]
}
```



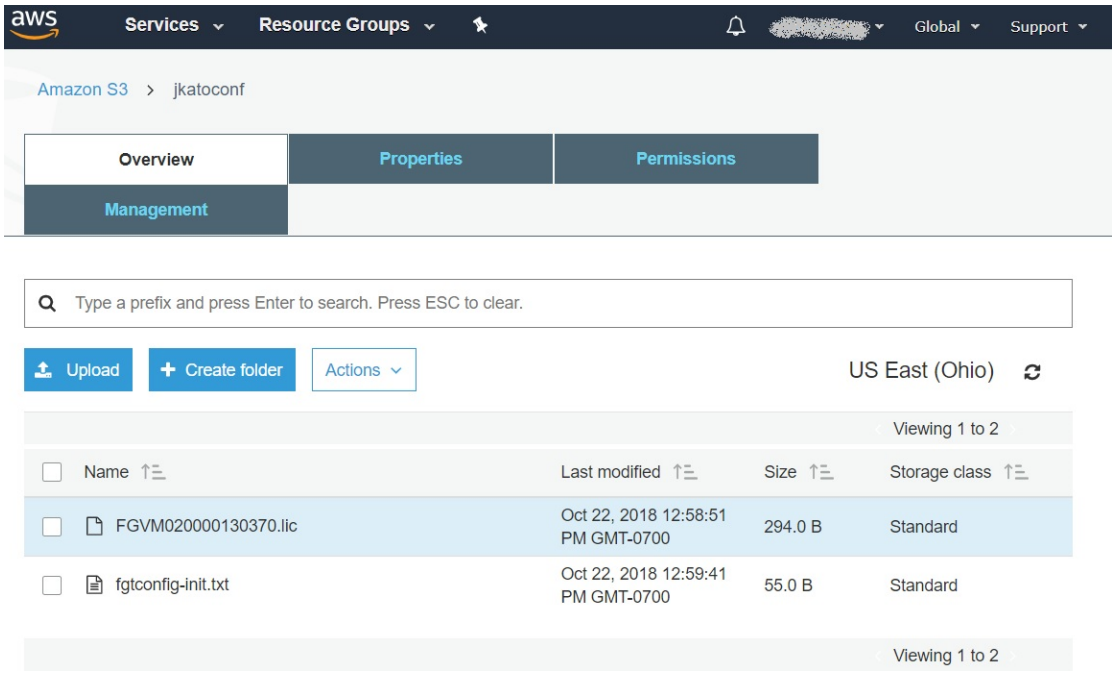
If you need further instructions, please refer to the AWS documentation on IAM Roles for Amazon EC2.

Creating S3 buckets with license and firewall configurations

1. On the AWS console, create an Amazon S3 bucket at the root level for the bootstrap files.
2. Upload the license file and configuration file(s) to the S3 bucket. In this example, one license file and two configuration files are uploaded. For example, let's have the following FortiOS CLI command statement in the config file:
   ```
   config sys global
      set hostname jkatocloudinit
   end
   ```
   This is to set a hostname as part of initial configuration at first-time launch.
3. Amazon S3 creates the bucket in a region you specify. You can choose any AWS region that is geographically close to you to optimize latency, minimize costs, or address regulatory requirements. To choose a region, use the following code:
   ```
   {
   ```

```
   "bucket" : "jkatoconf",
   "region" : "us-east-2",
   "license" : "/FGVM020000130370.lic",
   "config" : "/fgtconfig-init.txt"
}
```

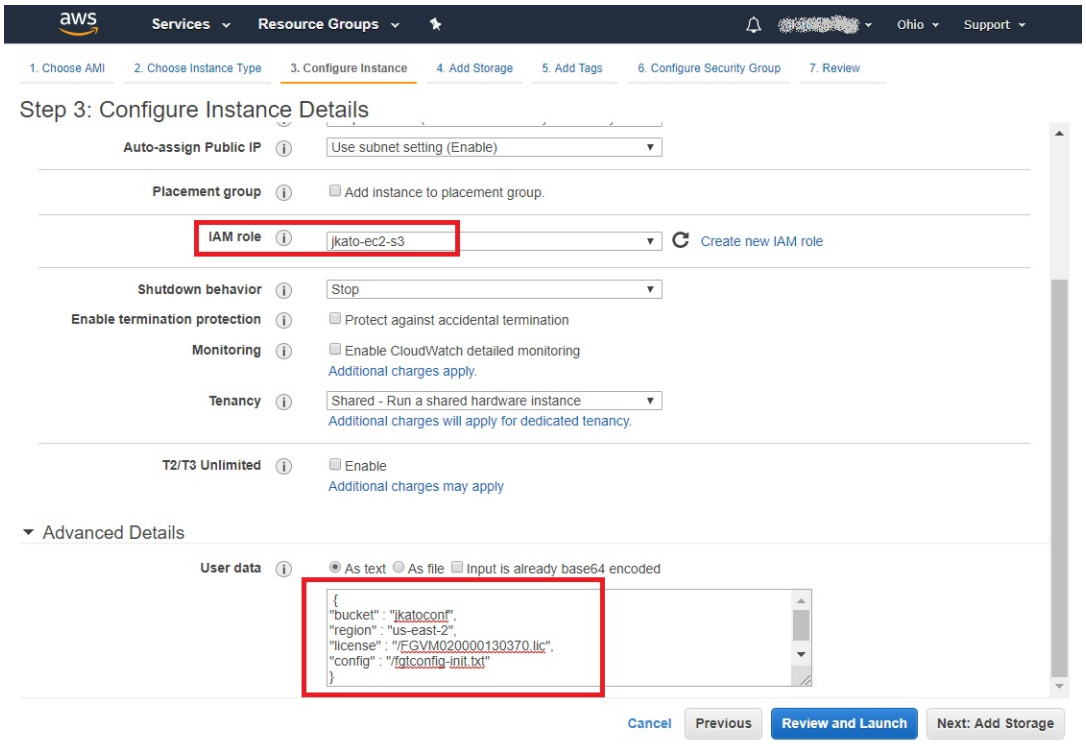
Although the S3 bucket and the firewall can be in different regions, it is highly recommended that they are in the same region in order to speed up the bootstrapping process.



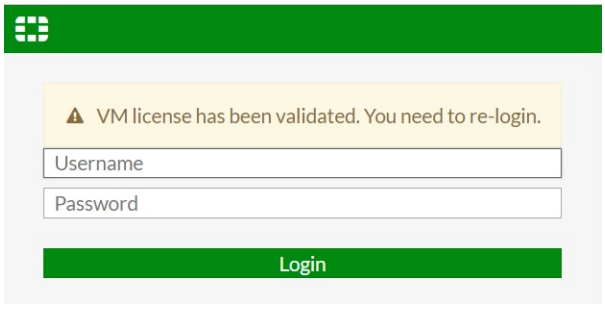
Launching the instance using roles and user data

Follow the normal procedure to launch the instance from the AWS marketplace.

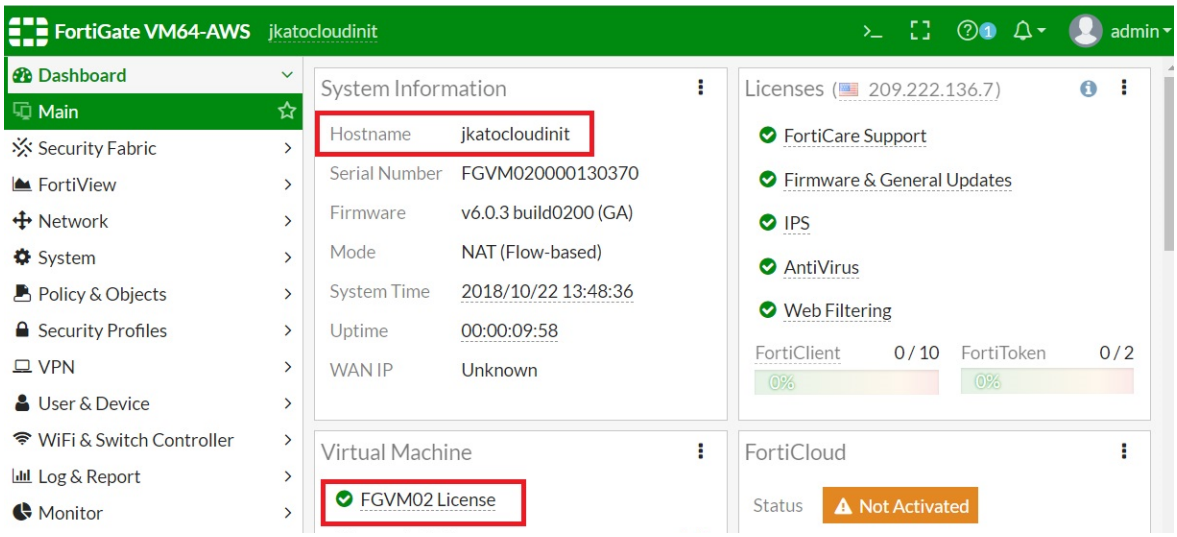
When selecting the VPC subnet, the instance must with the role that was created and specify the information about the license file and configuration file from the AWS S3 bucket previously configured under *Advanced Settings*. In this example, the role name is jkato-ec2-s3.



After you launch the FortiGate-VM for the first time and log into the management GUI, FortiOS validates the license instead of displaying the license upload prompt.



After logging in, you can see that the license was activated and that the specified hostname was configured.

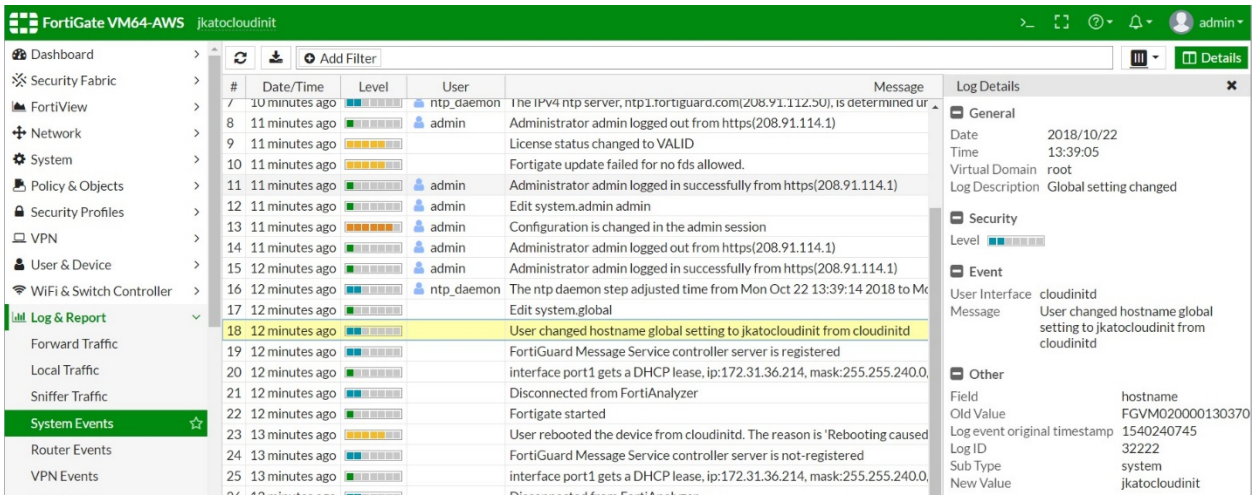


Check the serial number that you have with the license.

CLI Console

```
jkatocloudinit #
jkatocloudinit # get system status
Version: FortiGate-VM64-AWS v6.0.3,build0200,181009 (GA)
Virus-DB: 1.00000(2018-04-09 18:07)
Extended DB: 1.00000(2018-04-09 18:07)
Extreme DB: 1.00000(2018-04-09 18:07)
IPS-DB: 6.00741(2015-12-01 02:30)
IPS-ETDB: 0.00000(2001-01-01 00:00)
APP-DB: 6.00741(2015-12-01 02:30)
INDUSTRIAL-DB: 6.00741(2015-12-01 02:30)
Serial-Number: FGVM020000130370
IPS Malicious URL Database: 1.00001(2015-01-01 01:01)
Botnet DB: 1.00000(2012-05-28 22:51)
License Status: Valid
License Expires: 2018-12-22
VM Resources: 1 CPU/2 allowed, 2006 MB RAM
BIOS version: 04000002
Log hard disk: Available
Hostname: jkatocloudinit
Operation Mode: NAT
Current virtual domain: root
Max number of virtual domains: 10
Virtual domains status: 1 in NAT mode, 0 in TP mode
Virtual domain configuration: disable
FIPS-CC mode: disable
Current HA mode: standalone
--More--
```

You can view the cloud-init log in *Log & Report* > *System Events*.



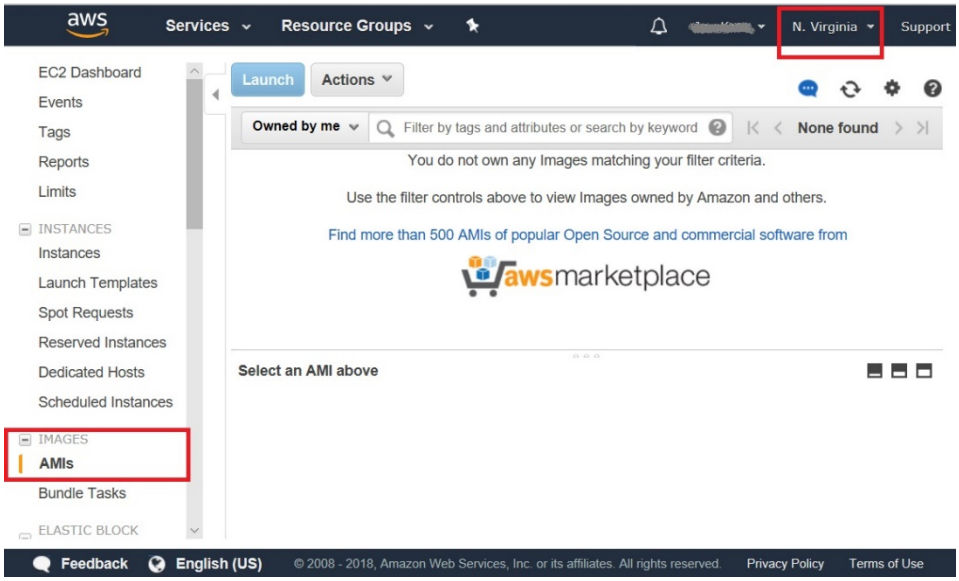
Deploying from BYOL AMI

You can deploy FortiGate-VM outside the marketplace launcher if you want to install it manually from the AMI for some reason, such as if your organization does not allow access to the AWS marketplace website. There are AMI images publicly available in various regions for the versions already listed in the marketplace. This deployment works only with AMI for BYOL licensing. Deploying from AMI designed for on-demand is not supported.

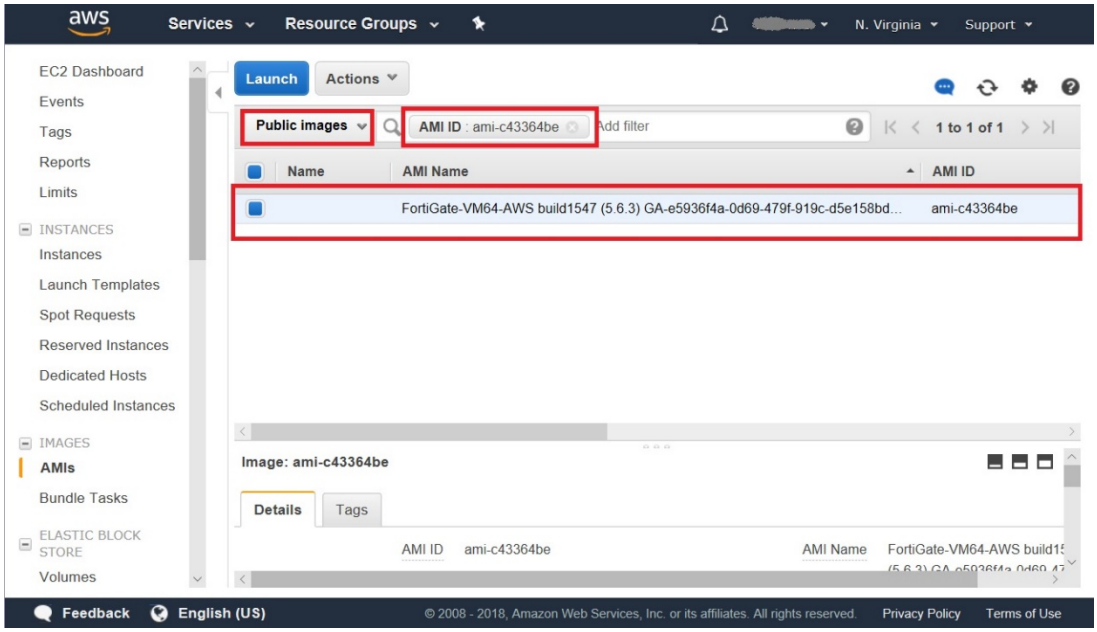
If you want to install the latest FortiGate-VM versions immediately after release from Fortinet but you do not see them published in the marketplace or publicly available in the AWS portal, you can always deploy older versions of FortiGate-VM available on the marketplace or the AWS portal as publicly available AMIs, then upgrade using the ".out" upgrade files, which are available at Customer Service & Support.

To deploy from BYOL AMI:

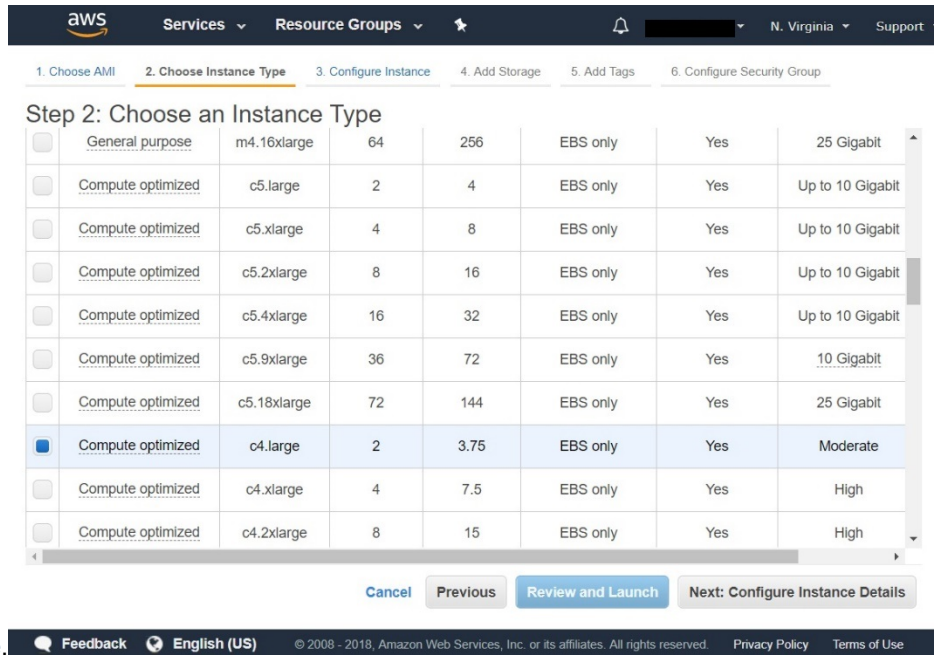
1. Select the desired AMI:
   a. Log into the AWS EC2 console and go to *IMAGES* > *AMIs*. Select the appropriate region.



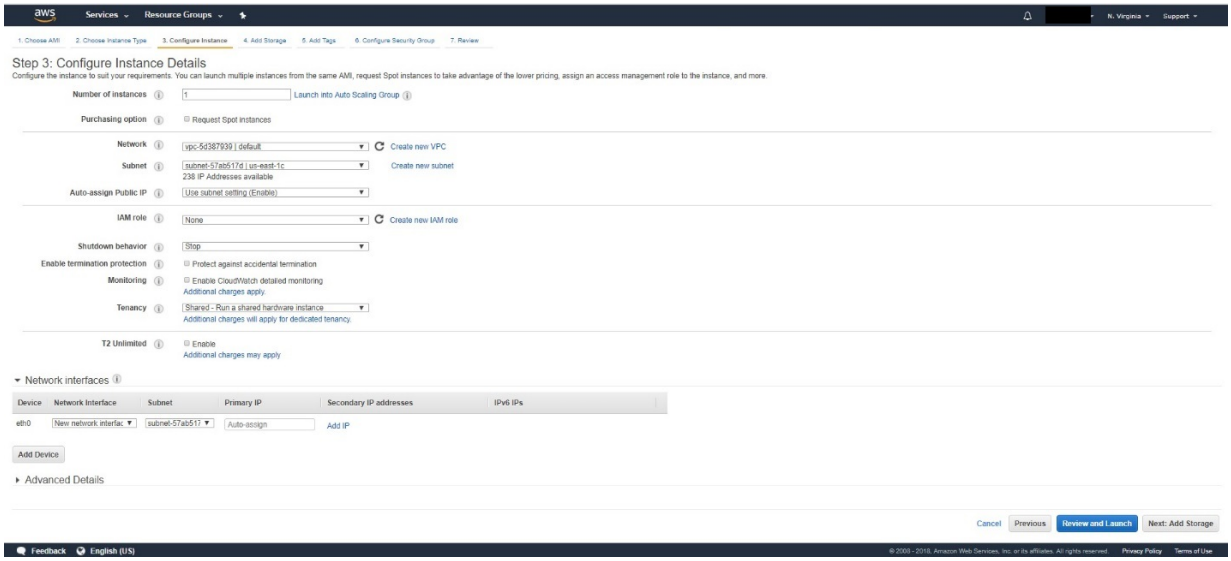
   b. Find the desired public AMI from the list of AMI IDs corresponding to your region.



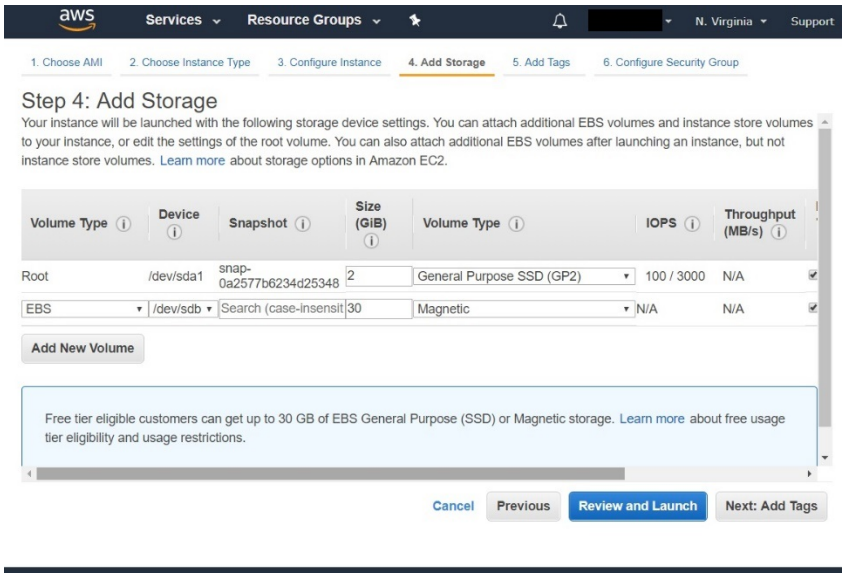
   c. Select the AMI and click *Launch*.



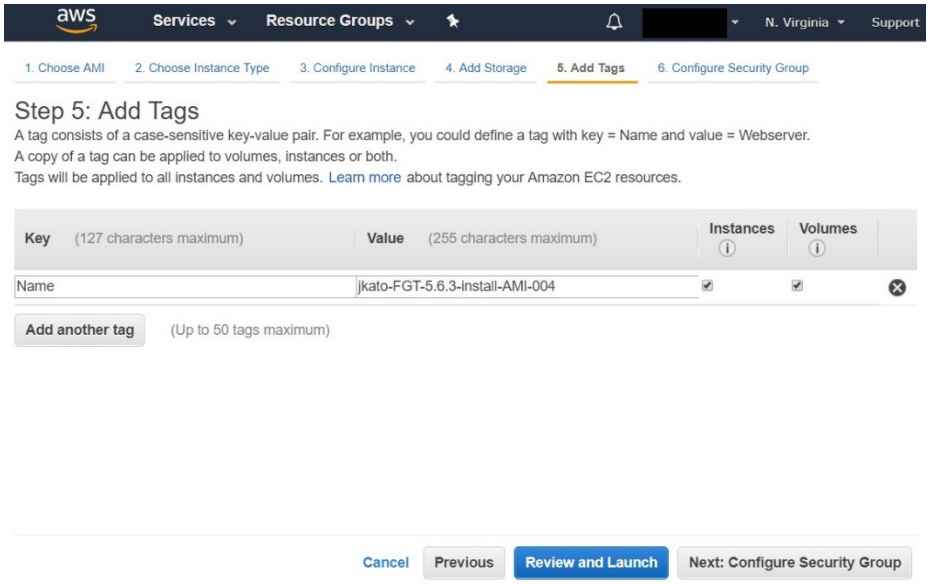
2. Choose a supported instance.
3. Click *Next: Configure Instance Details*.
4. Configure the instance details:
   a. In the *Network* field, select the VPC that you created.
   b. In the *Subnet* field, select the public subnet.
   c. In the *Network interfaces* section, you will see the entry for eth0 that was created for the public subnet. Select *Add Device* to add another network interface (in this example, eth1), and select the private subnet. It is recommended that you assign static IP addresses.
   d. When you have two network interfaces, a global IP address is not assigned automatically. You must manually assign a global IP address later. Select *Review and Launch*, then select *Launch*.



5. Click *Next: Add Storage*.
6. In *Step 4: Add Storage*, you can leave the fields as-is, or change the size of /dev/sdb as desired. The second volume is used for logging.



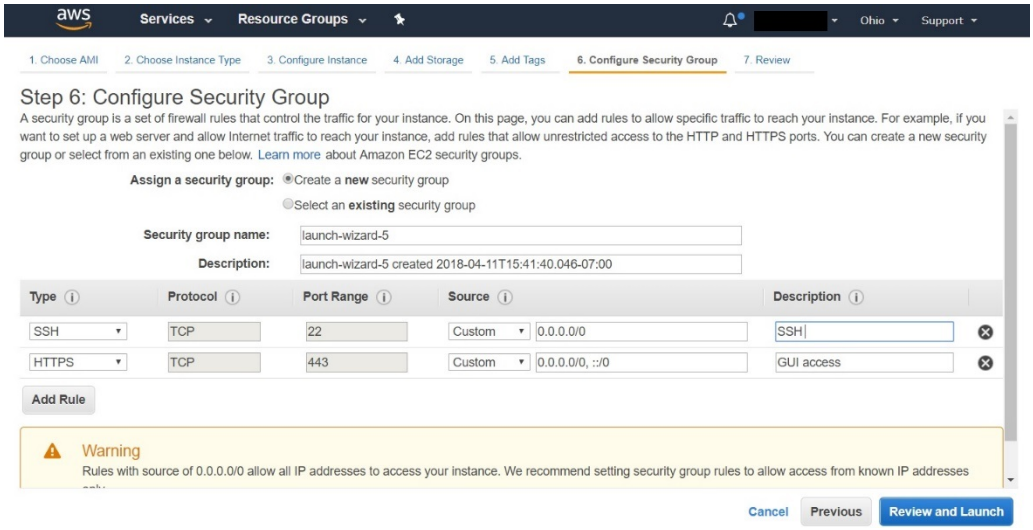
7. Click *Next: Add Tags*. You can add tags for convenient management.



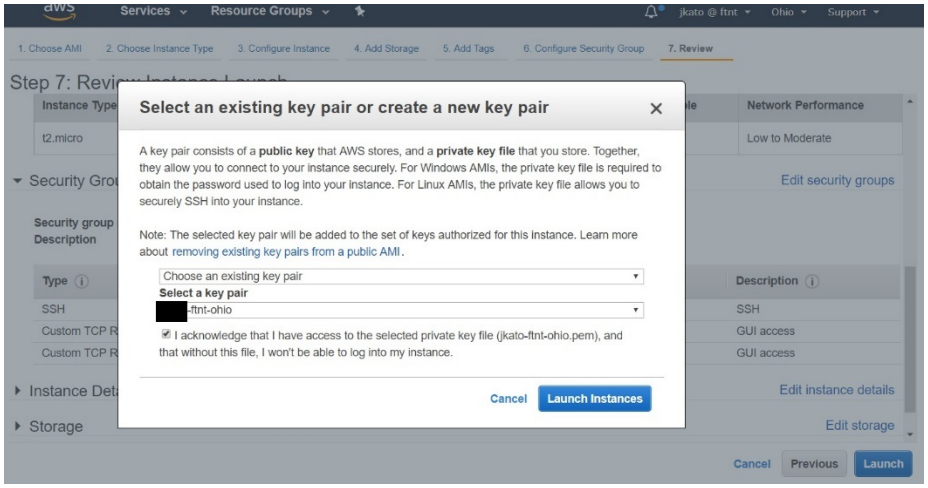
8. Click *Next: Configure Security Groups*. Here it is important to allow some incoming ports. Allow TCP port 8443 for management from the GUI. You can also allow TCP port 22 for SSH login. Allow other ports where necessary as noted. The use of ports is explained in the *FortiOS documentation*.

Incoming TCP ports allowed	Purpose
22	SSH
443	Management using the GUI
541	Management by FortiManager located outside AWS
8000	Fortinet Single Sign On
10443	SSLVPN

You can change the source address later.



9. Click *Review and Launch*. If everything looks good, go to next by clicking *Launch*.
10. Then select the appropriate keypair, then click *Launch Instance*. It may take 15 to 30 minutes to deploy the instance. To access the FortiGate and complete post-install setup, see Connecting to the FortiGate.



Deploying on AWS China

Deploying FortiGate-VM for AWS China has separate requirements than deploying FortiGate-VM for global AWS. To use AWS resources on AWS China, you must have an AWS China account separate from your global AWS account.

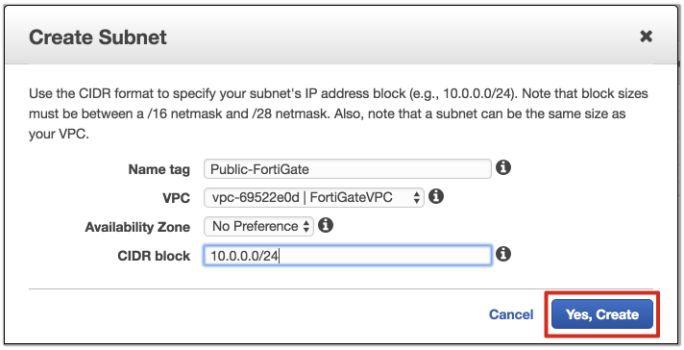
FortiGate-VM for AWS China only supports the BYOL licensing model. To activate it, you must obtain a license. Complete the following steps to deploy FortiGate-VM on AWS China:

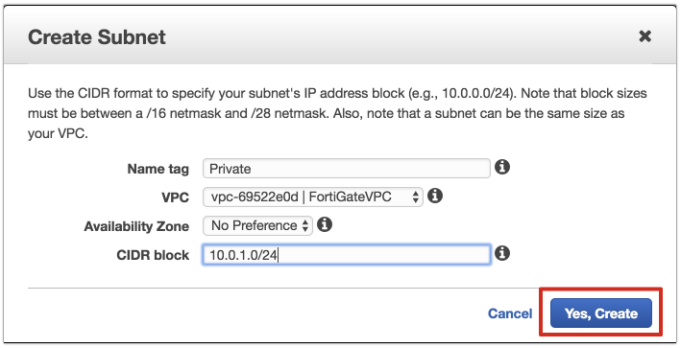
1. Creating a support account on page 12
2. Creating a VPC and subnets on page 31
3. Attaching the new VPC Internet gateway on page 31
4. Launching the instance with shared FortiGate-VM AMI on page 32
5. Connecting to the FortiGate-VM on page 33

Creating a VPC and subnets

This section shows you how to create an AWS VPC and create two subnets in it. For many steps, you have a choice to make that can be specific to your own environment.

1. Change your language to English and log into the AWS Management Console.
2. Go to *Services* > *Networking* > *VPC*.
3. Go to *Virtual Private Cloud* > *Your VPCs*, then select *Create VPC*.
4. In the *Name tag* field, set the VPC name.
5. In the *CIDR block* field, specify an IPv4 address range for your VPC.
6. In the *Tenancy* field, select *Default*.
7. Select *Yes, Create*.
8. Go to *Virtual Private Cloud* > *Subnets*, then select *Create Subnet*. Create a public subnet (in this example, *Subnet1*) and a private subnet (*Subnet2*), as shown in this example. Both subnets belong to the VPC that you created.





Attaching the new VPC Internet gateway

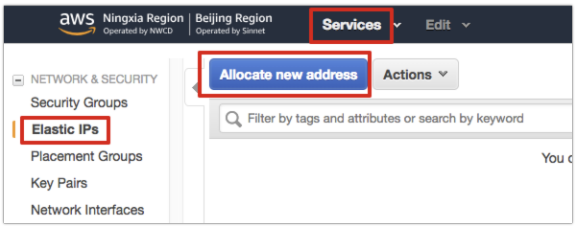
This section shows how to connect the new VPC to the Internet gateway. If you are using the default VPC, the Internet gateway should already exist.

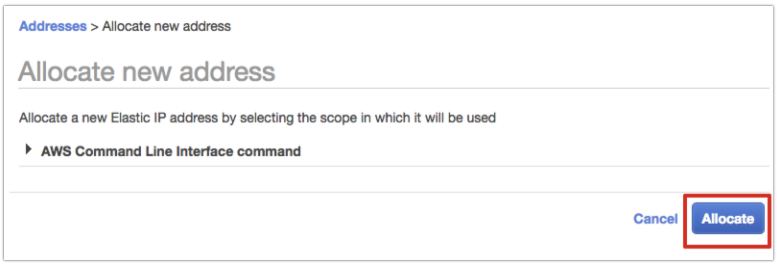
1. Go to *Virtual Private Cloud* > *Internet Gateways*, then select *Create internet Gateway*.
2. In the *Name tag* field, set the Internet gateway name, then select *Create*.
3. Select the Internet gateway, then select *Attach to VPC*.
4. Select the created VPC and select *Attach*. The Internet gateway state changes from *detached* to *attached*.

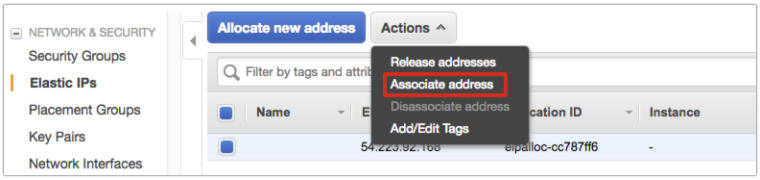
Launching the instance with shared FortiGate-VM AMI

To launch the instance with FortiGate-VM AMI:

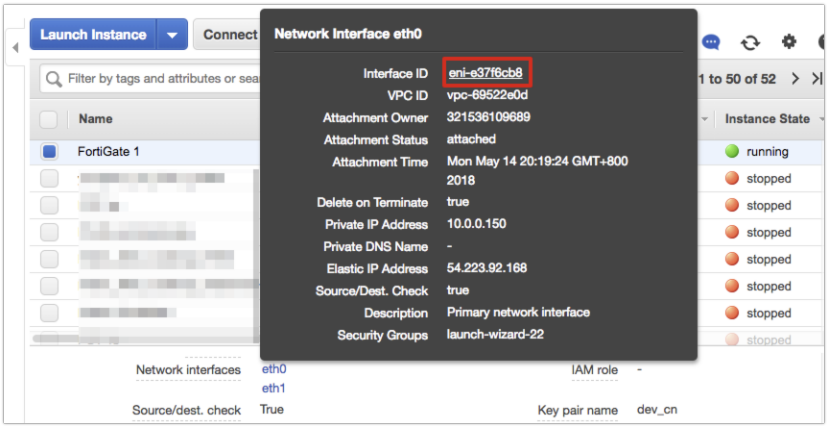
1. In the *Services*-EC2 Dashboard, go to *INSTANCES* > *Instances*, then select *Launch Instance*.
2. Select *AWS Marketplace*. Search for *FortiGate*. Click *Select*.
3. Select an instance type, then select *Next: Configure Instance Details*.
4. Configure the instance details:
   a. In the *Network* field, select the VPC you created.
   b. In the *Subnet* field, select the public subnet.
   c. In the *Network interfaces* section, you see the entry for eth0 that was created for the public subnet. Select *Add Device* to add another network interface (in this example, eth1), and select the private subnet.
   d. When you have two network interfaces, a global IP address is not assigned automatically. You must manually assign a global IP address later. Select *Review and Launch*, then select *Launch*.
   e. Select an existing key pair or create a new key pair. Select the acknowledgment checkbox. Select *Launch Instances*.
   f. To easily identify the instance, set a name for it in the *Name* field.
   g. Go to *NETWORK & SECURITY* > *Elastic IPs*, select a global IP address that is available for use. Select *Actions* > *Allocate new address*. If you do not have a global IP address available to use, create one.

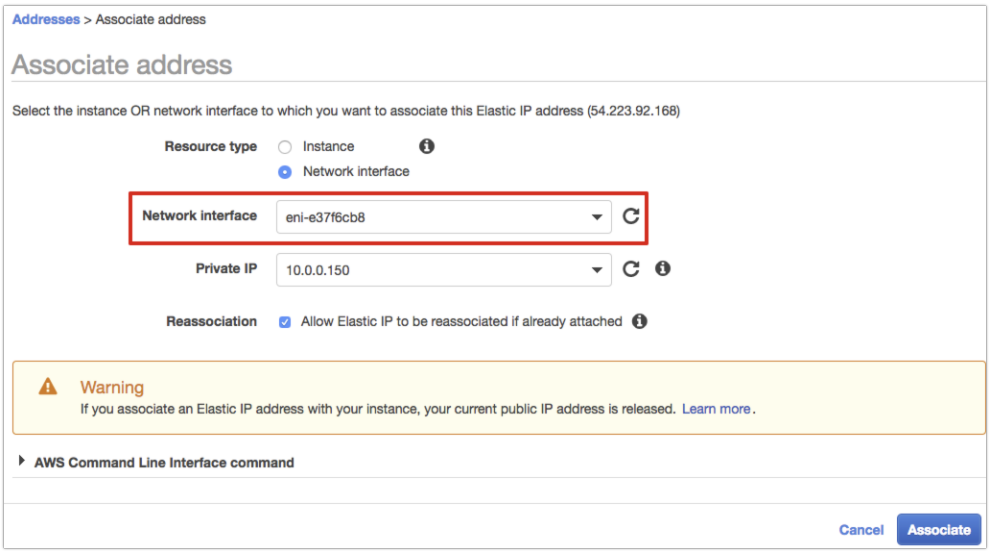






   h. In the *Resource type* section, select *Network Interface*.
   i. In the *Network interface* field, select the Interface ID of the network interface that you created for the public subnet (in this example, eth0). In the *Private IP* field, select the IP address that belongs to the public subnet. To find these values, go to the EC2 Management Console, select Instances, and select the interface in the *Network interfaces* section in the lower pane of the page (*Interface ID* and *Private IP Address* fields). Select *Associate*. A message is displayed indicating the address association was successful. Note that if the Internet Gateway isn't associated with a VPC, the elastic IP assignment will fail.



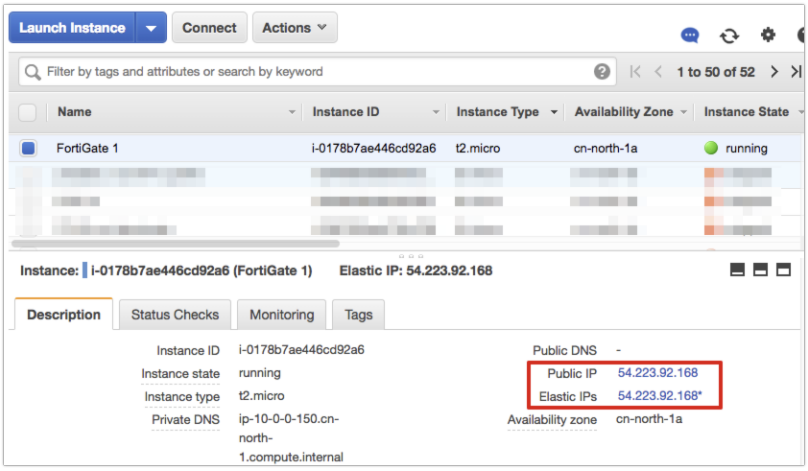


Connecting to the FortiGate-VM

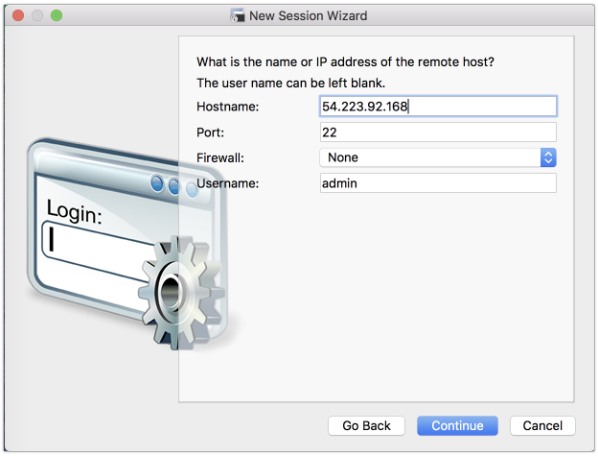
To connect to the FortiGate-VM, you need your login credentials, the FortiGate-VM's EIP, SSH client, and an FTP server.

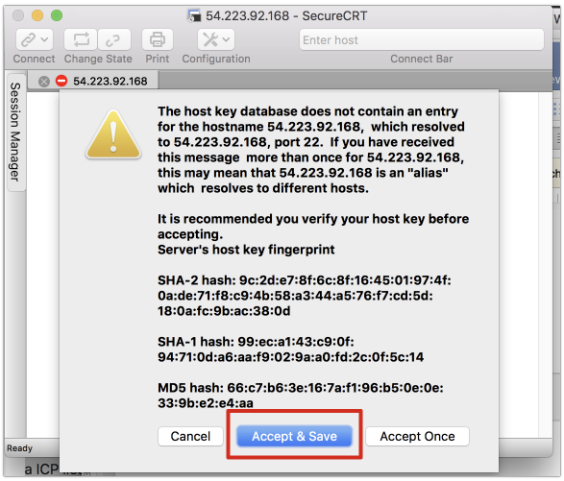
The default username is admin and the default password is the instance ID.

1. You can find the public IP address in the EC2 management console. Select *Instances* and look at the *Public IP* field in the lower pane.

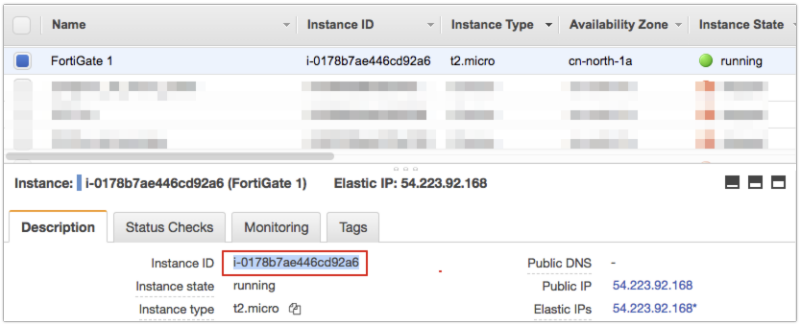


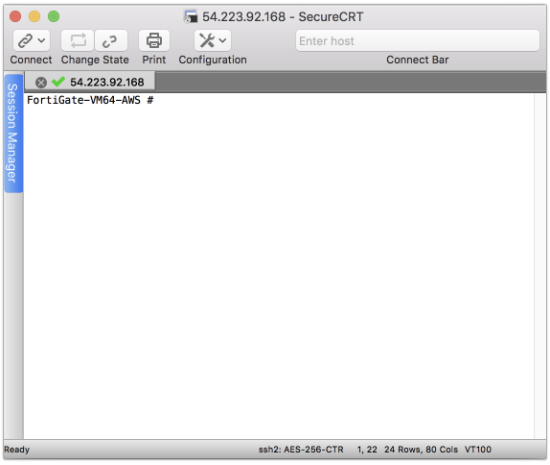
2. Each public IP address in China should obtain an ICP license. Otherwise it cannot be visited by ports 80, 443, and 8080. You cannot initially access the FortiGate-VM web GUI via the default HTTPS port. You can access the FortiGate-VM via SSH, then upload a BYOL license to the FortiGate-VM via FTP or TFTP. After activating the FortiGate-VM, you can modify the default admin HTTPS port to any port, such as 8443. Then you can go to the FortiGate-VM via https://<FortiGate-VM EIP>:8443.

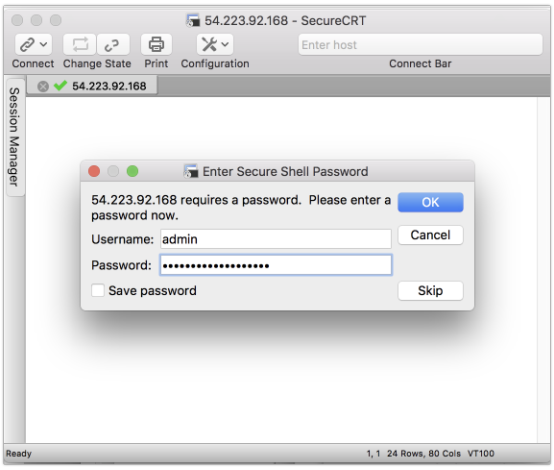




The default password is the instance ID as seen below.







3. Set up an FTP/TFTP server and ensure the FortiGate can log onto and download a BYOL license from it.
4. On the FortiGate, use one of the following CLI commands to restore the VM license.

```
exec restore vmlicense tftp <license file name> <IP address>
exec restore vmlicense ftp <license name (path) on the remote server> <ftp server
    address>[:ftp port]
```

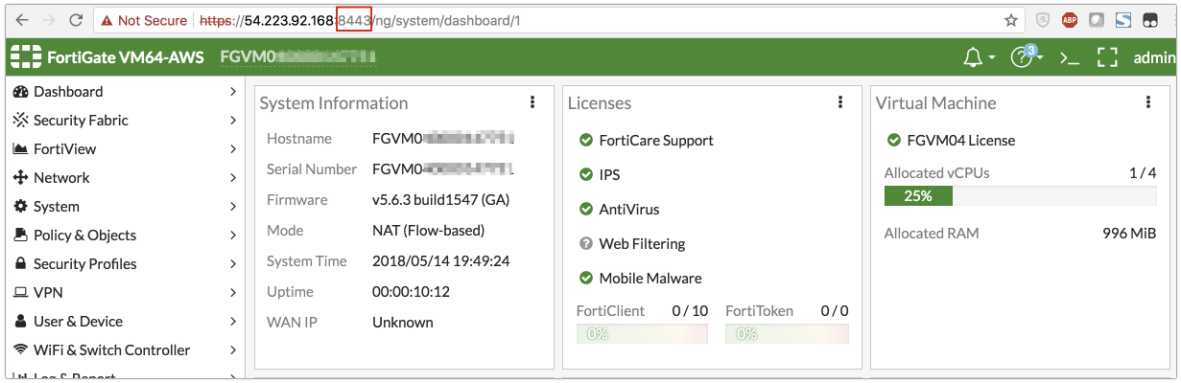
```
FortiGate-VM64-AWS # exec restore vmlicense tftp FGVM0[redacted].lic 1[redacted]3
This operation will overwrite the current VM license!
Do you want to continue? (y/n)y

Please wait...

Connect to tftp server 1[redacted]3 ...


Get VM license from tftp server OK.
VM license install succeeded. Rebooting firewall.
```

If the license installation is successful, the FortiGate-VM reboots automatically. After it restarts, log in.

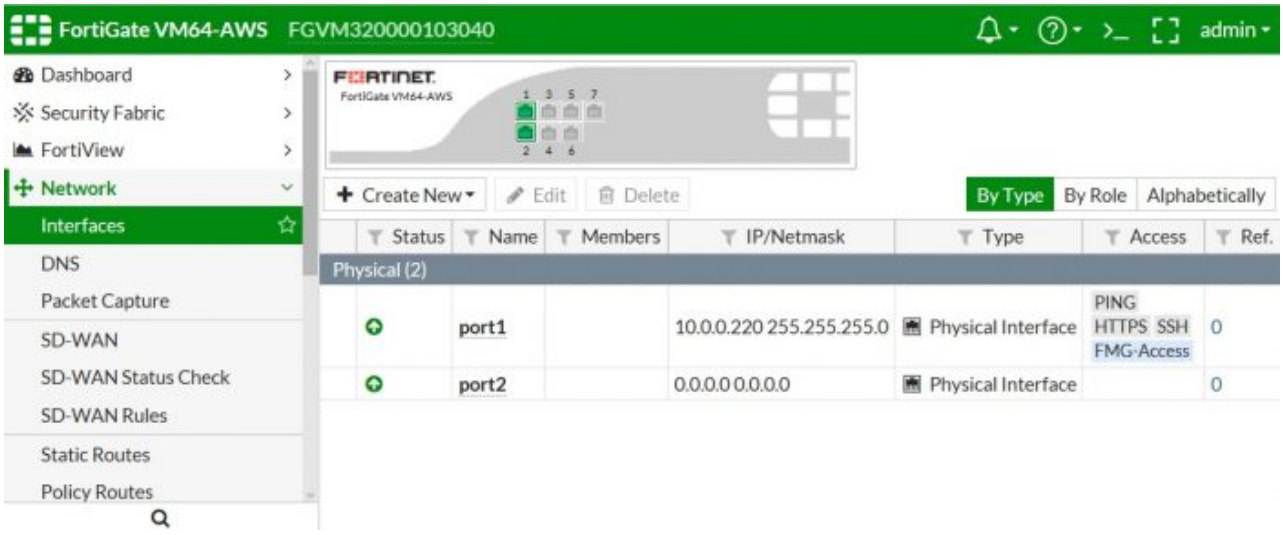
5. Change the default port to any port, such as 8443. Do not use ports 443, 8080, or 80.

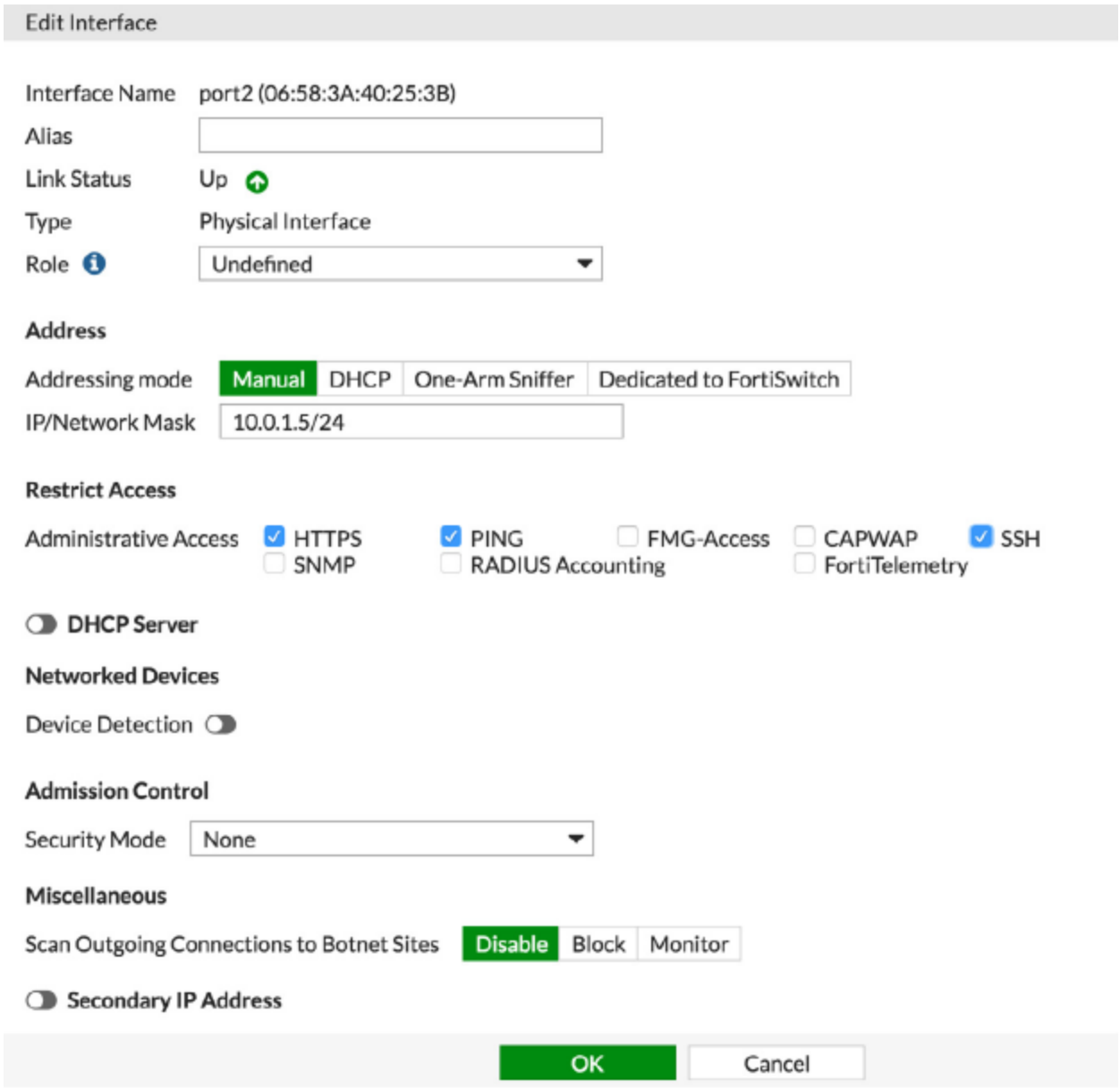
```
FGVM0[redacted] # config system global
FGVM0[redacted] (global) # set admin-sport 8443
FGVM0[redacted] (global) # end
```

6. You will now see the FortiGate-VM dashboard. Depending on your license type, the information in the license widget on the dashboard may vary.



7. Select *Network* > *Interfaces*, and edit the interfaces, if required. If the IP address or subnet mask is missing for port 1 or port 2, configure these values.





Upgrading the FortiGate-VM

For the recommended upgrade path, see the FortiOS Version Upgrade Path. Select *FortiGate-VM-AWS*, and the current and target upgrade versions.

For upgrade instructions, see *Upgrading the firmware*.

Backing up and restoring configuration

See Configuration backups.

Deploying auto scaling on AWS

You can deploy FortiGate virtual machines (VMs) to support Auto Scaling on AWS. Optionally, AWS Transit Gateway can be used to connect Amazon Virtual Private Clouds (Amazon VPCs) and their on-premises networks to a single gateway. This integration extends the FortiGate protection to all networks connected to the Transit Gateway. Consolidate logging and reporting for your FortiGate cluster by integrating FortiAnalyzer. Fortinet provides FortiGate Autoscale for AWS deployment packages to facilitate the deployment.

Multiple FortiGate-VM instances form an Auto Scaling group to provide highly efficient clustering at times of high workloads. FortiGate-VM instances can be scaled out automatically according to predefined workload levels. When a spike in traffic occurs, FortiGate-VM instances are automatically added to the Auto Scaling group. Auto Scaling is achieved by using FortiGate-native High Availability (HA) features that synchronize operating system (OS) configurations across multiple FortiGate-VM instances at the time of scale-out events.

FortiGate Autoscale for AWS is available with FortiOS 6.2.5, FortiOS 6.4.6, FortiOS 7.0.0, and FortiOS 7.0.1 and supports any combination of On-demand and Bring Your Own License (BYOL) instances. FortiAnalyzer 6.4.6 can be incorporated into Fortinet FortiGate Autoscale to use extended features that include storing logs into FortiAnalyzer.

Fees will be incurred based on the Amazon Elastic Compute Cloud (Amazon EC2) instance type. Additionally, a license is required for each FortiGate Bring Own License (BYOL) instance you may use.

FortiGate Autoscale for AWS uses AWS CloudFormation Templates (CFTs) to deploy components.

Deployments without Transit Gateway integration have:

- A highly available architecture that spans two Availability Zones.*
- An Amazon VPC configured with public and private subnets according to AWS best practices, to provide you with your own virtual network on AWS.*
- An Internet gateway to allow access to the Internet.*
- In the public subnets:
  - (Optional) A FortiAnalyzer instance, which consolidates logging and reporting for your FortiGate cluster.
  - Two or more FortiGate-VM instances, which complement AWS security groups. Security groups provide intrusion protection, web filtering, and threat detection to help protect your services from cyberattacks. Each instance also provides VPN access for authorized users. VPN connections use the Diffie-Hellman Group 14 and SHA256 (Secure Hash Algorithm 2).
  - A cluster of FortiGate-VM instances in the Auto Scaling groups, where one FortiGate-VM acts as the primary while the others act as secondary. The primary FortiGate-VM also acts as NAT gateway by default, allowing egress Internet access for resources in the private subnets.
- A public-facing network load balancer that distributes inbound traffic across the FortiGate-VM instances. An internal-facing network load balancer is optional.
- AWS Lambda, which provides the core Auto Scaling functionality between FortiGates-VM instances.
- Amazon Simple Storage Service (Amazon S3) to host artifacts for Lambda functions and logs.
- Amazon DynamoDB to store information about Auto Scaling condition states.

* When deploying into an existing VPC, the marked components in the above list are not created - you are prompted for your existing VPC configuration.

Deployments with Transit Gateway integration have:

- A highly available architecture that spans two Availability Zones.
- An Amazon VPC configured with public and private subnets according to AWS best practices, to provide you with your own virtual network on AWS.
- An Internet gateway to allow access to the Internet.
- In the public subnets:
  - (Optional) A FortiAnalyzer instance, which consolidates logging and reporting for your FortiGate cluster.
  - Two or more FortiGate-VM instances, which complement AWS security groups. Security groups provide intrusion protection, web filtering, and threat detection to help protect your services from cyberattacks. Each instance also provides VPN access for authorized users. VPN connections use the Diffie-Hellman Group 14 and SHA256 (Secure Hash Algorithm 2).
  - A primary FortiGate-VM instance in the Auto Scaling group(s).
- AWS Lambda, which provides the core Auto Scaling functionality between FortiGate-VM instances.
- Amazon Simple Storage Service (Amazon S3) to host artifacts for Lambda functions and logs.
- Amazon DynamoDB to store information about Auto Scaling condition states.
- Site-to-Site VPN connections.

Planning

This deployment requires familiarity with the configuration of a FortiGate using the CLI as well as with the following AWS services:

- Amazon Elastic Cloud Compute (Amazon EC2)
- Amazon EC2 Auto Scaling
- Amazon VPC
- AWS CloudFormation
- AWS Lambda
- Amazon DynamoDB
- Amazon API Gateway
- Amazon CloudWatch
- Amazon S3

If deploying with Transit Gateway integration, knowledge of the following is also required:

- AWS Transit Gateway
- Border Gateway Protocol (BGP)
- Equal-cost multi-path (ECMP)

If you are new to AWS, visit the Getting Started Resource Center and the AWS Training and Certification website.

It is expected that FortiGate Autoscale for AWS will be deployed by DevOps engineers or advanced system administrators who are familiar with the above.

Technical requirements

To start the deployment, you must have an AWS account. If you do not already have one, create one at https://aws.amazon.com/ by following the on-screen instructions. Part of the sign-up process involves receiving a phone call and entering a PIN. Your AWS account is automatically signed up for all AWS services. You are charged only for the services you use.

Log into your AWS account and verify the following:

- *IAM permissions.* Ensure that the AWS user deploying the template has sufficient permissions to perform the required service actions on resources. At a minimum, the following are required: *Service*: IAM; *Actions*:CreateRole; *Resource*: *. The FortiGate Autoscale for AWS template increases the security level of the deployment stack by narrowing down the scope of access to external resources belonging to the same user account as well as restricting access to resources within the deployment.
- *Region*. Use the region selector in the navigation bar to choose the AWS region where you want to deploy FortiGate Autoscale for AWS.

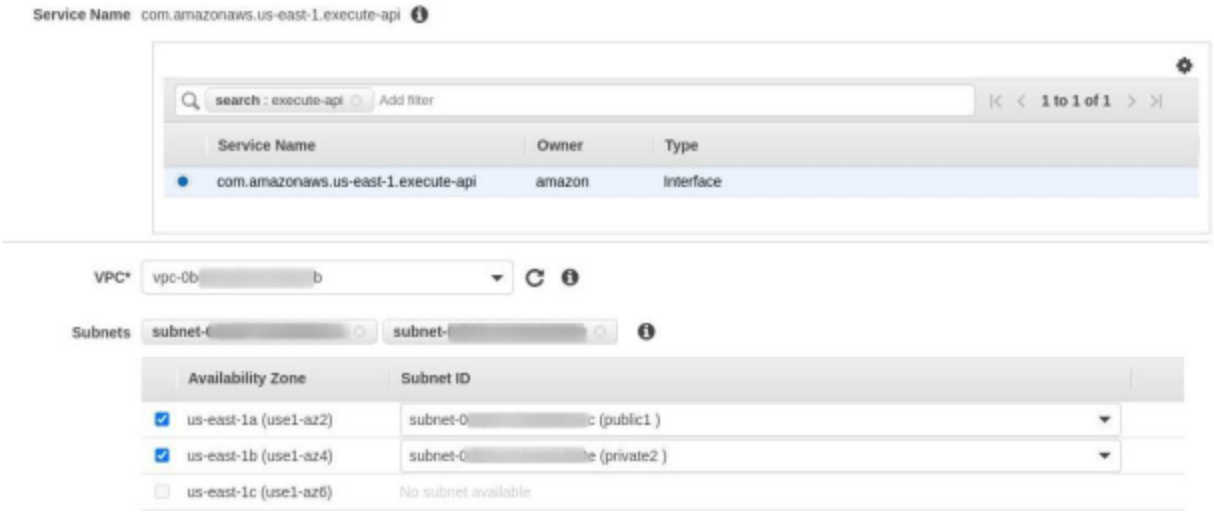
This deployment includes *AWS Auto Scaling*, which isn't currently supported in all AWS Regions. For a current list of supported Regions, refer to the AWS documentation Service Endpoints and Quotas.

- *Instance Type.* This deployment offers a range of instance types, some of which are not currently supported in all AWS Regions. Ensure that your desired instance type is available in your region by checking the Instance types page for your region.
- *FortiGate subscription(s).* Confirm that you have a valid subscription to the On-demand FortiGate and/or BYOL FortiGate marketplace listings, as required for your deployment.
  - If you are not subscribed, open the subscription page and click *Continue to Subscribe*.
  - Review the terms and conditions for software usage, and then choose *Accept Terms*. A confirmation page loads, and an email confirmation is sent to the account owner.
  - Exit out of AWS Marketplace without further action. Do not provision the software from AWS Marketplace.
- *Key pair.* Ensure at least one Amazon EC2 key pair exists in your AWS account in the region where you plan to deploy FortiGate Autoscale for AWS. Make note of the key pair name.
- *Resources.* If necessary, request service quota increases. This is necessary when you may exceed the default quotas with this deployment. The Service Quotas console displays your usage and quotas for some aspects of some services. For more information, see the AWSdocumentation. The default instance type is *c5.large*.
- *FortiGate licenses*. Ensure you have a license for each FortiGate BYOL instance you may use. Licenses can be purchased from FortiCare. In the section BYOL license files on page 44, you will place the license files in an S3 bucket for use by the deployment.

Requirements when using an existing VPC

When using an existing VPC, there are additional requirements:

- The VPC must have the *DNS hostnames* option enabled.
- Each availability zone in the VPC must have at least one public subnet and at least one private subnet.
- A VPC endpoint for the `execute-api` service under the *AWS services* category is required. This VPC endpoint must have the *Private DNS Name* option enabled and must be associated with the VPC:



After deployment, you must associate the created security group with the VPC endpoint. For details, see Post-deployment activities on page 62.

Obtaining the deployment package

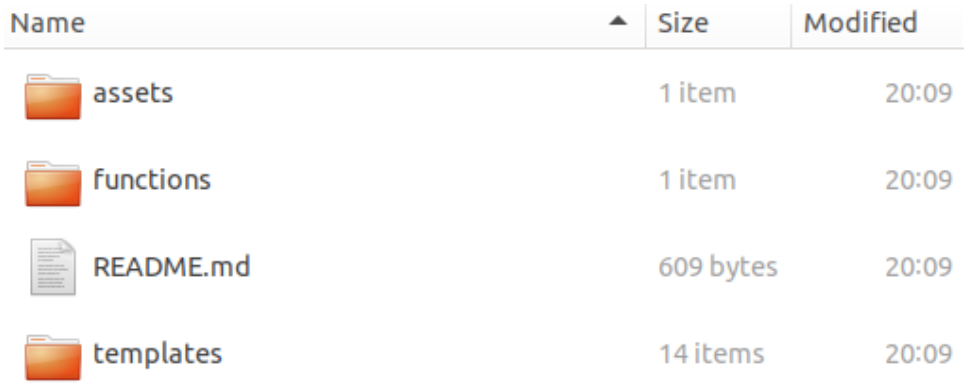
The FortiGate Autoscale for AWS deployment package is located in the Fortinet GitHub project.

To obtain the deployment package, use one of the following:

- Download the package *aws-cloudformation.zip* directly from the GitHub project release page.
- Manually generate the deployment package in your local workspace:
  a. From the GitHub project release page, download the source code (.zip or .tar.gz) for the latest version.
  b. Extract the source code into the project directory in your local workspace.
  c. Run `npm install` to initialize the project at the project root directory.
  d. Run `npm run build-artifacts` to generate the local deployment package.
  The deployment package *aws-cloudformation.zip* will be available in the *dist/artifacts* directory.

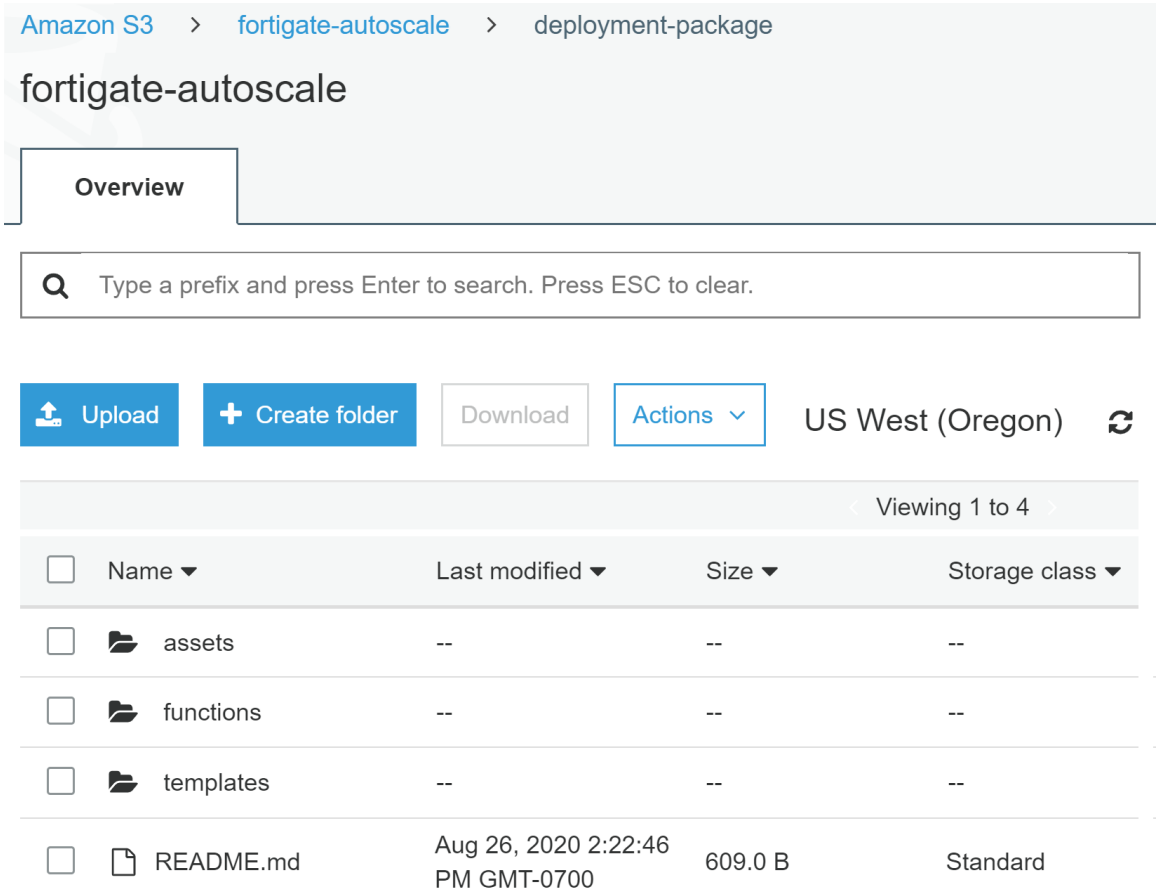
Once you have the deployment package *aws-cloudformation.zip*:

1. Unzip the file on your local PC. The following files and folders will be extracted:



2. Log into your AWS account.
3. In the Amazon S3 service, create an S3 bucket as the root folder for your deployment. In the example below, the folder is named *fortigate-autoscale*.
4. Inside this folder, create another folder to store the deployment resources. In the example below, this folder is named *deployment-package*.
5. Navigate to this second folder and upload the files and folders you extracted in step 2 to this location. In the example

below, we navigate to *Amazon S3* > *fortigate-autoscale* > *deployment-package*.



This *assets* folder contains configuration files that can be modified as needed to meet your network requirements. For details, refer to the Appendix > Major components on page 81 >*The "assets" folder in the S3 bucket*.

BYOL license files

If you will be using BYOL instances, the deployment package will look for FortiGate license files in a location that ends with *license-files* > *fortigate*. This location can created within the *assets* folder of the deployment package location or within a custom asset location.

If a custom asset location is used, you must specify the location in the parameters described in the table Custom asset location configuration on page 58.

Examples:

- If the deployment package is located at *Amazon S3* > *fortigate-autoscale* > *deployment-package*, license files would be uploaded to *Amazon S3* > *fortigate-autoscale* > *deployment-package* > *assets*> *license-files* > *fortigate*.

- If you will be storing license files in a custom S3 location and you have created the S3 bucket *custom-s3-bucket-name* with the directory *custom-asset-directory*, you would upload the license files to *Amazon S3* > *custom-s3-bucket-name* > *custom-asset-directory* > *license-files* > *fortigate*.

Deploying the CloudFormation templates

FortiGate Autoscale for AWS can be deployed:

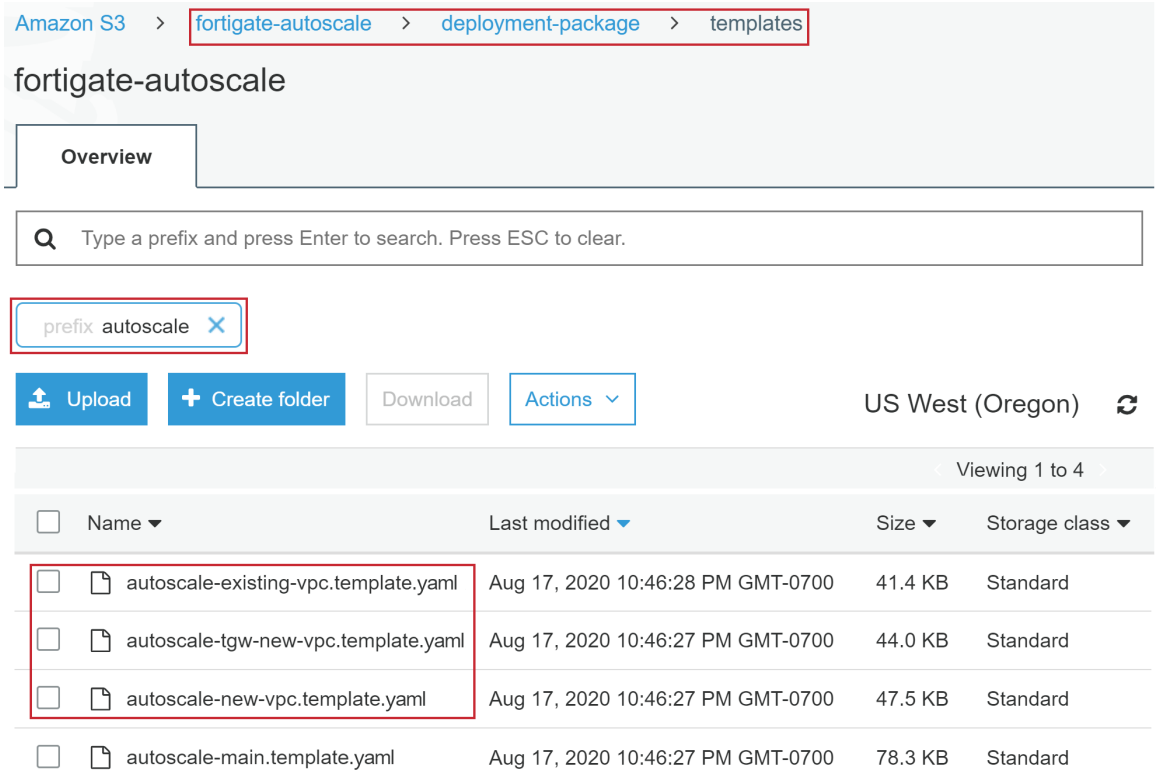
- *with Transit Gateway integration* (using a new Transit Gateway or integrating with your existing Transit Gateway). This option builds a new AWS environment consisting of the VPC, subnets, security groups, and other infrastructure components. It then deploys FortiGate Autoscale into this new VPC and attaches this new VPC to the Transit Gateway.
- *without Transit Gateway integration*. This option allows for deployment into a new VPC or into an existing VPC.

Deployment notes

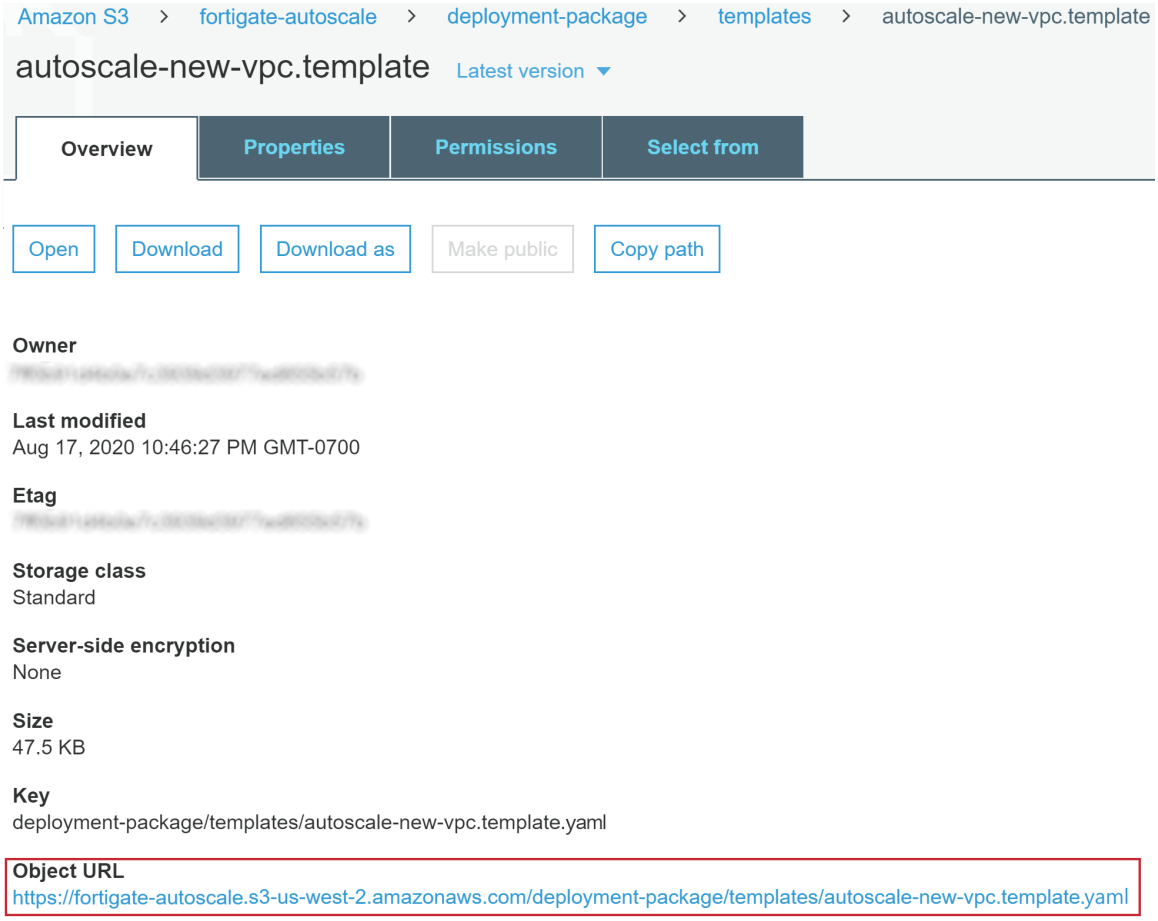
Deployment option	Notes
with Transit Gateway integration (new VPC only)	One inbound route domain and one outbound route domain will be created for the new or existing Transit Gateway. FortiGate Autoscale for AWS will be attached to the Transit Gateway.
into an existing VPC	• Incoming requests go through a connection that flows through the internet gateway, Network Load Balancer, and FortiGate Auto Scaling group before reaching the protected instances in the private subnets in your existing VPC. The protected instances return a response using the same connection. • Outgoing requests from the protected instances go through one FortiGate-VM instance in an Auto Scaling group and the internet gateway to the public network. The public network returns the response using the same path. •  FortiGate Autoscale will manage the `0.0.0.0/0` route for overall egress traffic. For details on using other NAT gateways refer to the section How to partially route egress traffic on page 83.

To deploy the CloudFormation templates:

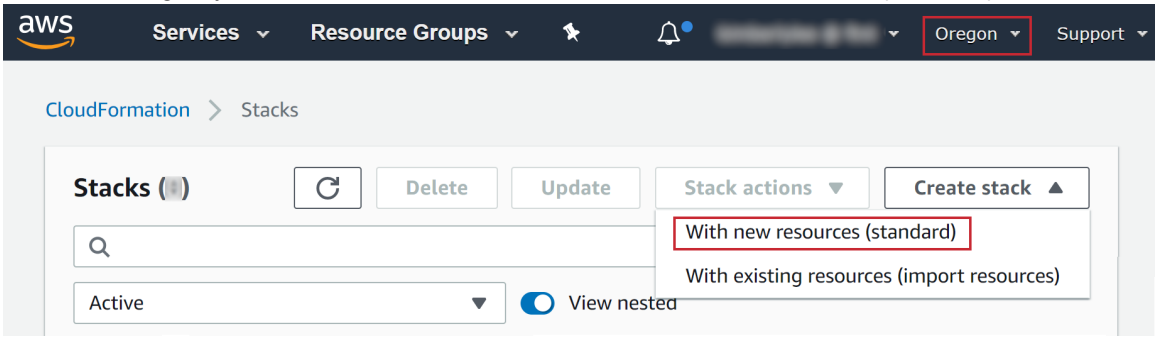
1. Navigate to the S3 folder you uploaded files to in the previous section. In the example below, we navigate to *Amazon S3* > *fortigate-autoscale* > *deployment-package*.
2. Click *templates* and select the appropriate entry template to start the deployment. To deploy:
   - with Transit Gateway integration, click `autoscale-tgw-new-vpc.template.yaml`
   - without Transit Gateway integration, click `autoscale-new-vpc.template.yaml` to deploy into a new VPC
   - without Transit Gateway integration, click `autoscale-existing-vpc.template.yaml` to deploy into an existing VPC


Amazon S3 > fortigate-autoscale > deployment-package > templates
fortigate-autoscale
Overview
Type a prefix and press Enter to search. Press ESC to clear.
prefix autoscale
Upload
Create folder
Download
Actions
US West (Oregon)
Viewing 1 to 4
Name
Last modified
Size
Storage class
autoscale-existing-vpc.template.yaml
Aug 17, 2020 10:46:28 PM GMT-0700
41.4 KB
Standard
autoscale-tgw-new-vpc.template.yaml
Aug 17, 2020 10:46:27 PM GMT-0700
44.0 KB
Standard
autoscale-new-vpc.template.yaml
Aug 17, 2020 10:46:27 PM GMT-0700
47.5 KB
Standard
autoscale-main.template.yaml
Aug 17, 2020 10:46:27 PM GMT-0700
78.3 KB
Standard

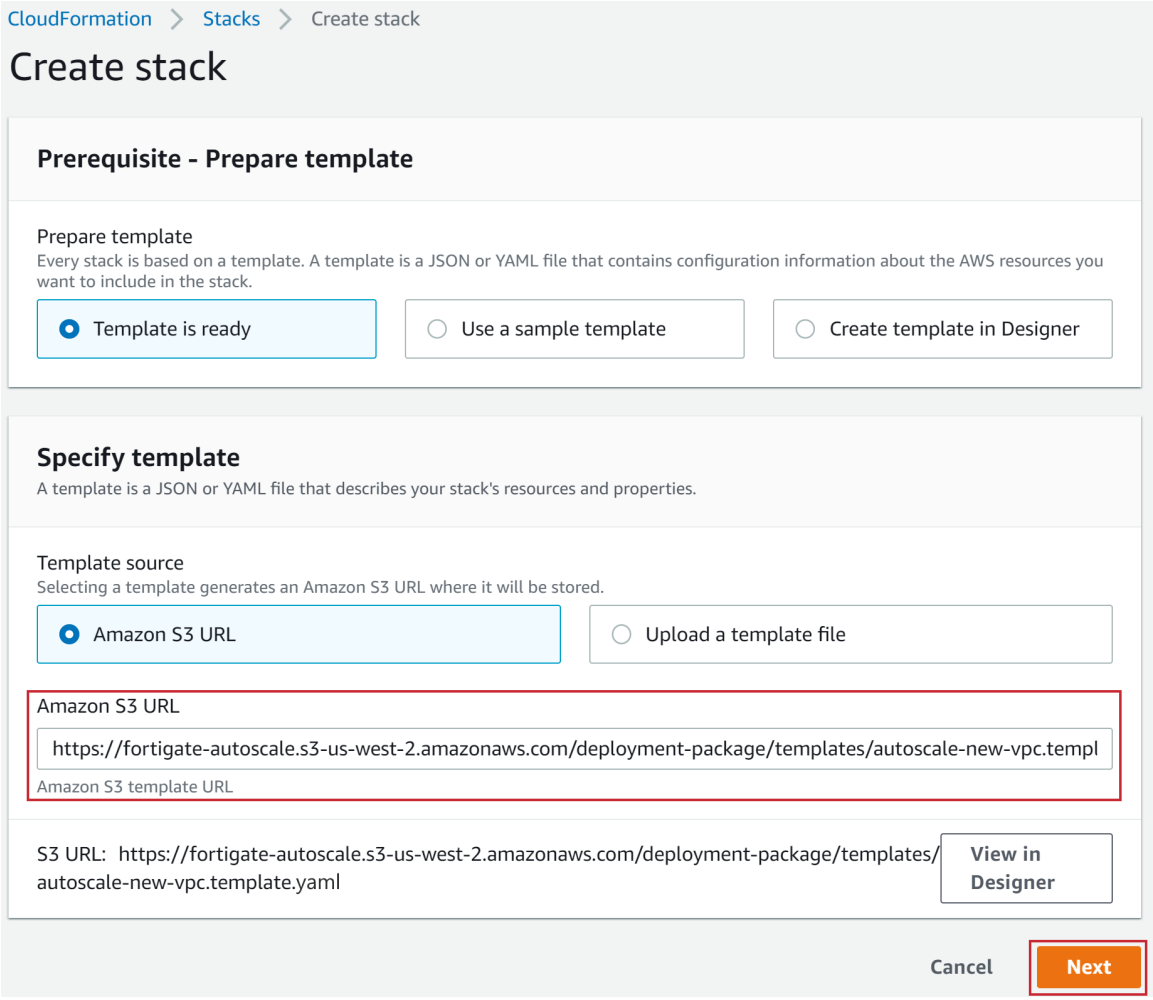
3. Copy the *Object URL* of the template you picked in the previous step. In our example, the template chosen is for deploying into a new VPC.



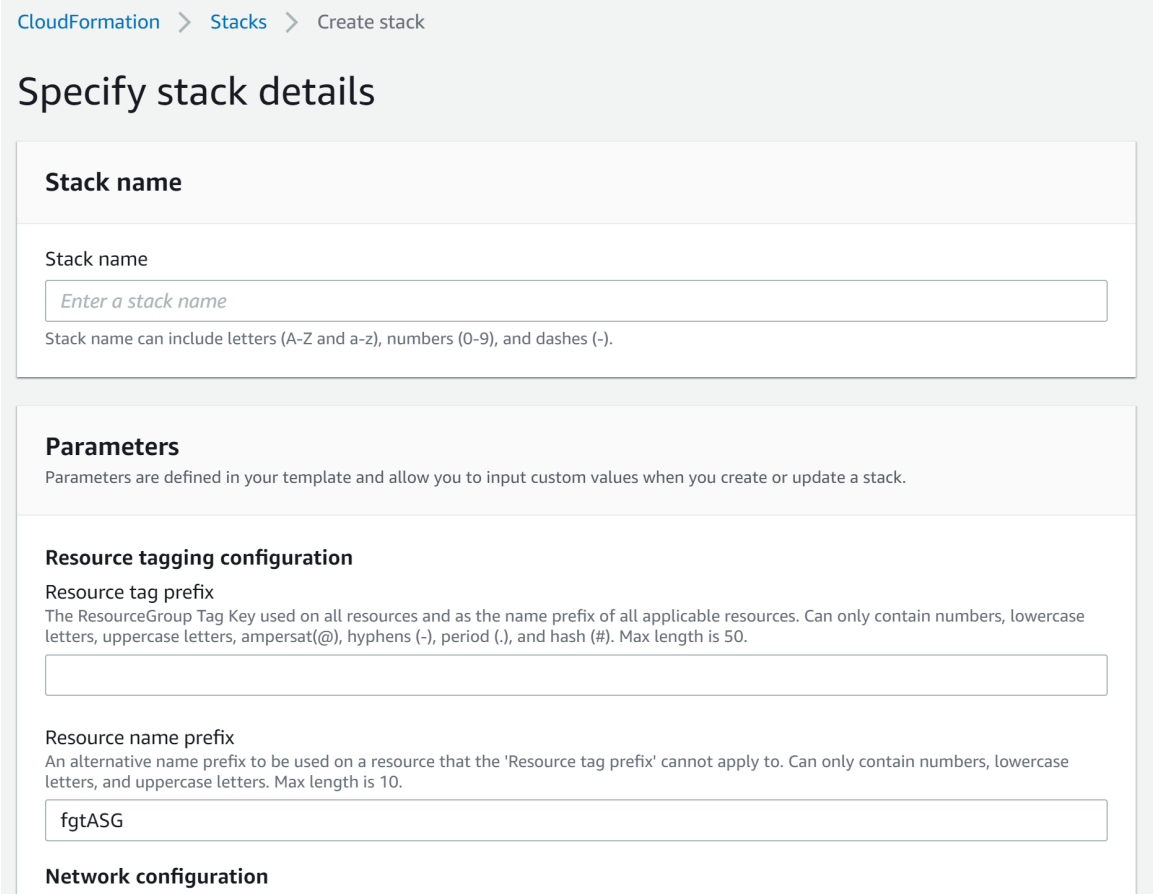
4. Click *Services*, and then *Management & Governance* > *CloudFormation*.
5. Confirm the region you are in and then click *Create Stack* > *With new resources (standard)*.



6. Paste the *Object URL* from step 3 into the *Amazon S3 URL* field as shown below.



7. Click *Next*.
8. On the *Specify stack details* page, enter a stack name and review parameters for the template, providing values for parameters that require input.



CFT parameters

The following sections provide descriptions of the available parameters. Some parameters are specific to certain templates, and are only displayed when that template is selected.

After entering required parameters, click *Next*.

Resource tagging configuration

Parameter label (`name`)	Default	Description
Resource tag prefix (`ResourceTagPrefix`)	*Requires input*	*ResourceGroup* Tag Key used on all resources and as the name prefix of all applicable resources. Can only contain uppercase letters, lowercase letters, numbers, ampersand (@), hyphens (-), period (.), and hash (#). Maximum length is 50.
Resource name prefix (`CustomIdentifier`)	`fgtASG`	Alternative name prefix to be used on a resource that the *Resource tag prefix* cannot apply to. Can only contain uppercase letters, lowercase letters, and numbers. Maximum length is 10.

Network configuration (New VPC, no Transit Gateway)

Parameter label (name)	Default	Description
Availability Zones (AvailabilityZones)	*Requires input*	List of Availability Zones to use for the subnets in the VPC. The FortiGate Autoscale solution uses two Availability Zones from your list and preserves the logical order you specify.
VPC CIDR (VpcCidr)	192.168.0.0/16	Classless Inter-Domain Routing (CIDR) block for the FortiGate Autoscale VPC.
Autoscale subnet 1 CIDR (PublicSubnet1Cidr)	192.168.0.0/24	CIDR block for the subnet located in Availability Zone 1 where FortiGate Autoscale instances will be deployed to.
Autoscale subnet 2 CIDR (PublicSubnet2Cidr)	192.168.1.0/24	CIDR block for the subnet located in Availability Zone 2 where FortiGate Autoscale instances will be deployed to.
Protected subnet 1 CIDR (PrivateSubnet1Cidr)	192.168.2.0/24	CIDR block for the private subnet located in Availability Zone 1 where it is protected by the FortiGate-VMs in the public subnet of the same Availability Zone.
Protected subnet 2 CIDR (PrivateSubnet2Cidr)	192.168.3.0/24	CIDR block for the private subnet located in Availability Zone 2 where it is protected by the FortiGate-VMs in the public subnet of the same Availability Zone.

Network configuration (Existing VPC, no Transit Gateway)

Parameter label (name)	Default	Description
VPC ID (VpcId)	*Requires input*	ID of the existing VPC where FortiGate Autoscale will be deployed. The VPC must have the option *DNS hostnames* enabled and each of the two Availability Zones in the VPC must have at least 1 public subnet and at least 1 private subnet.
VPC CIDR (VpcCidr)	*Requires input*	CIDR block of the selected existing VPC into which FortiGate Autoscale will be deployed. This can be found in parentheses in the VPC ID parameter selection.
Private VPC Endpoint (VpcEndpointId)	*Requires input*	ID of the Private VPC Endpoint associated with the existing VPC. A Private VPC Endpoint is required for FortiGate Autoscale and is a VPC Endpoint that has enabled *Private DNS names*.
Autoscale subnet 1 ID (PublicSubnet1)	*Requires input*	ID of the public subnet 1 located in Availability Zone 1 of the selected existing VPC. The FortiGate Autoscale instances will be deployed here.
Autoscale subnet 2 ID (PublicSubnet2)	*Requires input*	ID of the public subnet 2 located in Availability Zone 2 of the selected existing VPC. The FortiGate Autoscale instances will be deployed here.

Parameter label (name)	Default	Description
Private subnet 1 ID (PrivateSubnet1)	*Requires input*	ID of the private subnet 1 located in Availability Zone 1 of the selected existing VPC. This subnet will be protected by the FortiGate-VMs in the public subnet of the same Availability Zone.
Private subnet 2 ID (PrivateSubnet2)	*Requires input*	ID of the private subnet 2 located in Availability Zone 2 of the selected existing VPC. This subnet will be protected by the FortiGate-VMs in the public subnet of the same Availability Zone.
Private subnet route table (PrivateSubnetRouteTable)	*Requires input*	ID of the route table associated with the two private subnets.

Network configuration (Transit Gateway integration)

Parameter label (name)	Default	Description
Availability Zones (AvailabilityZones)	*Requires input*	The list of Availability Zones to use for the subnets in the VPC. The FortiGate Autoscale solution uses two Availability Zones from your list and preserves the logical order you specify.
VPC CIDR (VpcCidr)	192.168.0.0/16	The Classless Inter-Domain Routing (CIDR) block for the FortiGate Autoscale VPC.
Autoscale subnet 1 CIDR (PublicSubnet1Cidr)	192.168.0.0/24	The CIDR block for the subnet located in Availability Zone 1 where FortiGate Autoscale instances will be deployed to.
Autoscale subnet 2 CIDR (PublicSubnet2Cidr)	192.168.1.0/24	The CIDR block for the subnet located in Availability Zone 2 where FortiGate Autoscale instances will be deployed to.

FortiGate configuration

Parameter label (name)	Default	Description
Instance type (FortiGateInstanceType)	c5.large	Instance type for the FortiGate-VMs in the Auto Scaling group. There are t2.small and compute-optimized instances such as c4 and c5 available with different vCPU sizes and bandwidths. For more information about instance types, see Instance Types.
FortiOS version (FortiOSVersion)	7.0.1	FortiOS version supported by FortiGate Autoscale for AWS.  Requires one or more subscriptions to Fortinet FortiGate on-demand or BYOL AMIs.
FortiGate PSK secret (FortiGatePskSecret)	*Requires input*	Secret preshared key used by the FortiGate-VM instances to securely communicate with each other. Must contain numbers and letters and may contain special characters. Maximum length is 128.

Parameter label (name)	Default	Description
		 Changes to the PSK secret after FortiGate Autoscale for AWS has been deployed are not reflected here. For new instances to be spawned with the changed PSK secret, this environment variable will need to be manually updated.
Admin port (FortiGateAdminPort)	8443	A port number for FortiGate administration. Minimum is 1. Maximum is 65535. Do not use the FortiGate reserved ports 443, 541, 514, or 703.
Admin CIDR block (FortiGateAdminCidr)	*Requires input*	CIDR block for external administrator management access.  0.0.0.0/0 accepts connections from any IP address. Use a constrained CIDR range to reduce the potential of inbound attacks from unknown IP addresses.
Key pair name (KeyPairName)	*Requires input*	Amazon EC2 Key Pair for admin access.
BGP ASN (BgpAsn)	65000	The Border Gateway Protocol (BGP) Autonomous System Number (ASN) of the Customer Gateway of each FortiGate-VM instance in the Auto Scaling group. This value ranges from 64512 to 65534.  Only for deployments with Transit Gateway integration.

FortiGate Auto Scaling group configuration

Parameter label (name)	Default	Description
Desired capacity (BYOL) (FgtAsgDesiredCapacityByol)	2	Number of FortiGate-VM instances the BYOL Auto Scaling group should have at any time. For High Availability in BYOL-only and Hybrid use cases, ensure at least 2 FortiGate-VMs are in the group. For specific use cases, set to 0 for On-demand-only, and >= 2 for BYOL-only or hybrid licensing.
Minimum group size (BYOL) (FgtAsgMinSizeByol)	2	Minimum number of FortiGate-VM instances in the BYOL Auto Scaling group. For specific use cases, set to 0 for On-demand-only, and >= 2 for BYOL-only or hybrid licensing.

Parameter label (`name`)	Default	Description
		 For BYOL-only and hybrid licensing deployments, this parameter must be at least 2. If it is set to 1 and the instance fails to work, the current FortiGate-VM configuration will be lost.
Maximum group size (BYOL) (`FgtAsgMaxSizeByol`)	`2`	Maximum number of FortiGate-VM instances in the BYOL Auto Scaling group. For specific use cases, set to 0 for On-demand-only, and >= 2 for BYOL-only or hybrid licensing. This number must be greater than or equal to the *Minimum group size (BYOL)*.
Desired capacity (On-demand instances) (`FgtAsgDesiredCapacityPayg`)	`0`	Number of FortiGate-VM instances the On-demand Auto Scaling group should have at any time. For High Availability in an On-demand-only use case, ensure at least 2 FortiGate-VMs are in the group. For specific use cases, set to 0 for BYOL-only, >= 2 for On-demand-only, and >= 0 for hybrid licensing.
Minimum group size (On-demand instances) (`FgtAsgMinSizePayg`)	`0`	Minimum number of FortiGate-VM instances in the On-demand Auto Scaling group. For specific use cases, set to 0 for BYOL-only, >= 2 for On-demand-only, and >= 0 for hybrid licensing.  For On-demand-only deployments, this parameter must be at least 2. If it is set to 1 and the instance fails to work, the current FortiGate-VM configuration will be lost.
Maximum group size (On-demand instances) (`FgtAsgMaxSizePayg`)	`0`	Maximum number of FortiGate-VM instances in the On-demand Auto Scaling group. For specific use cases, set to 0 for BYOL-only, >= 2 for On-demand-only, and >= 0 for hybrid licensing. This number must be greater than or equal to the *Minimum group size (On-demand instances)*.
Scale-out threshold (`FgtAsgScaleOutThreshold`)	`80`	Threshold (in percentage) for the FortiGate Auto Scaling group to scale out (add) 1 instance. Minimum is 1. Maximum is 100.
Scale-in threshold (`FgtAsgScaleInThreshold`)	`25`	Threshold (in percentage) for the FortiGate Auto Scaling group to scale in (remove) 1 instance. Minimum is 1. Maximum is 100.
Primary election timeout (`PrimaryElectionTimeout`)	`300`	Maximum time (in seconds) to wait for the election of the primary instance to complete. Minimum is 30. Maximum is 3600.

Parameter label (name)	Default	Description
Get license grace period (GetLicenseGracePeriod)	600	Minimum time (in seconds) permitted before a distributed license can be revoked from a non-responsive FortiGate-VM and re-distributed. Minimum is 300.
Health check grace period (FgtAsgHealthCheckGracePeriod)	300	Length of time (in seconds) that Auto Scaling waits before checking an instance's health status. Minimum is 60.
Scaling cooldown period (FgtAsgCooldown)	300	Auto Scaling group waits for the cooldown period (in seconds) to complete before resuming scaling activities. Minimum is 60. Maximum is 3600.
Instance lifecycle timeout (LifecycleHookTimeout)	480	Amount of time (in seconds) that can elapse before the FortiGate Autoscale lifecycle hook times out. Minimum is 60. Maximum is 3600.

Transit Gateway configuration (Transit Gateway integration)

Parameter label (name)	Default	Description
Transit Gateway support (TransitGatewaySupportOptions)	create one	Create a Transit Gateway for the FortiGate Autoscale VPC to attach to, or specify to use an existing one.
Transit Gateway ID (TransitGatewayId)	*Conditionally requires input*	ID of the Transit Gateway that the FortiGate Autoscale VPC will be attached to. Required when *Transit Gateway support* is set to "use an existing one".

Load balancing configuration (no Transit Gateway integration)

Parameter label (name)	Default	Description
Traffic protocol (LoadBalancingTrafficProtocol)	HTTPS	Protocol used to load balance traffic.
Traffic port (LoadBalancingTrafficPort)	443	Port number used to balance web service traffic if the internal web service load balancer is enabled. Minimum is 1. Maximum is 65535.
Health check threshold (LoadBalancingHealthCheckThreshold)	3	Number of consecutive health check failures required before considering a FortiGate-VM instance unhealthy. Minimum 3.
Internal ELB options (InternalLoadBalancingOptions)	add a new internal load balancer	(Optional) Predefined Elastic Load Balancer (ELB) to route traffic to web service in the private subnets. You can optionally use your own one or decide to not need one.

Parameter label (name)	Default	Description
Health check path (InternalTargetGroupHealthCheckPath)	/	(Optional) Destination path for health checks. This path must begin with a forward slash (/) and can be at most 1024 characters in length.
Internal ELB DNS name (InternalLoadBalancerDnsName)	*Requires input*	(Optional) DNS name of an existing internal load balancer used to route traffic from a FortiGate-VM to targets in a specified target group. Leave it blank if you don't use an existing load balancer.

Failover management configuration

Parameter label (name)	Default	Description
Heart beat interval (HeartBeatInterval)	30	Length of time (in seconds) that a FortiGate-VM instance waits between sending heartbeat requests to the Autoscale handler. Minimum is 30. Maximum is 90.
Heart beat loss count (HeartBeatLossCount)	10	Number of consecutively lost heartbeats. When the Heartbeat loss count has been reached, the FortiGate-VM is deemed unhealthy and failover activities will commence.
Heart beat delay allowance (HeartBeatDelayAllowance)	2	Maximum amount of time (in seconds) allowed for network latency of the FortiGate-VM heartbeat arriving at the FortiGate Autoscale handler. Minimum is 0.
Autoscale notifications subscriber email (AutoscaleNotificationSubscriberEmail)	-	The email address (AWS SNS Topic subscriber) to receive Autoscale notifications. If provided, the template can only accept one email address. An email will be sent to the address to confirm the subscription.
Terminate unhealthy VM (TerminateUnhealthyVm)	no	Set to *yes* to terminate any VM that is deemed unhealthy by FortiGate Autoscale.
Autoscale sync recovery count (SyncRecoveryCount)	3	Number of consecutive on-time heartbeats required for a VM to become healthy again. This parameter is only used when *Terminate unhealthy VM* is set to no and allows for the VM to recover from an unhealthy state.

FortiAnalyzer integration

Parameter label (`name`)	Default	Description
FortiAnalyzer integration (`FortiAnalyzerIntegrationOptions`)	`yes`	Set to *no* if you do not want to incorporate FortiAnalyzer into FortiGate Autoscale to use extended features that include storing logs into FortiAnalyzer.
FortiAnalyzer version (`FortiAnalyzerVersion`)	`6.4.6`	FortiAnalyzer version supported by FortiGate Autoscale.  Requires a subscription to the "Fortinet FortiAnalyzer Centralized Logging/Reporting (10 managed devices)" AMI.
FortiAnalyzer instance type (`FortiAnalyzerInstanceType`)	`m5.large`	Instance type to launch as FortiAnalyzer on-demand instances. There are compute-optimized instances, such as m4 and c4, available with different vCPU sizes and bandwidths. For more information about instance types, see Instance Types.
Autoscale admin user name (`FortiAnalyzerAutoscaleAdminUsername`)	*Requires input*	Name of the secondary administrator-level account in the FortiAnalyzer, which FortiGate Autoscale uses to connect to the FortiAnalyzer to authorize any FortiGate device in the Auto Scaling group. To conform to the FortiAnalyzer naming policy, the user name can only contain numbers, lowercase letters, uppercase letters, and hyphens. It cannot start or end with a hyphen (-).
Autoscale admin password (`FortiAnalyzerAutoscaleAdminPassword`)	*Requires input*	Password for the "Autoscale admin user name." The password must conform to the FortiAnalyzer password policy and have a minimum length of 8 and a maximum length of 128. If you need to enable KMS encryption, refer to the documentation.
FortiAnalyzer private IP address (`FortiAnalyzerCustomPrivateIpAddress`)	*Requires input*	Custom private IP address to be used by the FortiAnalyzer. Must be within the public subnet 1 CIDR range. Required if "FortiAnalyzer integration" is set to *yes*. If "FortiAnalyzer integration" is set to *no*, any input will be ignored.

Custom asset location configuration

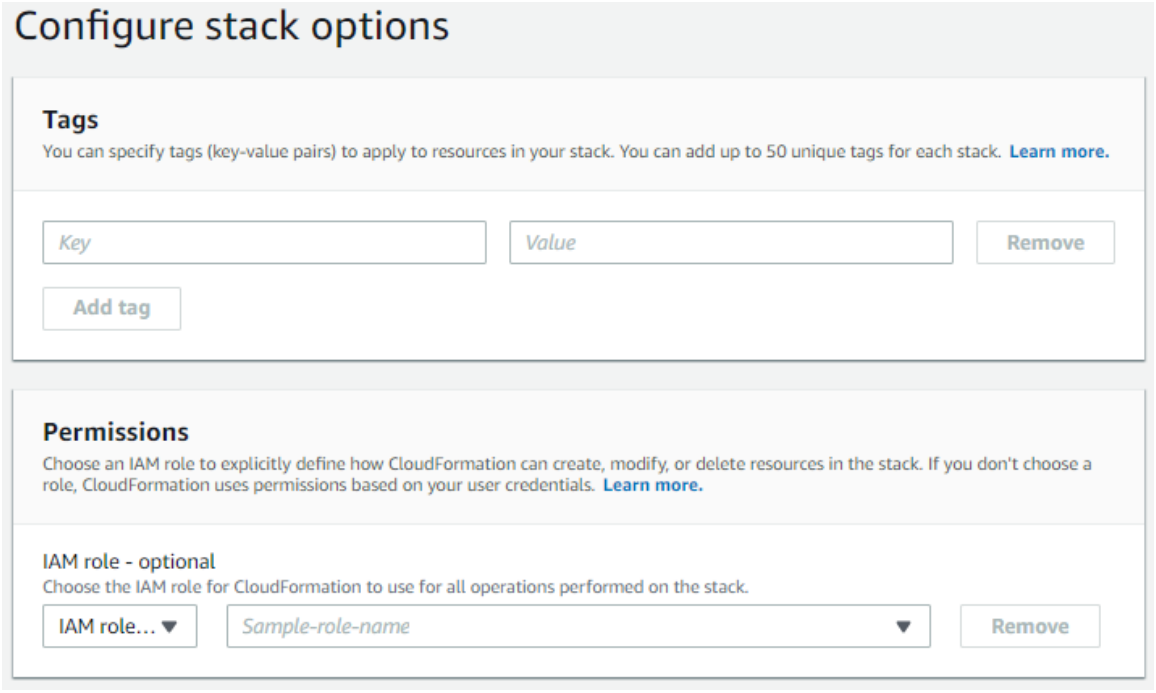
Parameter label (name)	Default	Description
Use custom asset location (UseCustomAssetLocation)	no	Set to *yes* to use a custom S3 location for custom assets such as licenses and customized configsets.
Custom asset S3 bucket (CustomAssetContainer)	*Requires input*	Name of the S3 bucket that contains your custom assets. Required if 'Use custom asset location' is set to *yes*. Can only contain numbers, lowercase letters, uppercase letters, and hyphens (-). It cannot start or end with a hyphen (-).
Custom asset folder (CustomAssetDirectory)	*Requires input*	The sub path within the 'custom asset container' that serves as the top level directory of all your custom assets. If 'Use custom asset location' is set to *yes*, and this value is left empty, the 'custom asset container' will serve as the top level directory. Can only contain numbers, lowercase letters, uppercase letters, hyphens (-), and forward slashes (/). If provided, it must end with a forward slash (/).

Deployment resources configuration

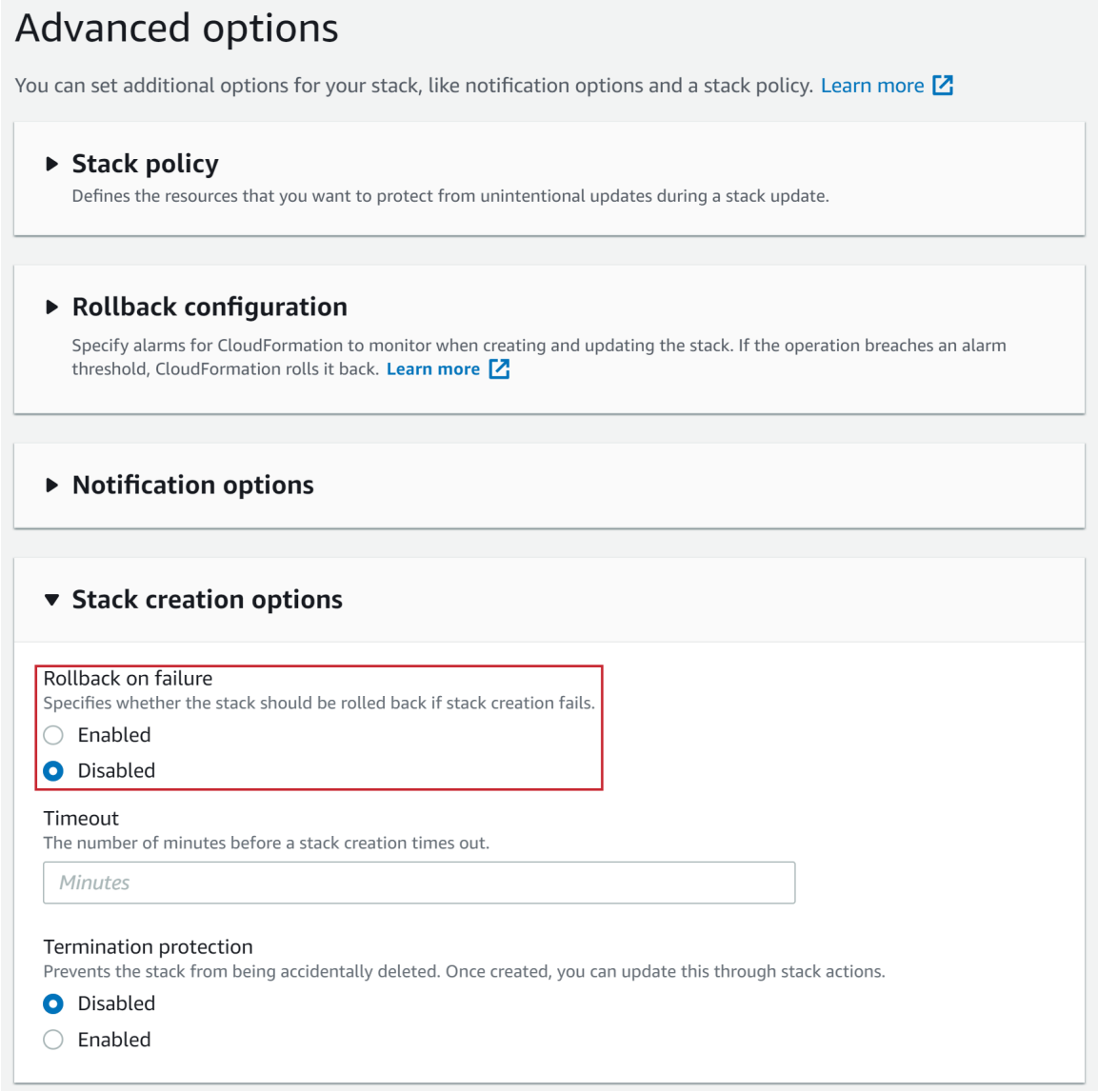
Parameter label (name)	Default	Description
S3 bucket name (S3BucketName)	*Requires input*	Name of the S3 bucket (created in step 4 of Obtaining the deployment package on page 43) that contains the FortiGate Autoscale deployment package. Can only contain numbers, lowercase letters, uppercase letters, and hyphens (-). It cannot start or end with a hyphen (-).
S3 resource folder (S3KeyPrefix)	*Requires input*	Name of the S3 folder (created in step 5 of Obtaining the deployment package on page 43) that stores the FortiGate Autoscale deployment resources. Can only contain numbers, lowercase letters, uppercase letters, hyphens (-), and forward slashes (/). If provided, it must end with a forward slash (/).

Optional settings

Optional settings are configured on the *Configure stack options* page:



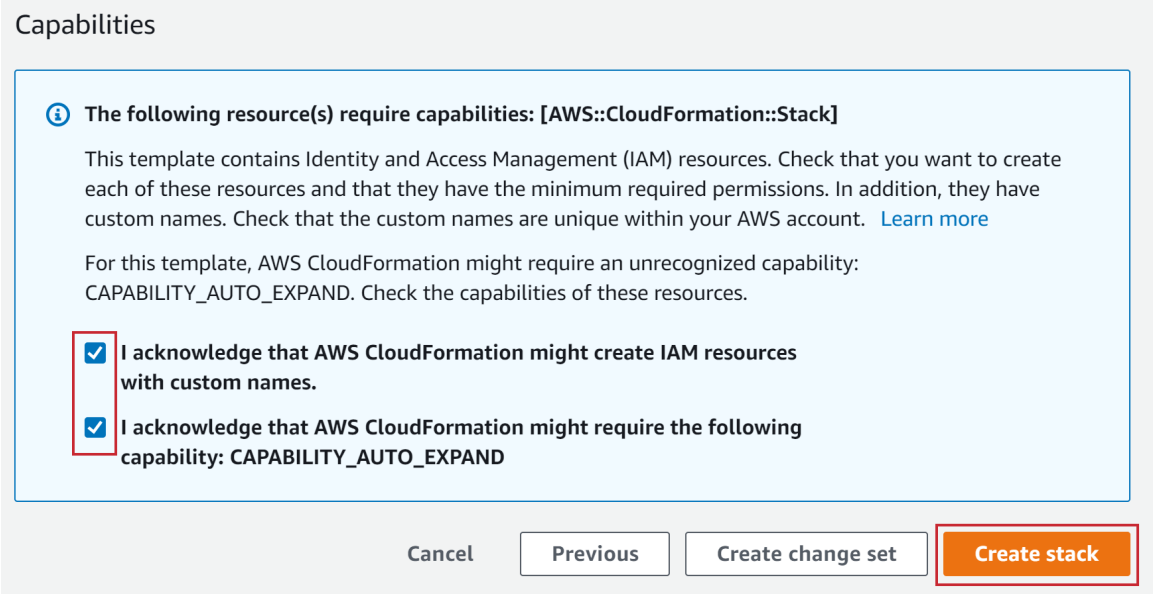
1. Specify *Tags* and *Permissions* as desired:
   a. *Tags*: Key-Value pairs for resources in your stack.
   b. *Permissions*: An IAM role that AWS CloudFormation uses to create, modify, or delete resources in your stack.
2. Under *Advanced options*, it is recommended that you disable the Stack creation option *Rollback on failure* to allow for a better troubleshooting experience.



3. Other advanced options can be specified as desired.
4. When done, click *Next*.

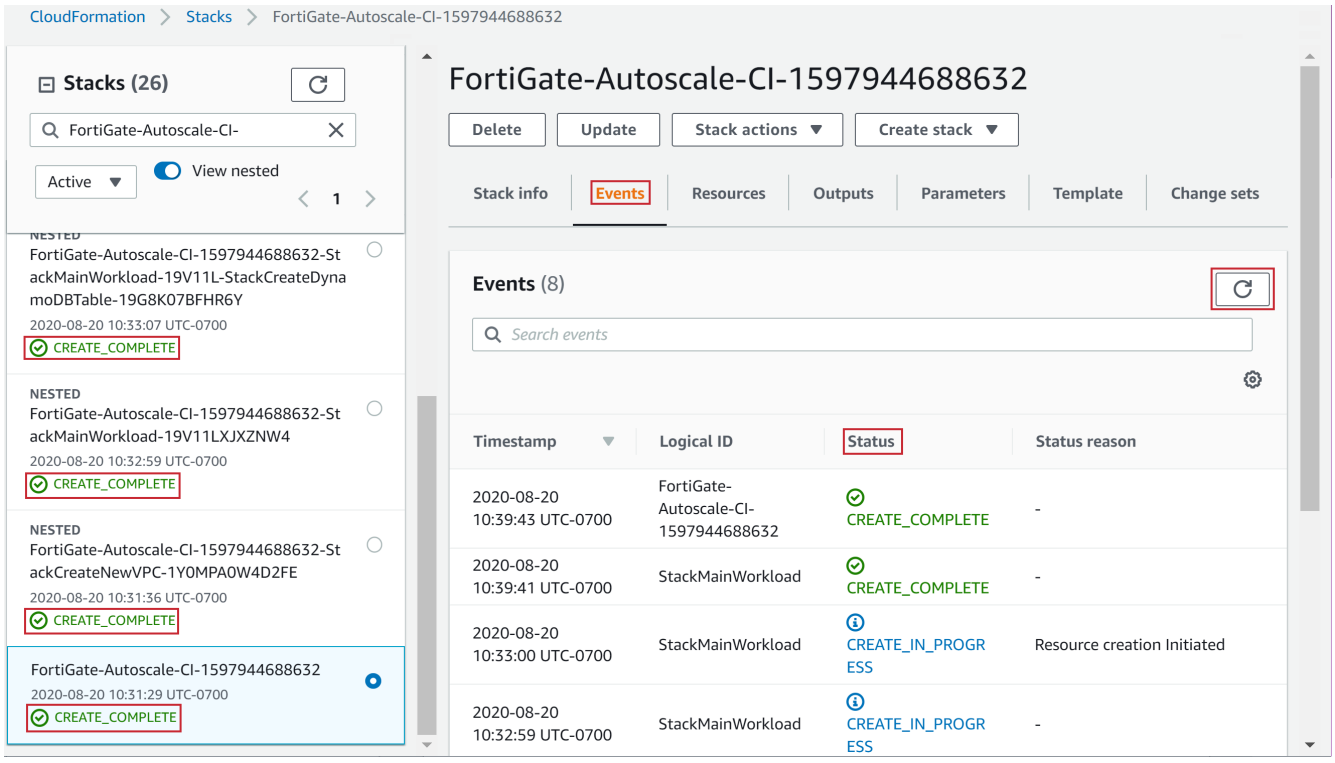
Completing the deployment

On the *Review* page, review and confirm the template, the stack details, and the stack options. Under *Capabilities*, select both checkboxes to acknowledge that the template creates IAM resources and may require the ability to automatically expand macros.



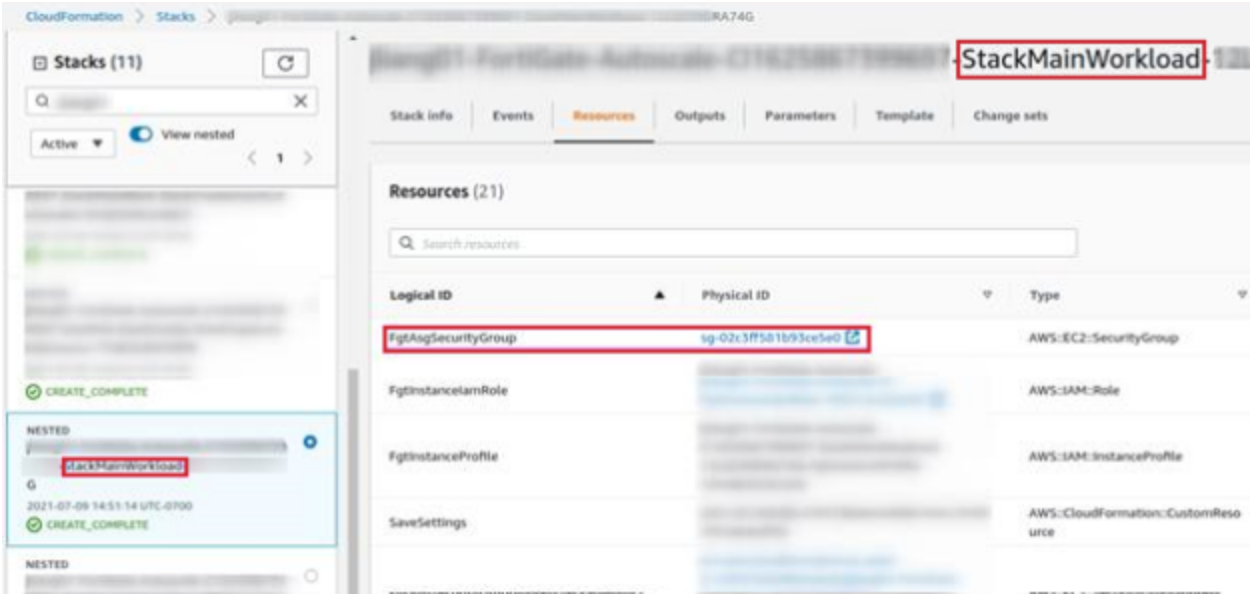
Click *Create stack* to deploy the stack.

The *Status* column shows creation status. To see the latest status, refresh the view. Creating the stacks takes about ten minutes. Deployment has completed when each stack, including the main stack and all nested stacks, has a status of *CREATE_COMPLETE*.

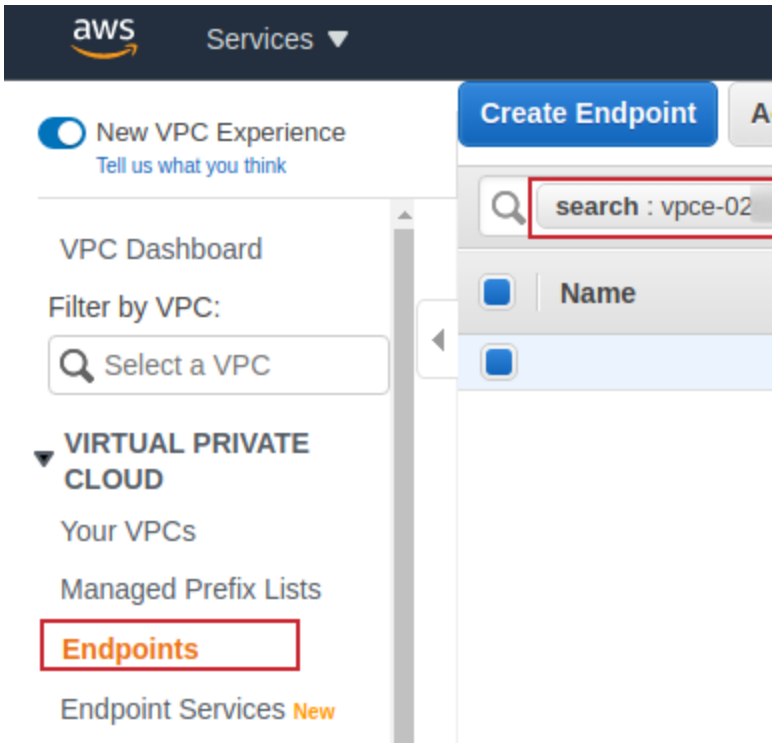


Post-deployment activities

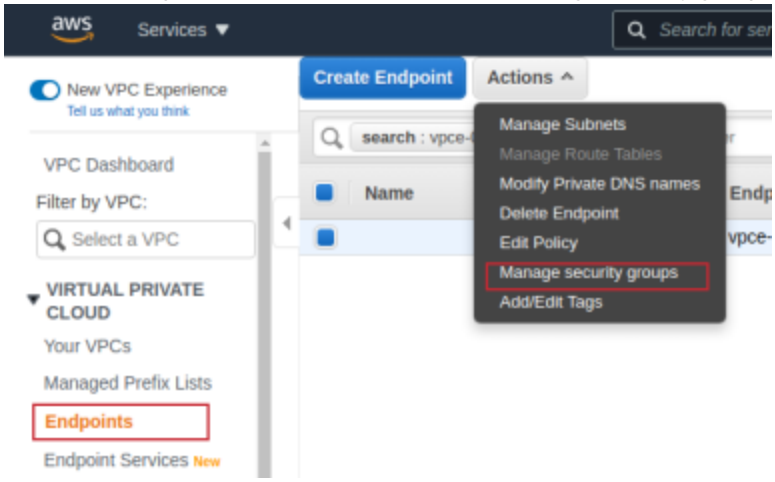
If you deployed into an existing VPC, locate and select *StackMainWorkload* from the left column. Note the *Physical ID* for the *Logical ID* `FgtAsgSEcurityGroup`. You must associate this security group with the Private VPC Endpoint of your existing VPC.



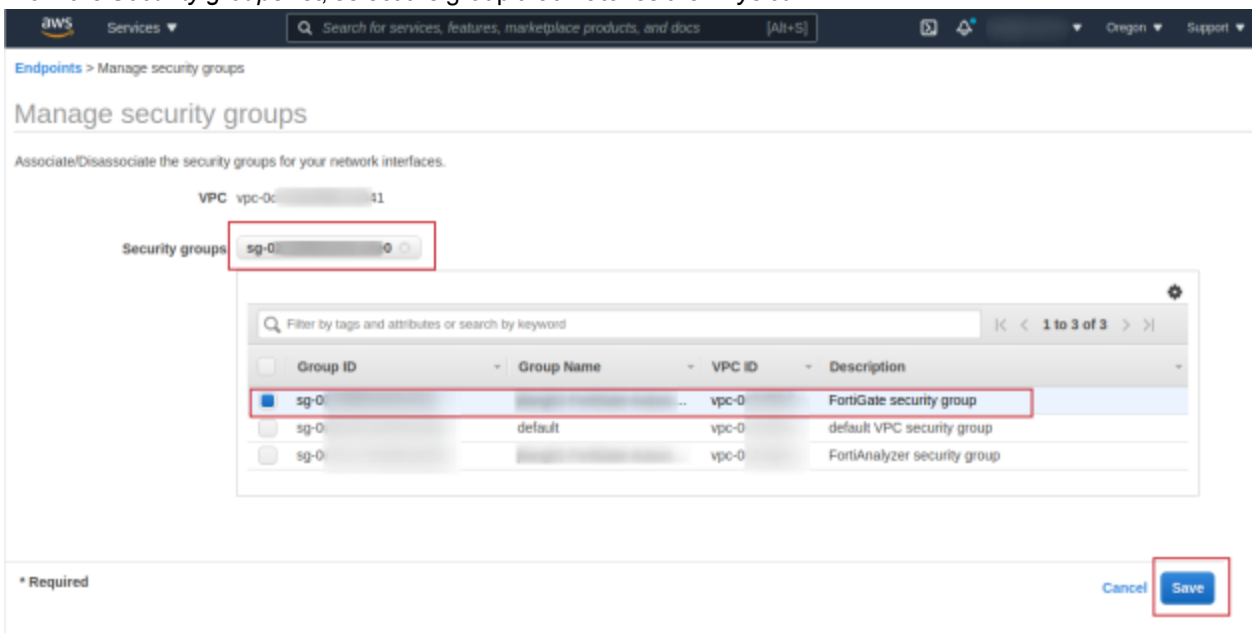
1. In the AWS console, select *Services* > *Network & Content Delivery* > *VPC*.
2. In the left navigation tree, click *Endpoints*.
3. Click the filter box and search for the VPC Endpoint created in Requirements when using an existing VPC on page 42.



4. Select the endpoint and under *Actions*, select *Manage security groups*.



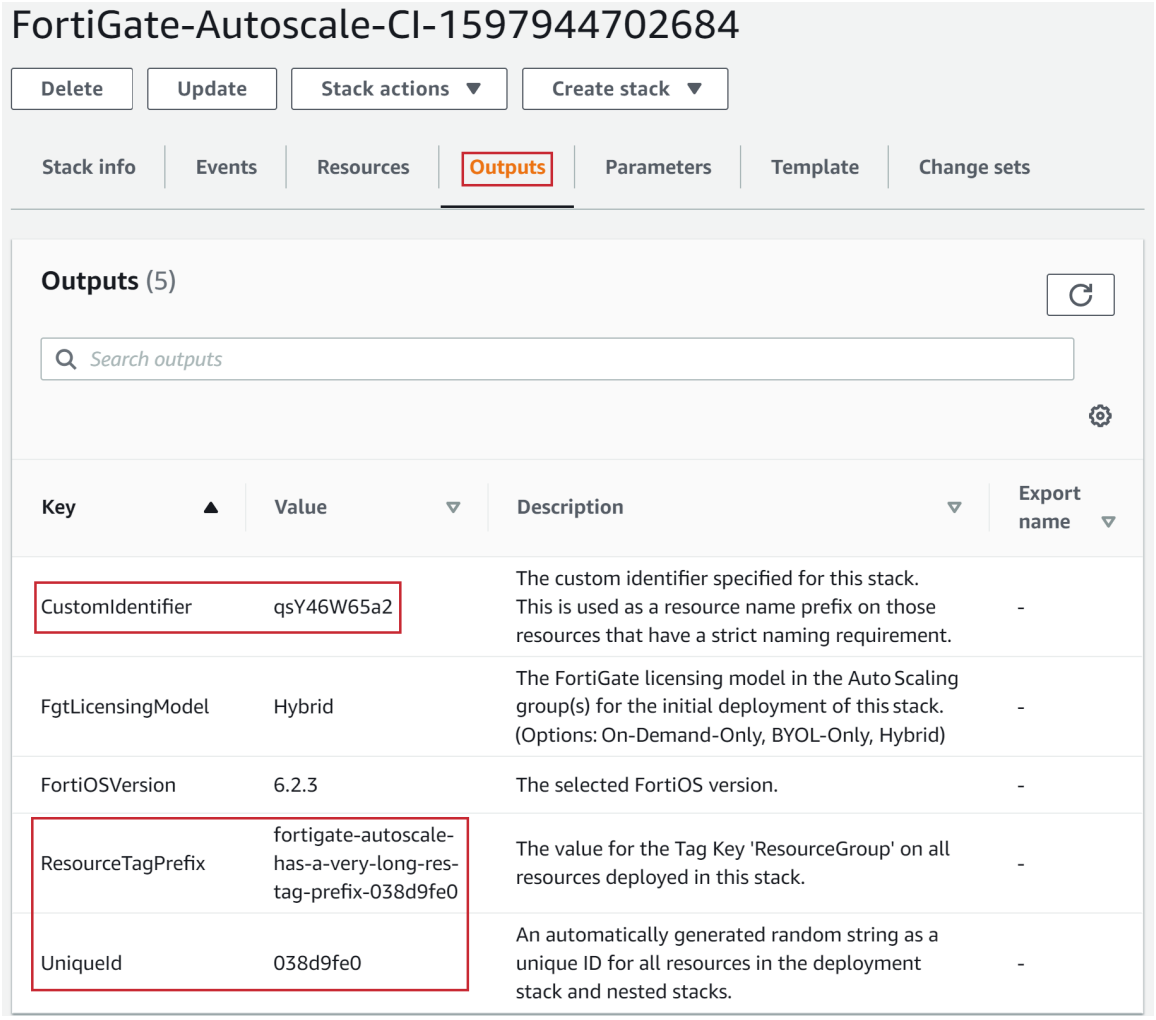
5. From the *Security groups* list, select the group that matches the *Physical ID*.



6. Click *Save*.

Locating deployed resources

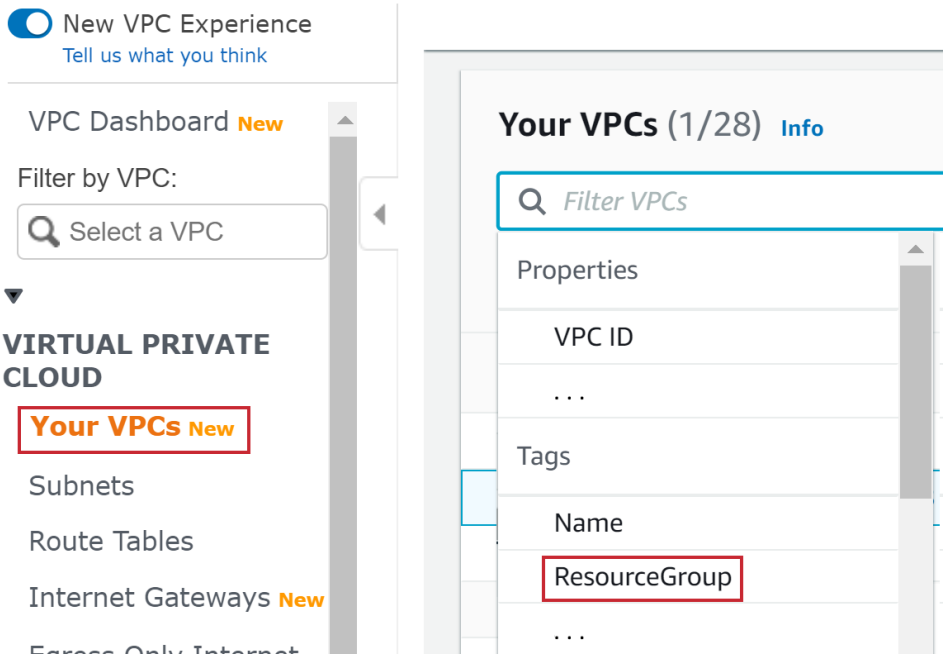
To locate a newly deployed resource, it is recommended to search for it using the *ResourceTagPrefix*, also referred to as the *ResourceGroup Tag Key*. Alternatively, the *UniqueID* can be used. For items that need a shorter prefix, the *CustomIdentifier* can be used. These keys are found on the *Outputs* tab as shown below. Note that the *UniqueID* is at the end of the *ResourceTagPrefix*.



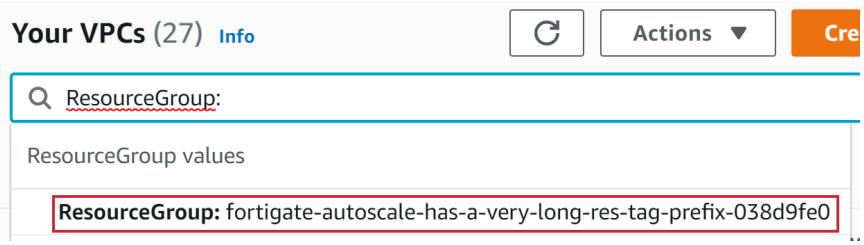
To look up the newly deployed VPC using the ResourceGroup Tag Key:

1. In the AWS console, select *Services* > *Network & Content Delivery* > *VPC*.
2. In the left navigation tree, click *Your VPCs*.

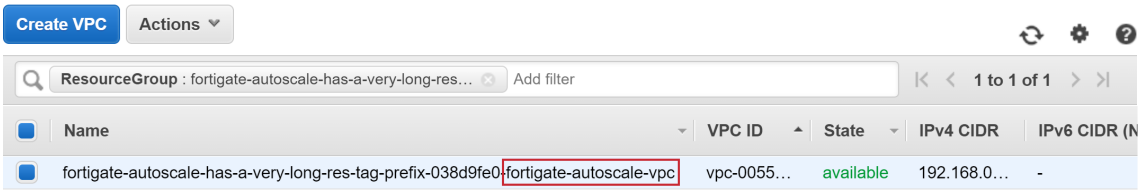
3. Click the filter box and under *Tags*, select *ResourceGroup*.



4. Select your *ResourceTagPrefix* from the list of Tags.



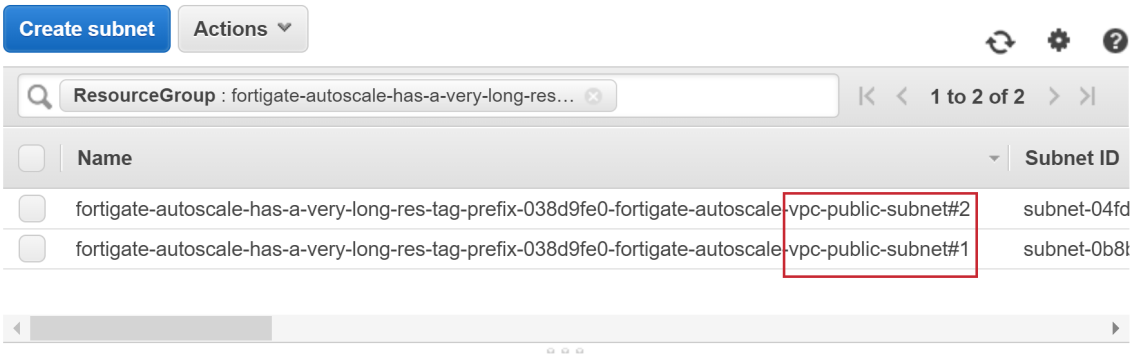
Your VPC will be displayed. The *Name* of VPC is of the format *<ResourceTagPrefix>-fortigate-autoscale-vpc*.



To look up the newly deployed VPC subnets using the ResourceGroup Tag Key:

1. In the AWS console, select *Services* > *Network & Content Delivery* > *VPC*.
2. In the left navigation tree, click *VIRTUAL PRIVATE CLOUD* > *Subnets*.
3. Click the filter box and select *Tag Keys* > *ResourceGroup*.
4. Select your *ResourceTagPrefix* from the list of Tag Keys.

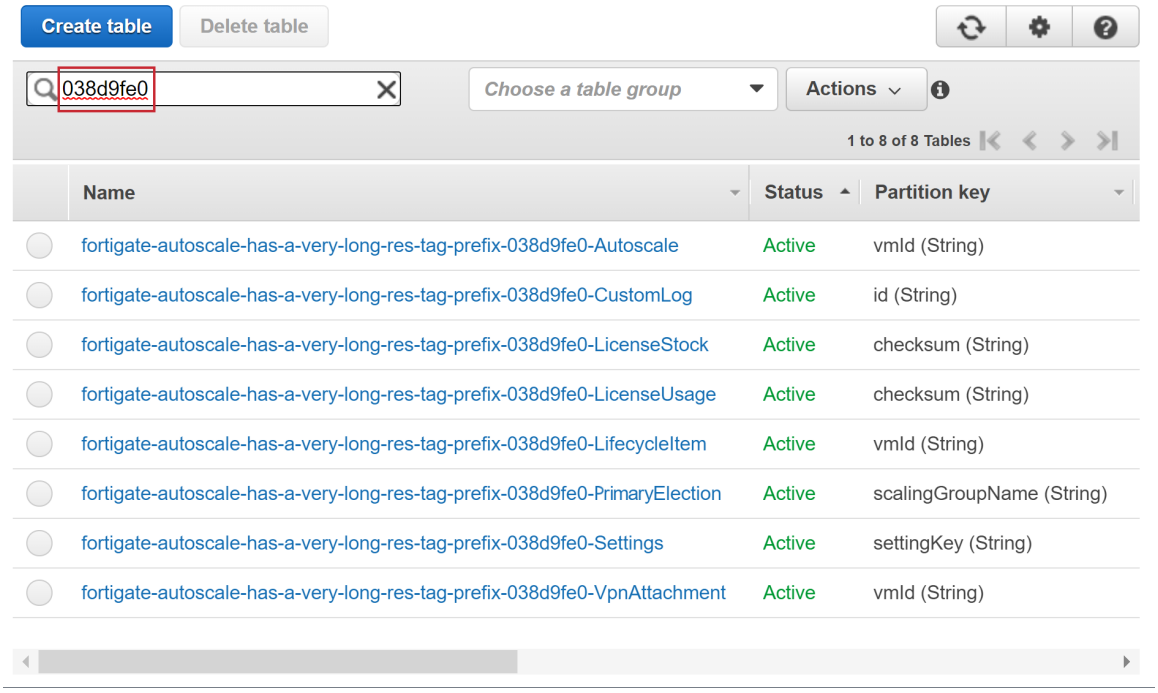
Your VPC subnets will be displayed. The *Name* of each subnets will be of the format *<ResourceTagPrefix>-fortigate-autoscale-vpc-subnet#<#>*.



To look up the newly deployed DynamoDB tables using the UniqueID

1. In the AWS console, select *Services* > *Database* > *DynamoDB*.
2. In the left navigation tree, click *Tables*.
3. Click the filter box and enter the *UniqueID*.

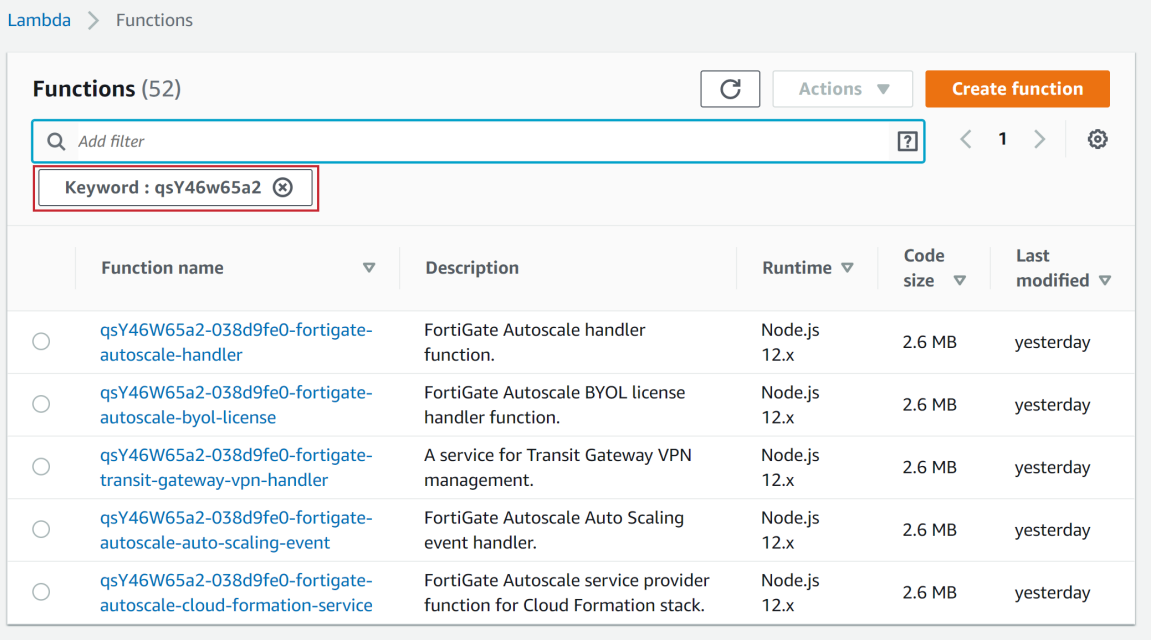
The DynamoDB tables will be displayed. The *Name* of each DynamoDB table will be of the format <*ResourceTagPrefix*>-<*table-name*>.



To look up the newly deployed Lambda Functions using the CustomIdentifier or the UniqueID:

1. In the AWS console, select *Services* > *Compute* > *Lambda*.
2. In the left navigation tree, click *Functions*.
3. Click the filter box and enter the *CustomIdentifier* or the *UniqueID*.

The Lambda Functions will be displayed. Each *Function name* will be of the format <*CustomIdentifier*>-<*UniqueID*>-*LambdaFunctionName*.



Click the *Function name* to go directly to the function.

Verifying the deployment

FortiGate Autoscale for AWS creates two Auto Scaling groups with instances as specified in the CFT parameters. One of theses instances is the elected primary instance. Verify the following:

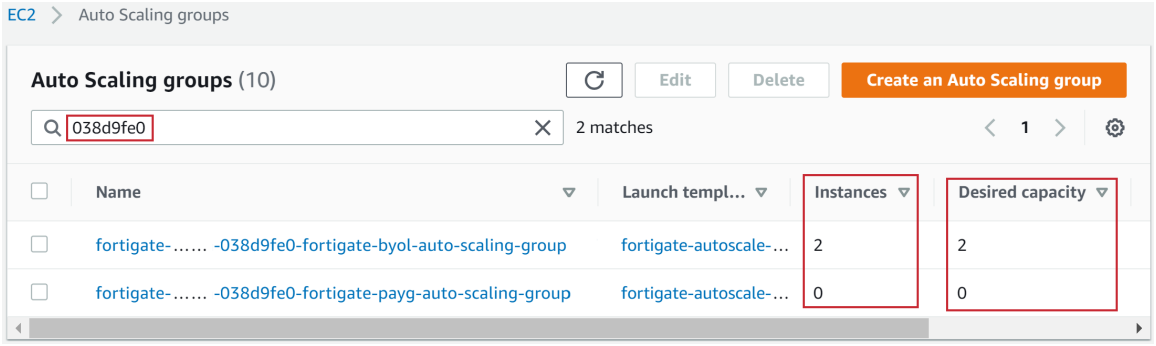
- the Auto Scaling groups
- the primary election

If deploying with Transit Gateway integration, you will also need to verify:

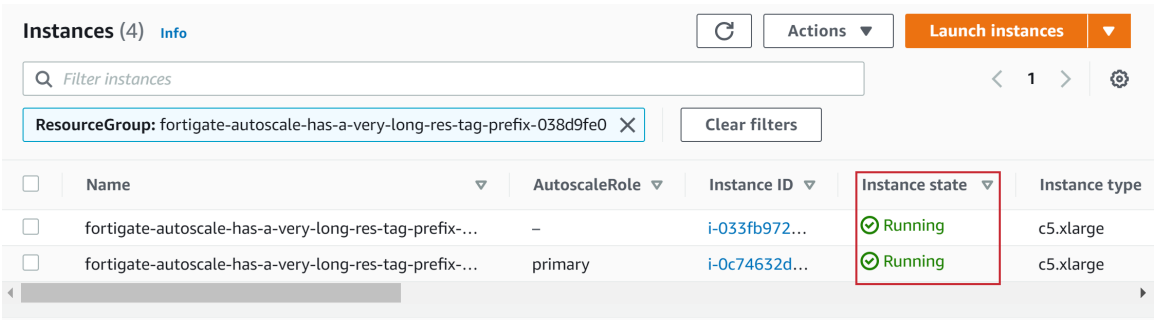
- the Transit Gateway

To verify the Auto Scaling groups:

1. In the AWS console, select the *Services* > *Compute* > *EC2*.
2. In the left navigation tree, click *AUTO SCALING* > *Auto Scaling Groups*.
3. Click the filter box and look up the Auto Scaling groups using the *Unique ID*.
4. The name of each group will start with the prefix you specified in *Resource tag prefix*. Confirm that the number in the *Instances* column is equal to or greater than the *Desired capacity* you specified.

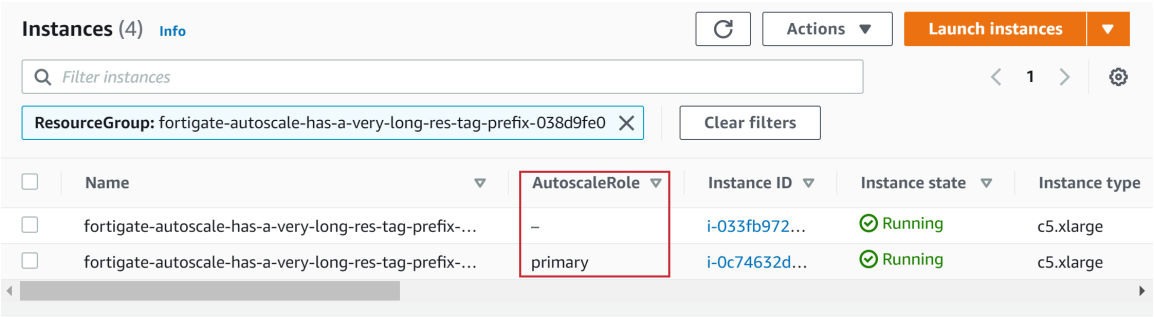


5. In the left navigation tree, click *INSTANCES* > *Instances*.
6. Click the filter box and look up instances using the *ResourceTagPrefix*.
7. Instances will be listed with their current state.

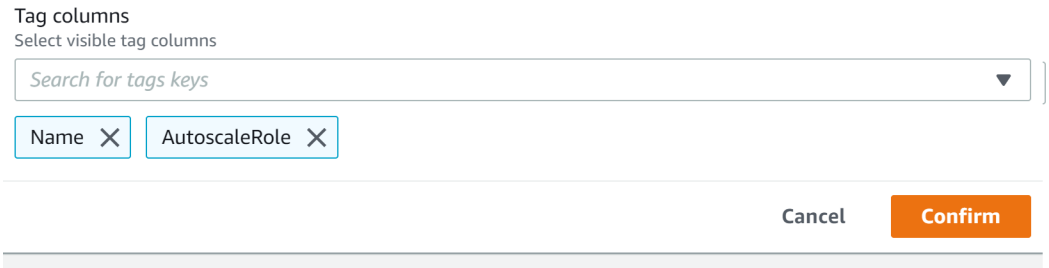


To verify the primary election:

The primary instance is noted in the *AutoscaleRole* column:

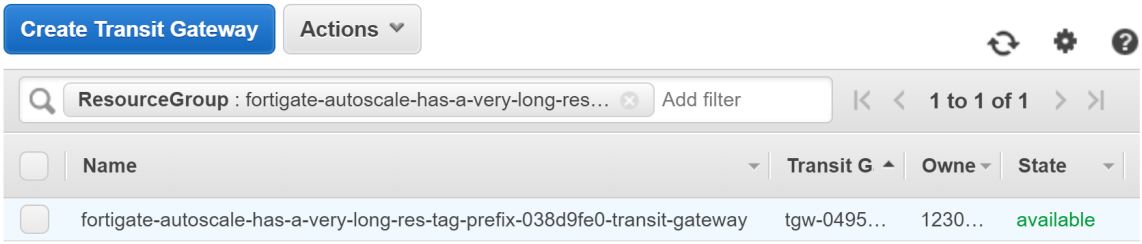


If the *AutoscaleRole* column is not displayed, click the *Preferences* cog and locate the *Tag columns*dropdown. Select *AutoscaleRole* and then click *Confirm*.

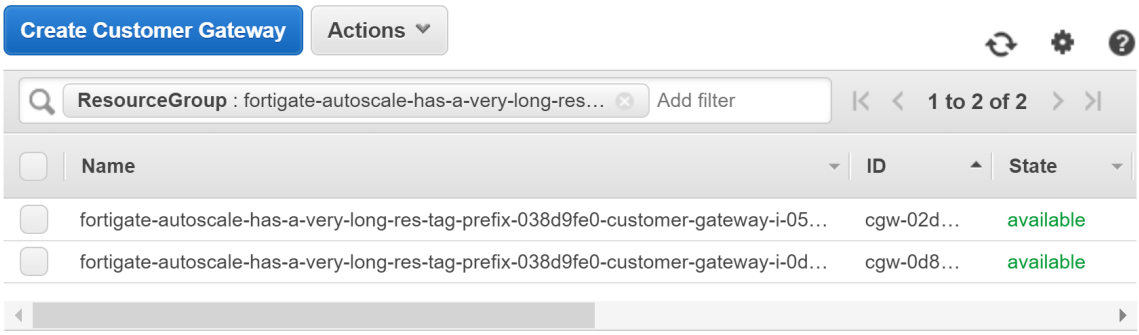


To verify the Transit Gateway:

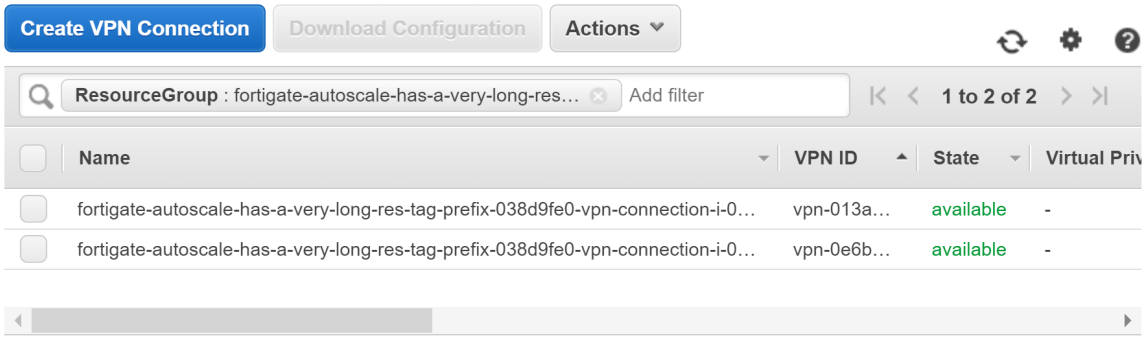
1. In the AWS console, select the *Services* > *Network & Content Delivery* > *VPC*.
2. In the left navigation tree, click *TRANSIT GATEWAYS* > *Transit Gateways*.
3. Filter by the Tag Key *ResourceGroup*. There should be one result.



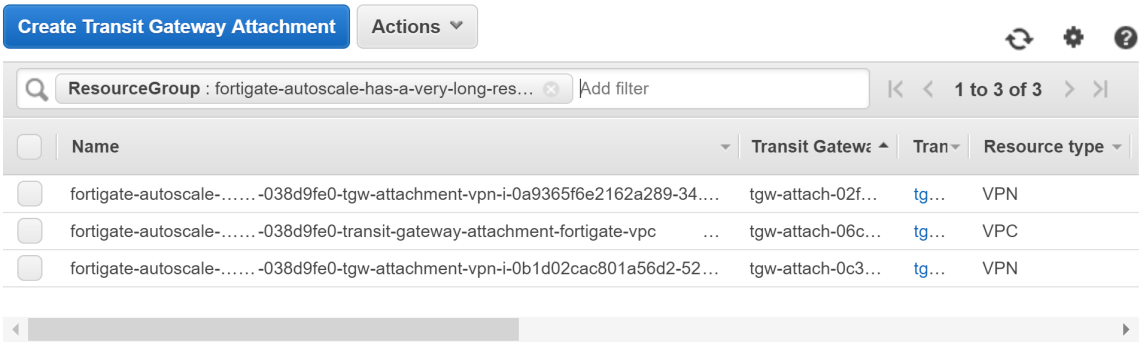
4. In the left navigation tree, click *VIRTUAL PRIVATE NETWORK (VPN)* > *Customer Gateways*.
5. Filter by the Tag Key *ResourceGroup*. There should be one customer gateway per running FortiGate-VM instance (2 at the start).



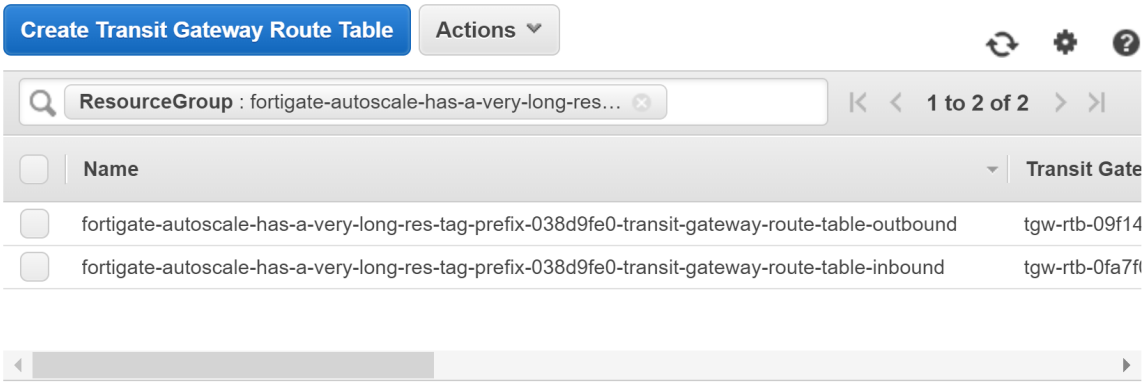
6. In the left navigation tree, click *VIRTUAL PRIVATE NETWORK (VPN)* > *Site-to-Site VPN Connections*.
7. Filter by the Tag Key *ResourceGroup*. There should be two items, 1 per FortiGate-VM instance, each with a corresponding Transit Gateway attachment.



8. In the left navigation tree, click *TRANSIT GATEWAYS* > *Transit Gateway Attachments*.
9. Filter by the Tag Key *ResourceGroup*. There should be one VPC, and one VPN per running FortiGate-VM instance in the Auto Scaling group. (2 at the start, one primary and one secondary). The VPN name will contain the public IP address of the VPN.



10. In the left navigation tree, click *TRANSIT GATEWAYS* > *Transit Gateway Route Tables*.
11. Filter by the Tag Key *ResourceGroup*. There should be two items, one for inbound and one for outbound. For diagrams, refer to the Appendix on page 81.

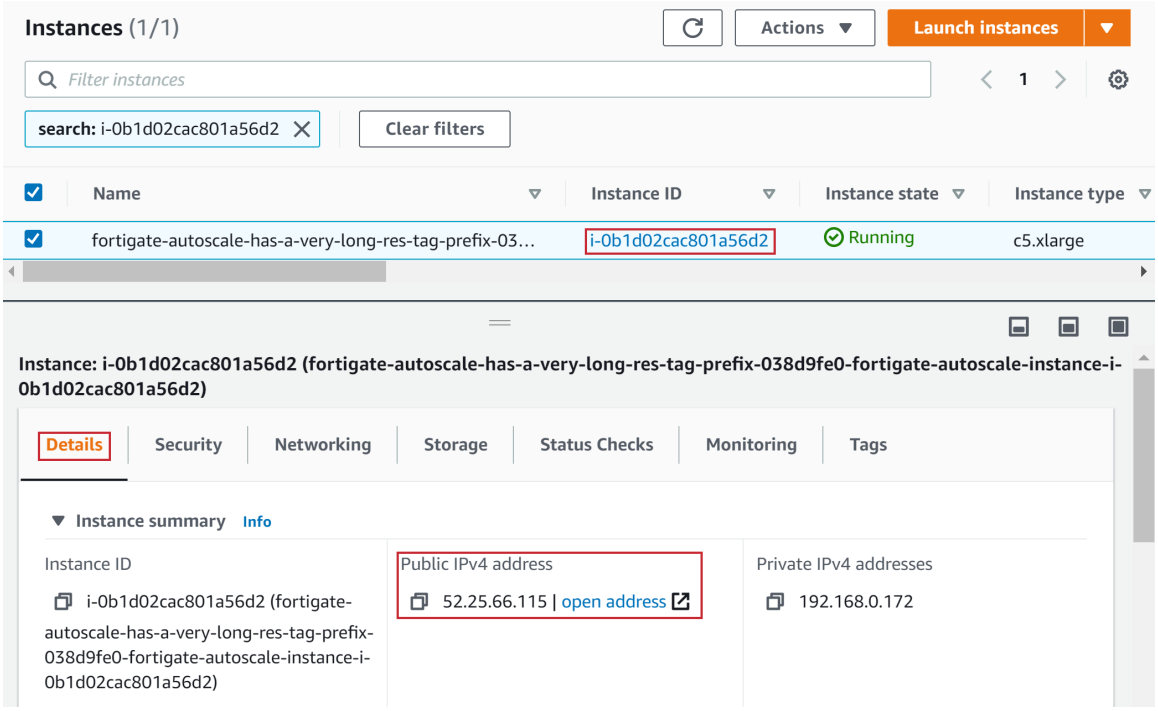


Connecting to the primary FortiGate-VM

To connect to the primary FortiGate-VM instance, you will need a login URL, a username, and a password.

1. Construct a login URL in this way: *https://<IPAddress>:<Port>/*, where:
   - *Port* refers to the *Admin port* specified in the section FortiGate configuration on page 52.
   - *IPAddress* refers to the *Public IPv4 address* of the FortiGate-VM and is listed on the *Details* tab for the instance. In the EC2 Management console, locate the primary instance as described in the section To verify the

primary election: on page 68. Click the *Instance ID* for the primary instance.



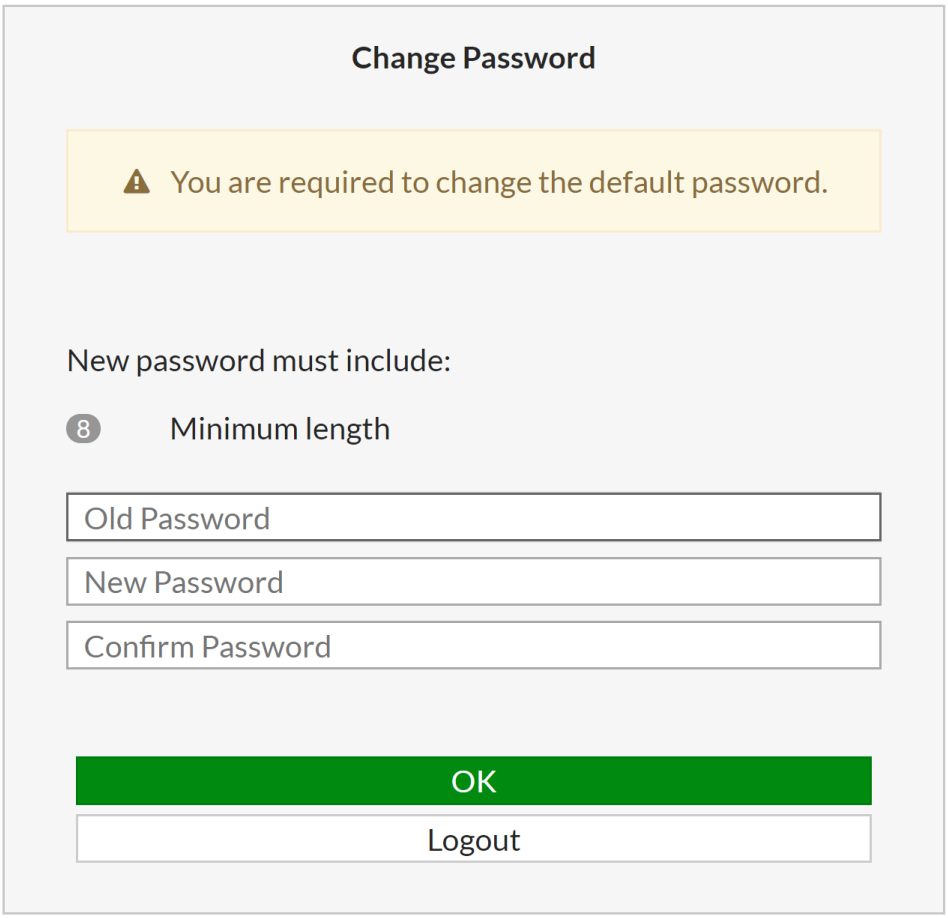
Make note of the *InstanceID* as you will need it to log in.

2. Open an HTTPS session in your browser and go to the login URL. Your browser will display a certificate error message. This is normal because the default FortiGate certificate is self-signed and not recognized by browsers. Proceed past this error. At a later time, you can upload a publicly signed certificate to avoid this error.
3. Log in with the username *admin* and the *Instance ID* of the primary FortiGate-VM instance.

As the primary FortiGate-VM propagates the password to all secondary FortiGate instances, this is the initial password for all FortiGate-VM instances.

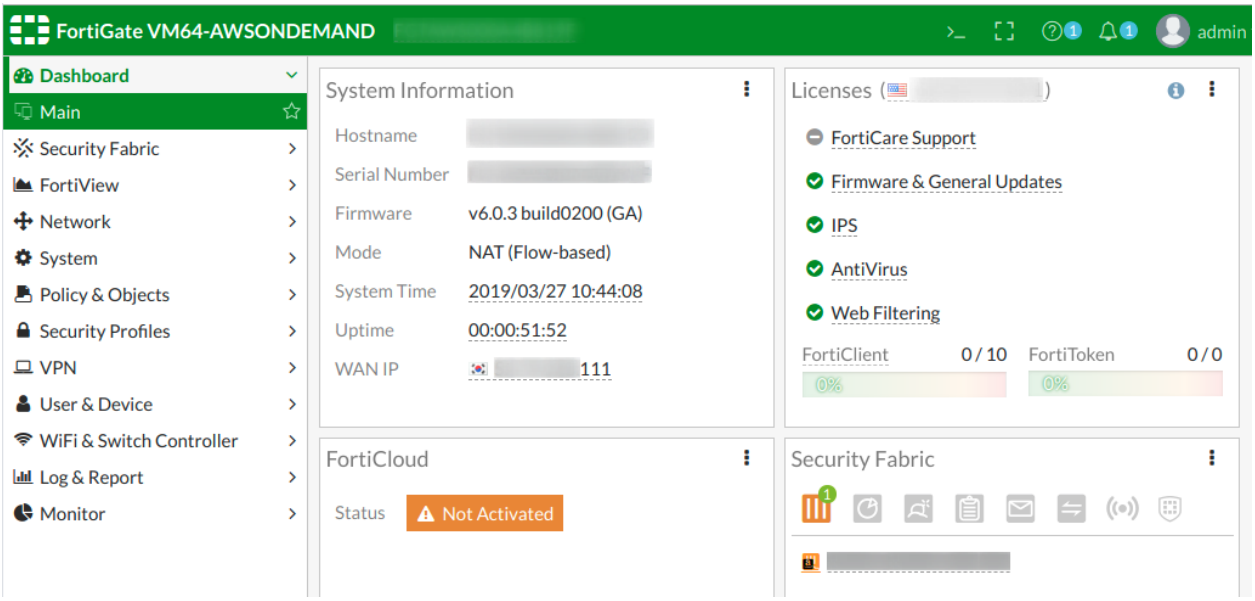
You will need this initial password if failover occurs prior to the password being changed, as the newly elected primary FortiGate-VM will still have the initial password of the previous primary .

4. You will be prompted to change the password at the first-time login. It is recommended that you do so at this time.



You should only change the password on the primary FortiGate-VM. The primary FortiGate-VM will propagate the password to all secondary FortiGate-VMs. Any password changed on a secondary FortiGate-VM will be overwritten.

5. You will now see the FortiGate-VM dashboard. The information displayed in the license widget of the dashboard depends on your license type.



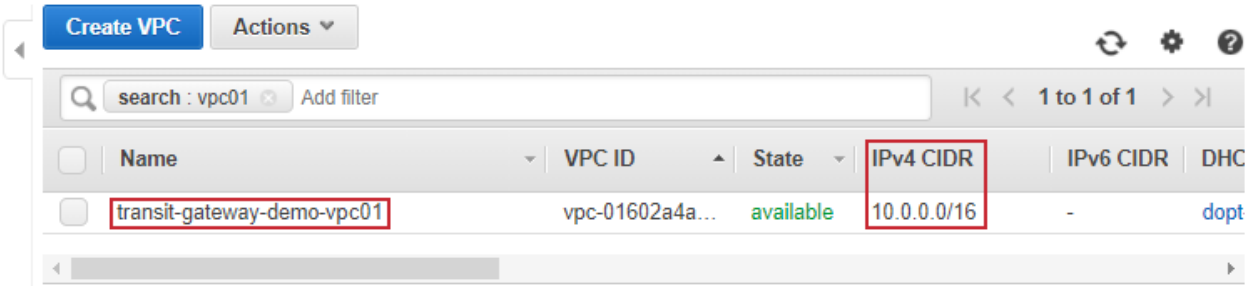
Attaching a VPC to the Transit Gateway

You can attach an existing VPC to the FortiGate Autoscale with Transit Gateway environment by manually creating a Transit Gateway attachment and adding the necessary routes, propagations, and associations:

1. Create a Transit Gateway attachment.
2. Create a route to the Transit Gateway.
3. Create a propagation in the inbound route table.
4. Create an association in the outbound route table.

The CIDR block for the VPC you are attaching must differ from that of the FortiGate Autoscale VPC.

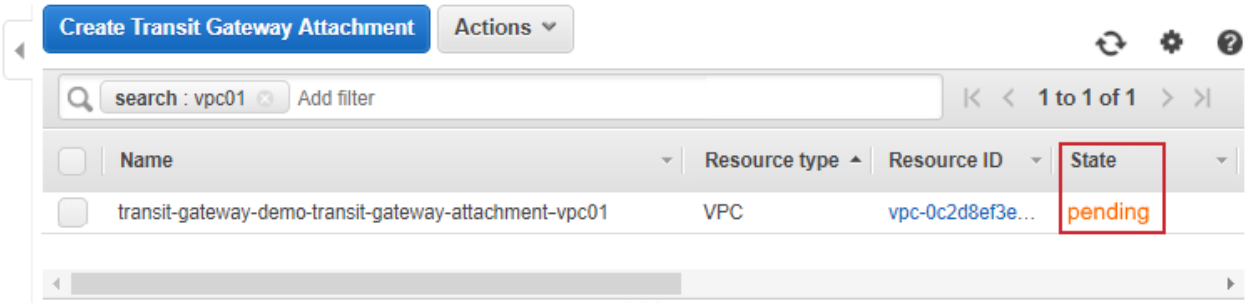
In the instructions that follow, the VPC *transit-gateway-demo-vpc01* with CIDR *10.0.0.0/16* will be attached to the FortiGate Autoscale with Transit Gateway environment.



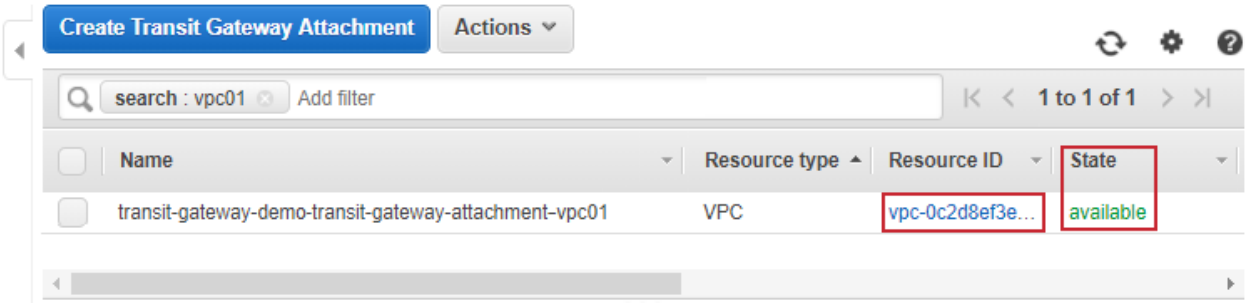
To create a Transit Gateway attachment:

1. In the left navigation tree, click *TRANSIT GATEWAYS* > *Transit Gateway Attachment*.
2. Click *Create Transit Gateway Attachment*.
3. Specify information as follows:
   a. *Transit Gateway ID*: Select from the dropdown menu
   b. *Attachment type*: VPC
   c. *Attachment name tag:* Enter a tag of your choice
   d. *VPC ID*: Select from the dropdown menu
   e. *Subnet ID*s: This option appears once the *VPC ID* has been selected. Check the Availability Zone checkbox (es) and choose 1 subnet per Availability Zone.

   For everything else, use the default settings.
4. Click *Create attachment*.
5. Wait for the *State* to change from *pending* to *available*.

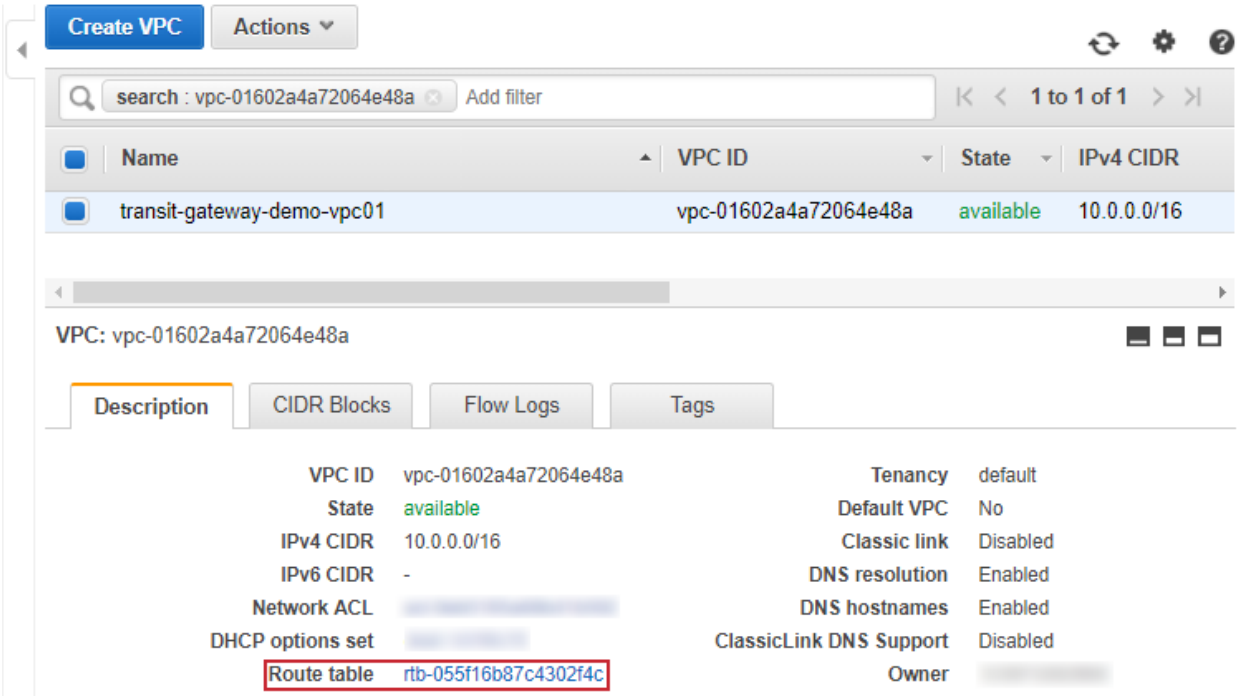


   The *Name* is what you specified for the *Attachment name tag*.
6. When the *State* is *available*, click on the *Resource ID* to go to the VPC.

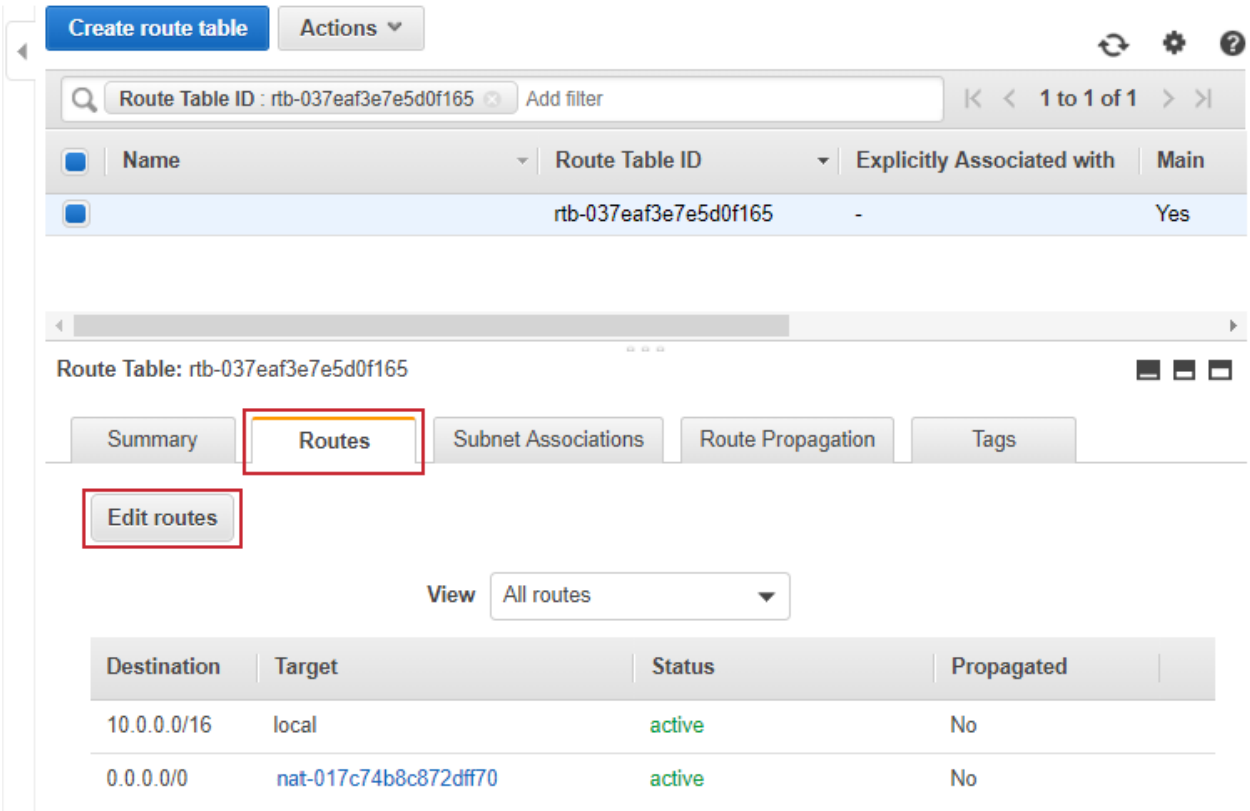


To create a route to the Transit Gateway:

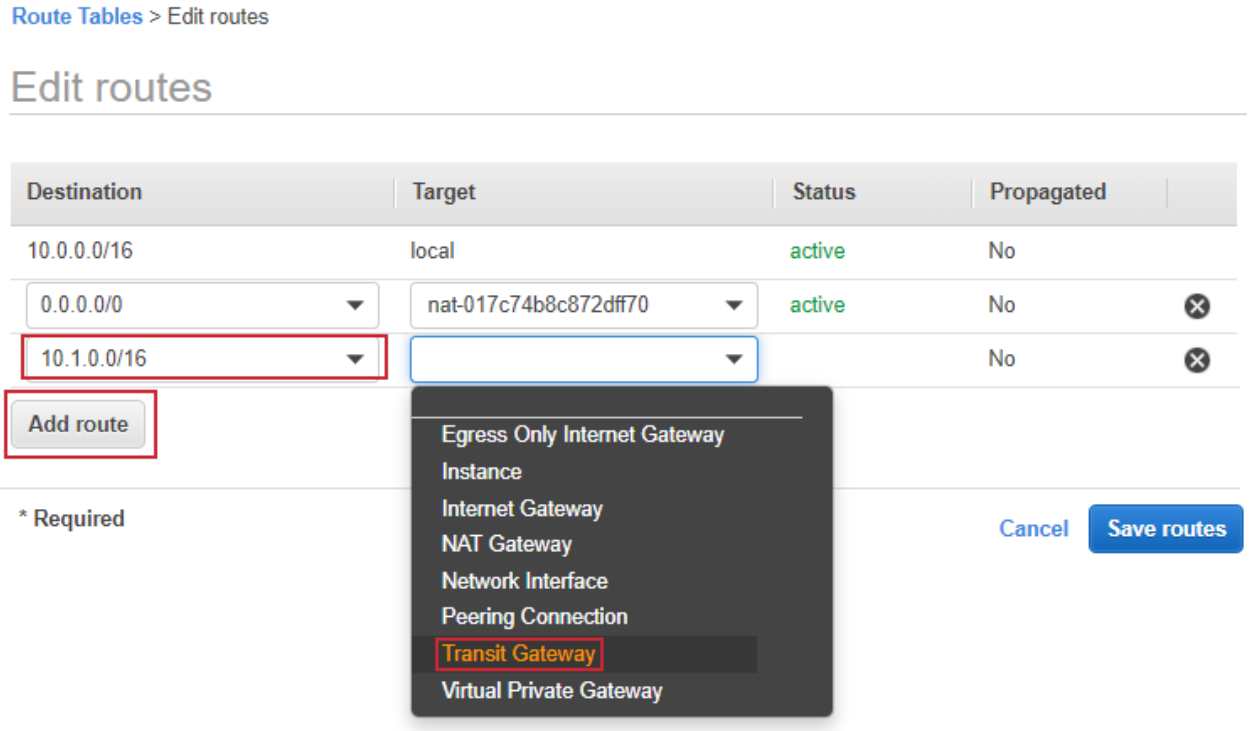
1. In the VPC, click on the *Route table*.



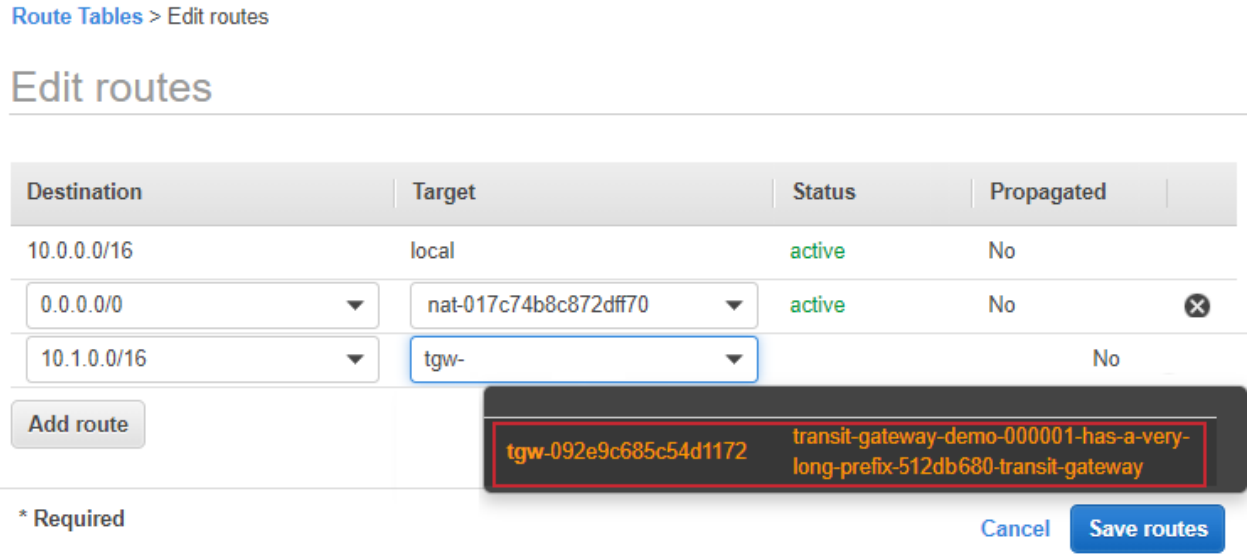
2. Click the *Routes* tab and then click *Edit routes*.



3. Click *Add route* and specify the *Destination*, for example, 10.1.0.0/16. Under *Target*, select *Transit Gateway*.



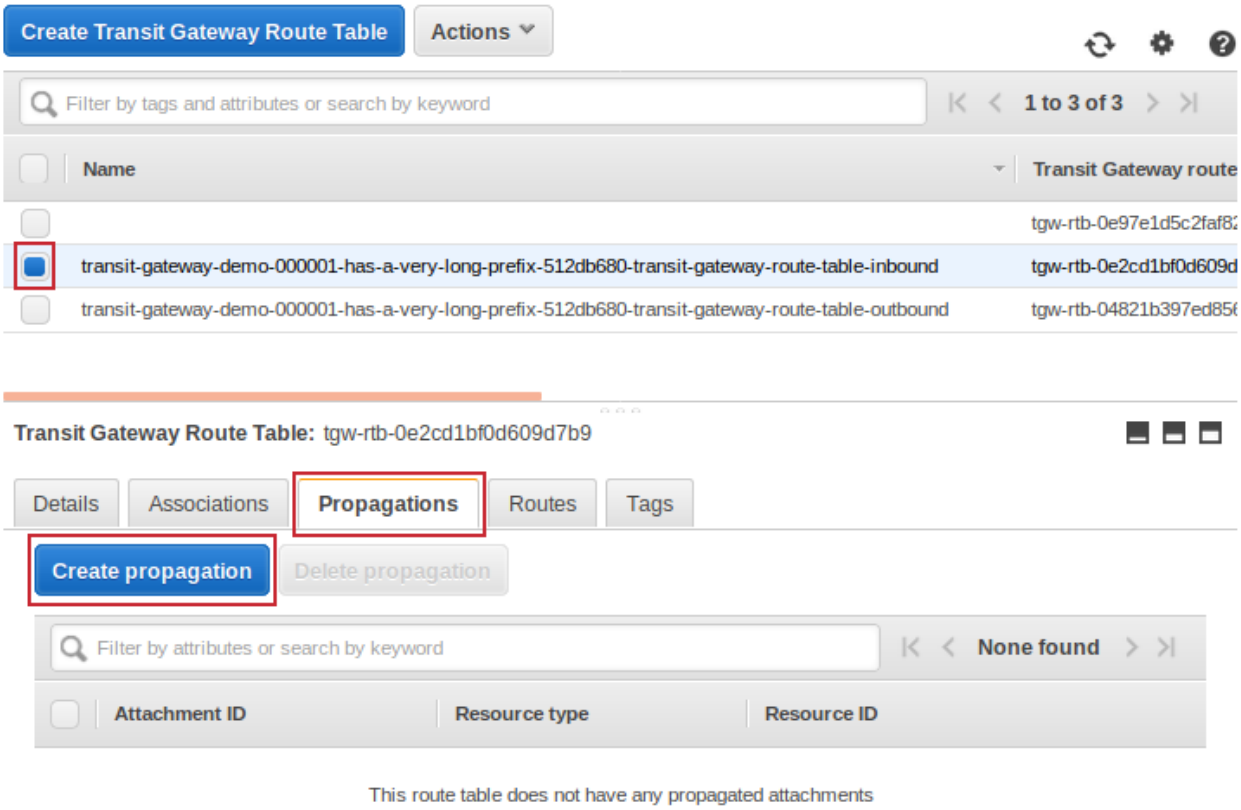
4. Then dropdown will change to display available Transit Gateways. Select the one created by the deployment stack and then click *Save routes*.



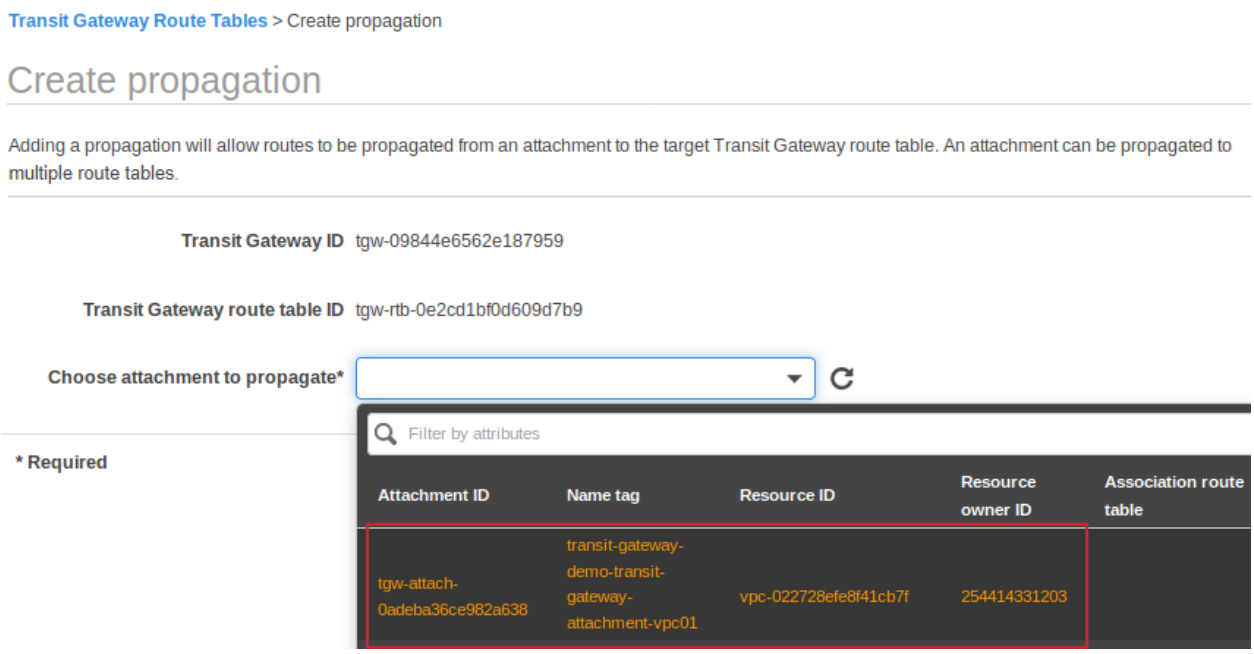
If you want to route all traffic to the Transit Gateway, you should add a new route for destination 0.0.0.0/0. If this route already exists, simply remove the route and add a new one for the same destination with the target set to the Transit Gateway created by the deployment stack.

To create a propagation in the inbound route table:

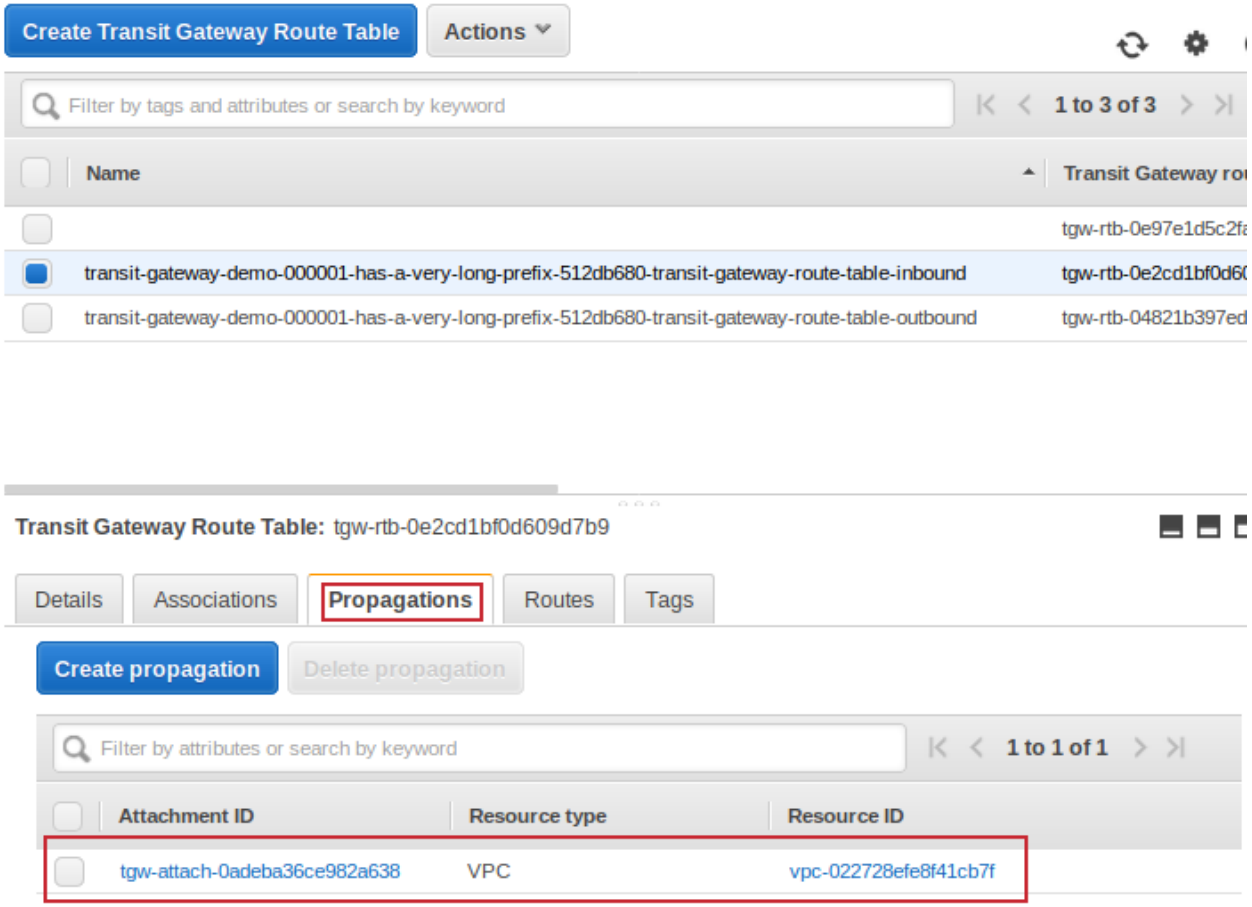
1. In the left navigation tree, click *Transit Gateways* > *Transit Gateway Route Tables*.
2. Select the <*ResourceTagPrefix*>-*transit-gateway-route-table-inbound* route table.



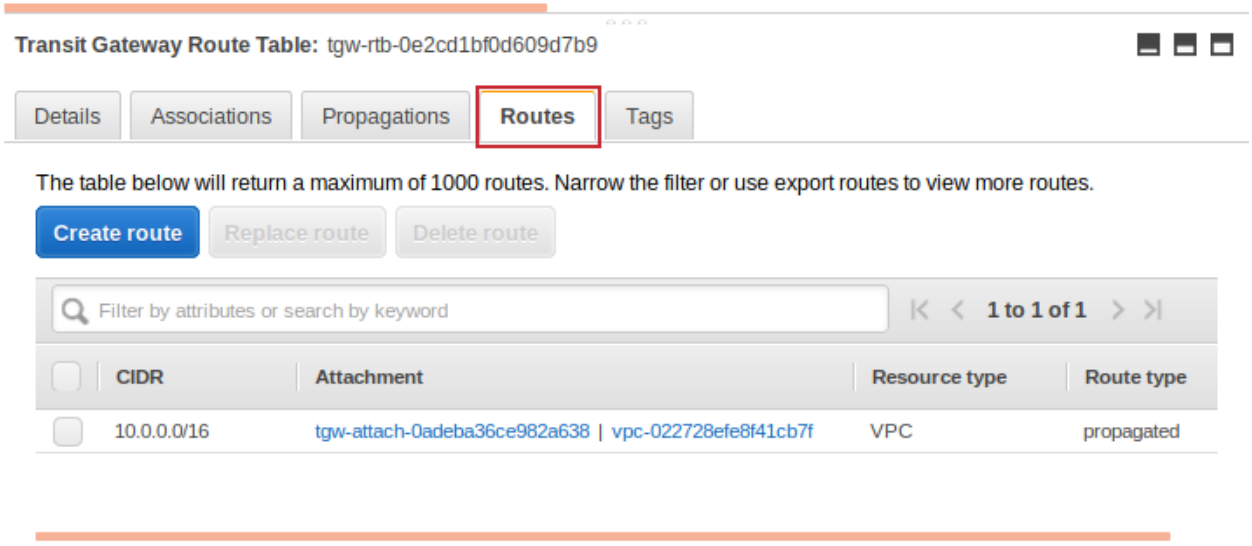
3. Click the *Propagations* tab and then click *Create propagation*.
4. From *Choose attachment to propagate*, select the attachment created in the section To create a Transit Gateway attachment: on page 74.



5. Click *Create propagation* and then click *Close*.
6. The new propagation with *Resource type* VPC is now listed on the *Propagations* tab.

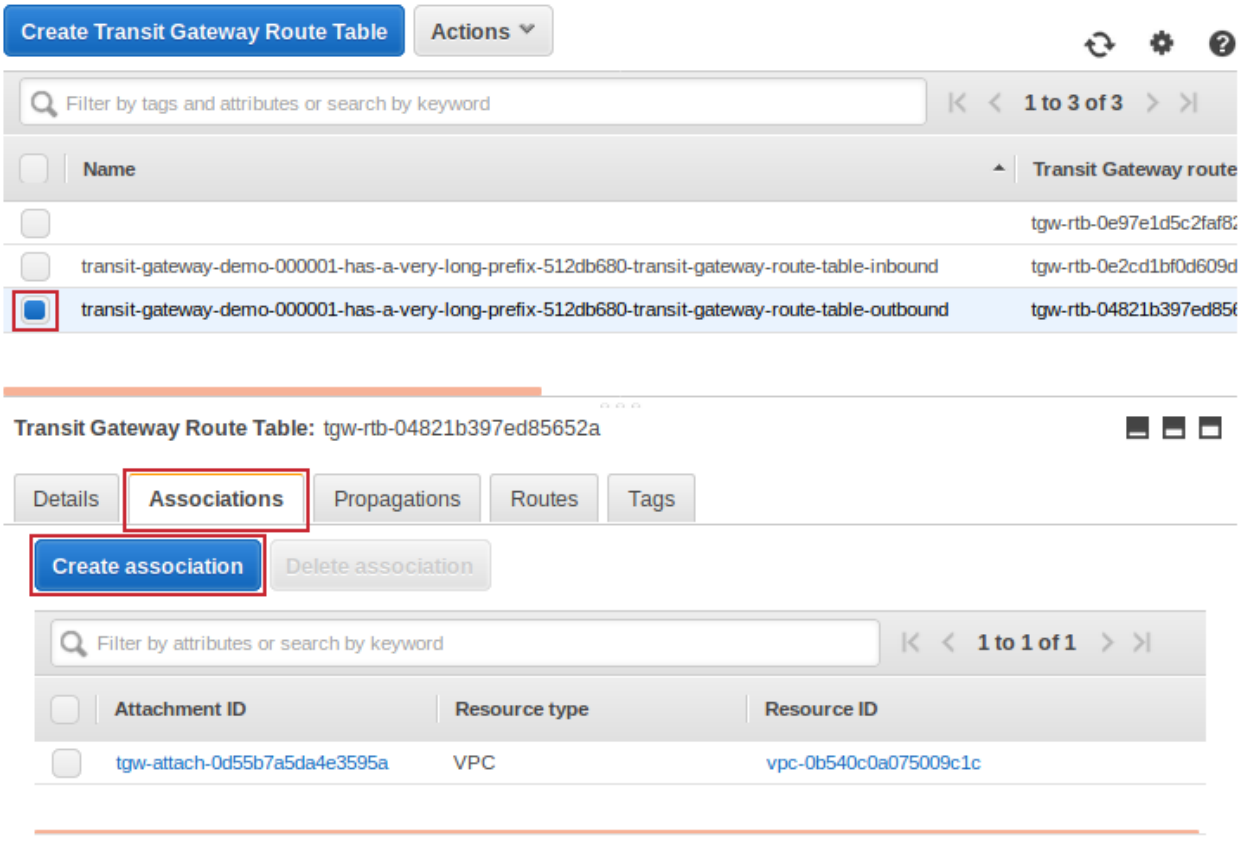


7. Click on the *Routes* tab to see that the route for your VPC has been automatically propagated.



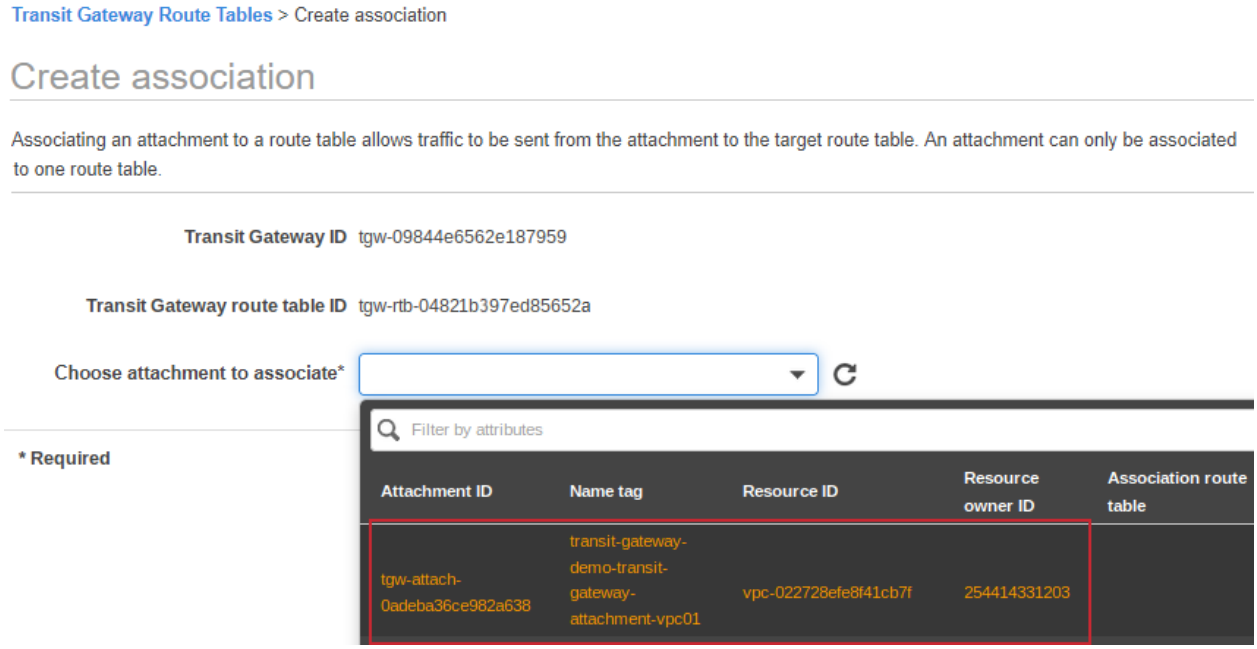
To create an association in the outbound route table:

1. In the left navigation tree, click *Transit Gateways* > *Transit Gateway Route Tables*.
2. Select the <*ResourceTagPrefix*>*-transit-gateway-route-table-outbound* route table.

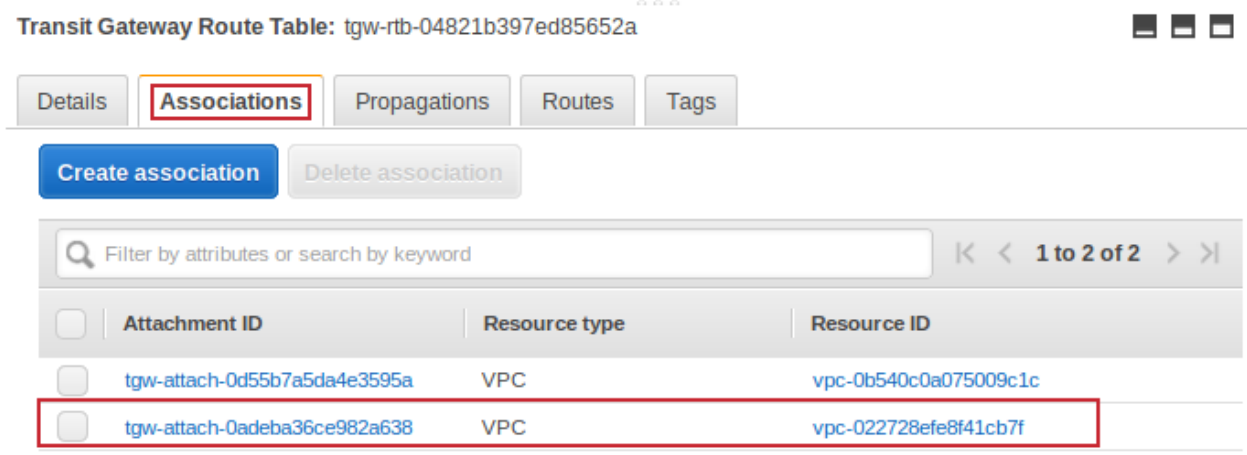


3. Click the *Associations* tab and then click *Create association*.

4. From *Choose attachment to associate*, select the attachment created in the section To create a Transit Gateway attachment: on page 74.



5. Click *Create association* and then click *Close*.
6. The new association with *Resource type* VPC is now listed on the *Associations* tab.



The VPC is now connected to the FortiGate Autoscale Transit Gateway. For a technical view of attaching VPCs to the FortiGate Autoscale Transit Gateway, please refer to the architectural diagram .

Troubleshooting

CREATE_FAILED error in CloudFormation stack

If you encounter a CREATE_FAILED error when you launch the Quick Start, it is recommended that you relaunch the template with *Rollback on failure* set to *Disabled*. (This setting is under *Advanced options* in the AWS CloudFormation console, *Configuring option settings* page.) With this setting, the stack's state is retained and the instance is left running, so you can troubleshoot the issue.

When you set *Rollback on failure* to *Disabled*, you continue to incur AWS charges for this stack. Please make sure to delete the stack when you finish troubleshooting.

For additional information, see Troubleshooting AWS CloudFormation on the AWS website.

The deployment will also fail if you select an instance type that is not supported in the region that was selected. Your desired instance type is available in your region if it is listed on the Instance types page for your region.

The election of the primary FortiGate-VM was not successful

If the election of the primary FortiGate-VM is not successful, reset the elected primary FortiGate-VM. If the reset does not solve the problem, please contact support.

How to reset the elected primary-VM FortiGate

To reset the elected primary FortiGate-VM, navigate to the DynamoDB table <*ResourceTagPrefix*>-*FortiGatePrimaryElection*. Click the *Items* tab and delete the only item in the table.

A new primary FortiGate-VM will be elected and a new record will be created as a result.

For details on locating the DynamoDB table <*ResourceTagPrefix*>-*FortiGatePrimaryElection*, refer to the section Locating deployed resources on page 63.

Appendix

FortiGate Autoscale for AWS features

Major components

- *The BYOL Auto Scaling group*. This Auto Scaling group contains 0 to many FortiGate-VMs of the BYOL licensing model and will dynamically scale-out or scale-in based on the scaling metrics specified by the parameters *Scale-out threshold* and *Scale-in threshold*. For each instance you must provide a valid license purchased from FortiCare.

For BYOL-only and hybrid licensing deployments, the Minimum group size (`FgtAsgMinSizeByol`) must be at least 2. These FortiGate-VMs are the main instances and are fixed and running 7x24. If it is set to 1 and the instance fails to work, the current FortiGate-VM configuration will be lost.

- *The On-demand Auto Scaling group*. This Auto Scaling group contains 0 to many FortiGate-VMs of the On-demand licensing model and will dynamically scale-out or scale-in based on the scaling metrics specified by the parameters *Scale-out threshold* and *Scale-in threshold*.

  For On-demand-only deployments, the Minimum group size (`FgtAsgMinSizePayg`) must be at least 2. These FortiGate-VMs are the main instances and are fixed and running 7x24. If it is set to 1 and the instance fails to work, the current FortiGate-VM configuration will be lost.

- *The "assets" folder in the S3 Bucket*.
  - The *configset* folder contains files that are loaded as the initial configuration for a new FortiGate-VM instance.
    - *baseconfig* is the base configuration. This file can be modified as needed to meet your network requirements. Placeholders such as {SYNC_INTERFACE} are explained in the Configset placeholders on page 82 table below
    - *httproutingpolicy* and *httpsroutingpolicy* are provided as part of the base configset - for a common use case - and specify the FortiGate firewall policy for VIPs for *http* routing and *https* routing respectively. This common use case includes a VIP on port 80 and a VIP on port 443 with a policy that points to an internal load balancer. The port numbers are configurable and can be changed during CFT deployment. Additional VIPs can be added here as needed.

      In FortiOS 6.2.3, any VIPs created on the primary instance will not sync to the secondary instances. Any VIP you wish to add must be added as part of the base configuration.

      If you set the *Internal ELB options* parameter to `do not need one`, then you must include your VIP configuration in the base configuration.

  - The *... >license-files > fortigate* folder contains BYOL license files.
- *Tables in DynamoDB*. These tables are required to store information such as health check monitoring, primary election, state transitions, etc. These records should not be modified unless required for troubleshooting purposes.
- *Networking Components* These are the network load balancers, the target group, and the VPC and subnets. You are expected to create your own client and server instances that you want protected by the FortiGate-VM.

Configset placeholders

When the FortiGate-VM requests the configuration from the Auto Scaling Handler function, the placeholders in the table below will be replaced with actual values about the Auto Scaling group.

Placeholder	Type	Description
{SYNC_ INTERFACE}	Text	The interface for FortiGate-VMs to synchronize information. Specify as port1, port2, port3, etc. All characters must be lowercase.
{CALLBACK_URL}	URL	The endpoint URL to interact with the auto scaling handler script.

Placeholder	Type	Description
		Automatically generated during CloudFormation deployment.
{PSK_SECRET}	Text	The Pre-Shared Key used in FortiOS. Specified during CloudFormation deployment.
{ADMIN_PORT}	Number	A port number specified for admin login. A positive integer such as 443 etc. Specified during CloudFormation deployment.
{HEART_BEAT_INTERVAL}	Number	The time interval (in seconds) that the FortiGate-VM waits between sending heartbeat requests to the Autoscale handler function.

Auto Scaling Handler environment variables

Variable name	Description
UNIQUE_ID	Reserved, empty string.
CUSTOM_ID	Reserved, empty string.
RESOURCE_TAG_PREFIX	The value of the CFT parameter *Resource tag prefix* which is described in the section Resource tagging configuration on page 50.

AWS GovCloud (US) support

The AWS GovCloud (US) regions `us-gov-east-1` and `us-gov-west-1` are supported.

AWS may have service limitations, restrictions, or different implementations for these regions. Please review AWS documentation for more information.

As service is provided differently than it is for commercial regions, if you encounter errors when deploying to these regions, please report them on the Issues tab of the FortiGate Autoscale for AWS GitHub project.

How to partially route egress traffic

By default, FortiGate Autoscale manages the route `0.0.0.0/0` in the route table associated with the FortiGate-VM cluster. As such, all egress traffic will be routed to the primary FortiGate-VM. If desired, you can add firewall policies to the FortiGate-VM with more customized egress rules.

In addition to the `0.0.0.0/0` route via FortiGate Autoscale, egress traffic can be also routed via other NAT gateways. This is done by creating a route with a specific destination with the NAT device as the target. This route must be next to the route `0.0.0.0/0` in the Autoscale route table and the route destination must be a valid CIDR. For example, for egress traffic to the IP address range `10.0.0.0/16` to use a different NAT device, create a route with destination `10.0.0.0/16` and the NAT device as the target. Egress traffic to `10.0.0.0/16` will now flow through the NAT device while the rest will still flow through FortiGate.

However, you cannot use the route with destination `0.0.0.0/0` because FortiGate Autoscale is managing it and will overwrite it whenever the FortiGate primary role has been switched.

Deployment templates

Deploying FortiGate Autoscale for AWS requires the use of deployment templates. There are two types of templates:

- *Entry template*. This template could run as the entry point of a deployment.
- *Dependency template*. This template is automatically run by the deployment process as a Nested Stack. It cannot be run as an entry template. A dependency template is run based on user selected options.

Following are descriptions of the templates included in the FortiGate Autoscale for AWS deployment package.

Template	Type	Description
autoscale-new-vpc.template.yaml	Entry template	Deploys the Auto Scaling solution to a new VPC.
autoscale-existing-vpc.template.yaml	Entry template	Deploys the Auto Scaling solution to an existing VPC.
autoscale-tgw-new-vpc.template.yaml	Entry template	Deploys the Auto Scaling solution with Transit Gateway Integration to a new VPC.
autoscale-main.template.yaml	Dependency template	Does the majority of the work for deploying FortiGate Autoscale.
configure-fortianalyzer-service.template.yaml	Dependency template	Configure the FortiAnalyzer integration additional services.
copy-objects.template.yaml	Dependency template	Creates an S3 bucket in the same region where the stack is launched and copies deployment related objects to this S3 bucket.
create-autoscale-handler.template.yaml	Dependency template	Creates a FortiGate Autoscale Handler Lambda function and an API Gateway.
create-db-table.template.yaml	Dependency template	Creates all necessary DynamoDB tables for the FortiGate Autoscale solution.
create-fortianalyzer-components.template.yaml	Dependency template	Deploys a FortiAnalyzer to a selected subnet and configures all FortiGates to connect to it.
create-fortigate.template.yaml	Dependency template	Deploys a FortiGate EC2 instance to a subnet using a given FortiGate AMI, security group, and instance profile.
create-hybrid-auto-scaling-group.template.yaml	Dependency template	Deploys the hybrid licensing FortiGate Auto Scaling groups.
create-load-balancer.template.yaml	Dependency template	Deploys network traffic Load Balancers and components for FortiGate Autoscale.
create-new-vpc.template.yaml	Dependency template	Creates a new VPC in which to deploy the FortiGate Autoscale solution.
create-transit-gateway-components.template.yaml	Dependency template	Creates a Transit Gateway for FortiGate Autoscale for AWS.
create-tgw-vpn-handler.template.yaml	Dependency template	Creates a service for Transit Gateway VPN management.

Cloud-init

In Auto Scaling, a FortiGate-VM uses the `cloud-init` feature to pre-configure the instances when they first come up. During template deployment, an internal API Gateway endpoint will be created.

A FortiGate-VM sends requests to the endpoint to retrieve necessary configuration after initialization.

Use this FortiOS CLI command to display information for your devices:

```
# diagnose debug cloudinit show
```

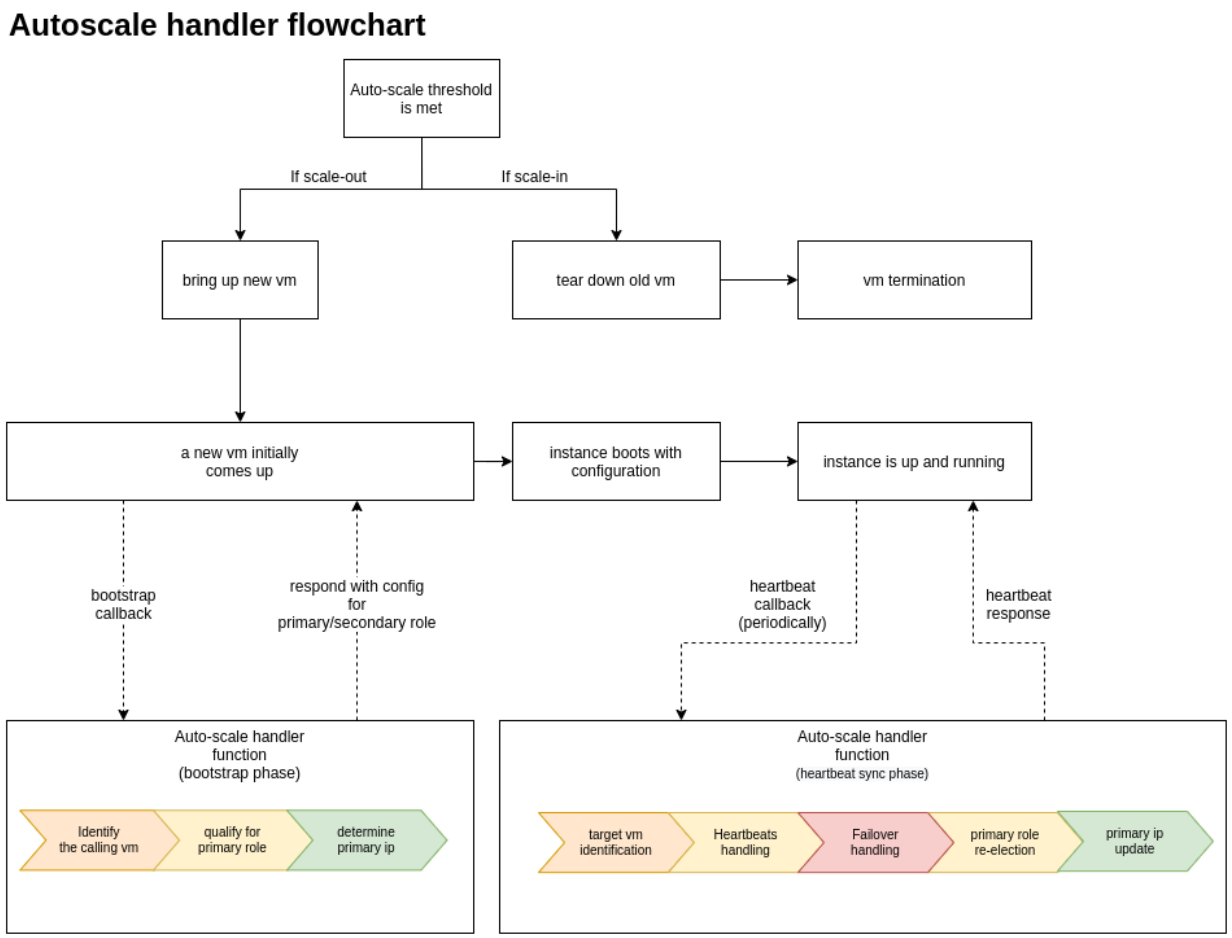
VPN output can be retrieved with this FortiOS CLI command:

```
# diagnose vpn tun list
```

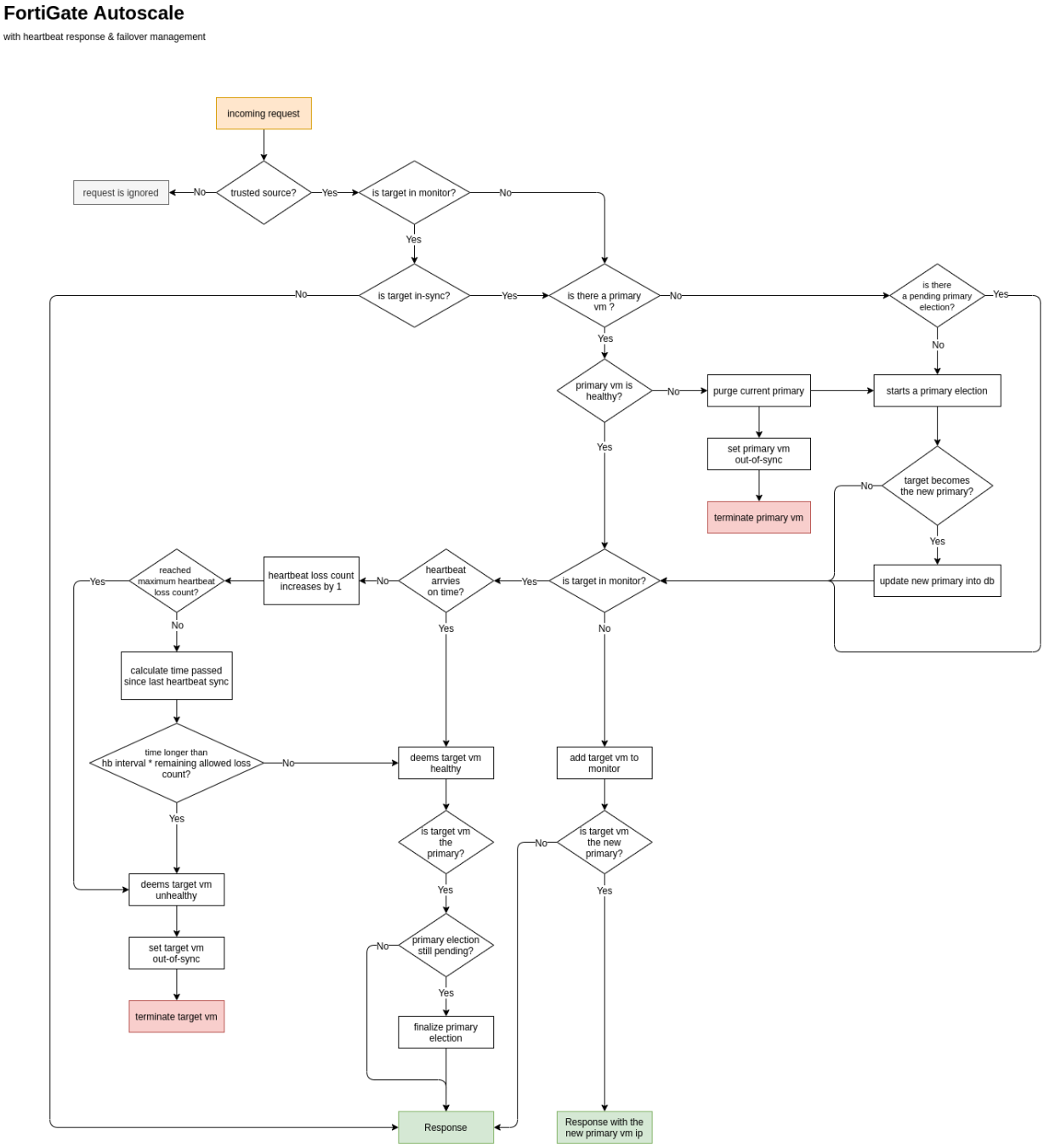
Architectural diagrams

The following diagrams illustrate the different aspects of the architecture of FortiGate Autoscale for AWS.

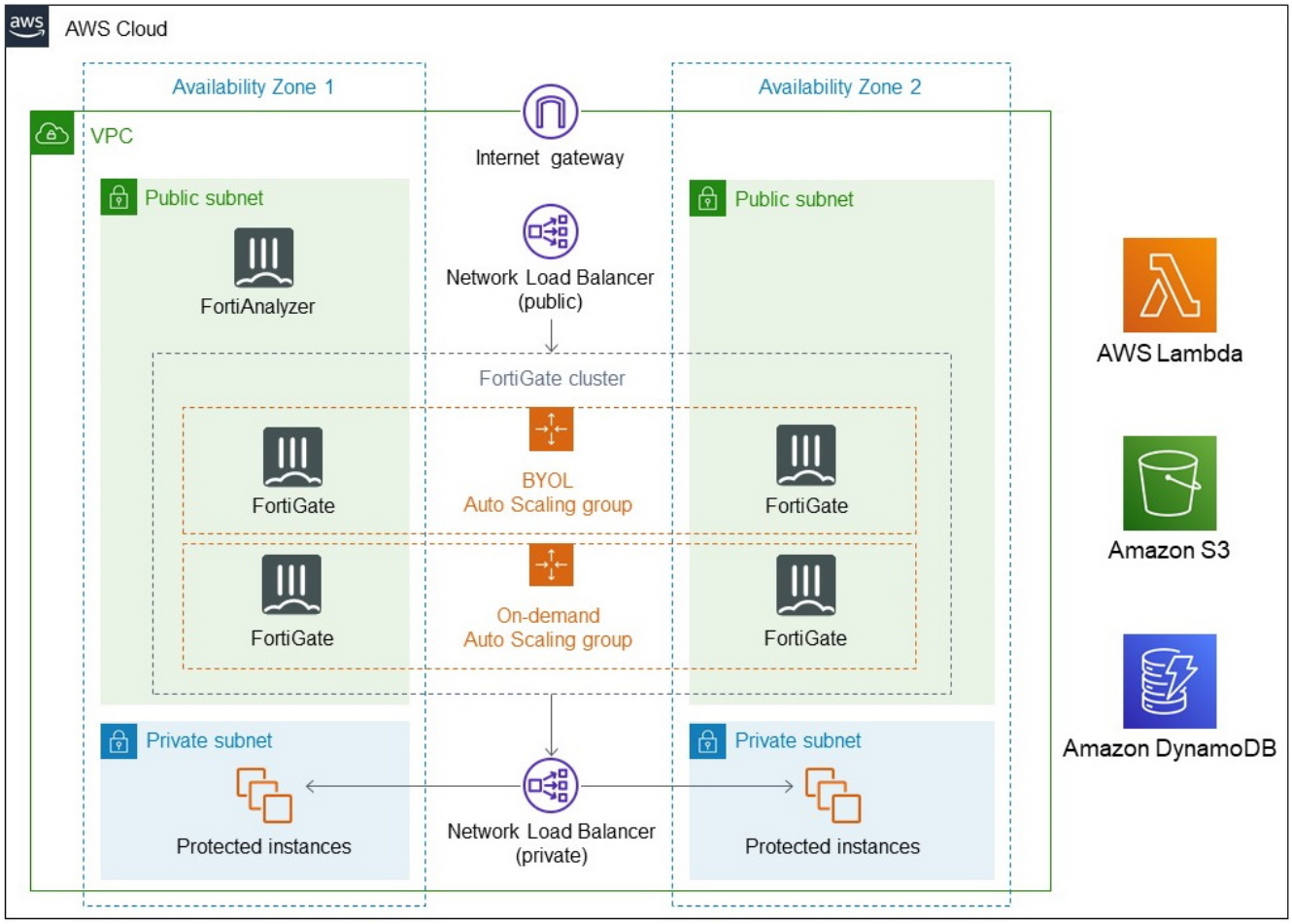
Autoscale handler flowchart



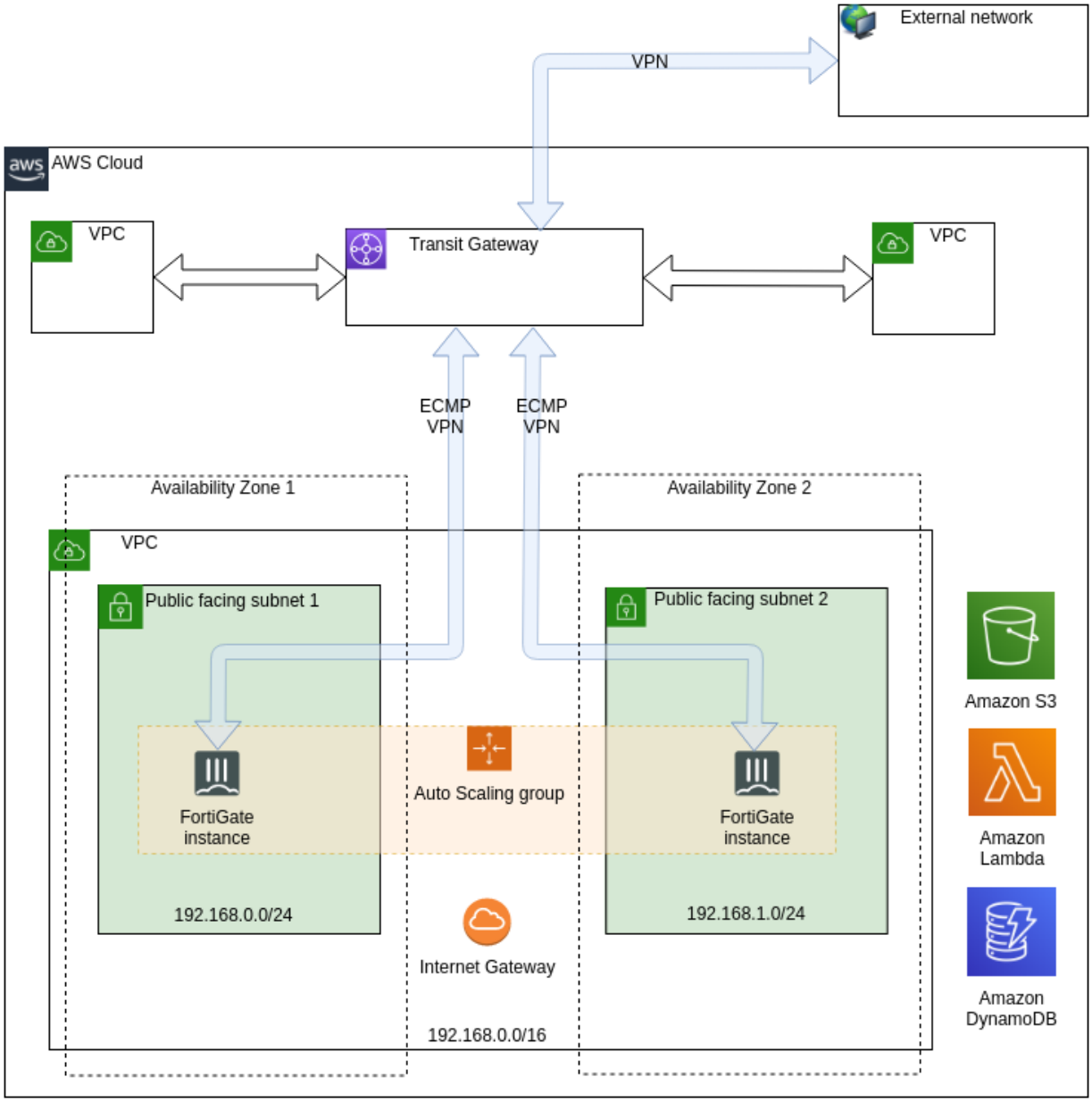
Primary election



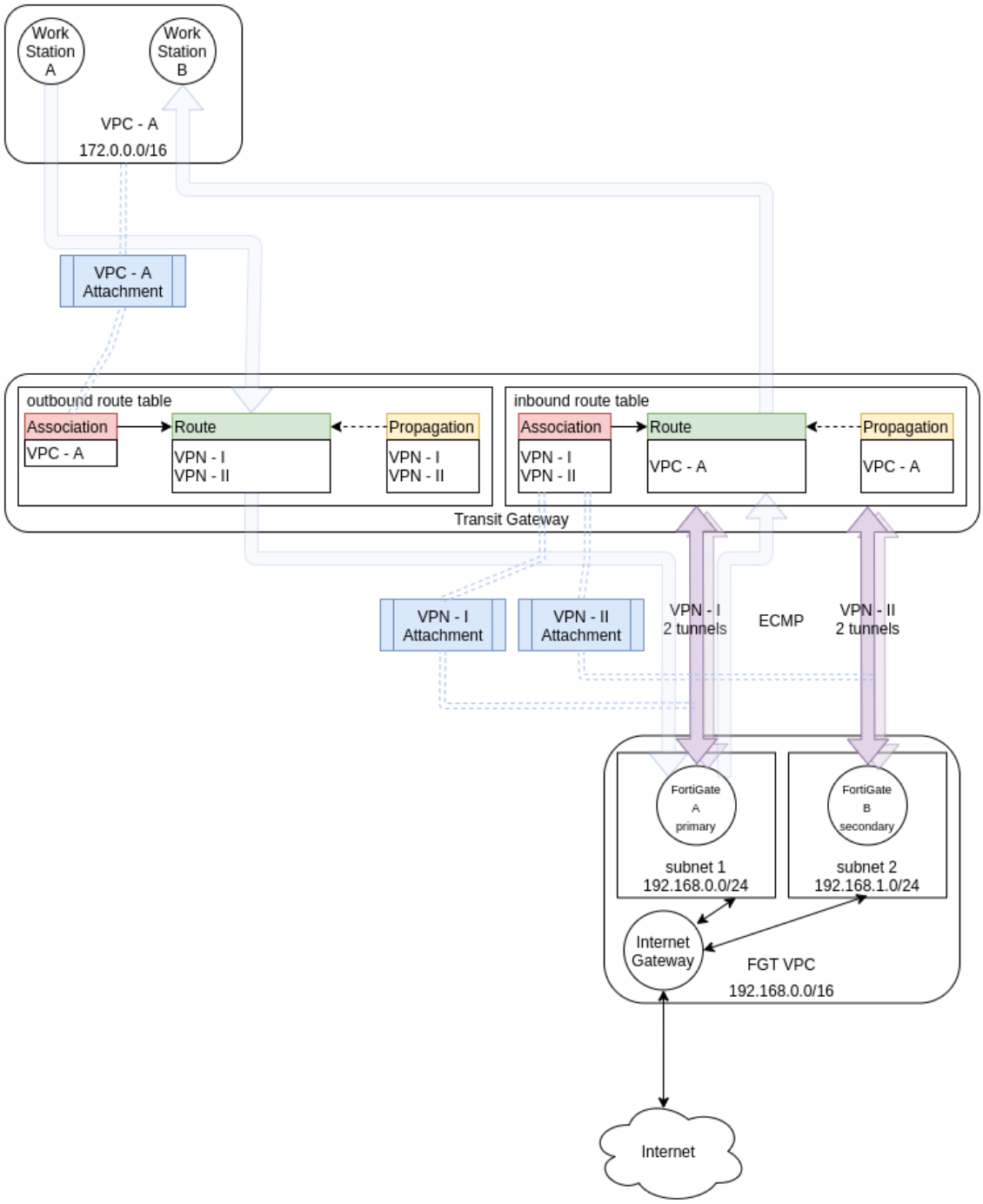
FortiGate Autoscale VPC



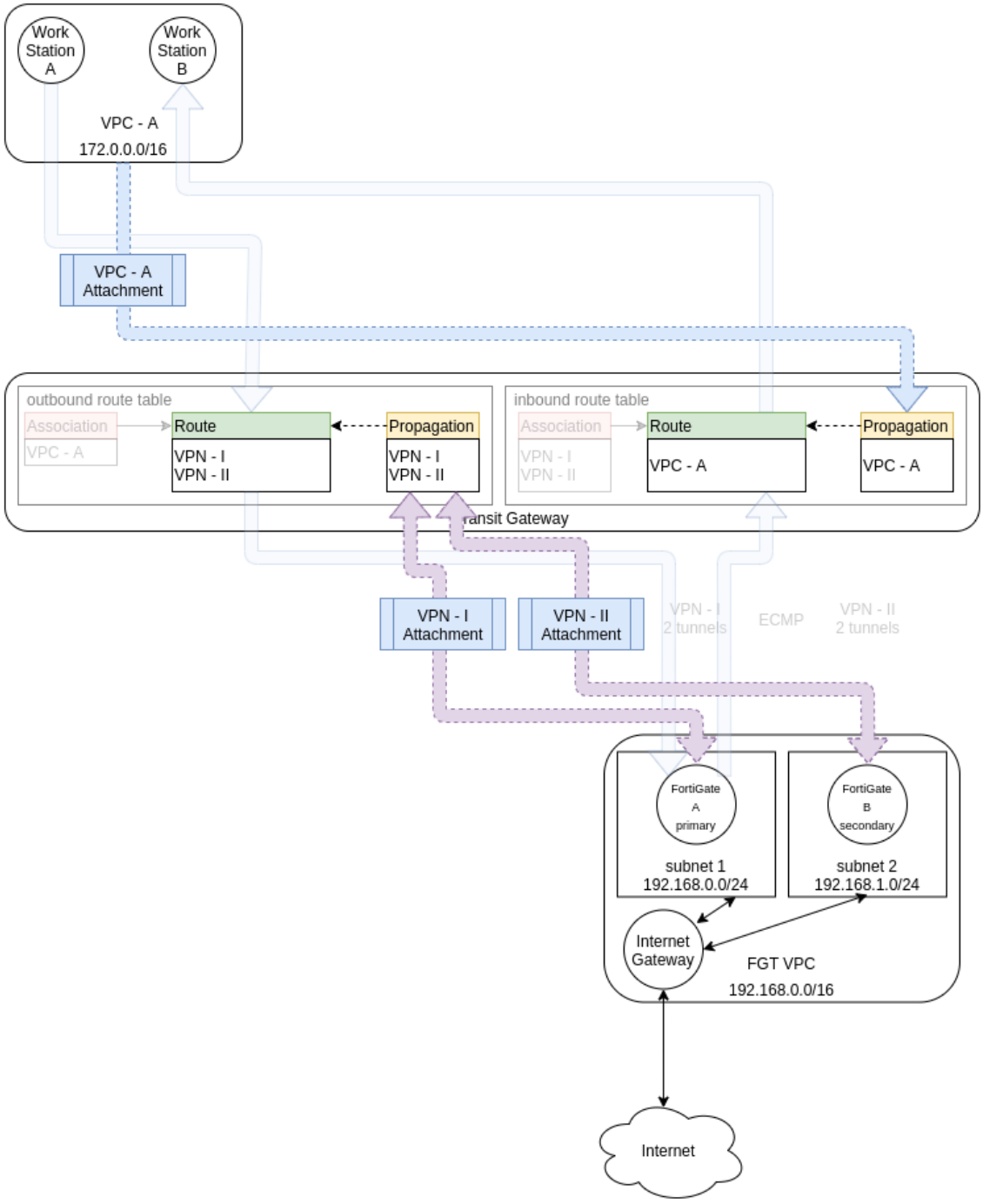
FortiGate Autoscale VPC attached to a Transit Gateway



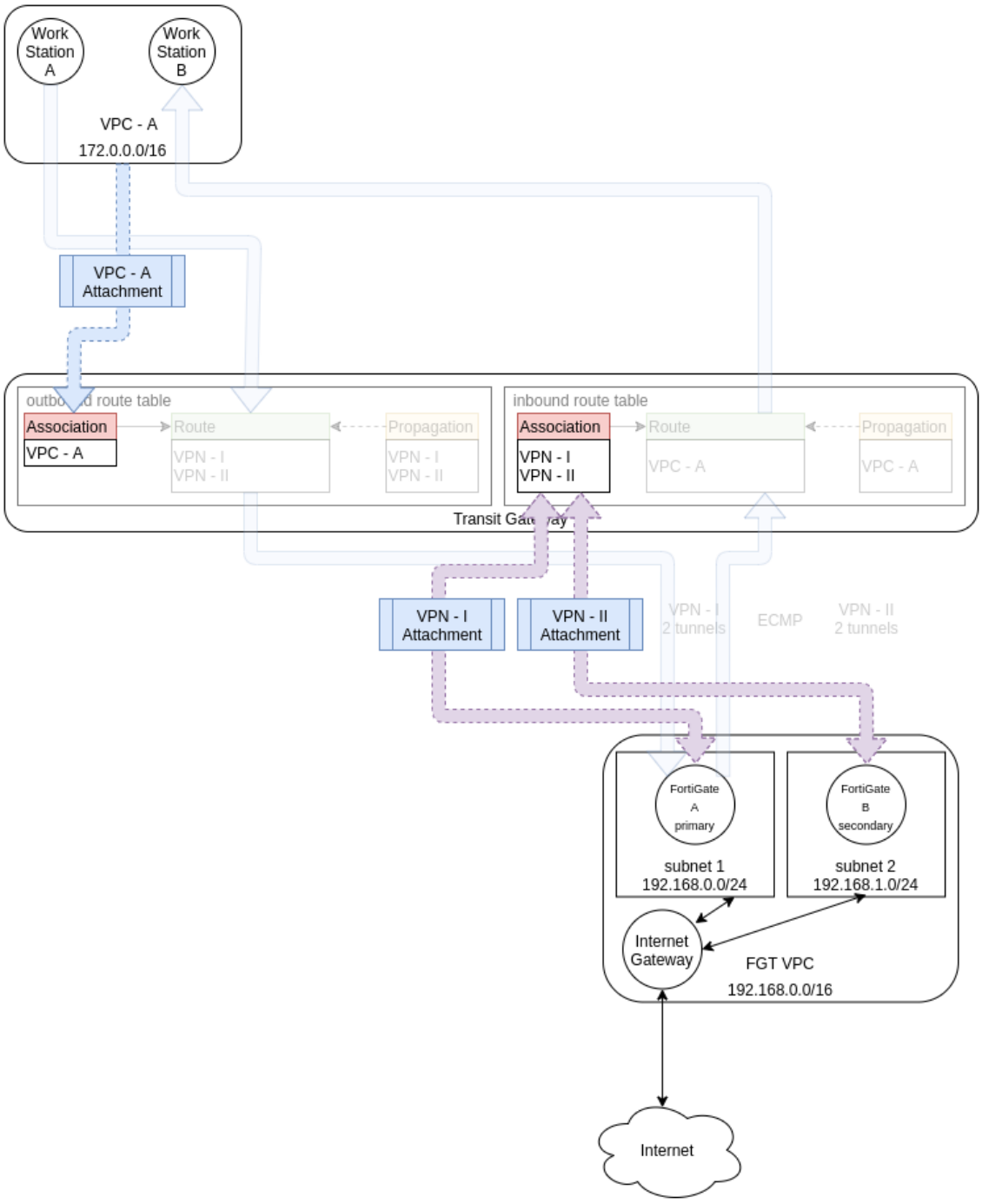
FortiGate Autoscale VPC integration with Transit Gateway



Route propagation



Route associations



Upgrading the deployment

The following provides steps to apply firmware updates to the FortiGate instances that the AWS Autoscaling deployment deployed.

Back up all FortiGate configurations prior to upgrading the FortiGate instances.

To upgrade the deployment:

1. Edit the autoscaling group to suspend the health check, launch, and terminate processes:
   a. In the AWS management console, go to *EC2* > *Auto Scaling* > *Auto Scaling Groups*.
   b. Edit the desired pay-as-you-go (PAYG) and/or bring your own license (BYOL) autoscaling group.
   c. On the *Details* tab, go to *Advanced Configurations*, then click *Edit*.
   d. From the *Suspended Processes* dropdown list, select *Health Check*, *Launch*, and *Terminate*.
   e. Click *Update* to save the changes.

   Using the *Instance Refresh* option is not recommended, as this is designed for truly ephemeral instances, which the FortiGate instances may not be.

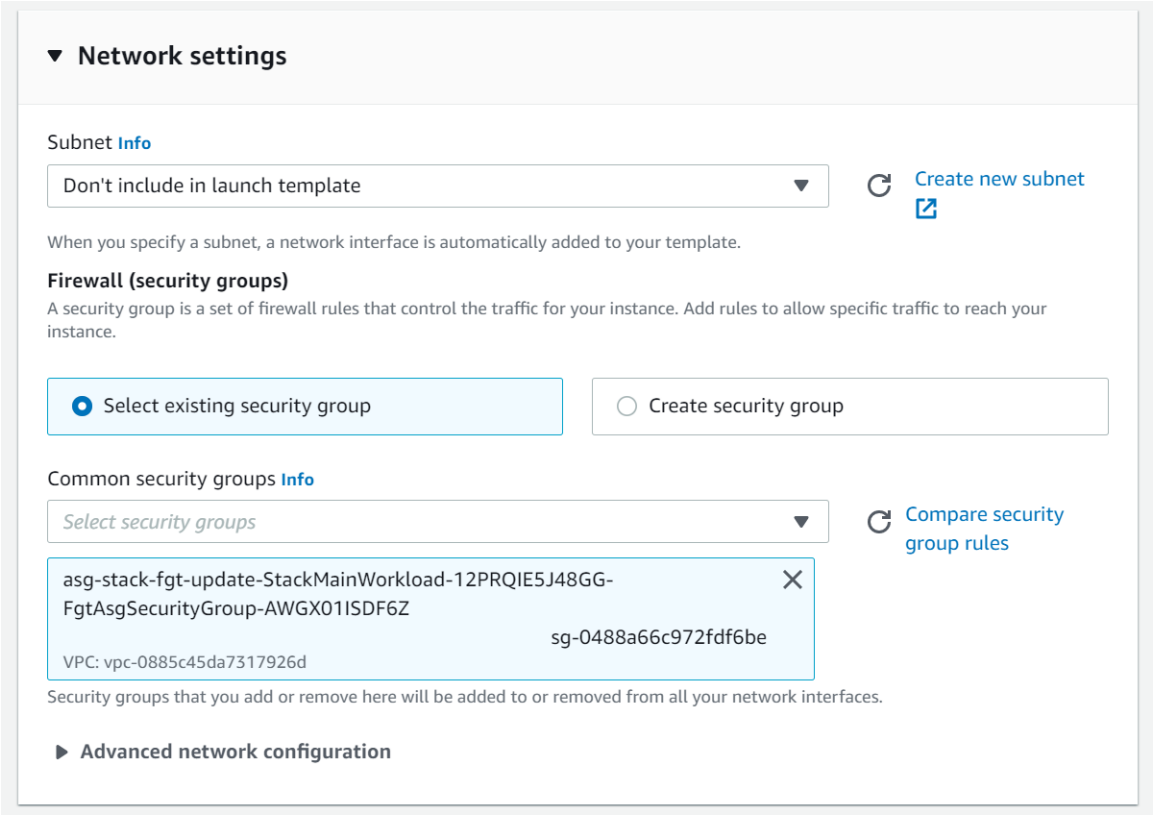
2. Confirm the new AMI ID for PAYG or BYOL as desired for your region.

   You can find the specific FortiGate AMI ID by going to the marketplace listing for FortiGate PAYG or BYOL, selecting *Subscribe*, continuing to configuration and confirming the desired region, then copying the AMI ID.

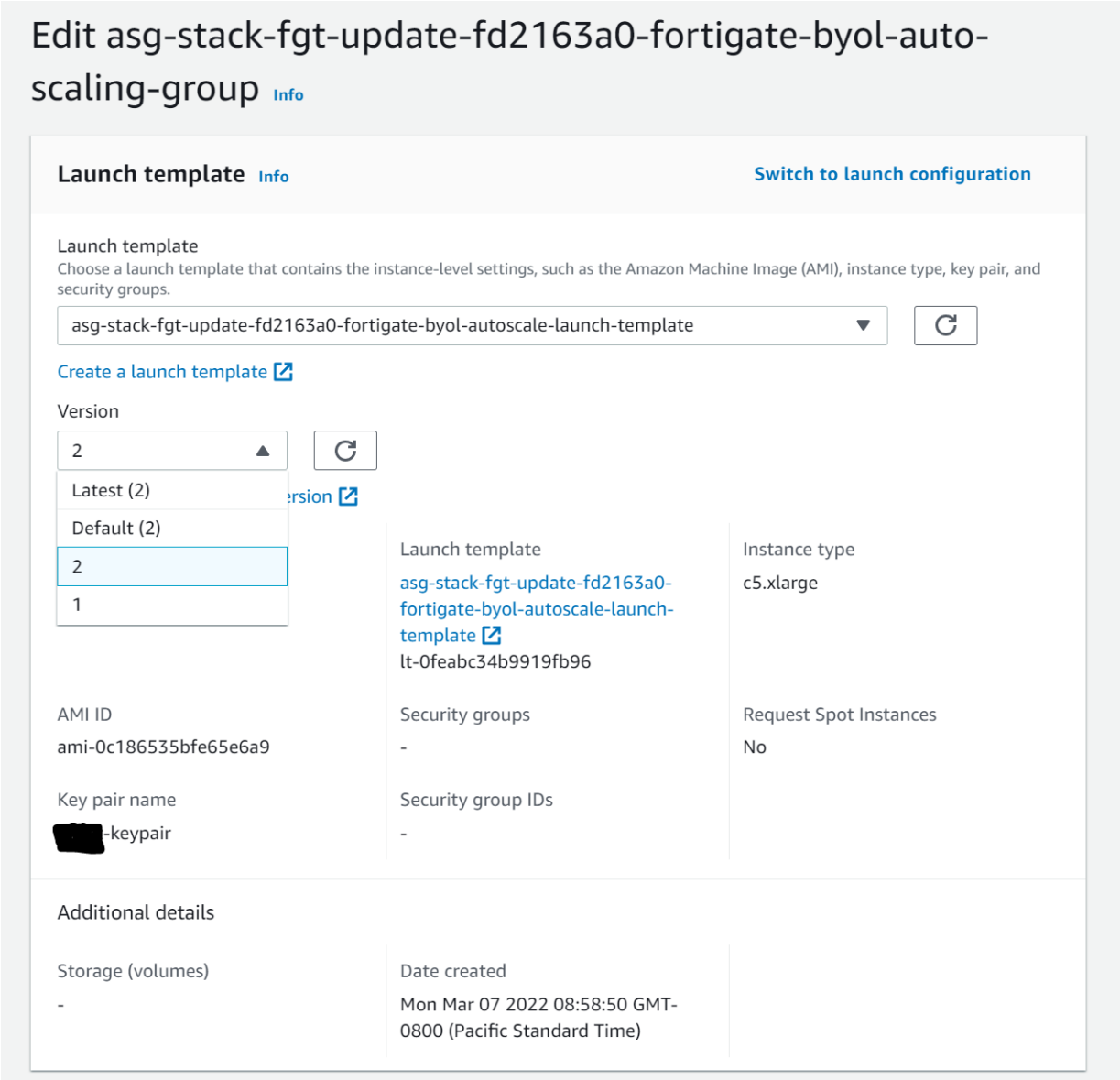
3. Edit the launch template or create a new one. You will need to create a new template version that references the new FortiGate version's AMI ID, so that autoscaling uses the new version for new instances:
   a. Go to *EC2* > *Instances* > *Launch Templates*.
   b. Select the desired launch template for FortiGate BYOL and PAYG.
   c. From the *Actions* menu, select *Modify Template (Create new version)*.
   d. Under *Application and OS Images*, paste the AMI ID that you confirmed in step 2 in the searchbar.
   e. Select the desired FortiGate marketplace offering.
   f. Click *Continue*. EC2 may display a warning that your security group rules may be overridden if you proceed. Under *Network settings* > *Firewall (security groups)*, click *Select existing security group*, and select the

previously selected security group before saving or creating a new version of the launch template.



4. Edit the BYOL or PAYG autoscaling group and update the launch template version to the new version:
   a. Go to *EC2 > Auto Scaling > Auto Scaling Groups*.
   b. Select the desired scaling group.
   c. In *LAUNCH TEMPLATE*, select *Edit*.

d. From the *Version* dropdown list, select the new version.



e. Click *Update*.

5. Manually apply the update to existing instances. Starting the upgrade process on the secondary autoscale FortiGates, then the primary FortiGate, is recommended. The firmware upgrade option is only available when logged in with administrator read-write privileges. Do one of the following:
   a. In FortiOS, go to *System* > *Firmware*. Select *FortiGuard Firmware*, then *Backup Config*. Upgrade to the latest available firmware.
   b. Log in to FortiOS as the admin user. Go to *System Firmware*. Under *Upload Firmware*, browse to and locate the previously downloaded firmware image file. Click *Backup config and upgrade*. The FortiGate backs up the current configuration to the management computer, uploads the firmware image file, upgrades to the new firmware version, and restarts. This process takes a few minutes.
6. Resume health check, launch, and terminate processes:
   a. Go to *EC2* > *Auto Scaling Groups*.
   b. Edit the desired autoscaling group.

c. Go to *Advanced Configuration* > *Edit*.
d. Deselect *Health Check*, *Launch Instances*, and *Terminate Instances*.
e. Click *Update*.

FortiGate Autoscale for AWS document history

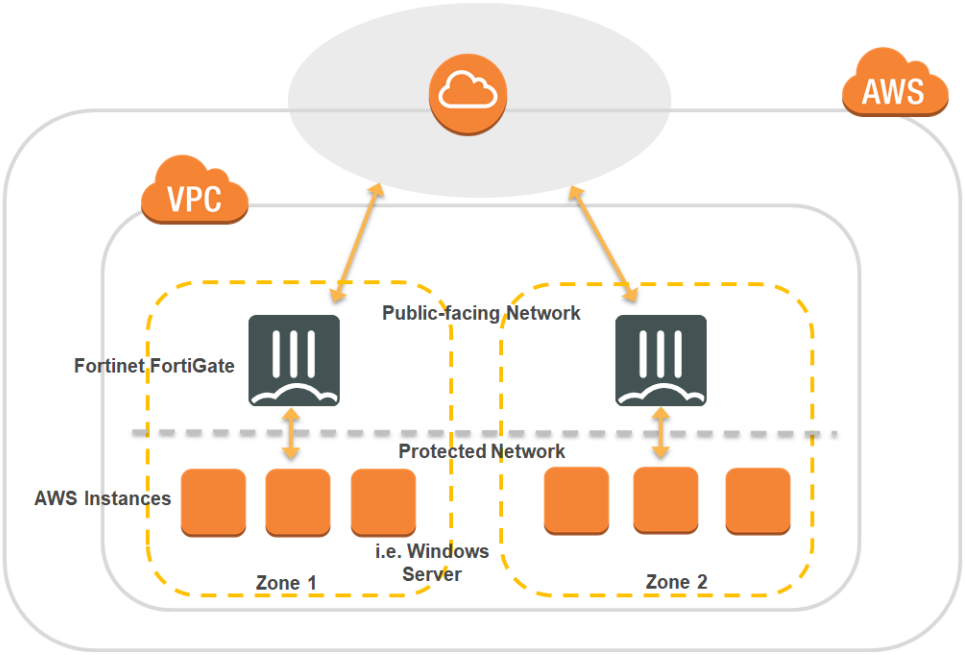
Template	Details
3.4.0	Updated the CFT parameters on page 50.
3.3.2 (latest)	Added support for FortiOS 6.2.5, FortiOS 6.4.6, FortiOS 7.0.0, and FortiOS 7.0.1. Removed support for FortiOS 6.2.3 and FortiOS 6.4.4. Added support for FortiAnalyzer 6.4.6. Removed support for FortiAnalyzer 6.2.5 and FortiAnalyzer 6.4.4.
3.3.1	Added Requirements when using an existing VPC on page 42.
3.3	Added support for AWS GovCloud (US). VPN connections now use Diffie-Hellman Group 14 and SHA256 (Secure Hash Algorithm 2); increased stack security.
3.2	Added support for FortiOS 6.4.4. FortiAnalyzer can now be integrated into the deployment.
3.0	Supports any combination of BYOL and On-demand instances as well as the option for Transit Gateway integration. Requires FortiOS 6.2.3.

2.0	Added support for Hybrid Licensing (any combination of BYOL and/or On-demand instances) with no Transit Gateway integration. Transit Gateway support is only for On-demand instances.
	Documentation is no longer maintained and is only available as a PDF:
	• Deploying auto scaling on AWS without Transit Gateway integration 2.0
	  • Requires FortiOS 6.2.3.
	• Deploying auto scaling on AWS with Transit Gateway integration 1.0
	  • Requires FortiOS 6.2.1.

1.0	Supports auto scaling for On-demand instances; does not support Transit Gateway integration. Requires FortiOS 6.0.6 or FortiOS 6.2.1. Documentation is no longer maintained and is only available as a PDF: • Deploying auto scaling on AWS 1.0

Single FortiGate-VM deployment

You can deploy the FortiGate-VM enterprise firewall for AWS as a virtual appliance in AWS (IaaS). This section shows you how to install and configure a single instance FortiGate-VM in AWS to provide a full NGFW/unified threat management security solution to protect your workloads in the AWS IaaS.



Networking is a core component in using AWS services, and using VPCs, subnets, and virtual gateways help you to secure your resources at the networking level.

This section covers the deployment of simple web servers, but you can use this type of deployment for any type of public resource protection, with only slight modifications. With this architecture as a starting point, you can implement more advanced solutions, including multitiered solutions.

In the example, two subnets are created: Subnet1, which is used to connect the FortiGate-VM to the AWS virtual gateway on the public-facing side, and Subnet2, which is used to connect the FortiGate-VM and the Windows server on the private side.

Determining your licensing model

On-demand users do not need to register from the FortiGate-VM GUI console. If you are using an on-demand licensing model, once you create the FortiGate-VM instance in AWS, contact Fortinet Customer Support with the following information:

- Your FortiGate-VM instance serial number
- Your Fortinet account email ID. If you do not have a Fortinet account, you can create one at Customer Service & Support.

If you are deploying a FortiGate-VM in the AWS marketplace with BYOL, you must obtain a license to activate it.

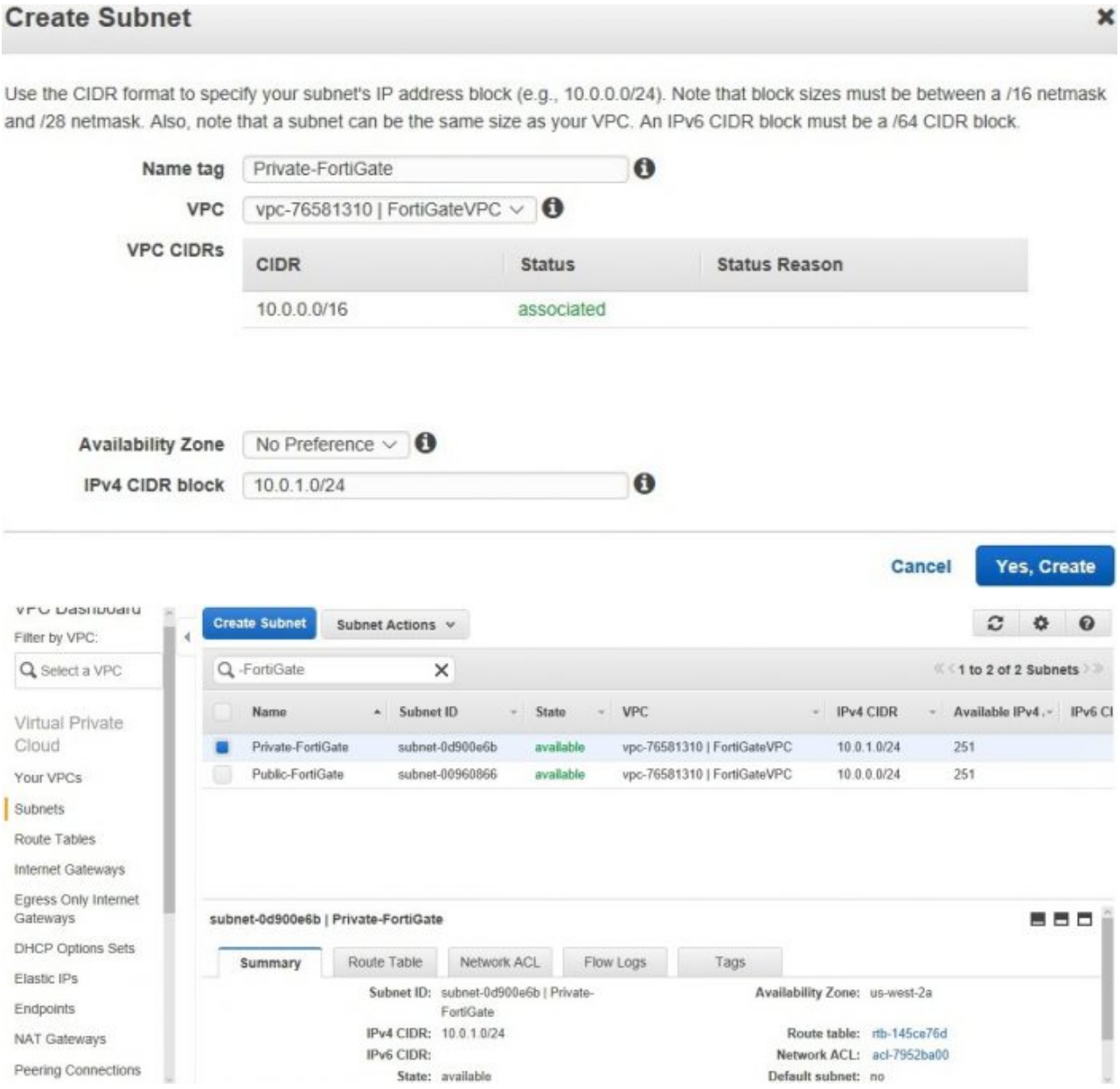
See Creating a support account on page 12.

Creating a VPC and subnets

This section shows you how to create an AWS VPC and create two subnets in it. For many steps, you have a choice to make that can be specific to your own environment.

To create a VPC and subnets:

1. Log in to the AWS Management Console.
2. Go to *Networking & Content Delivery* > *VPC*.
3. Go to *Virtual Private Cloud* > *Your VPCs*, then select *Create VPC*.
4. In the *Name tag* field, set the VPC name.
5. In the *CIDR block* field, specify an IPv4 address range for your VPC.
6. In the *Tenancy* field, select *Default*.
7. Select *Yes, Create*.
8. In the *Virtual Private Cloud* menu, select *Subnets*, then select *Create Subnet*. Create a public subnet (in this example, *Subnet1*) and a private subnet (*Subnet2*), as shown in this example. Both subnets belong to the VPC that you created.



Attaching the new VPC Internet gateway

If you are using the default VPC, the Internet gateway should already exist.

To attach the new VPC Internet gateway:

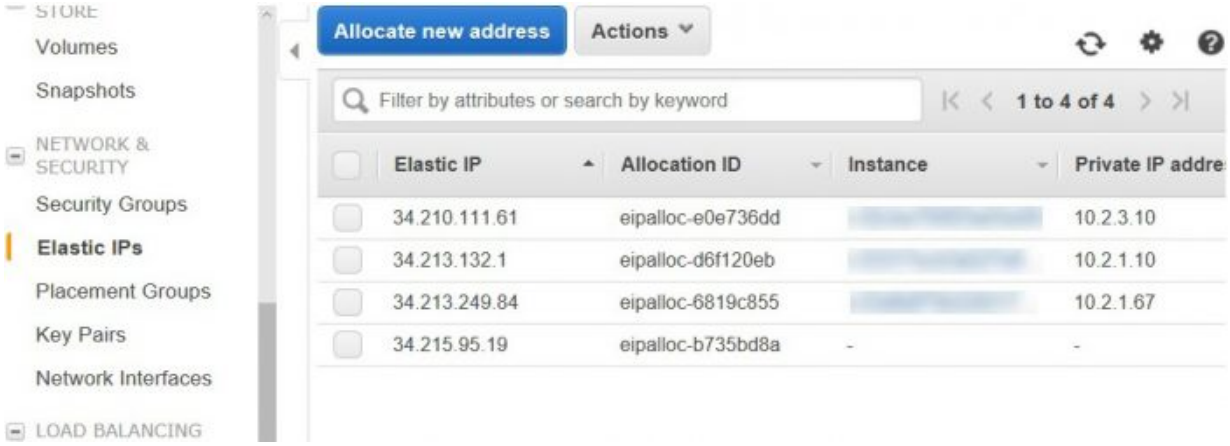
1. In the *Virtual Private Cloud* menu, select *Internet Gateways*, then select *Create Internet Gateway*.
2. In the *Name tag* field, set the Internet gateway name, then select *Yes, Create*.
3. Select the Internet gateway, then select *Attach to VPC*.

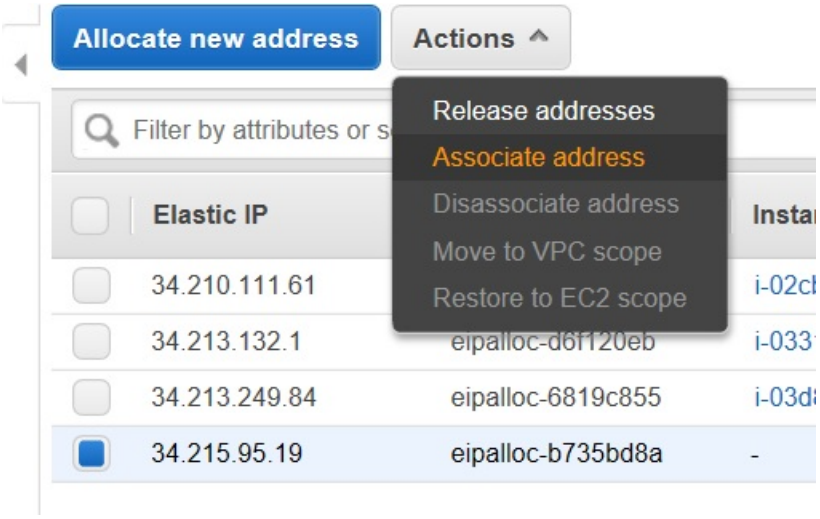
4. Select the VPC that you created and select *Yes, Attach*. The Internet gateway state changes from *detached* to *attached*.

Subscribing to the FortiGate

To subscribe to the FortiGate:

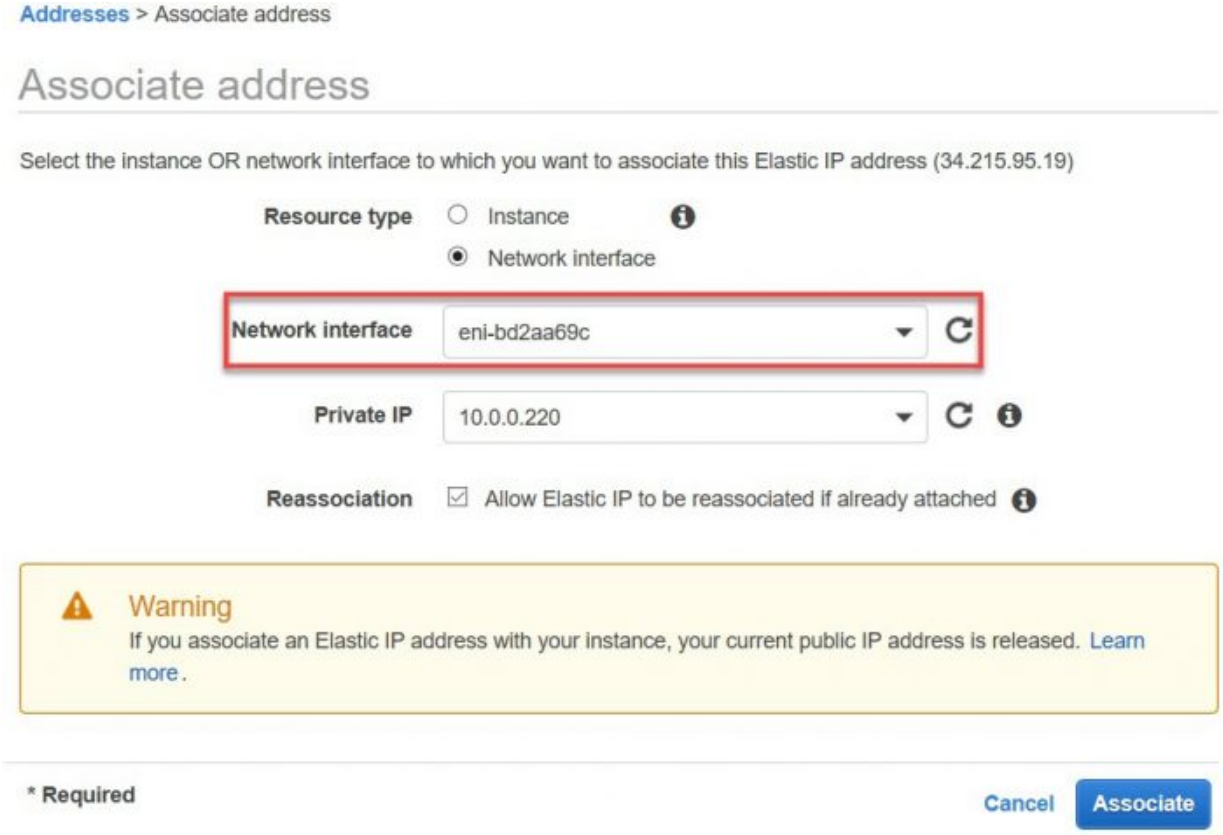
1. Go to the AWS Marketplace's page for Fortinet FortiGate-VM (BYOL) or FortiGate-VM (on-demand). Select *Continue*.
2. Select *Manual Launch*.
3. Select *Launch with EC2 Console* beside the region you want to launch.
4. Select an instance type, then select *Next: Configure Instance Details*.
5. Configure instance details:
   a. In the *Network* field, select the VPC that you created.
   b. In the *Subnet* field, select the public subnet.
   c. In the *Network interfaces* section, you will see the entry for *eth0* that was created for the public subnet. Select *Add Device* to add another network interface (in this example, *eth1*), and select the private subnet. It is recommended that you assign static IP addresses.
   d. When you have two network interfaces, an EIP is not assigned automatically. You must manually assign one later. Select *Review and Launch*, then select *Launch*.
6. Select an existing key pair or create a new key pair. Select the acknowledgment checkbox. Select *Launch Instances*.
7. To easily identify the instance, set a name for it in the *Name* field.
8. Since FortiOS 6.2.2, on-demand FortiGate-VMs require connectivity to FortiCare to obtain a valid license. Without connectivity to FortiCare, the FortiGate-VM shuts down for self-protection. Ensure the following:
   a. Outgoing connectivity to https://directregistration.fortinet.com:443 is allowed in security groups and ACLs.
   b. You have assigned a public IP address (default or EIP). If you have not enabled a public address during instance creation, follow the remaining steps to assign an EIP and bring up the FortiGate-VM again.
9. Configure an EIP:
   a. In the *Network & Security* menu, select *Elastic IPs*, then select one that is available for you to use or create one. Select *Actions* > *Associate Address*. If you do not have one available to use, create one.

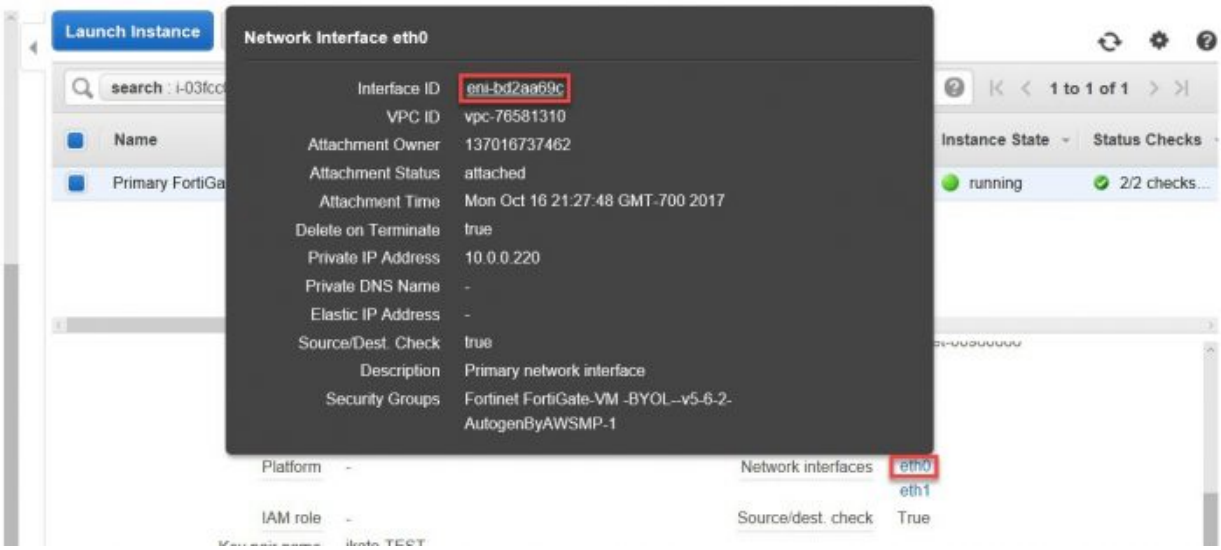




b. In the *Resource type* section, select *Network Interface*.

c. In the *Network interface* field, select the interface ID of the network interface that you created for the public subnet (in this example, *eth0*). In the *Private IP* field, select the IP address that belongs to the public subnet. To find these values, go to the EC2 Management Console, select *Instances*, and select the interface in the *Network interfaces* section in the lower pane of the page (*Interface ID* and *Private IP Address* fields). Select *Associate*. A message is displayed indicating the address association was successful. Note that if the Internet Gateway isn't associated with a VPC, the elastic IP assignment will fail.

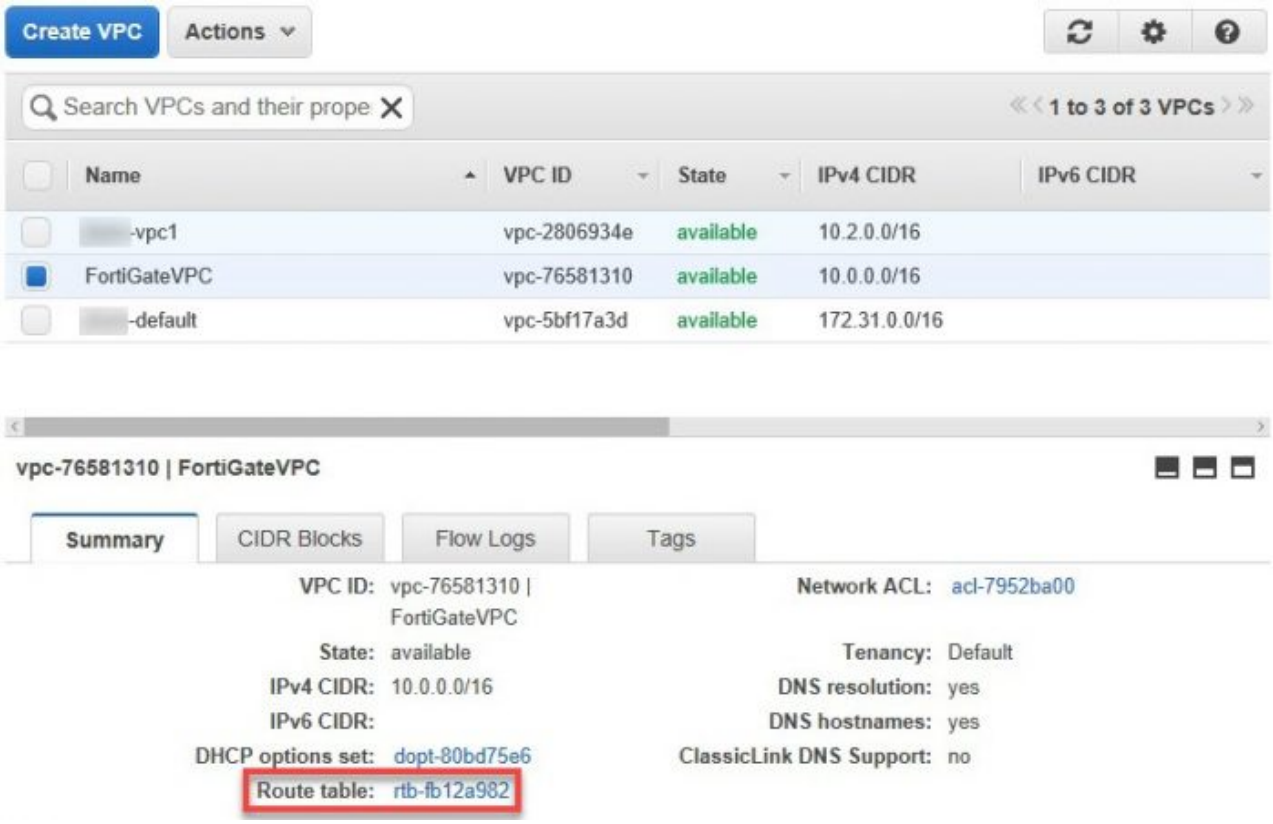




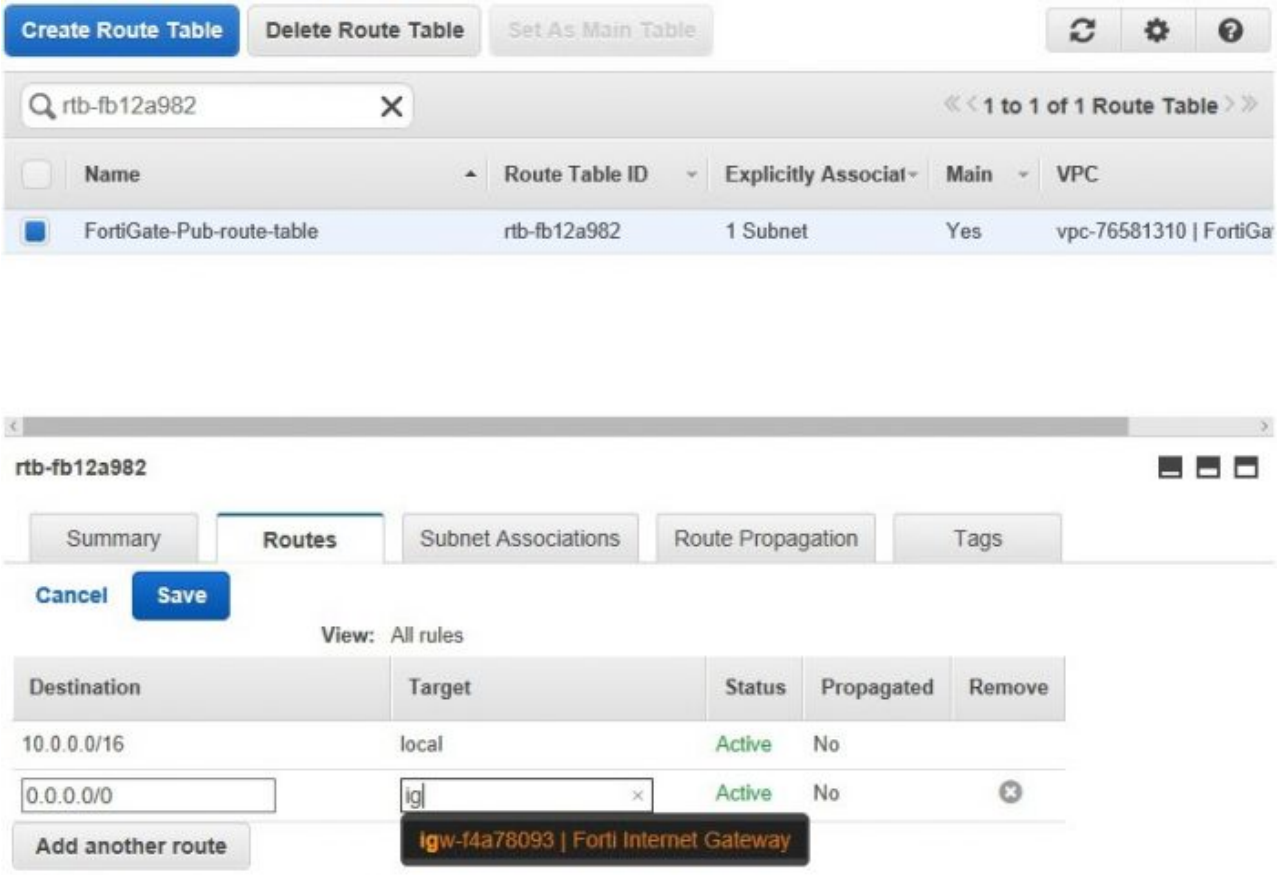
Creating routing tables and associate subnets

Configure the routing tables. Since the FortiGate-VM has two interfaces, one for the public subnet and one for the private subnet, you must configure two routing tables.

1. To configure the public subnet's routing table, go to *Networking & Content Delivery* > *VPC* in the AWSmanagement console. In the VPC Dashboard, select *Your VPCs*, and select the VPC you created. In the *Summary* tab in the lower pane, select the route table ID located in the *Route table* field. To easily identify the route table, set a name for it in the *Name* field.



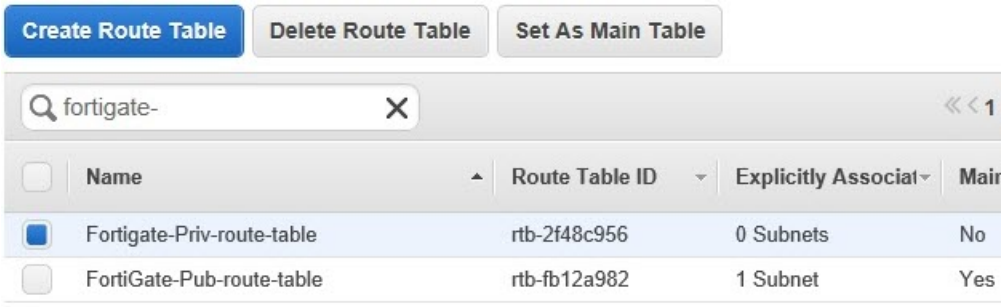
2. In the *Routes* tab, select *Edit*, then select *Add another route*. In the *Destination* field, type *0.0.0.0/0*. In the *Target* field, type *igw* and select the Internet Gateway from the auto-complete suggestions. Select *Save*. The default route on the public interface in this VPC is now the Internet Gateway.

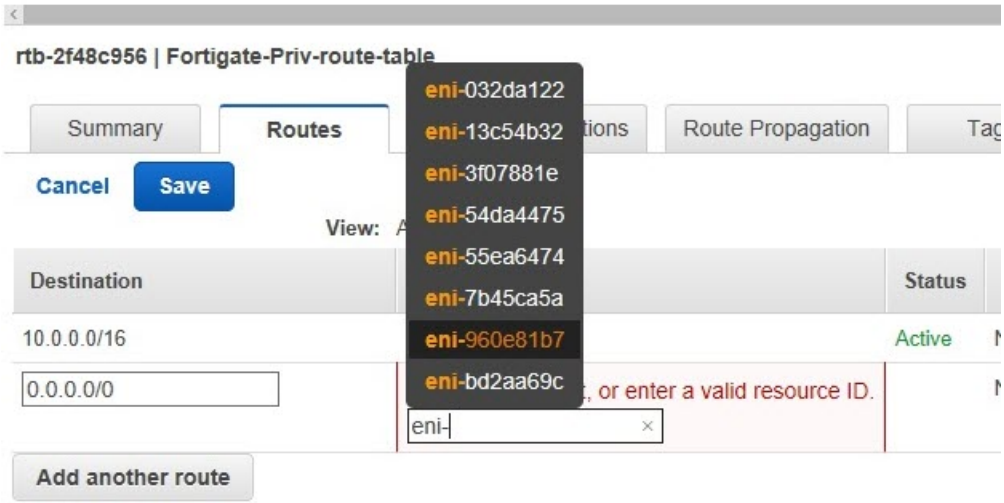


3. In the *Subnet Associations* tab, select *Edit*, and select the public subnet to associate it with this routing table. Select *Save*.

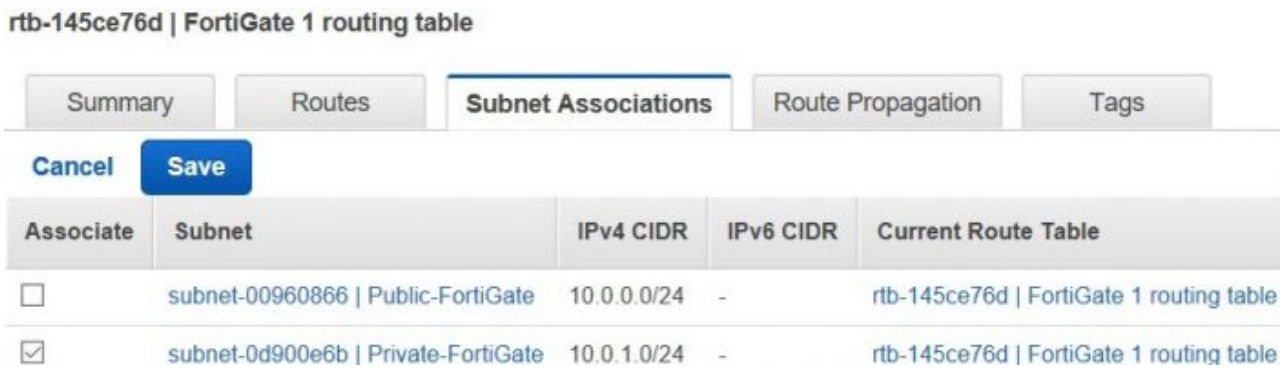
4. To configure the routing table for the private subnet, select *Create Route Table*. To easily identify the route table, set a name for it in the *Name* field. Select the VPC you created. Select *Yes, Create*.

5. In the *Routes* tab, select *Edit*, then select *Add another route*. In the *Destination* field, type *0.0.0.0/0*. In the *Target* field, enter the interface ID of the private network interface. To find the interface ID, go to the EC2 Management Console, select *Instances*, and select the interface in the *Network interfaces* section in the lower pane of the page (*Interface ID* field). Select *Save*. The default route on the private subnet in this VPC is now the private network interface of the FortiGate.

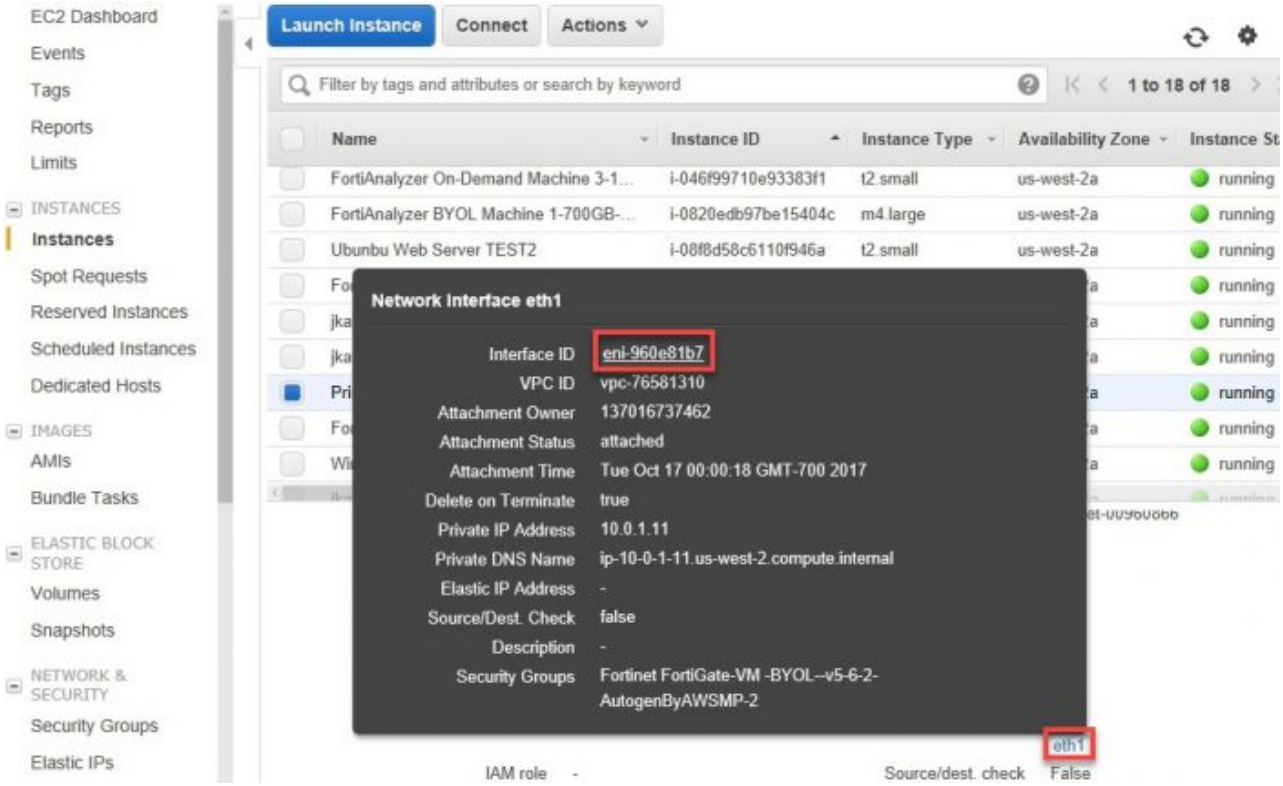




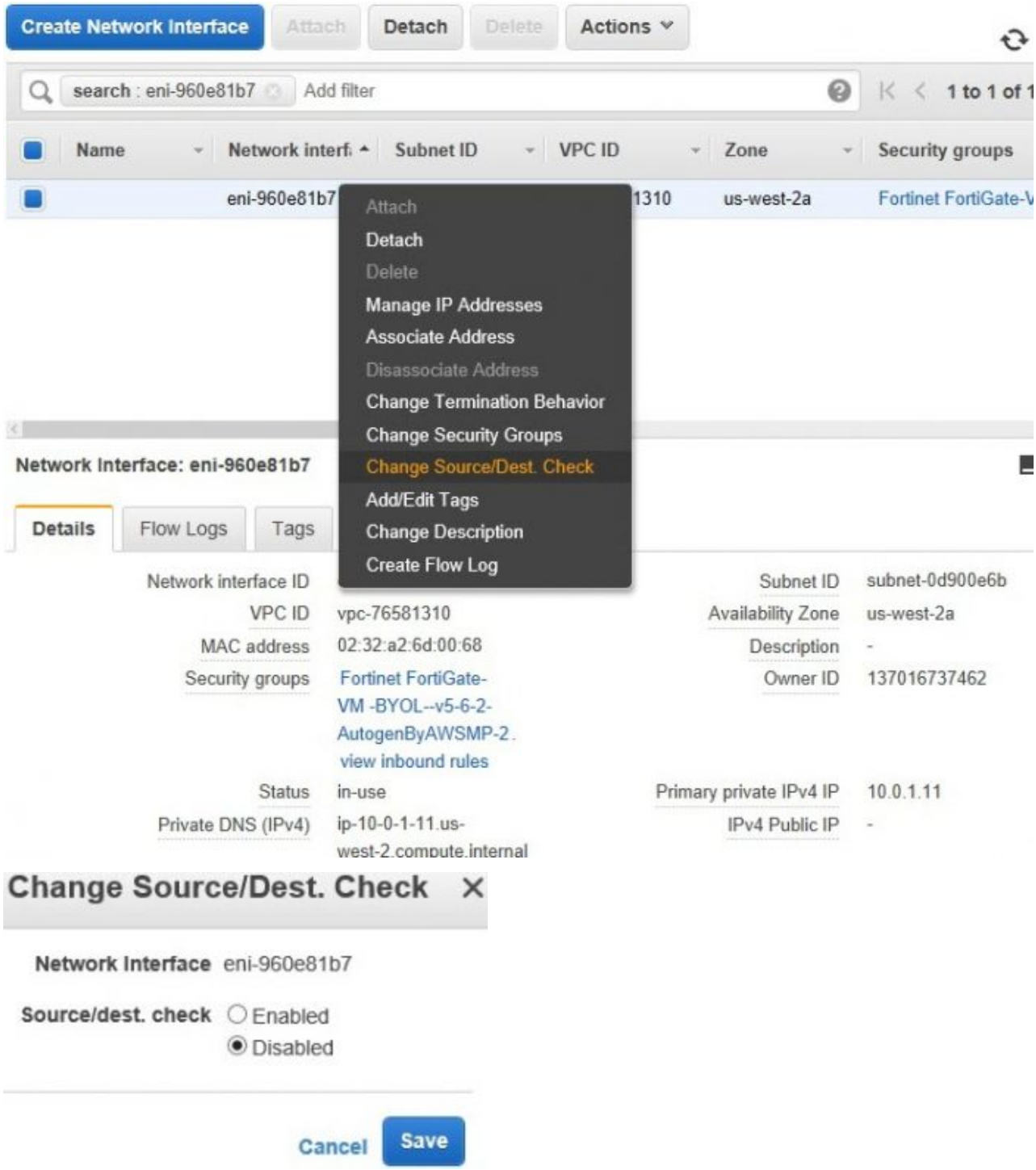
6. In the *Subnet Associations* tab, select *Edit*, select the private subnet to associate it with this routing table. Select *Save*. Two routing tables, one for the public segment and one for the private segment, have now been created with default routes.



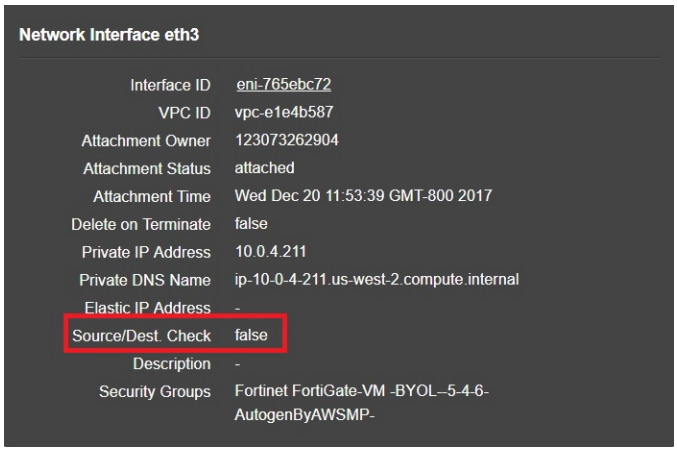
7. In the EC2 Management Console, select *Instances*, and select the network interface that you created for the private subnet (in this example, *eth1*) in the *Network interfaces* section in the lower pane. Select the interface ID.



8. Select the network interface, select the *Actions* dropdown list, select *Change Source/Dest. Check*. Select *Disabled*. Select *Save*.



If you have multiple network interfaces, Source/Dest. Check needs to be disabled in each interface. You can confirm by looking at the interface information shown as *false*.

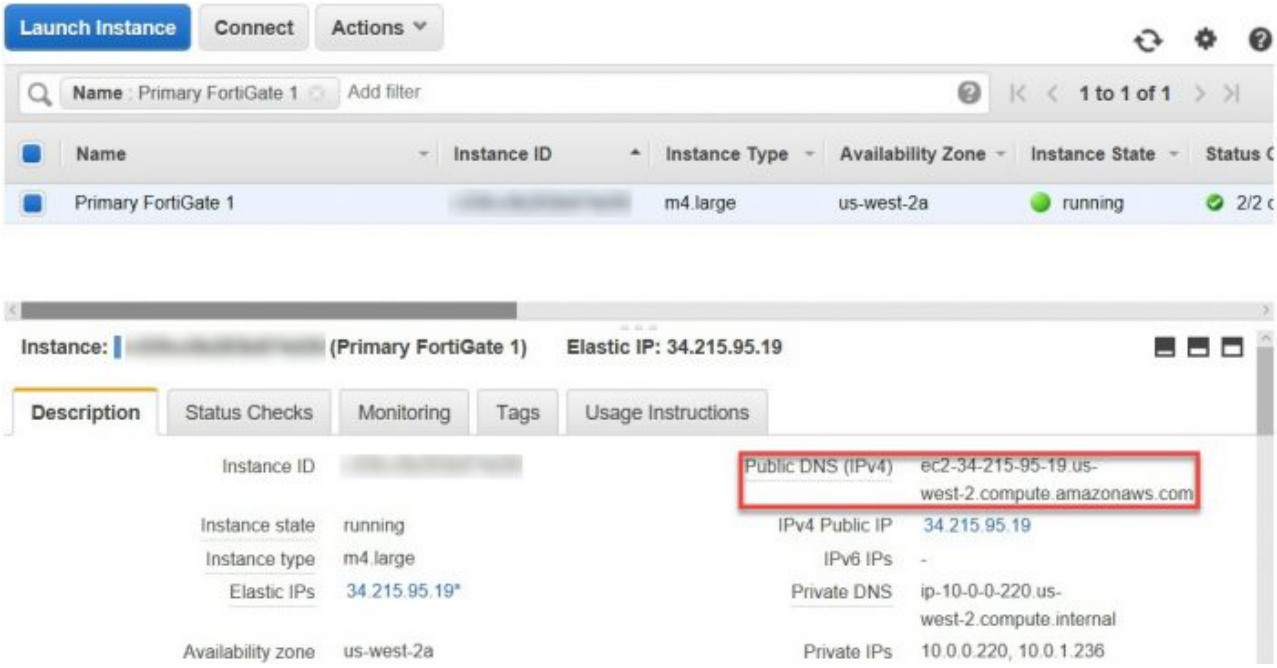


Connecting to the FortiGate-VM

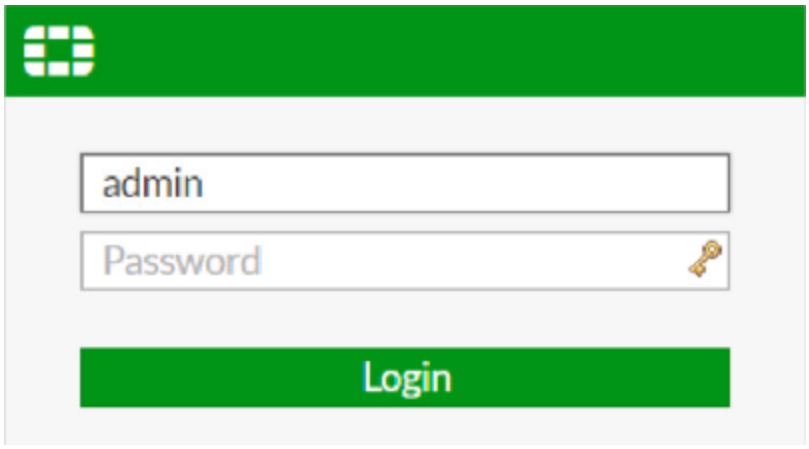
To connect to the FortiGate-VM, you need your login credentials and its public DNS address.

The default username is admin and the default password is the instance ID.

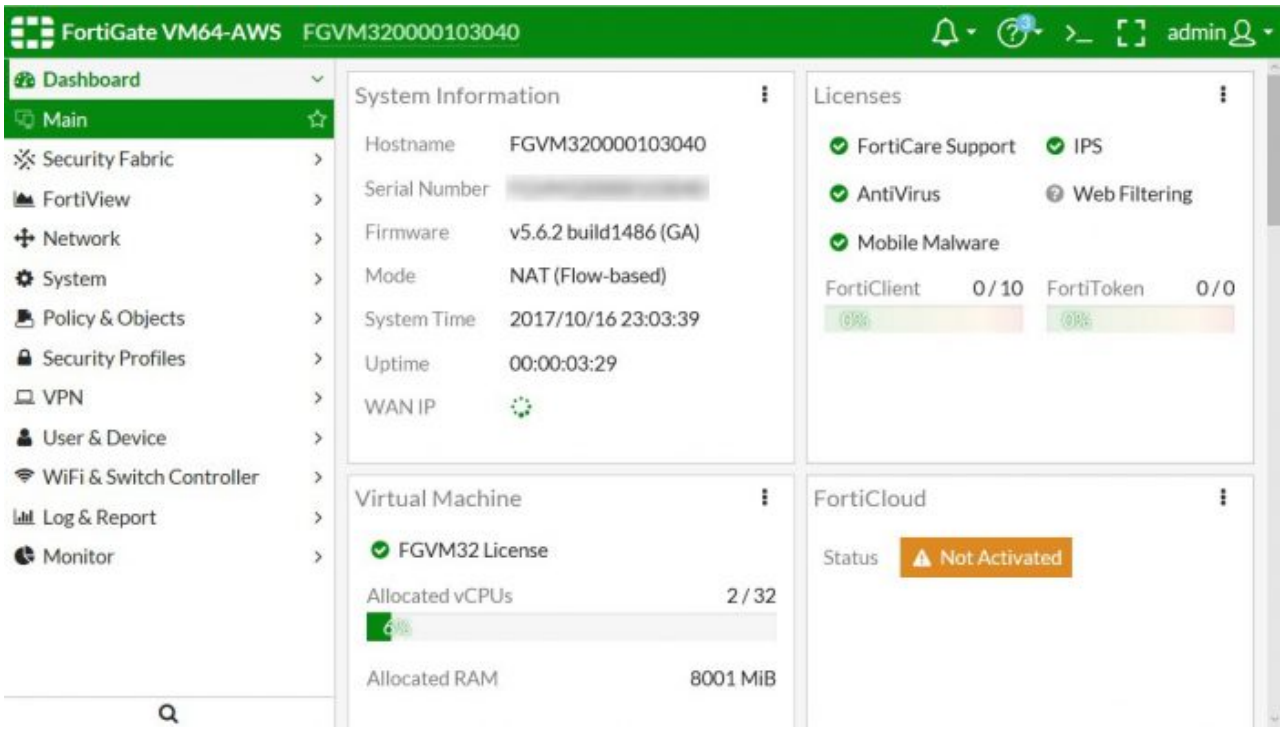
1. You can find the public DNS address in the EC2 management console. Select *Instances* and look at the *Public DNS (IPv4)* field in the lower pane. If you do not see the DNS address, you may need to enable DNS host assignment on your VPC. In this case, go back to the VPC management console, select *Your VPCs*, and select your VPC. Select the *Action* dropdown list, and select *Edit DNS Hostnames*. Select *Yes*. Select *Save*.



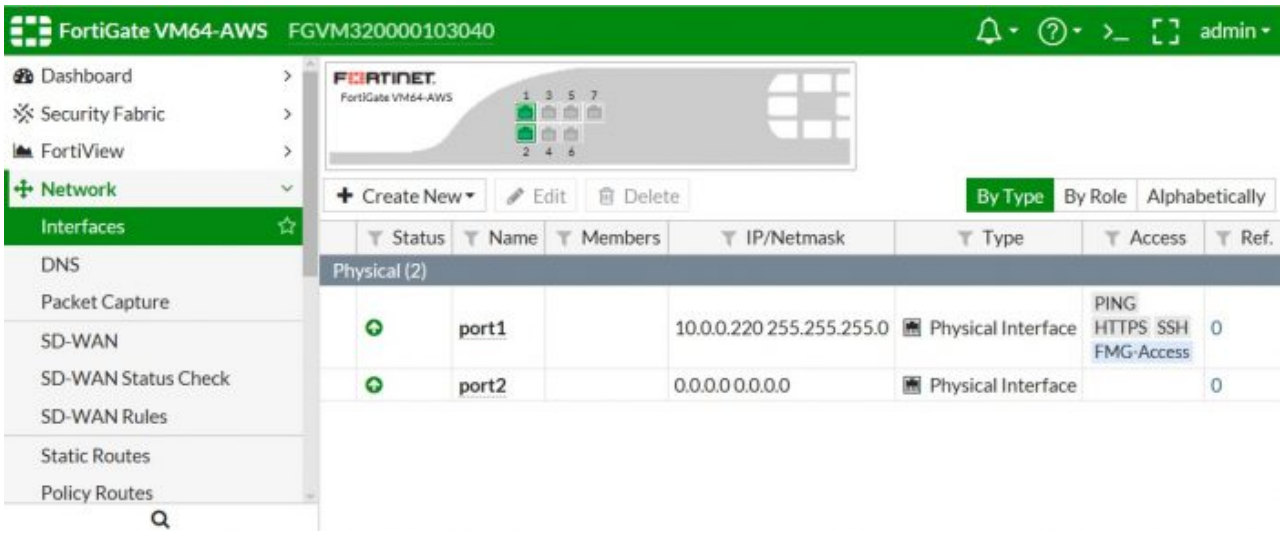
2. Open an HTTPS session using the public DNS address of the FortiGate-VM in your browser (https://<public DNS>). You will see a certificate error message from your browser, which is normal because the default FortiGate certificate is self-signed and isn't recognized by browsers. Proceed past this error. At a later time, you can upload a publicly-signed certificate to avoid this error. Log in to the FortiGate-VM with your username and password (the login credentials mentioned above).

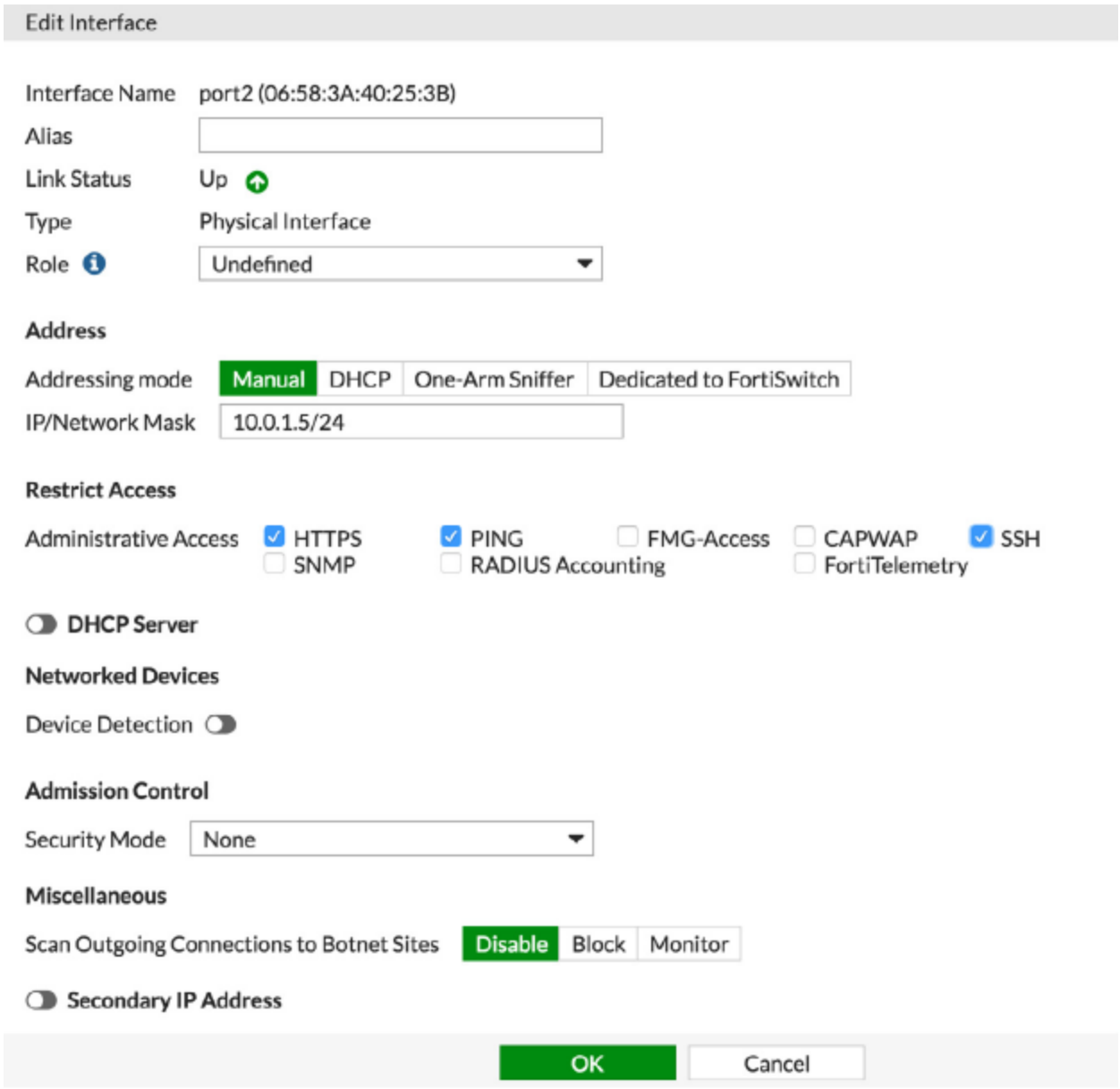


3. If you're using a BYOL license, upload your license (.lic) file to activate the FortiGate-VM. The FortiGate-VM will automatically restart. After it restarts, log in again.
4. You will now see the FortiGate-VM dashboard. Depending on your license type, the information in the license widget on the dashboard may vary.



5. Select *Network* > *Interfaces*, and edit the interfaces, if required. If the IP address or subnet mask is missing for port 1 or port 2, configure these values.

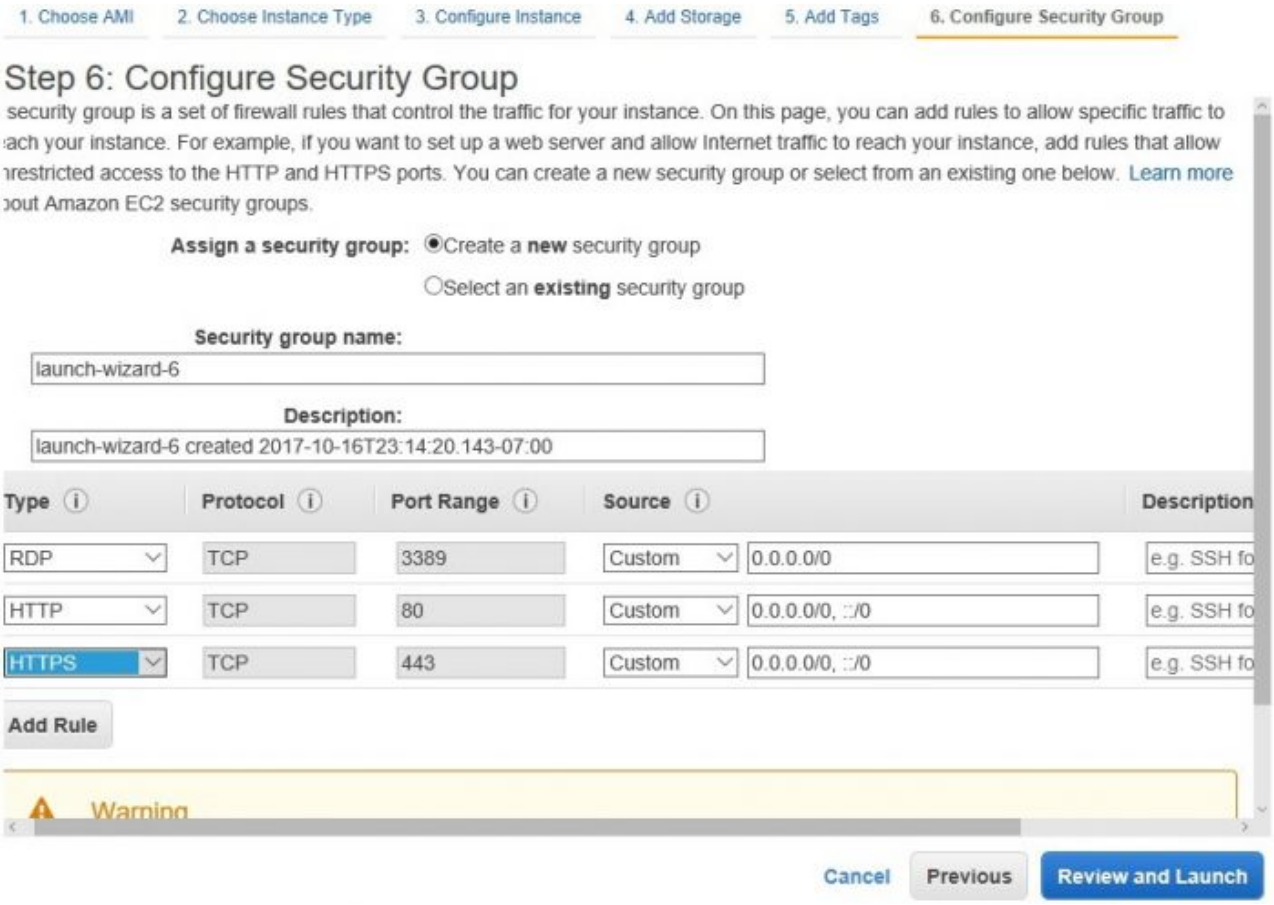

FortiGate VM64-AWS
FGVM320000103040
admin
Dashboard
Security Fabric
FortiView
Network
Interfaces
DNS
Packet Capture
SD-WAN
SD-WAN Status Check
SD-WAN Rules
Static Routes
Policy Routes
FORTINET
FortiGate VM64-AWS
Create New
Edit
Delete
By Type
By Role
Alphabetically
Status
Name
Members
IP/Netmask
Type
Access
Ref.
Physical (2)
port1
10.0.0.220 255.255.255.0
Physical Interface
PING
HTTPS SSH
FMG-Access
0
port2
0.0.0.0 0.0.0.0
Physical Interface
0



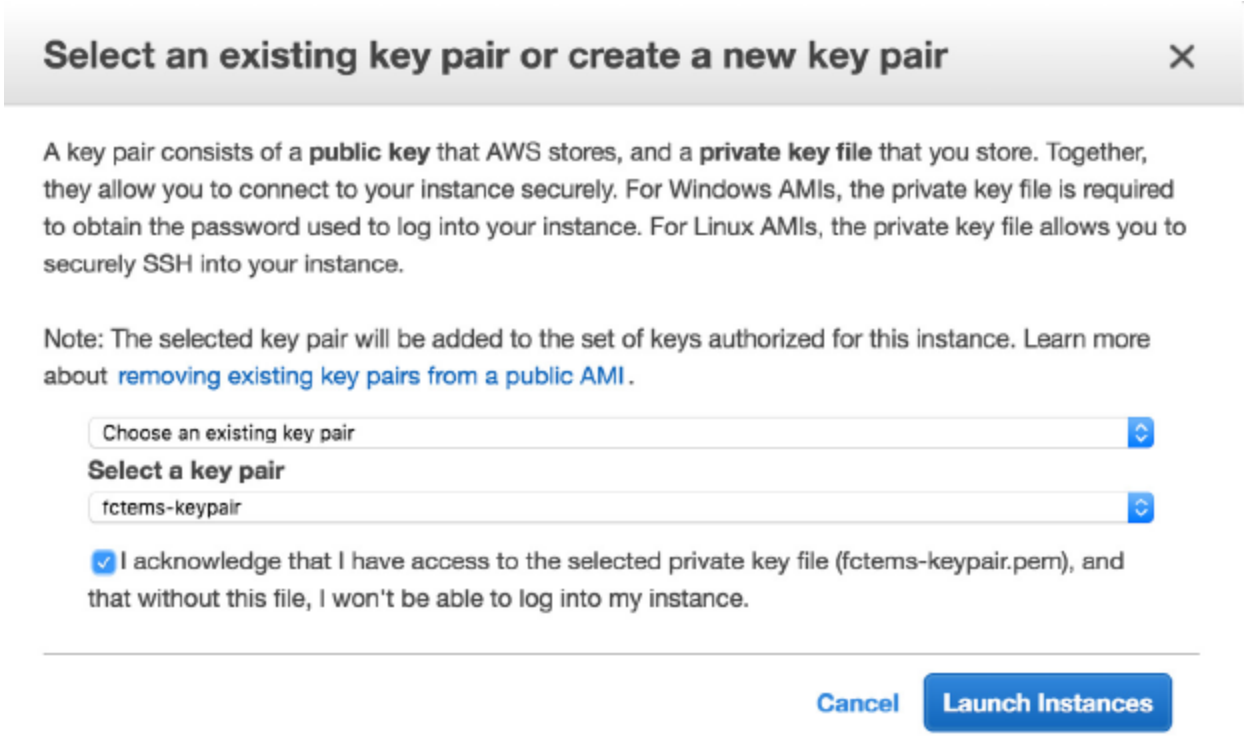
Setting up a Windows Server in the protected network

1. In the AWS management console, select EC2. Select *Launch Instance*, then select the *Microsoft Windows Server 2012 R2* that applies to your environment. You will use this to test connectivity with remote desktop access.

2. In the *Configure Instance Details* step, in the *Network* field, select the FortiGate-VM's VPC. In the *Subnet* field, select the private subnet.
3. In the *Configure Security Group* step, configure a security group for the Windows server so that it allows Internet access. In this example, we use Remote Desktop TCP port 3389, and other ports are optional. Select *Review and Launch*.



4. Select a key pair, select the acknowledgment checkbox, and select *Launch Instances*.



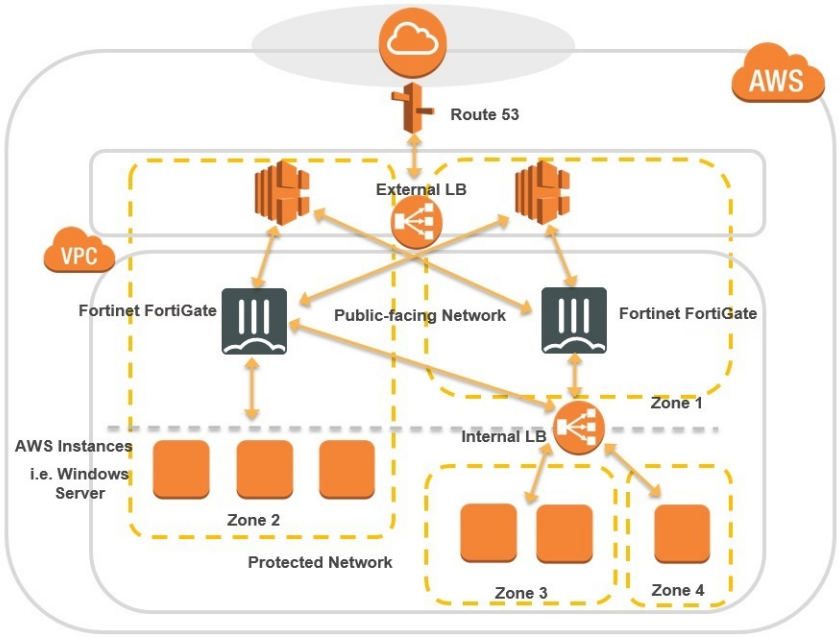
HA for FortiGate-VM on AWS

Deploying and configuring FortiGate-VM active-active HA

See GitHub for details on this configuration. Although GitHub only refers to 6.0, you can deploy this HA configuration for 6.4.

Deploying and configuring ELB-based HA/load balancing

FortiGate-VM can achieve HA using AWS ELB. You can deploy two FortiGate-VMs and associate them with an ELB, and traffic is balanced between the two. If one FortiGate-VM fails, the other handles traffic. This provides more security and reliability to the existing cloud infrastructure.



External and internal ELBs are required if you want to serve incoming and outgoing traffic for protected VMs. An external ELB is normally accessible from the Internet and distributes traffic as it enters a VPC. An internal ELB has similar capabilities but is only accessible within a VPC.

Like other load balancers, ELB can be configured as an external ELB that is accessible from the Internet and distributes traffic as it enters a VPC, or as an internal ELB which has similar functions and is only accessible inside a VPC. This section helps you get started with AWS ELB and FortiGate-VM configuration in an AWS environment.

Using this configuration, an IT administrator can place an application server inside a private subnet. The application server can provide web applications, terminal services, or general purpose Internet service. The access is fully protected and logged by the FortiGate-VM.

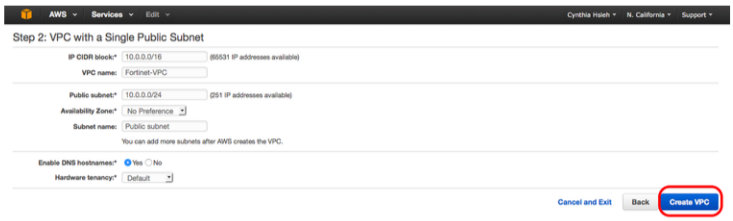
The design shows that application servers are fully separated between two subnets for active-active configuration. The load is divided evenly in this configuration.

You can protect and turn multiple AZs highly available depending on how you design the topology.

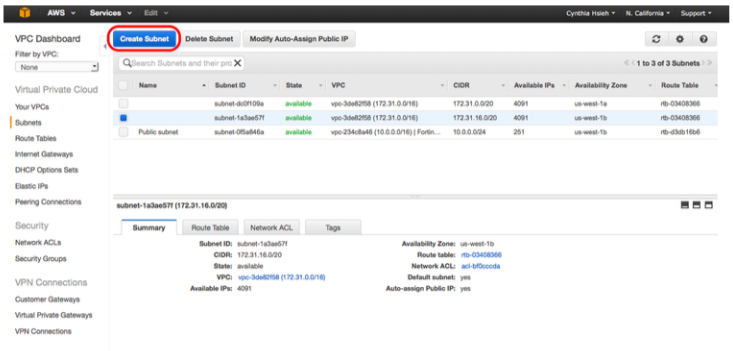
You can also combine AWS Route 53 to use DNS name together with ELB.

Creating two subnets on your Amazon VPC

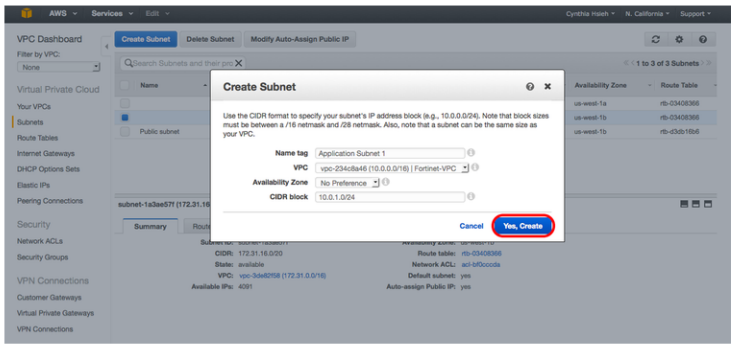
1. Log into AWS with your EC2 credentials and select *VPC*.
2. Select *Start VPC Wizard* to create a new VPC.
3. Select *VPC with a Single Public Subnet*.
4. Fill in the information as required and select *Create VPC*.



5. You have created a VPC with a single public subnet available. In this example, the subnet is referred to as Fortinet-VPC. To deploy the FortiGate-VM, you must also create a private subnet. Go to *Subnet* and select *Create Subnet*.



6. Fill in the information as required and select *Yes, Create*. In this example, the subnet will be referred to as "Application Subnet 1".



Creating a security group for the FortiGate-VM

1. Go to *Security Groups* and select *Create Security Group*. Set it to Fortinet-VPC. Select *Yes, Create*. In this example, this security group is referred to as *Allow everything*.
2. Edit the *Allow everything* group. Select the *Inbound Rules* tab and then select *Edit*.
3. Set *Type* to *ALL TCP*, *Protocol* to *TCP (6)*, *Port Range* to *ALL*, and *Source* to 0.0.0.0/0.
4. Select *Save*.

Allocating EIPs for the FortiGate-VM and for public access

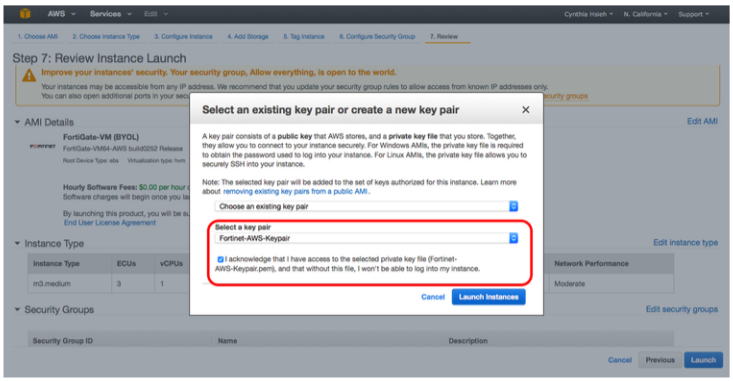
1. Go to *Elastic IPs* and select *Allocate New Address*.
2. Select *Yes, Allocate* to allocate an IP address.
3. Repeat to add two more address.

Deploying the FortiGate-VM

In this example, the FortiGate-VM instance is referred to as FortiGate 1.

1. Go to *Services* > *EC2* and select *Key Pairs*. Select *Create Key Pair*, fill in the key pair name, and select *Create*. This saves the key pair to your system. Remember where this file is, as it is used later.
2. Go to the *EC2 Dashboard* and select *Launch Instance*.
3. Select *AWS Marketplace* and search for Fortinet. Locate and select *FortiGate-VM (BYOL).**
4. Configure the FortiGate-VM deployment:
   a. Select *General purpose m3.medium* as the instance type and select *Next: Configure instance Details*.
   b. Set *Network* to Fortinet-VPC and *Subnet* to the public subnet.
   c. Under *Network Interface*, set *eth0* to the public subnet and *eth1* to Fortinet-VPC.
   d. Select *Next: Add Storage*.
   e. Review your storage options. This storage is used for logging. If you want more storage for logging, change the size from 10 to the desired value. Select *Next: Tag Instance*.
   f. Enter a *Name* tag and select *Configure Security Group*.
   g. Enable *Select an existing security group* and select the *Allow everything* security group. This allows the FortiGate-VM security features to be used, rather than the basic protection from Amazon.
5. Select *Review and Launch*. A review page will be shown with your configuration.
6. Review the settings and, if the configuration is correct, select *Launch*.

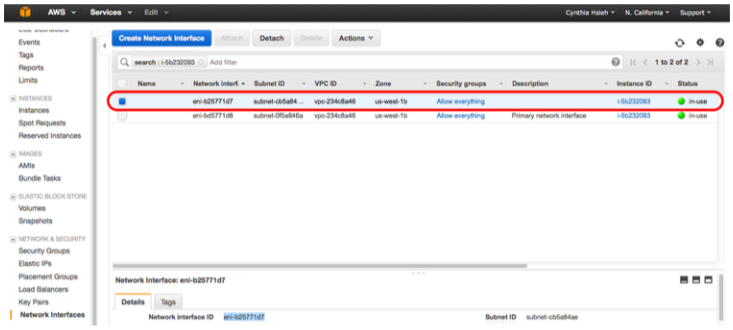
7. Select the *Fortinet-AWS-Keypair* that you previously created.



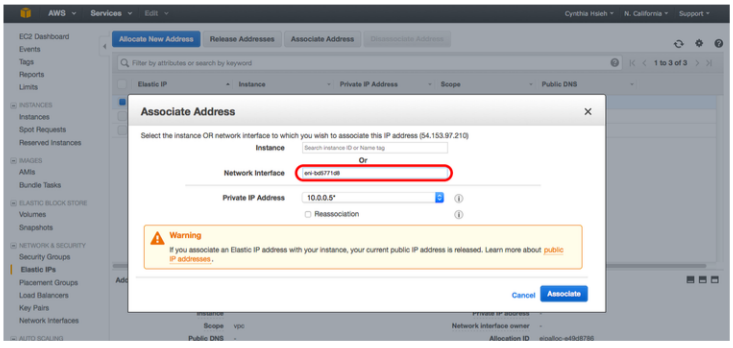
8. Review the information, then select *Launch Instance* to deploy the FortiGate-VM.

Assigning an IP address to the FortiGate-VM

1. Go to *Network Interface* and note the *Network Interface ID* of the private interface and the FortiGate-VM ID. In the example, these are eni-b25771d7 and eni-bd5771d8, respectively.



2. Go to *Elastic IPs* and select one of the IPs. Select *Associate Address*, then enter the network interface ID of the FortiGate-VM. Select *Associate*.



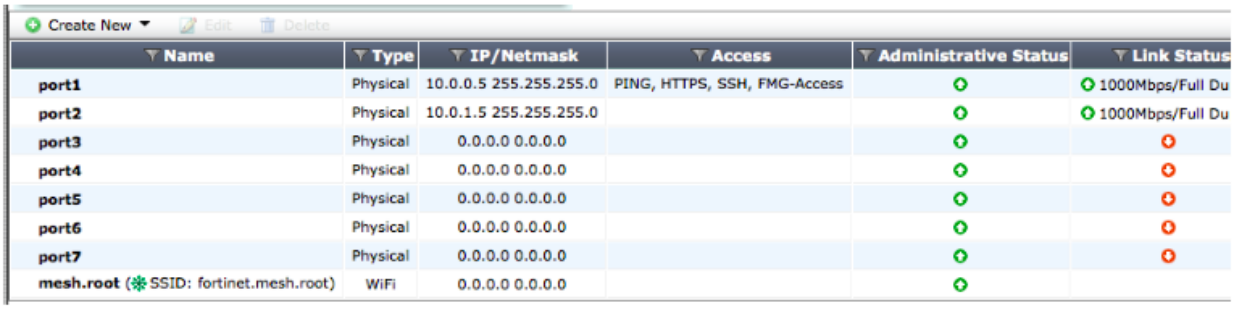
Creating a default route

1. Go to *VPC Dashboard* > *Route Tables* and select *Create Route Table*. Set *VPC* to the private subnet and select *Yes, Create*.

2. Select the new route, then select the *Routes* tab, then select *Edit*. Select *Add another route* and set *Destination* to 0.0.0.0/0 and *Target* to the network interface ID of the private interface.
3. Select the *Subnet Associations* tab, enable the private subnet, and select *Save*.

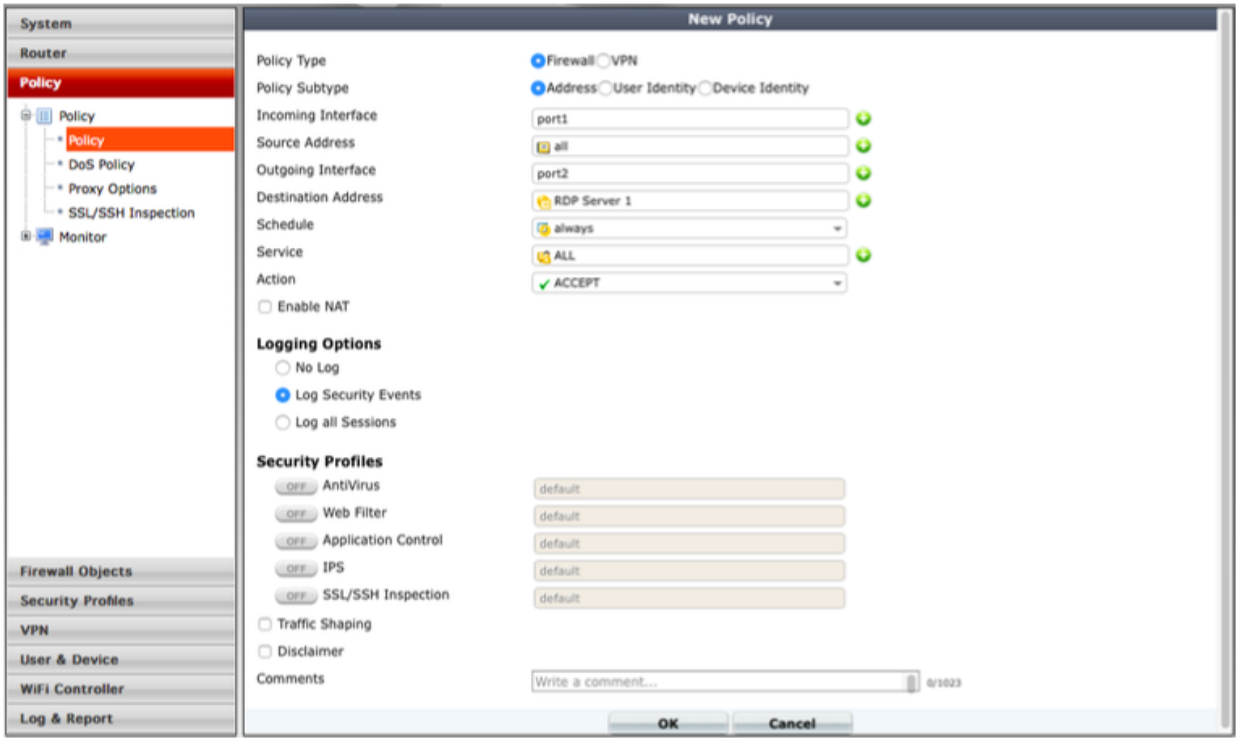
Configuring the FortiGate-VM

1. Log into the FortiGate-VM GUI using the default admin account. The default admin account has the username admin and no password. The license activation screen appears.
2. Select *Choose File*, select your license file, and select *OK*. The system restarts. After a few minutes, the login screen appears. Log back into the FortiGate-VM.
3. Using your terminal, enter the following commands to log into the server and enable disk logging:
```
ssh -i ./Fortinet-AWS-Keypair.pem admin@ FortiGate-VM64-AWS #execute update-now
       FortiGate-VM64-AWS #execute formatlogdisk
```
4. Go to *System* > *Admin* > *Administrators* and edit the default *admin* account. Select *Change Password* and enter a new password.
5. Go to *System* > *Network* > *Interfaces* and edit an internal interface (in the example, port2). Set *Addressing Mode* to DHCP.
6. This port's IP address has changed to the IP you entered using the terminal (in the example, 10.0.1.5).

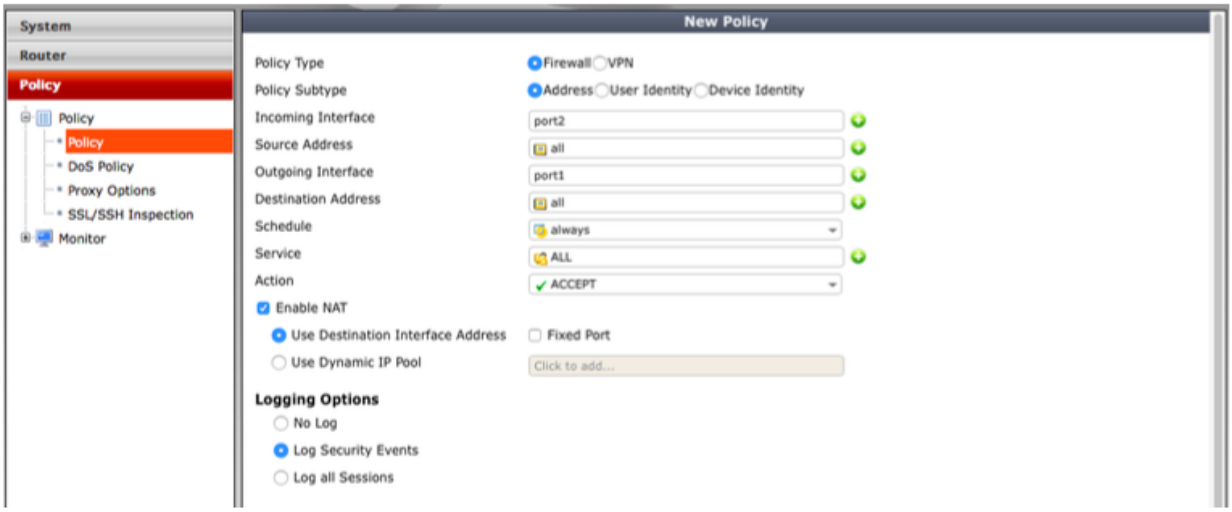


7. Go to *Firewall Objects* > *Virtual IPs* > *Virtual IPs* and create a new virtual IP that will map RDP (TCP port 3389) to a Windows server that will be deployed in the next step.

8. Go to *Policy* > *Policy* > *Policy* and create a new policy allowing traffic from the Internet-facing interface to the internal interface.

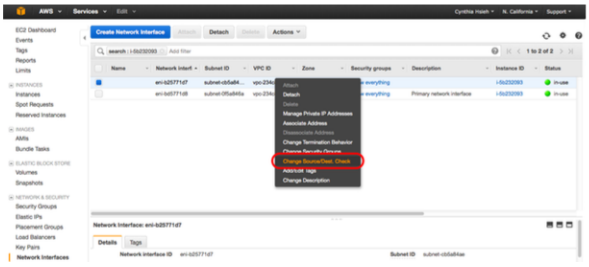


9. Create a second policy allowing traffic from the internal interface to the Internet-facing interface.

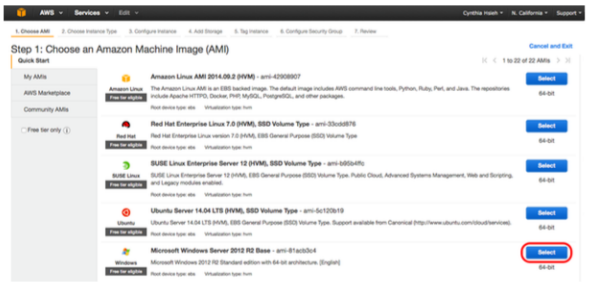


Deploying the Windows Server

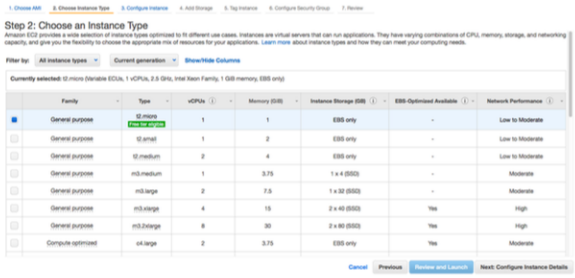
1. Connect to AWS and go to *Network Interfaces*. Right-click the private network interface, select *Change Source/Dest Check*, and select *Disable*. AWS now lets packets pass through instead of filtering them.



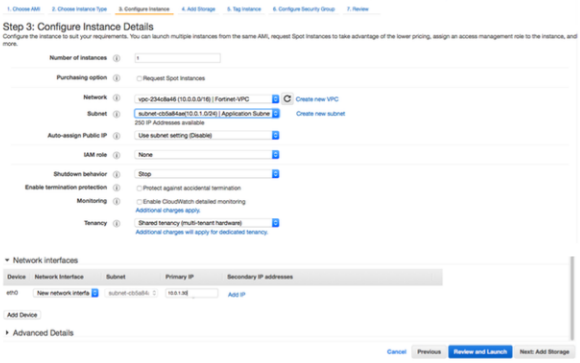
2. Go to *EC2 instances* and select *Launch Instance*.



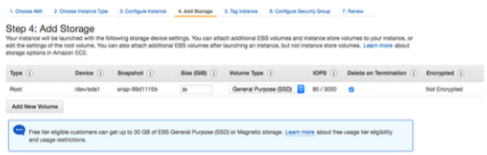
3. Select *t2.micro* for the instance type and select *Next: Configure Instance Details*.



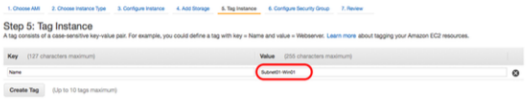
4. Set *Network* to Fortinet-VPC, subnet to Application Subnet 1, and *Network Interfaces* to eth0. Select *Next: Add Storage*.



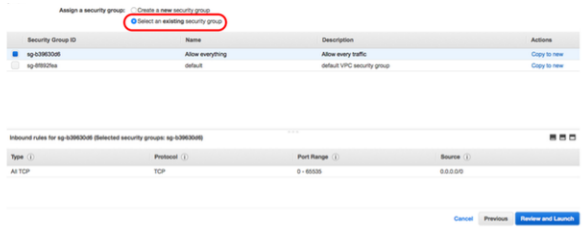
5. If necessary, change your storage option.



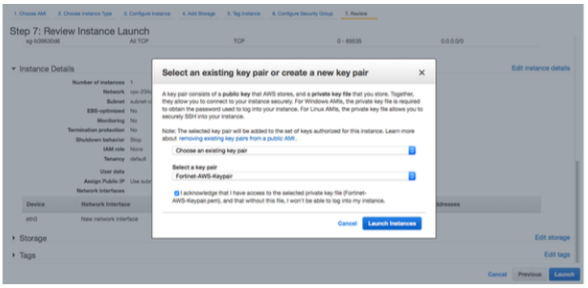
6. Select *Next: Tag Instance*. Enter a *Name* tag, then select *Next: Configure Security Group*.



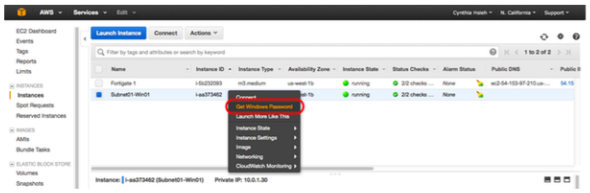
7. Enable *Select an existing security group* and select the *Allow everything* group.



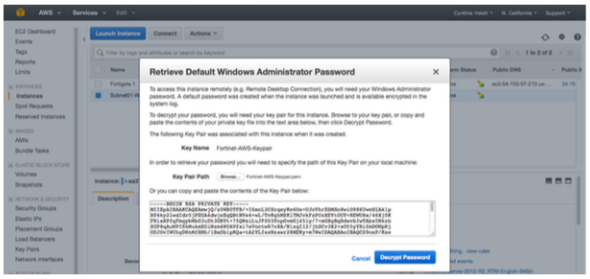
8. Select *Review and Launch*.



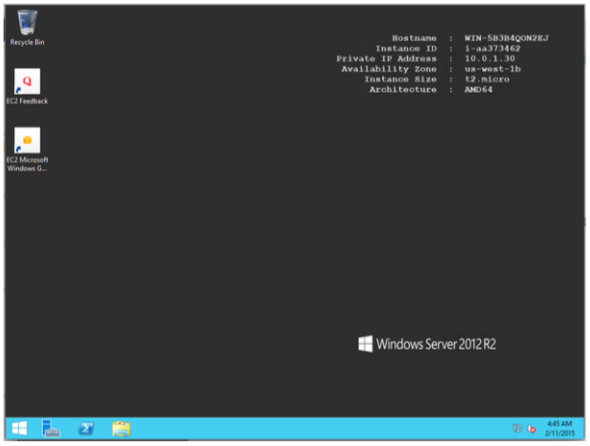
9. After you have reviewed the configuration, select the *Fortinet-AWS-Keypair* that you previously created.



10. Review the information, then select *Launch Instance* to deploy the server.



11. Go to *EC2 instance* and select the new subnet. Select *Get Windows Password*.



Click on *Key Pair Path Browse* and select the key pair file created earlier.

Select *Decrypt Password* to receive the administrator password for RDP connection.

12. Test the connection to your RDP server using your terminal and the following command:

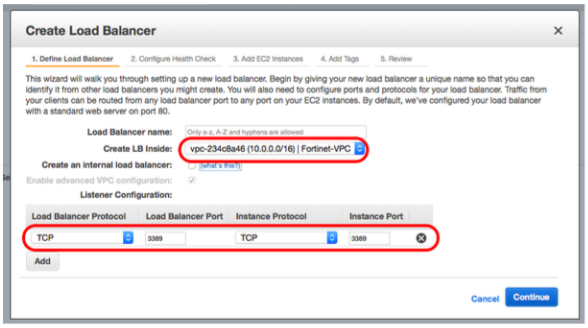
```
C:\> mstsc /v: /admin
```

Creating a second subnet and deploying a second FortiGate-VM

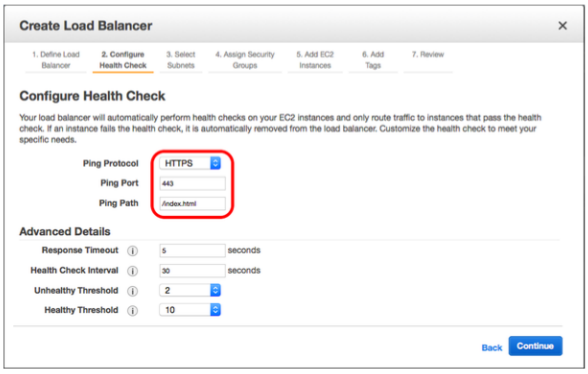
Repeat the above instructions to create a second AWS subnet and deploy a second FortiGate-VM on the subnet.

Creating an ELB between the FortiGate-VMs

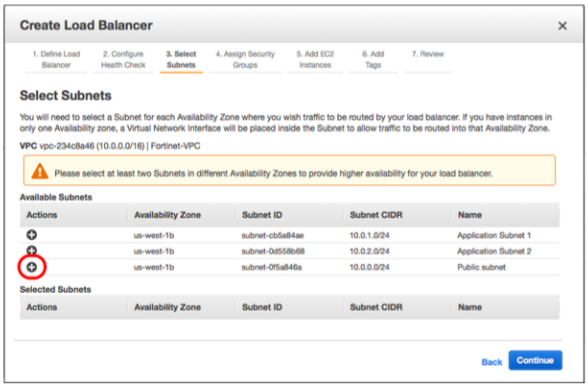
1. Go to the EC2 Dashboard, click *Load Balancers*, then *Create Load Balancer*.
2. Create a load balancer for RDP traffic within Fortinet-VPC. Select *Continue*.



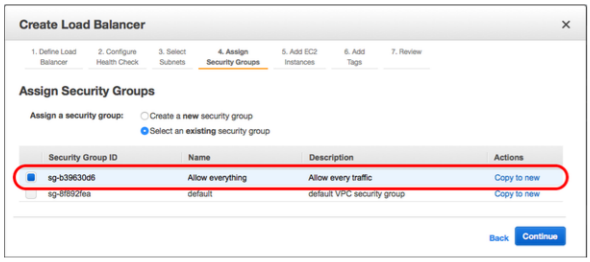
3. Set the *Ping Protocol* to use HTTPS. Select *Continue*.



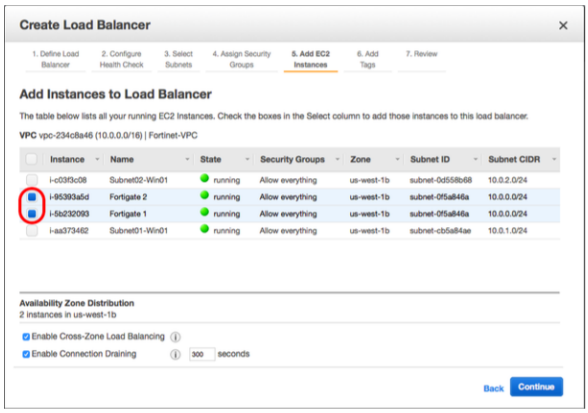
4. Use the + on *Public Subnet* and add it into the *Selected Subnets* list. Select *Continue*.



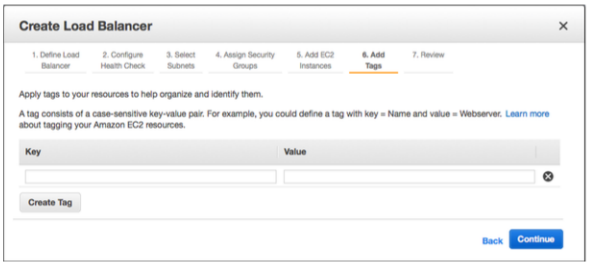
5. Select *Allow every traffic*. Select *Continue*.



6. *Select* both FortiGate-VMs. Select *Continue*.



7. Leave *Tags* as default and select *Continue*. A review page will appear. After you have reviewed the configuration, select *Create*.



8. Now that the ELB is created, you can use a domain name to test your connection via an RDP client.

Results

Go to the EC2 dashboard and right-click FortiGate 1. Select *Instance State* > *Stop* to stop this instance.

Connect via RDP to the Windows Server. All connections use the subnet for FortiGate 2.

You can also connect using your ELB DNS name. Connections only use the subnet for FortiGate 2.

Start FortiGate 2 and wait until the ELB status is 2/2. Connect to the server using multiple sources.

The load is balanced between the FortiGate-VM instances.

Deploying FortiGate-VM active-passive HA on AWS within one zone

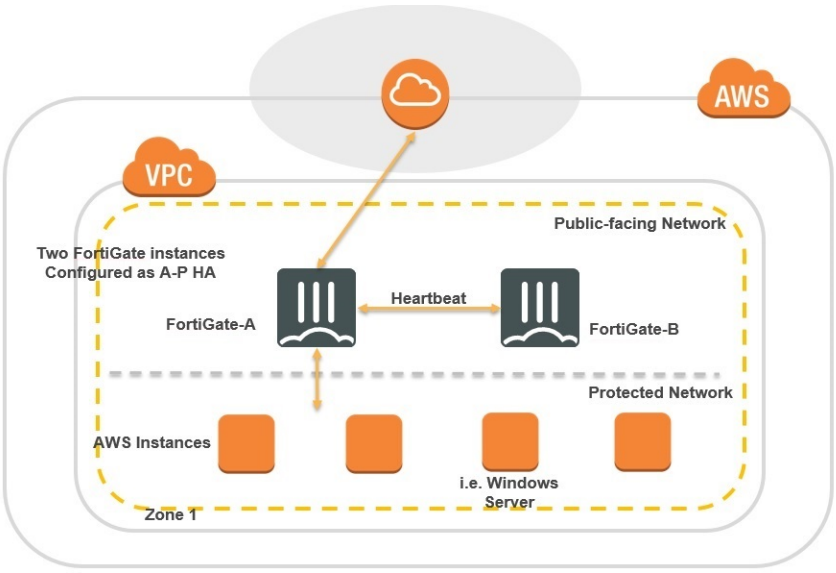
This guide provides sample configuration of active-passive FortiGate-VM high availability (HA) on AWS within one zone.

FortiGate's native HA feature (without using an AWS supplementary mechanism) can be configured with two FortiGate instances: one acting as the primary node and the other as the secondary node, located in two different availability zones (AZs) within a single VPC. This is called "Unicast HA" specific to the AWS environment in comparison to an equivalent feature provided by physical FortiGate units. The FortiGates run heartbeats between dedicated ports and synchronize OS configurations. When the primary node fails, the secondary node takes over as the primary node so endpoints continue to communicate with external resources over the FortiGate.

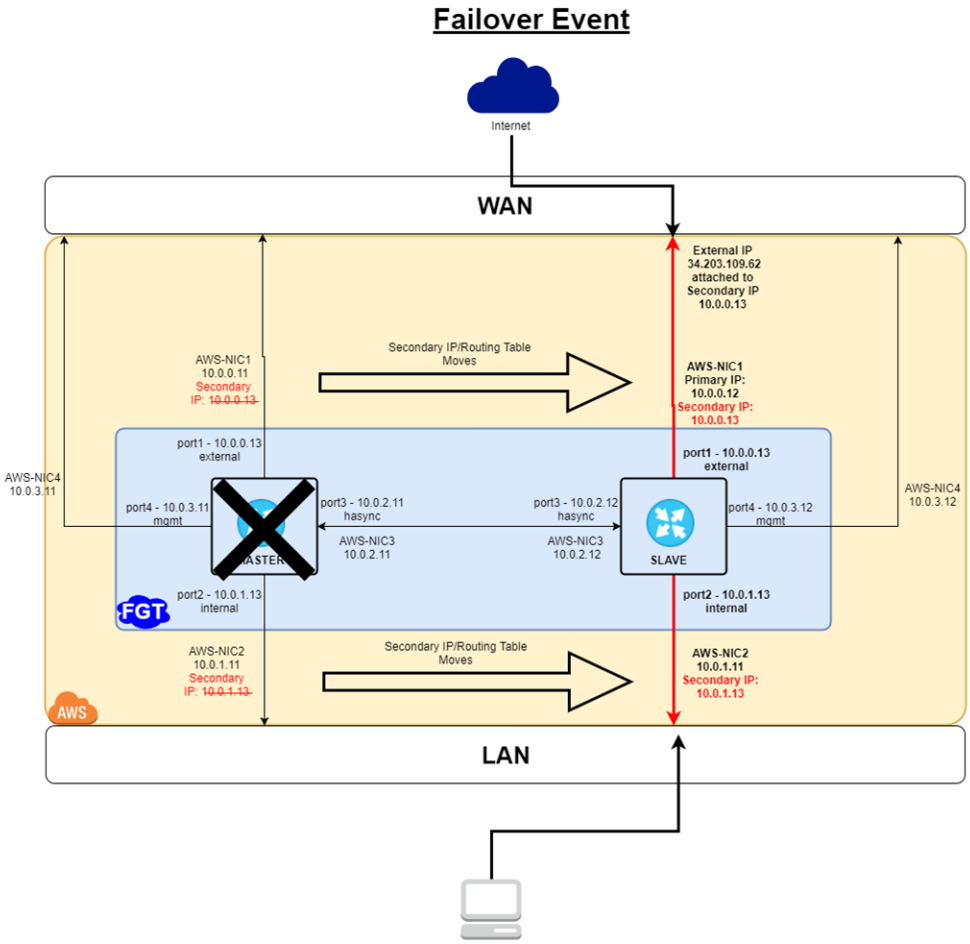
These paired FortiGate instances act as a single logical instance and share interface IP addressing. The main benefits of this solution are:

- Fast failover of FortiOS and AWS SDN without external automation/services
- Automatic AWS SDN updates to EIPs and route targets
- Native FortiOS configuration sync
- Ease of use as the cluster is treated as single logical FortiGate

The following depicts the network topology for this sample deployment:



The following depicts a failover event for this sample deployment:



The following lists the IP address assignments for this sample deployment for FortiGate A:

Port	AWS primary address	AWS secondary address
port1	10.0.0.11	10.0.0.13
port2	10.0.1.11	10.0.1.13
port3	10.0.2.11	N/A
port4	10.0.3.11	N/A

The following lists the IP address assignments for this sample deployment for FortiGate B:

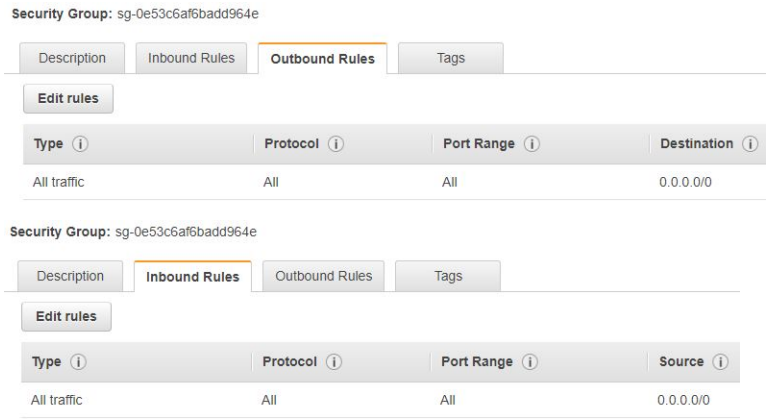
Port	AWS primary address
port1	10.0.0.12
port2	10.0.1.12
port3	10.0.2.12
port4	10.0.3.12

To check the prerequisites:

- Ensure that two FortiGates exist in the same VPC and AZ. The two FortiGates must also have the same build of FortiOS (FGT_VM64_AWS or FGT_VM64_AWSONDEMAND) installed.
- If using FGT_VM64_AWS, ensure that both FortiGates have valid licenses.

To configure FortiGate-VM HA in AWS:

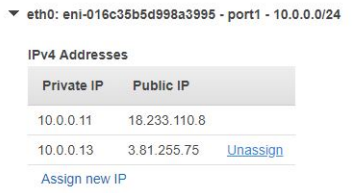
1. In the AWS management console, create a VPC. The VPC in this example has been created with 10.0.0.0/16 CIDR.
2. Create four subnets. In this example, the four subnets are as follows:
   a. Public WAN: 10.0.0.0/24
   b. Internal network: 10.0.1.0/24
   c. Heartbeat network: 10.0.2.0/24
   d. Management network: 10.0.3.0/24
3. Create a single, open security group as shown below:



4. Create an IAM role. The IAM role is necessary for HA failover. Ensure that the IAM role can read and write EC2 information to read, detach, and reattach network interfaces and edit routing tables.

5. Create five elastic IP addresses. Five elastic IP addresses are needed to set up the environment, but we will be left with three IP addresses at the end:
   a. One public WAN IP address. This will be attached to the instance NIC1's secondary IP address.
   b. One FortiGate A management IP address
   c. One FortiGate B management IP address
   d. Two temporary IP addresses
6. Create two FortiGate instances. You can use any instance type with at least four vCPUs, since four NICs are required:
   a. Configure FortiGate A:
      i. Attach the IAM role created earlier.
      ii. Create the instance in the VPC created earlier and in the public WAN subnet, with no ephemeral public IP address.
      iii. Configure an internal IP address of 10.0.0.11, and a secondary IP address of 10.0.0.13.

      iv. Attach a security group.
   b. Configure FortiGate B by repeating the steps for FortiGate A above. For FortiGate B, configure an internal IP address of 10.0.0.12, and no internal IP address.
   c. Attach three NICs to each FortiGate according to the IP assignment in the appropriate subnet:
      i. FortiGate A:
         i. port2 (AWS primary 10.0.1.11/AWS secondary 10.0.1.13) (internal network)
         ii. port3 (AWS primary 10.0.2.11) (Heartbeat network)
         iii. port4 (AWS primary 10.0.3.11) (management network)
      ii. FortiGate B:
         i. port2 (AWS primary 10.0.1.12) (internal network)
         ii. port3 (AWS primary 10.0.2.12) (Heartbeat network)
         iii. port4 (AWS primary 10.0.3.12) (management network)
7. Attach the two temporary elastic IP addresses to the port1 primary IP addresses of FortiGate A and FortiGate B. This allows access to the FortiGates via SSH for configuration purposes. The default password for the FortiGates is their instance IDs. The following shows the temporary elastic IP address assigned to FortiGate A:



The following shows the temporary elastic IP address assigned to FortiGate B:

To configure FortiGate A using the CLI:

Run the following commands in the FortiOS CLI on FortiGate A:

```
config sys glo
   set hostname master
end
config system interface
   edit port1
      set mode static
      set ip 10.0.0.13 255.255.255.0
      set allowaccess https ping ssh fgfm
      set alias external
   next
   edit port2
      set mode static
      set ip 10.0.1.13 255.255.255.0
      set allowaccess https ping ssh fgfm
      set alias internal
   next
   edit port3
      set mode static
      set ip 10.0.2.11 255.255.255.0
      set allowaccess https ping ssh fgfm
      set alias hasync
   next
   edit port4
      set mode static
      set ip 10.0.3.11 255.255.255.0
      set allowaccess https ping ssh fgfm
      set alias hamgmt
   next
   end
config router static
   edit 1
      set device port1
      set gateway 10.0.0.1
   next
   end
config system dns
   set primary 8.8.8.8
end
config firewall policy
   edit 0
      set name "outgoing"
      set srcintf "port2"
      set dstintf "port1"
      set srcaddr "all"
      set dstaddr "all"
      set action accept
      set schedule "always"
      set service "ALL"
      set logtraffic disable
      set nat enable
   next
end
config system ha
   set group-name "test"
```

```
   set mode a-p
   set hbdev "port3" 50
   set session-pickup enable
   set ha-mgmt-status enable
   config ha-mgmt-interfaces
      edit 1
         set interface "port4"
         set gateway 10.0.3.1
      next
   end
   set override disable
   set priority 1
   set unicast-hb enable
   set unicast-hb-peerip 10.0.2.12
end
```

To configure FortiGate B using the CLI:

Run the following commands in the FortiOS CLI on FortiGate B:

```
config sys glo
   set hostname slave
end
config system interface
   edit port1
      set mode static
      set ip 10.0.0.12 255.255.255.0
      set allowaccess https ping ssh fgfm
      set alias external
   next
   edit port2
      set mode static
      set ip 10.0.1.12 255.255.255.0
      set allowaccess https ping ssh fgfm
      set alias internal
   next
   edit port3
      set mode static
      set ip 10.0.2.12 255.255.255.0
      set allowaccess https ping ssh fgfm
      set alias hasync
   next
   edit port4
      set mode static
      set ip 10.0.3.12 255.255.255.0
      set allowaccess https ping ssh fgfm
      set alias hamgmt
   next
   end
config router static
   edit 1
      set device port1
      set gateway 10.0.0.1
   next
   end
config system dns
   set primary 8.8.8.8
```

```
end
config firewall policy
   edit 0
      set name "outgoing"
      set srcintf "port2"
      set dstintf "port1"
      set srcaddr "all"
      set dstaddr "all"
      set action accept
      set schedule "always"
      set service "ALL"
      set logtraffic disable
      set nat enable
   next
end
config system ha
   set group-name "test"
   set mode a-p
   set hbdev "port3" 50
   set session-pickup enable
   set ha-mgmt-status enable
   config ha-mgmt-interfaces
      edit 1
         set interface "port4"
         set gateway 10.0.3.1
      next
   end
   set override disable
   set priority 1
   set unicast-hb enable
   set unicast-hb-peerip 10.0.2.11
end
```

After completing configuration of FortiGate B, remove the two temporary IP addresses. You can connect to the FortiGates via the management ports instead.

To configure the routing tables in AWS:

You must configure three routing tables.

The following shows the public WAN routing table:

Destination	Target	Status	Propagated
10.0.0.0/16	local	active	No
0.0.0.0/0	igw-011ffd018b1e7227e	active	No

The following shows the internal network routing table. Ensure to point the 0.0.0.0/0 CIDR to FortiGate A's port2 NIC.

Destination	Target	Status	Propagated
10.0.0.0/16	local	active	No
0.0.0.0/0	eni-0f19c02934d82c086	active	No

The following shows the Heartbeat and management networks' routing table:

Destination	Target	Status	Propagated
10.0.0.0/16	local	active	No
0.0.0.0/0	igw-05af2c26b0e7ff1fa	active	No

To test FortiGate-VM HA:

1. Run `get system ha status` to check that the FortiGates are in sync:

```
master # get sys ha stat
HA Health Status: OK
Model: FortiGate-VM64-AWSONDEMAND
Mode: HA A-P
Group: 0
Debug: 0
Cluster Uptime: 0 days 0:42:46
Cluster state change time: 2019-01-15 17:23:02
Master selected using:
   <2019/01/15 17:23:02> FGTAWS000F19C1A0 is selected as the master because it has the
        largest value of uptime.
   <2019/01/15 17:09:47> FGTAWS000F19C1A0 is selected as the master because it's the only
        member in the cluster.
ses_pickup: enable, ses_pickup_delay=disable
override: disable
unicast_hb: peerip=10.0.2.12, myip=10.0.2.11, hasync_port='port3'
Configuration Status:
   FGTAWS000F19C1A0(updated 4 seconds ago): in-sync
   FGTAWS000ECBF4EF(updated 4 seconds ago): in-sync
System Usage stats:
   FGTAWS000F19C1A0(updated 4 seconds ago):
      sessions=2, average-cpu-user/nice/system/idle=0%/0%/0%/100%, memory=5%
   FGTAWS000ECBF4EF(updated 4 seconds ago):
      sessions=0, average-cpu-user/nice/system/idle=0%/0%/0%/100%, memory=5%
HBDEV stats:
   FGTAWS000F19C1A0(updated 4 seconds ago):
      port3: physical/1000full, up, rx-bytes/packets/dropped/errors=3135309/12092/0/0,
           tx=9539178/17438/0/0
   FGTAWS000ECBF4EF(updated 4 seconds ago):
      port3: physical/1000full, up, rx-bytes/packets/dropped/errors=9300105/17602/0/0,
           tx=3293016/11828/0/0
Master: master , FGTAWS000F19C1A0, HA cluster index = 0
Slave : slave , FGTAWS000ECBF4EF, HA cluster index = 1
number of vcluster: 1
vcluster 1: work 10.0.2.11
Master: FGTAWS000F19C1A0, HA operating index = 0
Slave : FGTAWS000ECBF4EF, HA operating index = 1
```

2. Ensure that failover functions as configured:
   a. Turn on debug mode on FortiGate B:

```
slave # di de en
slave # di de application awsd -1
Debug messages will be on for unlimited time.
```

   b. Shut down the primary FortiGate A. In the event of a successful failover, FortiGate B's CLI shows the following:

```
slave # Become HA master
send_vip_arp: vd root master 1 intf port1 ip 10.0.0.13
send_vip_arp: vd root master 1 intf port2 ip 10.0.1.13
awsd get instance id i-0ecbf4ef4c14ba1bb
awsd get iam role WikiDemoHARole
awsd get region us-east-1
awsd doing ha failover for vdom root
awsd moving secondary ip for port1
awsd moving secip 10.0.0.13 from eni-016c35b5d998a3995 to eni-0d318aeb7cdfe72fb
awsd move secondary ip successfully
```

```
awsd associate elastic ip allocation eipalloc-0e5ff7daabd5f46dc to 10.0.0.13 of eni
     eni-0d318aeb7cdfe72fb
awsd associate elastic ip successfully
awsd moving secondary ip for port2
awsd moving secip 10.0.1.13 from eni-0f19c02934d82c086 to eni-004d87ffb05329b28
awsd move secondary ip successfully
awsd update route table rtb-0bc0aaaea8fe56192, replace route of dst 0.0.0.0/0 to eni-
     004d87ffb05329b28
awsd update route successfully
```

c. Verify on AWS that the public and internal networks' secondary IP addresses moved, and that the routing table changes to point to FortiGate B's internal network ENI.

3. Initiate an SSH session (or another protocol with similar long keep-alive session characteristics) to an external IP address on Ubuntu or an internal VM used for testing purposes. Test failover again and check that the session continues to function without needing to reconnect, and that the session list on the primary and failed over secondary FortiGates are synced.

Deploying FortiGate-VM active-passive HA AWS between multiple zones

This guide provides sample configuration of active-passive FortiGate-VM high availability (HA) on AWS between multiple zones.

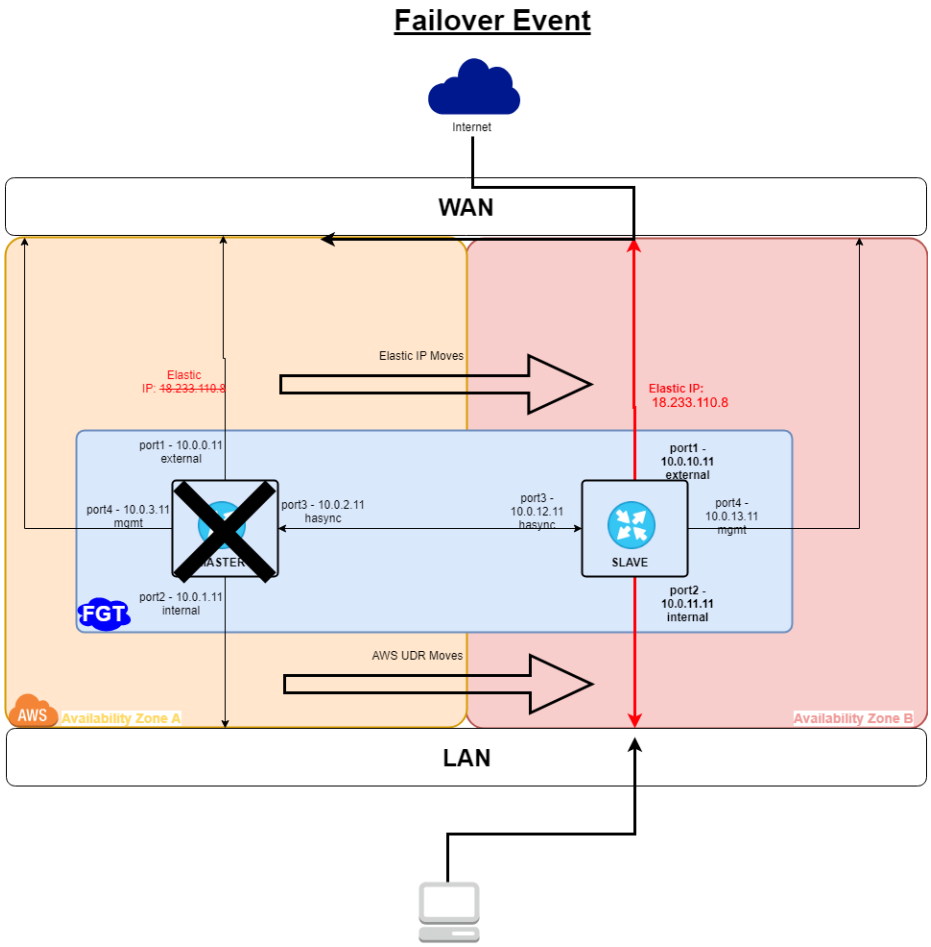
You can configure FortiGate's native HA feature (without using an AWS supplementary mechanism) with two FortiGate instances: one acting as the master/primary node and the other as the slave/secondary node, located in two different availability zones (AZs) within a single VPC. This is called "Unicast HA" specific to the AWS environment in comparison to an equivalent feature provided by physical FortiGate units. The FortiGates run heartbeats between dedicated ports and synchronize OS configurations. When the primary node fails, the secondary node takes over as the primary node so endpoints continue to communicate with external resources over the FortiGate.

This feature is important because it solves a critical issue of High Availability, which is the ability to recover in the event of a catastrophic failure. In the case that both FortiGates are located in the same Availability Zone and that AZ happens to fail, then both FortiGates would go down and HA would be useless. Thus, there is a need to support HA configuration where both FortiGates are in separate AZs.

These paired FortiGate instances act as a single logical instance and share interface IP addressing. The main benefits of this solution are:

- Fast failover of FortiOS and AWS SDN without external automation/services
- Automatic AWS SDN updates to EIPs and route targets
- Native FortiOS configuration sync
- Ease of use as the cluster is treated as single logical FortiGate

The following depicts the network topology for this sample deployment:



The following lists the IP address assignments for this sample deployment for FortiGate A:

Port	AWS primary address	Subnet
port1	10.0.0.11	10.0.0.0/24 EIP
port2	10.0.1.11	10.0.1.0/24
port3	10.0.2.11	10.0.2.0/24
port4	10.0.3.11	10.0.3.0/24 EIP

The following lists the IP address assignments for this sample deployment for FortiGate B:

Port	AWS primary address	Subnet
port1	10.0.10.11	10.0.10.0/24 EIP
port2	10.0.11.11	10.0.11.0/24
port3	10.0.12.11	10.0.12.0/24
port4	10.0.13.11	10.0.13.0/24 EIP

IPsec VPN phase 1 configuration does not synchronize between primary and secondary FortiGates across AZs. Phase 2 configuration does synchronize.

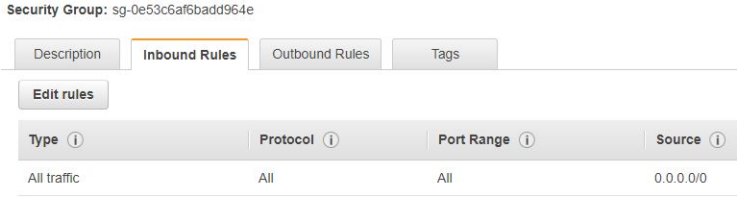
To check the prerequisites:

- Ensure that two FortiGates exist in the same VPC but different AZs. The two FortiGates must also have the same FortiOS build (FGT_VM64_AWS or FGT_VM64_AWSONDEMAND) installed.
- If using FGT_VM64_AWS, ensure that both FortiGates have valid licenses.
- The following summarizes minimum sufficient IAM roles for this deployment:

```
{
   "Version": "2012-10-17",
   "Statement": [
   {
   "Action": [
   "ec2:Describe*",
   "ec2:AssociateAddress",
   "ec2:AssignPrivateIpAddresses",
   "ec2:UnassignPrivateIpAddresses",
   "ec2:ReplaceRoute"
   ],
   "Resource": "*",
   "Effect": "Allow"
   }
   ]
}
```

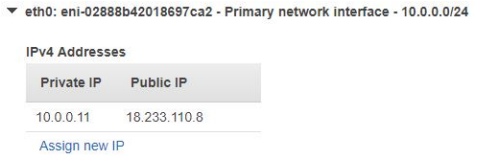
To configure FortiGate-VM HA in AWS:

1. In the AWS management console, create a VPC. The VPC in this example has been created with 10.0.0.0/16 CIDR.
2. Create eight subnets. In this example, the eight subnets are as follows:
   a. Four in AZ A:
      i. Public WAN: 10.0.0.0/24
      ii. Internal network: 10.0.1.0/24
      iii. Heartbeat network: 10.0.2.0/24
      iv. Management network: 10.0.3.0/24
   b. Four in AZ B:
      i. Public WAN: 10.0.10.0/24
      ii. Internal network: 10.0.11.0/24
      iii. Heartbeat network: 10.0.12.0/24
      iv. Management: 10.0.13.0/24
3. Create a single, open security group as shown below:

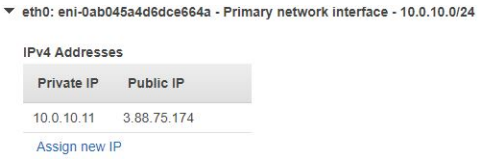


4. Create an IAM role. The IAM role is necessary for HA failover. Ensure that the IAM role can read and write EC2 information to read, detach, and reattach network interfaces and edit routing tables.
5. Create three elastic IP addresses:
   a. One public WAN IP address. This will be attached to the instance NIC1's secondary IP address.
   b. One FortiGate A management IP address
   c. One FortiGate B management IP address
6. Create two FortiGate instances. You can use any instance type with at least four vCPUs, since four NICs are required:
   a. Configure FortiGate A:
      i. Attach the IAM role created earlier.
      ii. Create the instance in the VPC created earlier and in the public WAN subnet, with no ephemeral public IP address.
      iii. Configure an internal IP address of 10.0.0.11.

      iv. Attach a security group.
   b. Configure FortiGate B by repeating the steps for FortiGate A above. For FortiGate B, configure the instance in the public WAN subnet in AZ B, and configure an internal IP address of 10.0.10.11.
   c. Attach three NICs to each FortiGate according to the IP assignment in the appropriate subnet:
      i. FortiGate A:
         i. port2 (AWS primary 10.0.1.11) (internal network)
         ii. port3 (AWS primary 10.0.2.11) (Heartbeat network)
         iii. port4 (AWS primary 10.0.3.11) (management network)
      ii. FortiGate B:
         i. port2 (AWS primary 10.0.11.11) (internal network)
         ii. port3 (AWS primary 10.0.12.11) (Heartbeat network)
         iii. port4 (AWS primary 10.0.13.11) (management network)
7. Attach the two elastic IP addresses to the port1 primary IP addresses of FortiGate A and FortiGate B. This allows access to the FortiGates via SSH for configuration purposes. The default password for the FortiGates is their instance IDs. The following shows the elastic IP address assigned to FortiGate A:



   The following shows the elastic IP address assigned to FortiGate B:



To configure FortiGate A using the CLI:

Run the following commands in the FortiOS CLI on FortiGate A:

```
config sys glo
   set hostname master
end
config system interface
   edit port1
      set mode static
      set ip 10.0.0.11 255.255.255.0
      set allowaccess https ping ssh fgfm
      set alias external
   next
   edit port2
      set mode static
      set ip 10.0.1.11 255.255.255.0
      set allowaccess https ping ssh fgfm
      set alias internal
   next
   edit port3
      set mode static
      set ip 10.0.2.11 255.255.255.0
      set allowaccess https ping ssh fgfm
      set alias hasync
   next
   edit port4
      set mode static
      set ip 10.0.3.11 255.255.255.0
      set allowaccess https ping ssh fgfm
      set alias hamgmt
   next
end
config router static
   edit 1
      set device port1
      set gateway 10.0.0.1
   next
   edit 2
      set device port2
      set gateway 10.0.1.1
      set dst 10.0.11.0/24
   next
end
config firewall policy
   edit 0
      set name "outgoing"
      set srcintf "port2"
      set dstintf "port1"
      set srcaddr "all"
      set dstaddr "all"
      set action accept
      set schedule "always"
      set service "ALL"
      set logtraffic disable
      set nat enable
   next
end
config system ha
   set group-name "test"
   set mode a-p
```

```
   set hbdev "port3" 50
   set session-pickup enable
   set ha-mgmt-status enable
   config ha-mgmt-interfaces
      edit 1
         set interface "port4"
         set gateway 10.0.3.1
      next
   end
   set override disable
   set priority 255
   set unicast-hb enable
   set unicast-hb-peerip 10.0.12.11
end
```

To configure FortiGate B using the CLI:

Run the following commands in the FortiOS CLI on FortiGate B:

```
config sys glo
   set hostname slave
end
config system interface
   edit port1
      set mode static
      set ip 10.0.10.11 255.255.255.0
      set allowaccess https ping ssh fgfm
      set alias external
   next
   edit port2
      set mode static
      set ip 10.0.11.11 255.255.255.0
      set allowaccess https ping ssh fgfm
      set alias internal
   next
   edit port3
      set mode static
      set ip 10.0.12.11 255.255.255.0
      set allowaccess https ping ssh fgfm
      set alias hasync
   next
   edit port4
      set mode static
      set ip 10.0.13.11 255.255.255.0
      set allowaccess https ping ssh fgfm
      set alias hamgmt
   next
end
config router static
   edit 1
      set device port1
      set gateway 10.0.10.1
   next
   edit 2
      set device port2
      set gateway 10.0.11.1
      set dst 10.0.1.0/24
```

```
   next
end
config firewall policy
   edit 0
      set name "outgoing"
      set srcintf "port2"
      set dstintf "port1"
      set srcaddr "all"
      set dstaddr "all"
      set action accept
      set schedule "always"
      set service "ALL"
      set logtraffic disable
      set nat enable
   next
end
config system ha
   set group-name "test"
   set mode a-p
   set hbdev "port3" 50
   set session-pickup enable
   set ha-mgmt-status enable
   config ha-mgmt-interfaces
      edit 1
         set interface "port4"
         set gateway 10.0.13.1
      next
   end
   set override disable
   set priority 1
   set unicast-hb enable
   set unicast-hb-peerip 10.0.2.11
end
```

After completing configuration of FortiGate B, remove the EIP to the FortiGate B public IP address. You can connect to the FortiGates via the management ports instead.

To configure the routing tables in AWS:

You must configure three routing tables.

The following shows the public WAN routing table. Ensure to point the 0.0.0.0/0 CIDR to the Internet gateway:

Destination	Target	Status	Propagated
10.0.0.0/16	local	active	No
0.0.0.0/0	igw-05af2c26b0e7ff1fa	active	No

The following shows the internal network routing table. Ensure to point the 0.0.0.0/0 CIDR to FortiGate A's port2 NIC.

Destination	Target	Status	Propagated
10.0.0.0/16	local	active	No
0.0.0.0/0	eni-0c4c085477aaff8c5	active	No

The following shows the Heartbeat and management networks' routing table:

Destination	Target	Status	Propagated
10.0.0.0/16	local	active	No
0.0.0.0/0	igw-05af2c26b0e7ff1fa	active	No

To configure a VDOM exception:

You must configure a VDOM exception to prevent interface synchronization between the two FortiGates. FortiOS 6.4.1 and later versions support the following commands. FortiOS 6.4.0 does not support these commands.

```
config system vdom-exception
   edit 1
      set object system.interface
   next
   edit 2
      set object router.static
   next
   edit 3
      set object firewall.vip
   next
end
```

To test FortiGate-VM HA:

1. Run `get system ha status` to check that the FortiGates are in sync:

```
master # get sys ha stat
HA Health Status: OK
Model: FortiGate-VM64-AWSONDEMAND
Mode: HA A-P
Group: 0
Debug: 0
Cluster Uptime: 3 days 1:50:18
Cluster state change time: 2019-01-31 18:20:47
Master selected using:
   <2019/01/31 18:20:47> FGTAWS0006AB1961 is selected as the master because it has the
        largest value of override priority.
   <2019/01/31 18:20:47> FGTAWS0006AB1961 is selected as the master because it's the only
        member in the cluster.
ses_pickup: enable, ses_pickup_delay=disable
override: disable
unicast_hb: peerip=10.0.12.11, myip=10.0.2.11, hasync_port='port3'
Configuration Status:
   FGTAWS0006AB1961(updated 3 seconds ago): in-sync
   FGTAWS000B29804F(updated 4 seconds ago): in-sync
System Usage stats:
   FGTAWS0006AB1961(updated 3 seconds ago):
      sessions=18, average-cpu-user/nice/system/idle=0%/0%/0%/100%, memory=10%
   FGTAWS000B29804F(updated 4 seconds ago):
      sessions=2, average-cpu-user/nice/system/idle=0%/0%/0%/100%, memory=10%
HBDEV stats:
   FGTAWS0006AB1961(updated 3 seconds ago):
      port3: physical/00, up, rx-bytes/packets/dropped/errors=430368/1319/0/0,
           tx=560457/1280/0/0
   FGTAWS000B29804F(updated 4 seconds ago):
      port3: physical/00, up, rx-bytes/packets/dropped/errors=870505/2061/0/0,
           tx=731630/2171/0/0
Master: master , FGTAWS0006AB1961, HA cluster index = 1
Slave : slave , FGTAWS000B29804F, HA cluster index = 0
number of vcluster: 1
vcluster 1: work 10.0.2.11
Master: FGTAWS0006AB1961, HA operating index = 0
Slave : FGTAWS000B29804F, HA operating index = 1
```

2. Ensure that failover functions as configured:
   a. Turn on debug mode on FortiGate B:
   ```
   slave # di de en
   slave # di de application awsd -1
   Debug messages will be on for unlimited time.
   ```
   b. Shut down the primary FortiGate A. In the event of a successful failover, FortiGate B's CLI shows the following:
   ```
   slave # Become HA master
   send_vip_arp: vd root master 1 intf port1 ip 10.0.10.11
   send_vip_arp: vd root master 1 intf port2 ip 10.0.11.11
   awsd get instance id i-0b29804fd38976af4
   awsd get iam role WikiDemoHARole
   awsd get region us-east-1
   awsd get vpc id vpc-0ade7ea6e64befbfc
   awsd doing ha failover for vdom root
   awsd associate elastic ip for port1
   awsd associate elastic ip allocation eipalloc-06b849dbb0f76555f to 10.0.10.11 of eni
        eni-0ab045a4d6dce664a
   awsd associate elastic ip successfully
   awsd update route table rtb-0a7b4fec57feb1a21, replace route of dst 0.0.0.0/0 to eni-
        0c4c085477aaff8c5
   awsd update route successfully
   ```
   c. Verify on AWS that the public and internal networks' secondary IP addresses moved to the new primary FortiGate, and that the routing table changes to point to the secondary FortiGate's internal network ENI.

Deploying FortiGate-VM active-passive HA AWS between multiple zones manually with Transit Gateway integration

This guide provides sample configuration of a manual build of an AWS Transit Gateway (TGW) with two virtual private cloud (VPC) spokes and a security VPC. The security VPC contains two FortiGate-VMs to inspect inbound and outbound traffic.

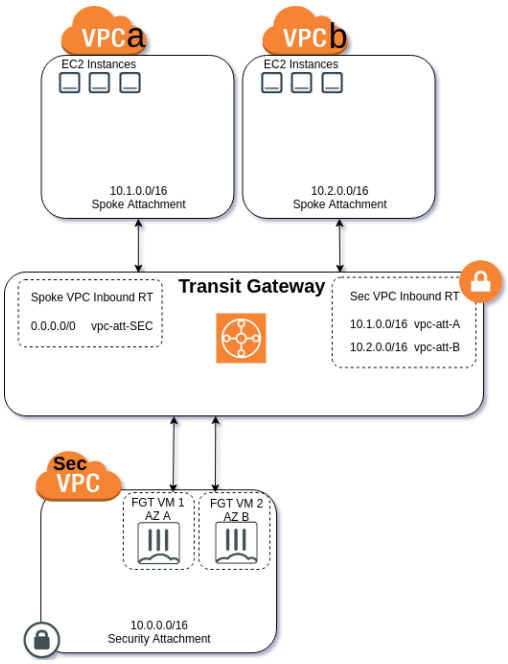
Before deploying FortiGate high availability (HA) for AWS with TGW integration, familiarity with the following AWS services is recommended:

- Transit Gateway
- Elastic Cloud Compute (EC2)
- VPC

If you are new to AWS, see Getting Started with AWS.

This deployment consists of the following steps:

1. Creating VPCs and subnets on page 140
2. Creating a Transit Gateway and related resources on page 141
3. Creating an Internet gateway on page 146
4. Creating VPC route tables on page 147
5. Deploying FortiGate-VM from AWS marketplace on page 148
6. Adding network interfaces and elastic IP addresses to the FortiGate-VMs on page 149
7. Configuring the FortiGate-VMs on page 151
8. Updating the route table and adding an IAM policy on page 152
9. Testing FortiGate-VM HA failover on page 153



Creating VPCs and subnets

Each VPC requires private subnets:

- Each spoke VPC must each have one private subnet.
- The security VPC hub must have ten subnets: five per availability zone (AZ). Each AZ contains a subnet for the following:
  - Management
  - FortiGate private interface
  - FortiGate public interface
  - FortiGate heartbeat interface
  - Transit Gateway (TGW) attachment

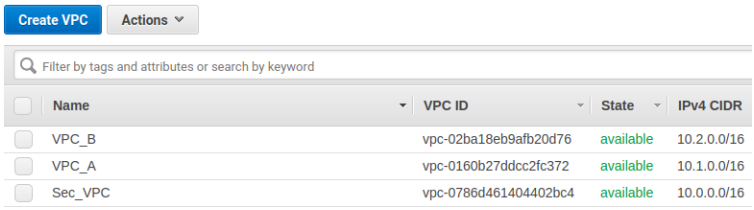
Create the spoke and security subnets in different AZs to demonstrate cross-AZ functionality. The example shows the following:

- Spoke 1 (A) has one subnet in the us-west-2a AZ.
- Spoke 2 (B) has one subnet in the us-west-2b AZ.
- The security hub has four subnets for each AZ in both the us-west-2a and us-west-2b AZs.

To create VPCs and subnets:

1. In the AWS console, open the VPC service.
2. Select *Your VPCs* and click the *Create VPC* button.
3. In the *Name tag* field, enter the desired name.
4. In the *IPv4 CIDR block* and *IPv6 CIDR block* fields, specify the desired CIDR for the spoke VPC.
5. Click *Create*.

6. Repeat the process to create another spoke VPC and a security VPC.



7. Create subnets:
   a. In the AWS console, go to the VPC service.
   b. Select *Subnets*, then click the *Create Subnet* button.
   c. In the *Name tag* field, enter the desired name.
   d. In the *VPC* field, enter the VPC ID of the desired spoke or security VPC.
   e. From the *Availability Zone* dropdown list, select the desired AZ.
   f. In the *IPv4 CIDR block* field, enter the desired CIDR block. Using default /24-sized subnets is recommended.
   g. Click *Create*.
   h. Repeat the process until you have the ten subnets.

After completing this process, the example has configured the following subnets:

- AZ A subnets in security VPC:
  - Public: 10.0.0.0/24
  - Internal: 10.0.1.0/24
  - Heartbeat: 10.0.2.0/24
  - Management: 10.0.3.0/24
  - TGW-Subnet: 10.0.4.0/24
- AZ B subnets in security VPC:
  - Public: 10.0.10.0/24
  - Internal: 10.0.11.0/24
  - Heartbeat: 10.0.12.0/24
  - Management: 10.0.13.0/24
  - TGW-Subnet: 10.0.14.0/24
- AZ A subnet in spoke 1 VPC: 10.1.1.0/24
- AZ B subnet in spoke 2 VPC: 10.2.1.0/24

Creating a Transit Gateway and related resources

To create a Transit Gateway and related resources:

1. Create a Transit Gateway (TGW):
   a. In the AWS console, open the VPC service.
   b. Select *Transit Gateways*, then click the *Create Transit Gateway* button.
   c. In the *Name tag* field, enter the desired name.
   d. Deselect *Default route table association* and *Default route table propagation* to prevent undesired association into the security route.
   e. Edit *Transit gateway CIDR blocks* to enter the same CIDR range as the TGW-Subnet that you configured in

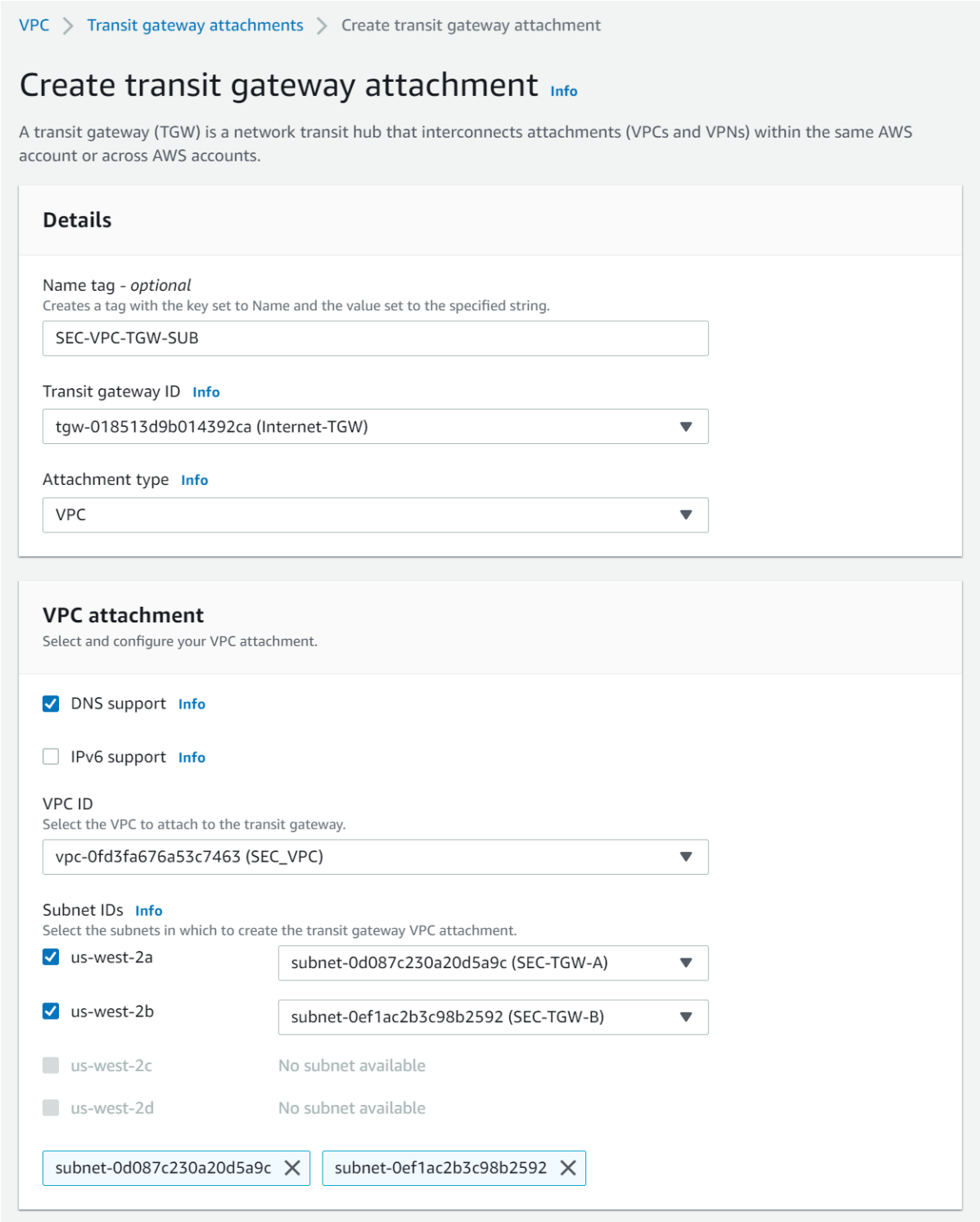
Creating VPCs and subnets on page 140.

f. Configure other fields as desired, then click *Create*.

g. Wait for the TGW state to change from *Pending* to *Available* before proceeding.

2. Create two TGW route tables: one for the security VPC and another for the spokes:
   a. In the AWS console, open the VPC service.
   b. Select *Transit Gateway Route Tables*, then click the *Create Transit Gateway Route Table* button.
   c. In the *Name tag* field, enter the desired name.
   d. From the *Transit Gateway ID* dropdown list, select the Transit Gateway ID.
   e. Click *Create*.
   f. Repeat the process for the spoke route table.
3. Create three TGW attachments, one for each VPC:
   a. In the AWS console, open the VPC service.
   b. Select *Transit Gateway Attachments*, then click the *Create Transit Gateway Attachment* button.
   c. From the *Transit Gateway ID* dropdown list, select the Transit Gateway ID.
   d. In the *Attachment type* field, select *VPC*.
   e. In the *Attachment name tag* field, enter the desired name.
   f. In the *VPC ID* field, enter the security VPC ID for the first attachment. This is TGW_Sec_VPC_Attachment in the screenshot.

g. For *Subnet IDs*, select the TGW-Subnet of each availability zone (AZ) for the security VPC.



h. Repeat the process for the other two VPC IDs, spokes A and B. For the subnet VPC attachment, select the corresponding AZ for each, then the *Subnet ID* dropdown list shows the spoke subnet that you created.

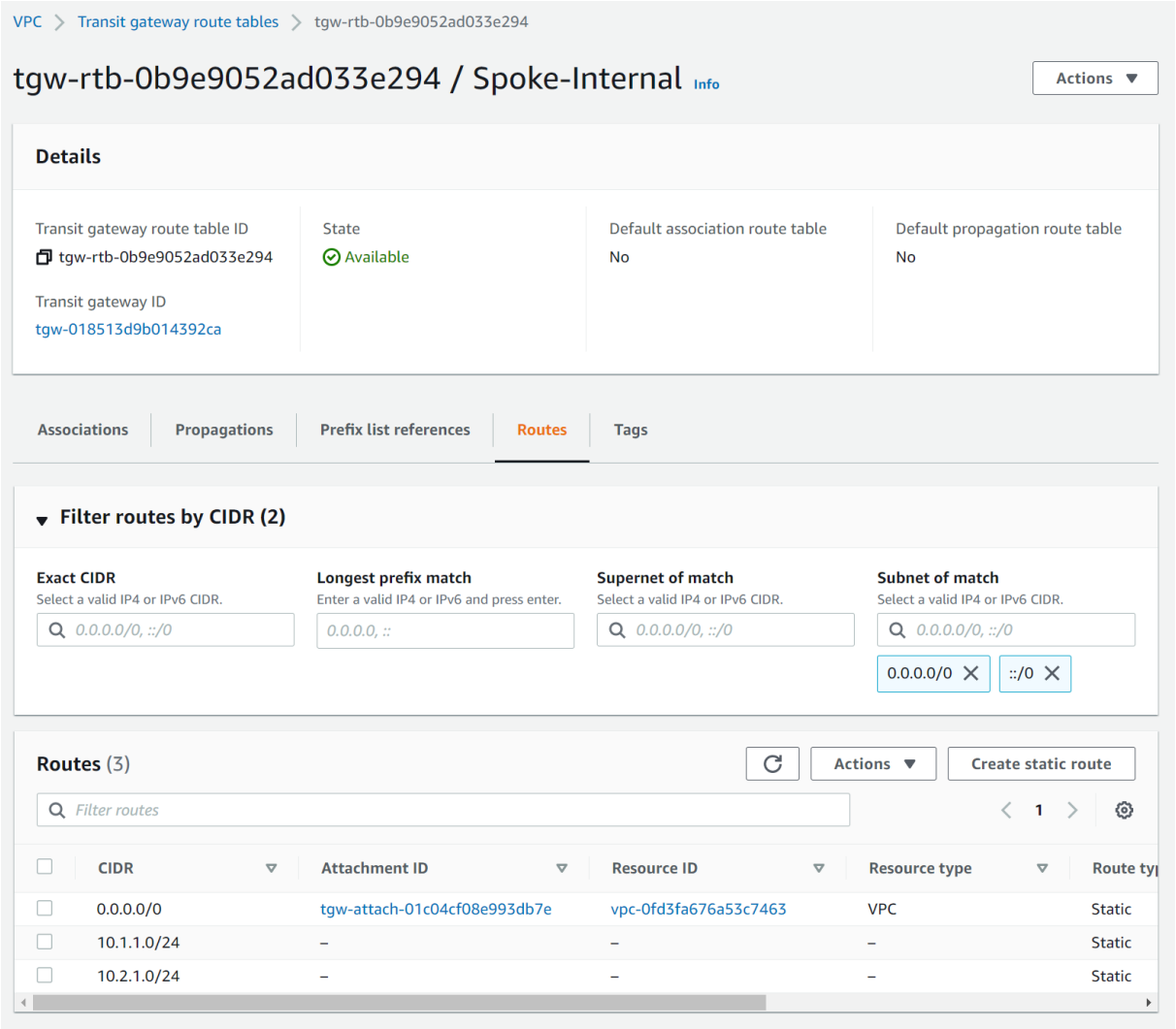
i. Wait for the State to become Available.

4. Create TGW associations:
   a. In the AWS console, open the VPC service.
   b. Select *Transit Gateway Route Tables*, then select the spoke route table.
   c. On the *Associations* tab, click the *Create Association* button.
   d. From the *Choose attachment to associate* dropdown list, select the spoke 1 VPC.
   e. Click *Create association*.
   f. Repeat the process for spoke B, which will be the second association for the route table.
   g. Wait for both associations to achieve the *Associated* state before proceeding.
   h. Next, select the security route table.
   i. Repeat the same as above to add the security VPC attachment to the security TGW route table. Click *Create association*.

You should associate the security attachment using the TGW-Subnets to the security route table. The spoke attachments will be associated to the spoke route table.

5. Add routes to the security TGW route table:
   a. In the AWS console, open the VPC service.
   b. Select *Transit Gateway Route Tables*, then select the security route table.
   c. Add a static route for each spoke subnet and select spoke VPC attachments.
6. Add routes to the spoke TGW route table:
   a. In the AWS console, open the VPC service.
   b. Select *Transit Gateway Route Tables*, then select the spoke route table.
   c. Add a static route for 0.0.0.0/0 to the security VPC attachment.

d. Add specific null routes: Spoke1(A) subnet, Spok2(B) Subnet, and SEC-Public Subnets.



Creating an Internet gateway

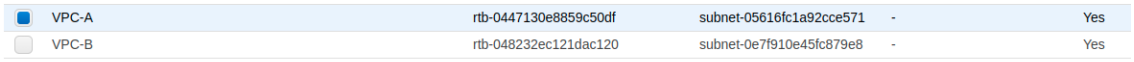
To create an Internet gateway:

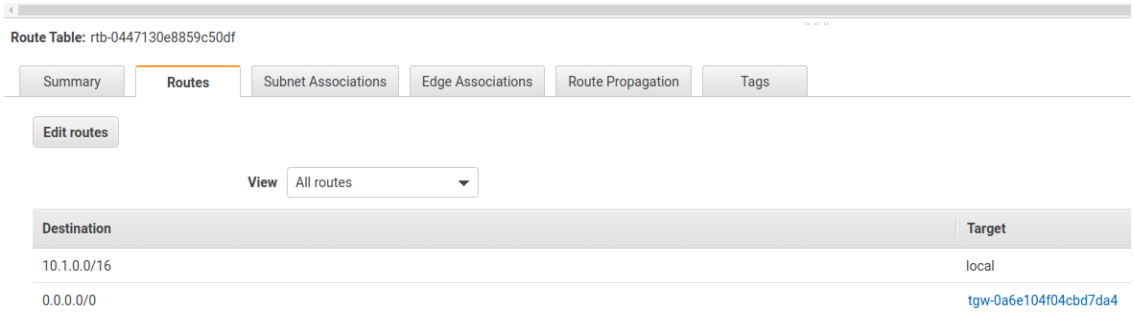
1. In the AWS console, open the VPC service.
2. Click the *Create Internet Gateway* button.
3. In the *Name tag* field, enter the desired name.
4. Click *Create*.
5. Attach the Internet gateway to the security VPC by selecting the Internet gateway and selecting *Attach to VPC* from the *Actions* menu.
6. Select the security VPC in the VPC dropdown list and click the *Attach* button to save.

Creating VPC route tables

To create a VPC route table:

1. In the AWS console, open the VPC service.
2. Configure two spoke VPC route tables:
   a. Select *Route Tables*, then click the *Create route table* button.
   b. Configure the desired name, then select the spoke A VPC. Click the *Create* button.
   c. Repeat the process for the spoke B VPC.
   d. Select the spoke A VPC route table. On the *Routes* tab, click the *Edit routes* button.
   e. Click *Add Route*.
   f. In the *Destination* field, specify 0.0.0.0/0.
   g. For the *Target*, specify the Transit Gateway (TGW). Click *Save Routes*.
   h. On the *Subnet Associations* tab, click the *Edit subnet associations* button.
   i. Select the spoke subnet that you just created, then click *Save*.





   j. Repeat the process for the spoke B route table.
3. Configure the security VPC internal route table:
   a. Click the *Create route table* button.
   b. Configure Sec_VPC_Internal as the name. This will be the route for internal traffic targeting the TGW. Select the security VPC.
   c. Click the *Create* button.
   d. Select the security VPC internal route table. On the *Routes* tab, click the *Edit routes* button.
   e. Click *Add Route*.
   f. In the *Destination* field, enter 0.0.0.0/0. Use the TGW as the target.
   g. Click *Save changes*.
   h. On the *Subnet Associations* tab, click the *Edit subnet associations* button.
   i. Select the internal/private subnets for both VPC availability zones (AZ) A and B, then click the *Save* button.
4. Configure the security VPC external route table:
   a. Click the *Create route table* button.
   b. Configure Sec_VPC_External as the name. This will be the Internet-facing route table. Select the security VPC.

c. Click the *Create* button.
d. Select the security VPC external route table. On the *Routes* tab, click the *Edit routes* button.
e. Add the following routes:

Destination	Target
0.0.0.0/0	Internet gateway
10.1.1.0/24	TGW
10.2.1.0/24	TGW

f. On the *Subnet Associations* tab, click the *Edit subnet associations* button.
g. Add the management, public, and heartbeat subnets for security VPC AZs, then click the *Save* button.

5. Configure the route table for return traffic to the spoke VPCs from the FortiGate:
a. Click the *Create route table* button.
b. Configure Sec_VPC_TGW as the name. Select the security VPC.
c. Click the *Create* button.
d. On the *Routes* tab, click the *Edit routes* button.
e. Add the following routes:

Destination	Target
10.1.1.0/24	TGW
10.2.1.0/24	TGW

f. On the *Subnet Associations* tab, click the *Edit subnet associations* button.
g. Select the TGW subnets for both AZs A and B, then click the *Save* button.

You will add a route that targets the ENI ID of port2 of the primary FortiGate in a later step.

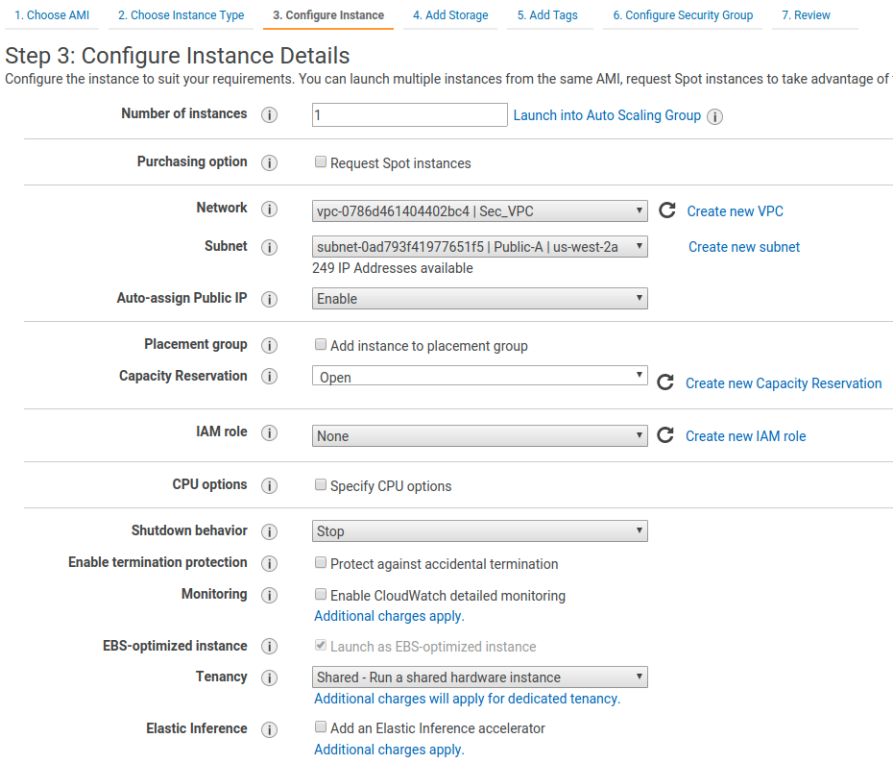
Deploying FortiGate-VM from AWS marketplace

To deploy the FortiGate-VM from the AWS marketplace:

1. On the AWS marketplace, find a FortiGate-VM listing and version available for selection. This example uses FortiGate-VM On-Demand 6.2.1, ami-0439b030915c59e67, on c5.xlarge instances. Available versions may change.

Deploying a high availability (HA) pair requires four network interfaces. Instances smaller than x.large do not support four network interfaces and do not work for this deployment type.

2. Launch the FortiGate-VM through Elastic Compute Cloud.



3. Deploy the VM with only one network interface with public IP address assignment enabled.
4. Repeat the steps for the second VM instance in a second availability zone.
5. To enable management access to the FortiGate-VMs and HA traffic flow, open the security group attached to the FortiGate-VMs:
   a. In the AWS console, select *Security Groups*.
   b. Click the *Create Security Group* button.
   c. Add a rule with a source of 0.0.0.0/0 for all traffic types.
   d. Assign the rule to all interfaces on both FortiGate-VMs. The next step in the process, Adding network interfaces and elastic IP addresses to the FortiGate-VMs on page 149, explains creating additional network interfaces. You can tighten the security group later.

Adding network interfaces and elastic IP addresses to the FortiGate-VMs

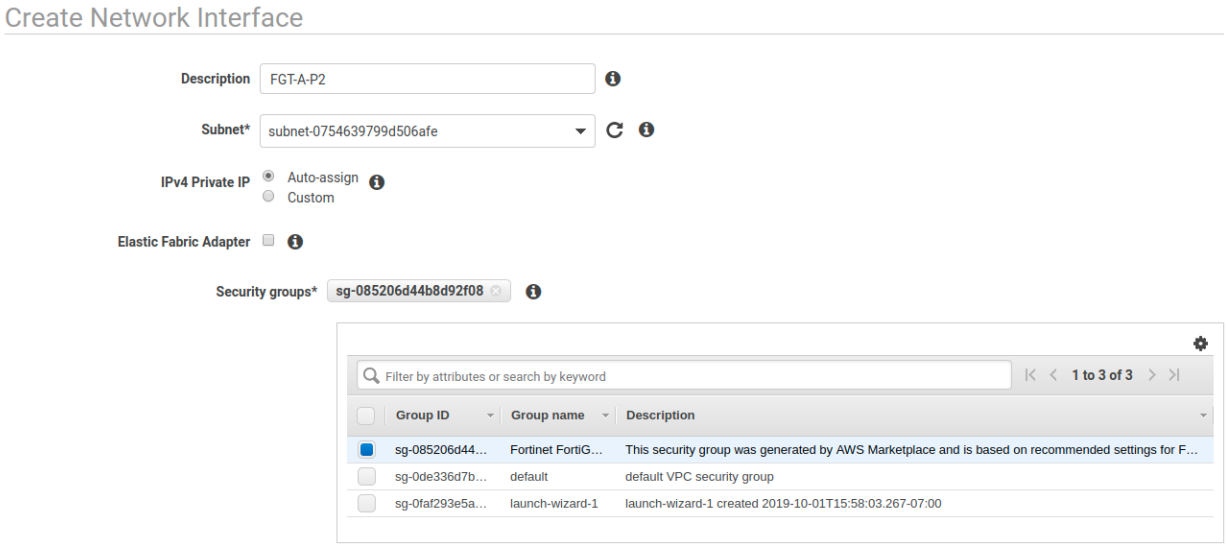
To add network interfaces and elastic IP addresses to the FortiGate-VMs:

1. Add network interfaces:
   a. In the AWS console, open the Elastic Compute Cloud (EC2) service.
   b. Select *Network Interfaces*, then click the *Create Network Interface* button.
   c. Provide a description of the interface, specify the private subnet in availability zone A and specify the security group created in Deploying FortiGate-VM from AWS marketplace on page 148.
   d. Click *Yes, Create*.
   e. Click the newly created interface. From the *Actions* dropdown list, select *Change Source/Dest Check*. Disable *Source/Dest Check* and save.
   f. From the *Actions* dropdown list, select *Attach*.

g. From the dropdown list, select the first FortiGate-VM. Click *Attach*.
h. Repeat the process for the second FortiGate-VM.

2. Repeat step 1 for the secondary FortiGate-VM. Each FortiGate-VM will be attached with four network interfaces:

Port	Purpose
Port1 (eth0)	Public network IP address. Elastic IP address (EIP) only for primary FortiGate in high availability group.
Port2 (eth1)	Private network IP address
Port3 (eth2)	Heartbeat network IP address
Port4 (eth3)	Management network IP address. EIP on each FortiGate.



3. Add elastic IP addresses (EIPs):
   a. In the AWS console, open the EC2 service.
   b. Select *Elastic IPs*, then click the *Allocate new address* button.
   c. Accept the defaults, then click the *Allocate* button.
   d. Repeat steps a-c twice for a total of three EIPs:
      - One EIP is for port1 that will move to the secondary FortiGate-VM during failover.
      - Two EIPs are for high availability (HA) management ports.
4. Attach three EIPs as follows:
   a. Port 1 of the primary FortiGate by selecting *Network Interface* as the *Resource Type* and its eth0 ENI network interface to associate.
   b. Port 4 of the primary FortiGate by selecting *Network Interface* as the *Resource Type* and its eth3 ENI network interface to associate.
   c. Port 4 of the secondary FortiGate by selecting *Network Interface* as the *Resource Type* and its eth 3ENI network interface to associate.

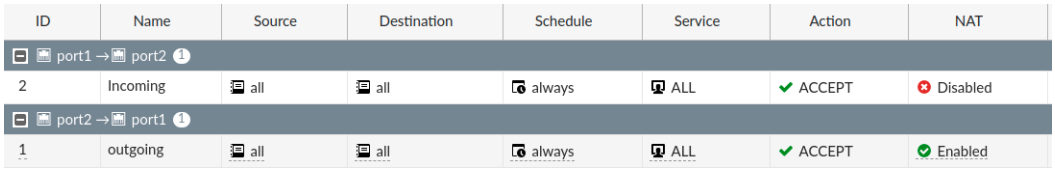
   The primary FortiGate port 1 EIP will fail over to the secondary FortiGate in case of failure.

   Port4 elastic IPs are not accessible until you form an HA cluster.

Configuring the FortiGate-VMs

To configure the FortiGate-VMs:

1. Log in to the primary FortiGate-VM:
   a. In the browser, enter https:// followed by the by the port1 (eth0) public IP address.
   b. Click *Advanced*, then proceed with the warning.
   c. Enter admin and the instance ID as the username and password, respectively, for the primary FortiGate-VM, and proceed to change the default password.
2. Configure the primary FortiGate-VM:
   a. Go to *Network* > *Interfaces*. Confirm all four port IP address settings.
   b. Go to *Network* > *Static Routes*. Set the static route for port1 and port2 to the corresponding gateway on each FortiGate-VM. Usually the last number is 1 for the same subnet (i.e. 10.0.0.1) on AWS.
   c. Ensure that the 10.2.1.0/24 and 10.1.1.0/24 (or your internal subnet CIDR) route has been created to forward internal traffic out of port2.
   d. Go to *System* > *HA*. Configure high availability (HA) settings. After enabling active-passive mode, you can only access the FortiGate-VM through the HA management port (elastic IP address on port4).
      i. From the *Mode* dropdown list, select *Active-Passive*.
      ii. In the *Device priority* field, enter a value that will be higher than the one you configure for the secondary node.
      iii. Configure the *Group name* and *Password* fields.
      iv. Enable *Session pickup*.
      v. For *Heartbeat interfaces*, select *port3*.
      vi. Enable *Management Interface Reservation*. From the *Interface* dropdown list, select *port4*. Specify the gateway for the same subnet.
      vii. Enable *Unicast Heartbeat*. Specify the port3 IP address of the peer FortiGate.
3. Log in to and configure the secondary FortiGate-VM by repeating steps 1-2. When configuring device priority in HA settings, set a lower value than that of the primary node.
4. Configure policies to forward internal traffic out from port1. You only need to configure such policies on the primary FortiGate-VM, as the policy configuration will synchronize between the FortiGate-VMs.



ID	Name	Source	Destination	Schedule	Service	Action	NAT
port1 → port2 1							
2	Incoming	all	all	always	ALL	ACCEPT	Disabled
port2 → port1 1							
1	outgoing	all	all	always	ALL	ACCEPT	Enabled

5. You must configure a VDOM exception to prevent interface synchronization between the two FortiGates. Run the following commands in the FortiOS CLI:

```
config system vdom-exception
   edit 1
      set object system.interface
   next
   edit 2
      set object router.static
   next
   edit 3
      set object firewall.vip
   next
end
```

6. (Optional) You an configure an AWS SDN connector to allow population of dynamic objects such as policy objects. See Configuring the SDN connector to populate dynamic objects.

Updating the route table and adding an IAM policy

To update the route table and add an IAM policy:

1. Update the route table:
   a. After configuring the internal network ports, you must route all internal traffic to the elastic network interface (ENI) of the primary FortiGate-VM port2. In the AWS console, open the Elastic Cloud Compute service.
   b. Select *Instances*, then select the primary FortiGate-VM.
   c. On the *Description* tab, select port2 (eth1) and copy the interface ID.
   d. Save the content into a text editor.
   e. In the AWS console, open the VPC service.
   f. Select *Route Tables*, then select the Sec_VPC_TGW route table.
   g. On the *Routes* tab, click the *Edit Routes* button.
   h. Add the following route:

Destination	Target
0.0.0.0/0	Paste the ENI ID of port2 of the primary FortiGate.

   i. Click *Save*.
   j. Ensure that the Sec_VPC_TGW route table has the following routes:

Destination	Target
10.1.1.0/24	Transit Gateway (TGW)
10.2.1.0/24	TGW
0.0.0.0/0	ENI ID of port2 of the primary FortiGate.
10.0.0.0/16	Local. Depends on the security VPC network settings.

Check that the TGW subnets (security VPC TGW subnets) for both availability zones A and B are associated with this routing table.

2. Both firewalls need an IAM policy attached to make API calls to AWS to move the elastic IP address on port1 and network interface on port2 between primary and secondary FortiGate-VMs. Go to the AMI service and create a role with the following policy: {

```
"Version": "2012-10-17",
"Statement": [
   {
      "Action": [
         "ec2:Describe*",
         "ec2:AssociateAddress",
         "ec2:AssignPrivateIpAddresses",
         "ec2:UnassignPrivateIpAddresses",
         "ec2:ReplaceRoute"
      ],
```

```
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

3. Attach the AMI role to both FortiGate-VMs by selecting the FortiGate EC2 instance and selecting *Attach/Replace IAM Role* in the *Actions* menu.

Testing FortiGate-VM HA failover

The following prerequisites are required for successful failover:

- Two FortiGates exist in the same virtual private cloud and different availability zones. The two FortiGates must also have the same FortiOS build (FGT_VM64_AWS or FGT_VM64_AWSONDEMAND) installed and the same instance shape. In this example, both FortiGate-VM instances were deployed as C5.xlarge.
- The high availability (HA) management port can resolve DNS and make API calls to AWS. The HA management port is not blocked by the security group and routed to the Internet gateway on all cluster members.
- If using FortiGate-VM BYOL instances, both FortiGate-VMs have valid licenses.
- Minimum sufficient IAM roles as shown in Updating the route table and adding an IAM policy on page 152

To test FortiGate-VM HA failover:

1. To ensure that the FortiGate-VMs are in sync, run `get system ha status`:

```
master # get sys ha stat
HA Health Status: OK
Model: FortiGate-VM64-AWSONDEMAND
Mode: HA A-P
Group: 0
Debug: 0
Cluster Uptime: 1 days 1:50:18
Cluster state change time: 2019-01-31 18:20:47
Master selected using:
<2019/01/31 18:20:47> FGTAWS0006AB1961 is selected as the master because it has the
     largest value of override priority.
<2019/01/31 18:20:47> FGTAWS0006AB1961 is selected as the master because it's the only
     member in the cluster.
ses_pickup: enable, ses_pickup_delay=disable
override: disable
Master: FGTAWS0006AB1961, HA operating index = 0
Slave : FGTAWS000B29804F, HA operating index = 1
```

2. Enable debug mode on the secondary FortiGate:

```
diagnose debug enable
diagnose debug application awsd -1
Debug messages will be on for unlimited time.
```

3. Shut down the primary FortiGate. In the event of a successful failover, the secondary FortiGate CLI shows the following:

```
slave # Become HA master
send_vip_arp: vd root master 1 intf port1 ip 10.0.10.11
send_vip_arp: vd root master 1 intf port2 ip 10.0.11.11
awsd get instance id i-0b29804fd38976af4
awsd get iam role WikiDemoHARole
awsd get region us-west-2
awsd get vpc id vpc-0ade7ea6e64befbfc
```

```
awsd doing ha failover for vdom root
awsd associate elastic ip for port1
awsd associate elastic ip allocation eipalloc-06b849dbb0f76555f to 10.0.10.11 of eni
     eni-0ab045a4d6dce664a
awsd associate elastic ip successfully
awsd update route table rtb-0a7b4fec57feb1a21, replace route of dst 0.0.0.0/0 to eni-
     0c4c085477aaff8c5
awsd update route successfully
```

4. Verify on AWS that the public EIP on port1 and the Sec_VPC_Internal route table point to the new primary FortiGate port2 ENI.

Deploying FortiGate-VM using Terraform

See the following:

- Single FortiGate-VM deployment
- Active-passive HA cluster deployment in the same availability zone
- Active-passive HA cluster deployment across two availability zones
- Active-active FortiGate-VM pair deployment using internal and external network load balancers
- AWS Transit Gateway hub-spoke FortiGate-VM deployment across two availability zones

Support

For issues, see this GitHub project's Issues tab. For other questions related to the GitHub project, contact github@fortinet.com.

SDN connector integration with AWS

Certificate-based SDN connector integration

See the *FortiOS Administration Guide*.

Configuring an AWS SDN connector using IAM roles

The following summarizes minimum sufficient IAM roles for this deployment:

```
{
   "Version": "2012-10-17",
   "Statement": [
   {
   "Action": [
   "ec2:Describe*"
   ],
   "Resource": "*",
   "Effect": "Allow"
   }
   ]
}
```

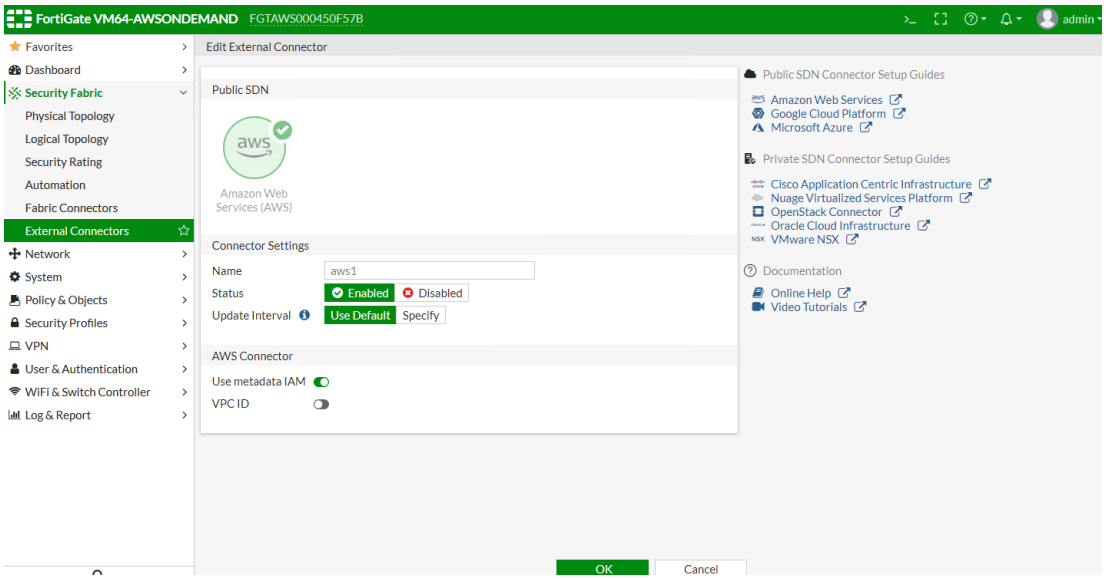
For instances running in AWS (on demand or BYOL), you can set up the AWS SDN connector using AWS Identify and Access Management (IAM) credentials.

IAM authentication is available only for FGT-AWS and FGT-AWSONDEMAND platforms.

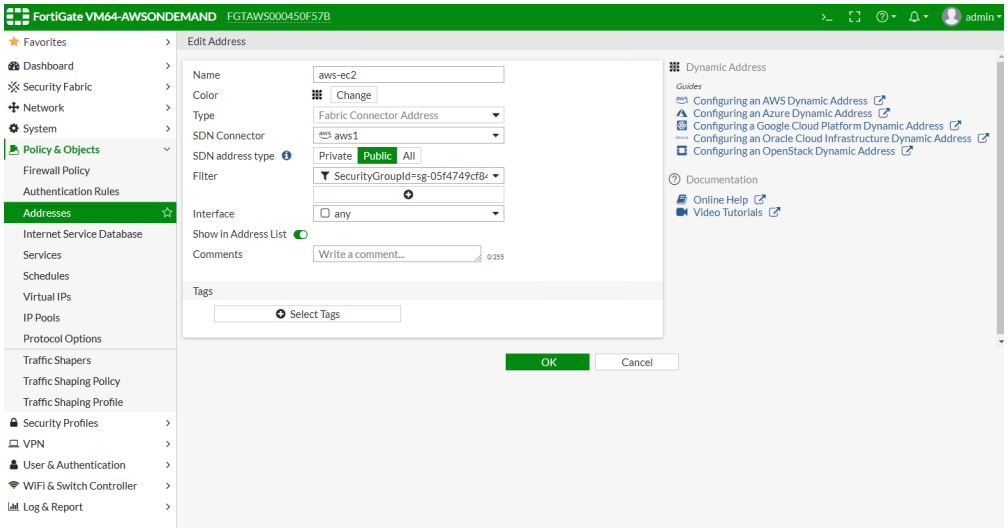
To configure AWS SDN connector using the GUI:

1. Configure the AWS SDN connector:
   a. Go to *Security Fabric* > *External Connectors*.
   b. Click *Create New*, and select *Amazon Web Services (AWS)*.

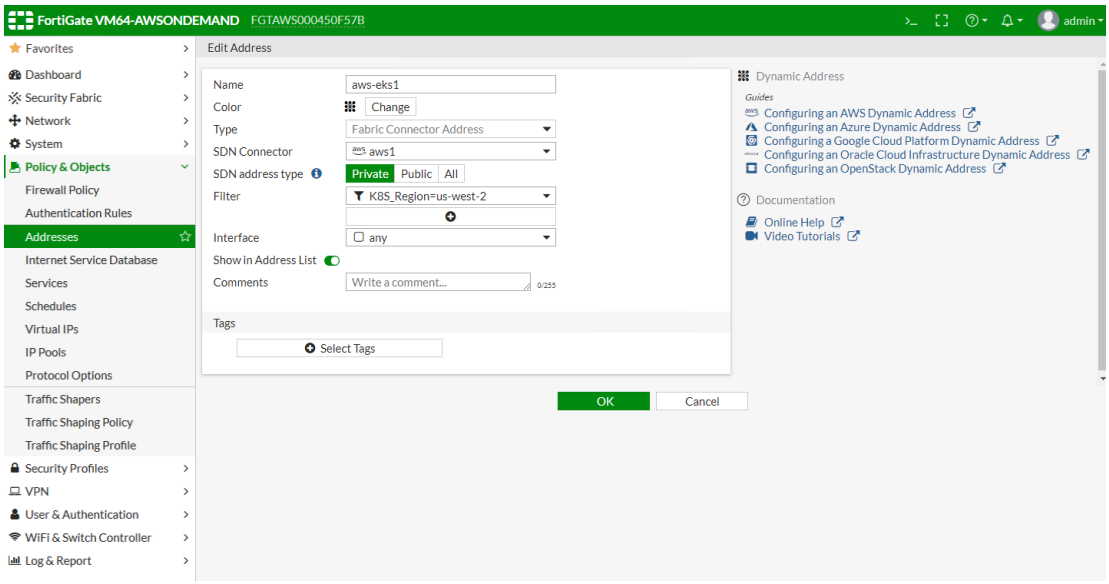
c. Configure as shown:



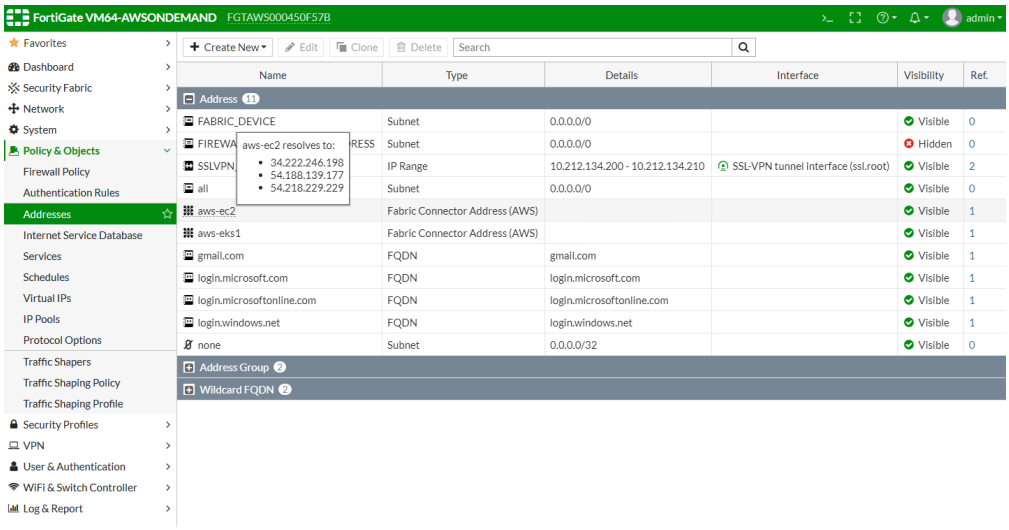
2. Create a dynamic firewall address for the configured AWS SDN connector:
   a. Go to *Policy & Objects* > *Addresses*.
   b. Click *Create New*, then select *Address*.
   c. Configure the address as shown, selecting the desired filter in the *Filter* dropdown list.
   The following is an example for a public SDN address type:



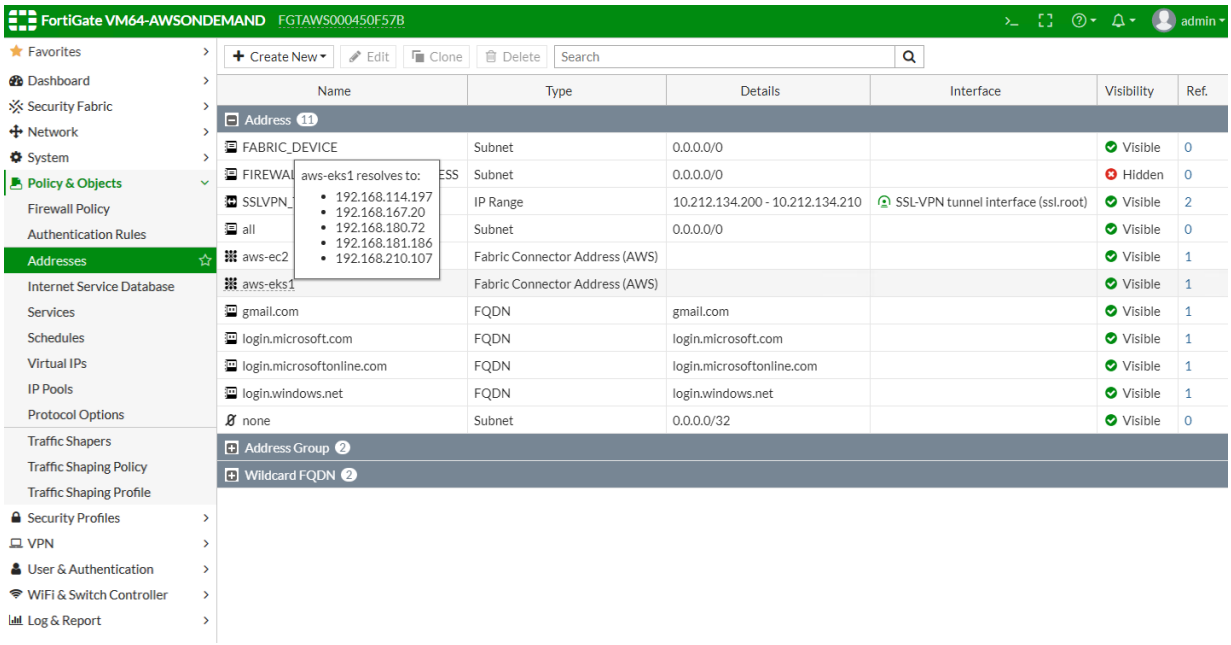
The following is an example for a private SDN address type:



3. Ensure that the AWS SDN connector resolves dynamic firewall IP addresses:
   a. Go to *Policy & Objects* > *Addresses*.
   b. Hover over the address created in step 2 to see a list of IP addresses for instances that belong to the security group configured in step 2.
   The following is an example for a public SDN address type:



   The following is an example for a private SDN address type:



To configure AWS SDN connector using CLI commands:

1. Configure the AWS connector:

```
config system sdn-connector
   edit "aws1"
      set status enable
      set type aws
      set use-metadata-iam enable
      set update-interval 60
   next
end
```

2. Create a dynamic firewall address for the configured AWS SDN connector with the supported filter: Dynamic firewall address IPs are resolved by the SDN connector.

```
config firewall address
   edit "aws-ec2"
      set type dynamic
      set sdn "aws1"
      set filter "SecurityGroupId=sg-05f4749cf84267548"
      set sdn-addr-type public
   next
   edit "aws-eks1"
      set type dynamic
      set sdn "aws1"
      set filter "K8S_Region=us-west-2"
   next
end
```

3. Confirm that the AWS SDN connector resolves dynamic firewall IP addresses using the configured filter:

```
config firewall address
   edit "aws-ec2"
      set type dynamic
      set sdn "aws1"
      set filter "SecurityGroupId=sg-05f4749cf84267548"
      set sdn-addr-type public
```

```
        config list
           edit "34.222.246.198"
           next
           edit "54.188.139.177"
           next
           edit "54.218.229.229"
           next
        end
     next
     edit "aws-eks1"
        set type dynamic
        set sdn "aws1"
        set filter "K8S_Region=us-west-2"
        config list
           edit "192.168.114.197"
           next
           edit "192.168.167.20"
           next
           edit "192.168.180.72"
           next
           edit "192.168.181.186"
           next
           edit "192.168.210.107"
           next
        end
     next
  end
```

AWS Kubernetes (EKS) SDN connector

AWS SDN connectors support dynamic address groups based on AWS Kubernetes (EKS) filters. The following summarizes minimum permissions for this deployment:

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "VisualEditor0",
         "Effect": "Allow",
         "Action": [
            "ec2:Describe*",
            "eks:DescribeCluster",
            "eks:ListClusters"
         ],
         "Resource": "*"
      }
   ]
}
```

Once you have the proper permissions for EKS, you must follow the steps at Managing Users or IAM Roles for your Cluster for EKS to properly pull data from the cluster. The following shows a successful pull of IP addresses from the EKS cluster:

```
awsd getting IPs from EKS cluster: dchao-cluster (us-west-2), endpoint:
     https://F57B834C1ADA8ED7FA3CAFB36073D384.gr7.us-west-2.eks.amazonaws.com
```

```
kube url: https://F57B834C1ADA8ED7FA3CAFB36073D384.gr7.us-west-
     2.eks.amazonaws.com/api/v1/services
kube host: F57B834C1ADA8ED7FA3CAFB36073D384.gr7.us-west-
     2.eks.amazonaws.com:443:100.21.79.123
kube url: https://F57B834C1ADA8ED7FA3CAFB36073D384.gr7.us-west-
     2.eks.amazonaws.com/api/v1/nodes
kube host: F57B834C1ADA8ED7FA3CAFB36073D384.gr7.us-west-
     2.eks.amazonaws.com:443:100.21.79.123
k8s node ip: 172.31.34.72, nodename: ip-172-31-34-72.us-west-2.compute.internal
cluster: dchao-cluster, region: us-west-2, zone: us-west-2b
k8s node ip: 18.237.109.243, nodename: ip-172-31-34-72.us-west-2.compute.internal
cluster: dchao-cluster, region: us-west-2, zone: us-west-2b
kube url: https://F57B834C1ADA8ED7FA3CAFB36073D384.gr7.us-west-
     2.eks.amazonaws.com/api/v1/pods
kube host: F57B834C1ADA8ED7FA3CAFB36073D384.gr7.us-west-
     2.eks.amazonaws.com:443:100.21.79.123
k8s pod ip: 172.31.34.72, podname: aws-node-7kbm5, namespace: kube-system
cluster: dchao-cluster, region: us-west-2, zone: us-west-2b
k8s pod ip: 172.31.45.127, podname: coredns-6f647f5754-85m88, namespace: kube-system
cluster: dchao-cluster, region: us-west-2, zone: us-west-2b
k8s pod ip: 172.31.38.147, podname: coredns-6f647f5754-87ch7, namespace: kube-system
cluster: dchao-cluster, region: us-west-2, zone: us-west-2b
k8s pod ip: 172.31.34.72, podname: kube-proxy-ks9pw, namespace: kube-system
cluster: dchao-cluster, region: us-west-2, zone: us-west-2b
```

After configuring the above, follow the instructions in the *FortiOS Cookbook* to complete configuration.

Populating threat feeds with GuardDuty

AWS GuardDuty is a managed threat detection service that monitors malicious or unauthorized behaviors/activities related to AWS resources. GuardDuty provides visibility of logs called "findings", and Fortinet provides a Lambda script called "aws-lambda-guardduty", which translates feeds from AWS GuardDuty findings into a list of malicious IP addresses in an S3 location, which a FortiGate-VM can consume as an external threat feed after being configured to point to the list's URL. To use this feature, you must subscribe to GuardDuty, CloudWatch, S3, and DynamoDB.

Installing and configuring GuardDuty requires knowledge of:

- CLI
- AWS Lambda function, DynamoDB, S3 bucket, and IAM
- Node.js

The Lambda script is available to download on GitHub.

Security implications

It is highly recommended that you create a dedicated AWS IAM role to run this Lambda function. The role should have limited permissions to restrict operation on a dedicated S3 bucket resource for only this project.

It is never suggested to attach a full control policy such as AmazonS3FullAccess, which has full permissions to all resources under your Amazon AWS account, to the role which runs the Lambda function. Allowing full-access permissions to all resources may put your resources at risk.

Following is a list of permissions required for the IAM role to run this project across the required AWS services:

AWS service	Permission
S3	ListBucket, HeadBucket, GetObject, PutObject, PutObjectAcl
DynamoDB	DescribeStream, ListStreams, Scan, GetShardIterator, GetRecords, UpdateItem

Parameters

GuardDuty findings give visibility on the following:

- Severity: high/medium/low (associated with scores)
- Where the behavior/activity occurred: Region, resource ID, account ID
- When: last seen date/time
- Count
- Detailed information
  - Affected resource: type/instance ID/image ID/port/resource type/image description/launch time/tags/network interfaces (public IP, private IP, subnet ID, VPC ID, security groups)
  - Action: type/connection direction
  - Actor
  - Additional

For more information about Amazon GuardDuty, see the Amazon GuardDuty official website.

There are five configurable environment variables in the Lambda function:

Variable name	Type	Description
MIN_SEVERITY	Integer	The minimum severity to block an IP address. Defaults to 3. Value ranges from 1 to 10 by AWS GuardDuty definition.
S3_BUCKET	Text	S3 bucket name to store the IP block list file. No default value. Must specify.
S3_BLOCKLIST_KEY	Text	Path to the IP block list file within the S3 bucket. No default value. Must specify. The relative file path to the S3 bucket.
REGION	Text	AWS region to run Lambda, DynamoDB services. Must specify.
DDB_TABLE_NAME	Text	DynamoDB table name which stores malicious IP addresses from findings. Must specify.

Installation

You can follow the installation steps below to setup this Lambda function:

Prerequisites

See below for a list of tools required to deploy this project before installation. Some prerequisites are platform-specific. Choose the right one for your OS (such as Windows, Linux, or macOS).

- Node.js (6.5.0 or later)
- npm. Although npm comes with Node.js, check here for how to install npm and manage the npm version.
- AWS account
- Git (latest version)
- *Git Bash (latest version). Git Bash is a solution for Windows platform users to run the following installation steps. The article Use git, ssh and npm on windows with Git Bash gives more information about setting up Git Bash on Windows.

Preparing the deployment package

When you have all prerequisites ready, you can continue the installation as below. The commands in each steps are intended to run in Terminal or Git Bash only.

You must create a deployment package from the local Git project repository, which will be uploaded for the Lambda function creation in a later step.

To prepare the deployment package:

1. Clone this project into the "guardduty" folder in your current local directory, and enter the project directory:
```
$ git clone https://github.com/fortinet/aws-lambda-guardduty.git guardduty
$ cd guardduty
```
2. Install project dependencies:
```
$ npm install
```
3. Build this project locally to create a deployment package .zip file. The file will be located in ./dist/aws_lambda_guardduty.zip:
```
$ npm run build
```

Setting up the S3 bucket

This project needs one S3 bucket. The example in the following steps creates an S3 bucket named "my-aws-lambda-guardduty". The example uses the bucket name in some configuration steps. Due to bucket naming limitations in S3, each bucket should have a globally unique name. Therefore, your bucket should have a different name than the example's. Write down your bucket name, since it is used in other configuration steps.

Create the S3 bucket to store the IP block list. In this example, the bucket is named my-aws-lambda-guardduty. This bucket is required to run this project. Although bucket creation is region-specific, once created, the bucket can be accessed from any region. Do not grant the bucket public access permissions. The Lambda function points to this bucket through its S3_BUCKET environment variable.

Setting up the DynamoDB table

One DynamoDB table with the stream feature enabled is required to store records of malicious IP addresses from GuardDuty findings. DynamoDB tables and Lambda functions are region-specific so you must create the table and the Lambda function in the same AWS region. A DynamoDB trigger on this table is created to cause the Lambda function to

execute. Since the Lambda function has not been created yet, instructions to create the trigger are provided later in Setting up the DynamoDB stream trigger.

1. Create the DynamoDB table. In this example, the table is named my-aws-lambda-guardduty-db.
   a. For the primary key, do the following:
      i. Input the value *finding_id*. This value is case-sensitive.
      ii. From the data type dropdown list, select *String*.
   b. Add a sort key:
      i. Input the value *ip*. This value is case-sensitive.
      ii. From the data type dropdown list, select *String*.
   c. Check used default settings for *Table settings*.
   d. Click *Create*.
2. Enable the Stream feature on the table.
   a. On the *Overview* tab, click *Manage Stream*, select *Keys only*, then click *Enable* to save.
   b. Write down the *Latest stream ARN*. This ARN is used in the IAM policy creation step.

Setting up the IAM role and policies

An IAM role is created to run the Lambda function. Three policies attach to the IAM role. The first one is a user-managed policy which grants permissions to operation on the S3 bucket my-aws-lambda-guardduty. The second one is a user-managed policy which grants permission to operation on the DynamoDB table my-aws-lambda-guardduty-db. The third one is an AWS-managed policy which allows the Lambda function to write logs to CloudWatch.

1. Create a policy to operate on the S3 bucket.
   a. Choose S3 as its service.
   b. In *Access level*, add *ListBucket* on *List*, *HeadBucket* and *GetObject* on *Read*, *PutObject* on *Write*, and *PutObjectAcl* on *Permissions management*.
   c. In *Resources*, choose *Specific*.
      i. For the *bucket* resource type, add the my-aws-lambda-guardduty S3 bucket ARN (for example, *arn:aws:s3:::my-aws-lambda-guardduty*) to restrict access to any file in the specific bucket only.
      ii. For the *object* resource type, add the my-aws-lambda-guardduty S3 bucket ARN and a /* wildcard (for example, *arn:aws:s3:::my-aws-lambda-guardduty/**) to restrict access to any file in the specific bucket only.
   d. Click *Review Policy*, then *Save Changes*. The policy in JSON form looks like the code snippet below:

```
{
        "Version": "2012-10-17",
        "Statement": [
           "{
              "Sid": "VisualEditor0",
              "Effect": "Allow",
              "Action": [
                 "s3:PutObject",
                 "s3:GetObject",
                 "s3:ListBucket",
                 "s3:PutObjectAcl"
              "],
              "Resource": [
                 "arn:aws:s3:::my-aws-lambda-guardduty",
                 "arn:aws:s3:::my-aws-lambda-guardduty/*"
              ]
           },
```

```
            {
               "Sid": "VisualEditor1",
               "Effect": "Allow",
               "Action": "s3:HeadBucket",
               "Resource": "*"
            }
         ]
      }
```

2. Create a policy to operate on the DynamoDB table.
   a. Choose DynamoDB as its service.
   b. In *Access level*, add *ListStreams* on *List*, *DescribeStream*, *GetRecords*, *GetShardIterator*, *Scan* on *Read*, and *UpdateItem* on *Write*.
   c. In *Resources*, choose *Specific*.
      i. For the *stream* resource type, add the my-aws-lambda-guardduty-db latest stream ARN (for example,arn:aws:dynamodb:us-east-1:888888888888:table/my-aws-lambda-guardduty-db/2018-07-20T10:30:10.888). Replace the Stream label content with the * wildcard to allow for access to any stream resource of the my-aws-lambda-guardduty-db table.
      ii. Forthe *table* resource type, add the my-aws-lambda-guardduty-db DynamoDB table ARN (for example, arn:aws:dynamodb:us-east-1:888888888888:table/my-aws-lambda-guardduty-db) to restrict access to the specific table only.
   d. Click *Review Policy*, then *Save Changes*. The policy in JSON form looks like the code snippet below:

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "VisualEditor0",
         "Effect": "Allow",
         "Action": [
            "dynamodb:GetShardIterator",
            "dynamodb:Scan",
            "dynamodb:UpdateItem",
            "dynamodb:DescribeStream",
            "dynamodb:GetRecords"
         ],
         "Resource": [
            "arn:aws:dynamodb:us-east-1:888888888888:table/my-aws-lambda-guardduty-
                 db/stream/*",
            "arn:aws:dynamodb:us-east-1:888888888888:table/my-aws-lambda-guardduty-db"
         ]
      },
      {
         "Sid": "VisualEditor1",
         "Effect": "Allow",
         "Action": "dynamodb:ListStreams",
         "Resource": "*"
      }
   ]
}
```

3. Create an IAM role to run the Lambda function.
   a. Choose the Lamba service that will use this role.
   b. Attach the two user-managed policies created in the previous steps to this role.
   c. Attach the AWS-managed policy AWSLambdaBasicExecutionRole to this role.

Creating the Lambda function

The Lambda function is created with the deployment package generated in Preparing the deployment package on page 163. This package is uploaded directly to this Lambda function. The Lambda function has five configurable environment variables for severity, AWS region, DynamoDB table name, and IP block list file entry point.

1. Create a function that authors from scratch.
   a. Give the function a unique name.
   b. For its *Runtime*, select *Node.js 6.10*.
   c. For *Role*, select *Choose an existing role*. Select the role created in Setting up the IAM role and policies on page 164.
2. Set up the function code.
   a. For code entry type, select *Upload a .ZIP file*. The *Function package* field appears.
   b. For *Function package*, click *Upload* to upload the deployment package .zip file generated in Preparing the deployment package.
   c. For *Handler*, enter *index.handler*.
3. Set up the environment variables. Note values for key fields are case-sensitive and should all be in upper case.
   a. Add a key *MIN_SEVERITY* and input a value of *3*.
   b. Add a key *S3_BUCKET* and paste the name of the S3 bucket created in Setting up the S3 bucket on page 163. In this example, the S3 bucket name is my-aws-lambda-guardduty.
   c. Add a key *S3_BLOCKLIST_KEY* and input a value of *ip_blocklist* or a different name as desired.
   d. Add a key *REGION* and input the AWS region where your Lambda function and DynamoDB table are situated. For example, the region of US East (N. Virginia) is us-east-1. For information about AWS Regions, please see AWS Regions and Endpoints.
   e. Add a key *DDB_TABLE_NAME* and input the name of the DynamoDB table created in Setting up the DynamoDB table on page 163. In this example, the DynamoDB table name is my-aws-lambda-guardduty-db.
4. Save the Lambda function.

Setting up the DynamoDB stream trigger

You must add a trigger to the DynamoDB table created in Setting up the DynamoDB table on page 163. This trigger is the key that causes the Lambda function to generate a full IP block list to a static file in the S3 bucket.

The following describes how to create a trigger on a DynamoDB table

1. In DynamoDB, click the table to toggle on its detail window.
2. On the *Triggers* tab, click *Create Trigger*, then *Existing Lambda function* from the dropdown list.
3. From the *Function* dropdown list, select the Lambda function created in Creating the Lambda function on page 166.
4. Leave the *Batch size* value at its default, which is normally 100.
5. Select the *Enable trigger* checkbox.
6. Click *Create*.

At this point, installation is complete, although the AWS CloudWatch and GuardDuty services need additional configuration to work with the Lambda function.

Setting up CloudWatch

In this section, a CloudWatch event rule is created to invoke the Lambda function based on events happening in GuardDuty findings. If you have not subscribed to GuardDuty yet, you must subscribe to it before moving on. For information about GuardDuty, see Amazon GuardDuty.

The following describes creating a new event rule.

1. For *Event Source*, choose *Event Pattern*, and select *Events by Service* from the dropdown list.
2. For *Service Name*, select *GuardDuty* from the dropdown list.
3. For *Event Type*, select *GuardDuty Finding* from the dropdown list.
4. Check that the *Event Pattern Preview* looks like the code snippet below.

```
{
   "source": [
      "aws.guardduty"
   ],
      "detail-type": [
         "GuardDuty Finding"
   ]
}
```

5. For the targets, click *Add Target** and select *Lambda function* from the dropdown list.
6. For the *Function*, select the Lambda function you created from the dropdown list.
7. Click *Configure rule details*.
8. Name the rule as desired.
9. For *State*, select the *Enabled* checkbox.
10. Click *Create Rule*.

Testing the setup

When all services have been created and configured properly, execute this simple test to verify your work.

1. Create and run the test event from the Lambda function:
   a. From the *Test Event* dropdown list, select *Configure test events*.
   b. Select *Create new test event* to add a test event with the content as the code snippet below.

```
{
   "id": "fa9fa4a5-0232-188d-da1c-af410bcfc344",
   "detail": {
      "service": {
         "serviceName": "guardduty",
         "action": {
            "networkConnectionAction": {
               "connectionDirection": "INBOUND",
               "remoteIpDetails": {
                  "ipAddressV4": "192.168.123.123"
               }
            }
         },
         "additionalInfo": {
            "threatListName": "GeneratedFindingThreatListName">
         },
         "eventLastSeen": "2018-07-18T22:12:01.720Z"
      },
```

```
        "severity": 3
      }
    }
```

c. From the *Test Event* dropdown list again, select the event you have just created, then click *Test* to execute this Lambda function with the given event.

2. Verify the test result.
   a. If everything was set up correctly, you will see *Execution result: succeeded* on the top of the page of this Lambda function.
   b. Check and see a record with *finding_id - fa9fa4a5-0232-188d-da1c-af410bcfc344* and *ip - 192.168.123.123* is in the DynamoDB table - my-aws-lambda-guardduty-db.
   c. Check and see the file *ip_blocklist* resides in the S3 bucket my-aws-lambda-guardduty.
   d. Check that the ip_blocklist file has a *Read object* permission for *Everyone* under the *Public access* section.
   e. Check that the ip_blocklist is accessible through its link in browser (e.g. https://s3-us-east-1.amazonaws.com/***my-aws-lambda-guardduty***/ip_blocklist)
   f. Check that the ip_blocklist file contains *192.168.123.123* in a single line in its content.

(Optional) Generating sample findings in GuardDuty

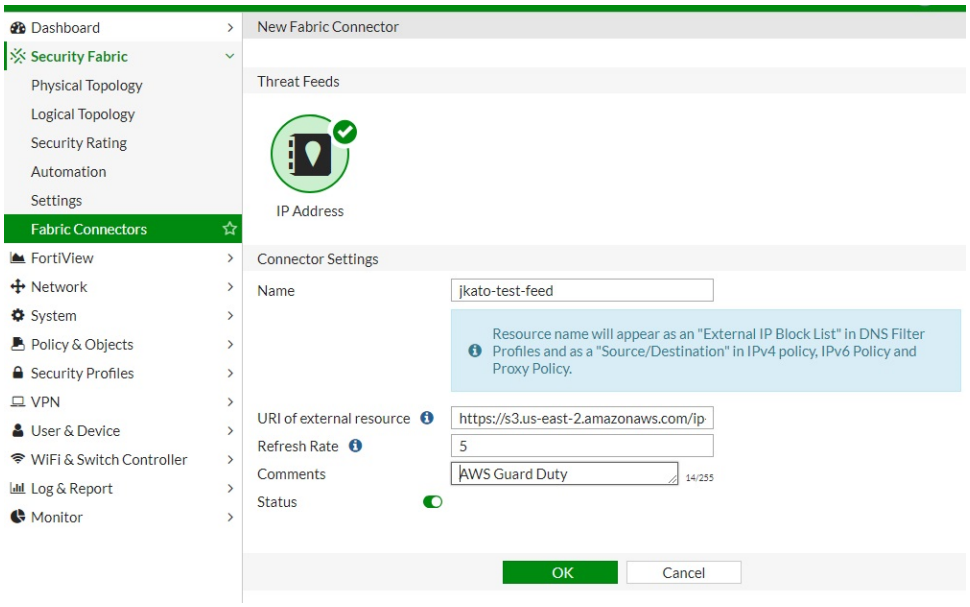
Amazon GuardDuty monitors your AWS infrastructures on a continuous basis to detect malicious or unauthorized behavior and creates records based on such findings. If you have just subscribed to GuardDuty for the first time, you will see no findings in the list. You can click *Generate sample findings* under *Settings* and get some samples. Then several dummy findings marked as “[SAMPLE]” are created. As long as you have set up the Lambda function and CloudWatch correctly, some of those sample findings trigger the CloudWatch event rule to run the Lambda function. A few new IP addresses eventually appear in the ip_blocklist.

Setting up the FortiGate(s)

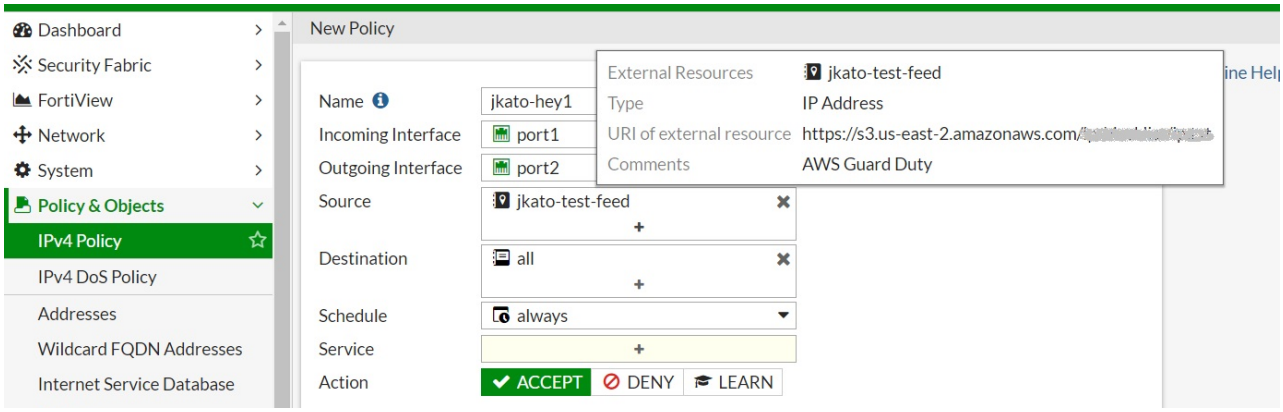
As a FortiGate-VM feature, GuardDuty integration introduces the ability to dynamically import external block lists from an HTTP server. You can use the block lists to enforce your organization's specialized security requirements. This can include long term policies, such as always blocking access to certain websites, or short term requirements to block access to known compromised locations. Since these lists are dynamically imported, the FortiGate-VM instantly imports any changes made to the list.

In this example, the FortiGate-VM integrates with AWS GuardDuty to populate a list, which is treated as a "threat feed". You can use a threat feed to deny access to a source or destination IP address in web filter and DNS filter profiles, SSL inspection exemptions, and as a source/destination in proxy policies. The block list is stored as an external resource, which is dynamically imported to the FortiGate-VM at a configured interval/refresh rate to maintain an updated list. The administrator can configure multiple threat feeds in each profile.

1. To configure a threat feed, go to *Security Fabric* > *Fabric Connectors*, then click *Create New*, then *IP Address* under *Threat Feeds*.
2. The following example creates an IP address connector. The resource name appears as an external IP block list in DNS filter profiles and as a source/destination in proxy policies. Configure the following:
   a. *URI of external resource*: link to an external resource file. The file should be a plain text file with one IP address on each line. In this example, the IP address is https://s3-us-east-1.amazonaws.com/***my-aws-lambda-guardduty***/ip_blocklist. The file size is up to 10 MB or 128000 lines of text, whichever is more restrictive.
   b. *Refresh Rate*: time interval to refresh the external resource. The rate can be between 1 to 43200 minutes.



3. Go to *Policy & Objects* > *IPv4 Policy* and create/edit a policy. In the *Source* and *Destination* fields, you should be able to add the new feed.



Cleanup

Since test events and sample findings can update the ip_blocklist with sample IP addresses, it is highly recommended to clean up the ip_blocklist for production use. This cleanup step removes the ip_blocklist from the S3 bucket and clears the DynamoDB table.

1. Delete all records from the DynamoDB table. In this example, the DynamoDB table is my-aws-lambda-guardduty-db.
2. Delete the ip_blocklist file in the my-aws-lambda-guardduty bucket.

Pipelined automation using AWS Lambda

With automation stitches, you can decrease response times to security events by automating activities between different device components in the Security Fabric. You can monitor events from any source in the Security Fabric and set up

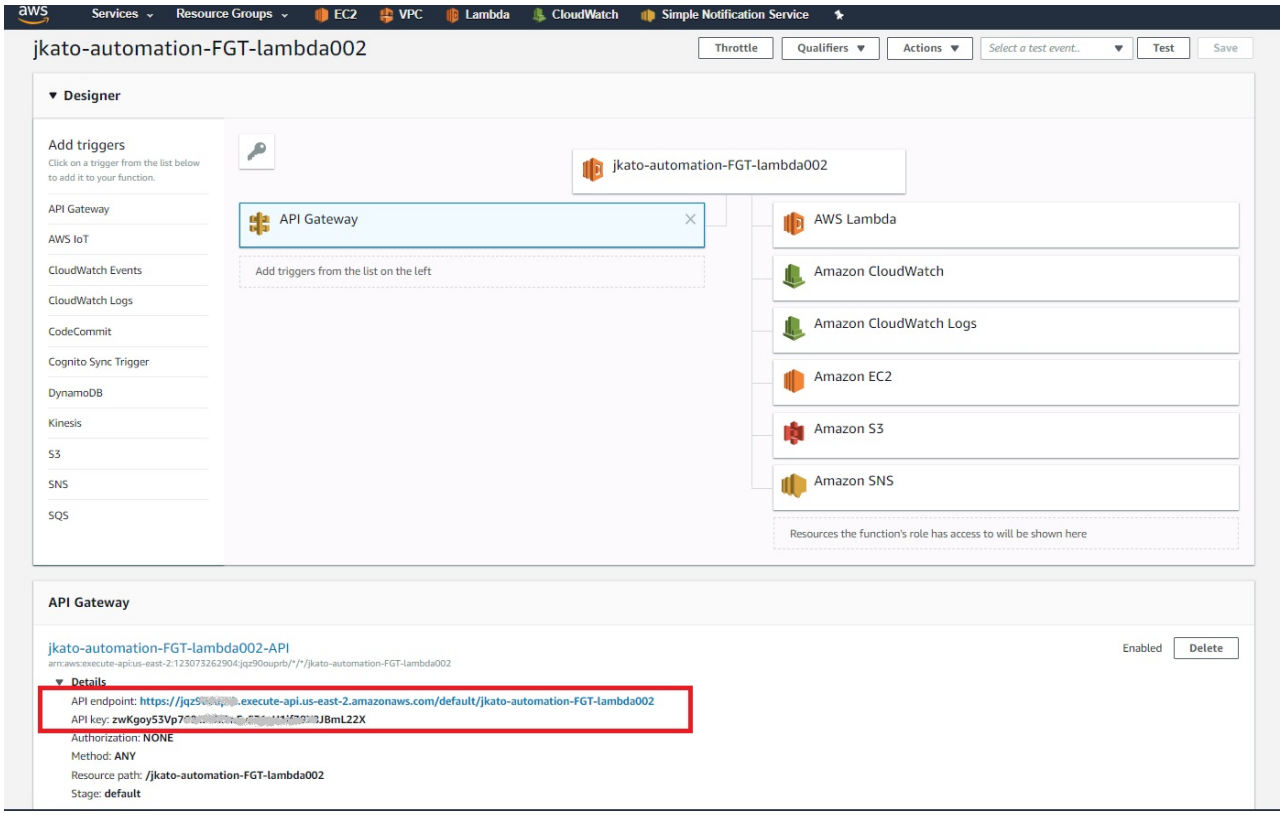
action responses to any destination.

FortiGate (both physical and virtual instances) supports AWS Lambda as an automated workflow.

- Creating an automation stitch on page 170
- Configuring an example automation stitch on page 171

Creating an automation stitch

1. In FortiOS, go to *Security Fabric* > *Automation*.
2. Under *Action*, select *AWS Lambda*.
3. In the *Name* field, enter the desired name for the stitch.
4. Under *AWS Lambda*, configure the following:
   a. In the *Name* field, enter the name of the action.
   b. For the *API Gateway* field, see the Lambda code configuration page, which shows the API gateway URL once added to the Lambda code.
   c. For the *API Key* field, see the Lambda code configuration page as above.



You must specify an AWS role that is sufficiently privileged to run the Lambda code and access CloudWatch/CloudWatch logs.

Configuring an example automation stitch

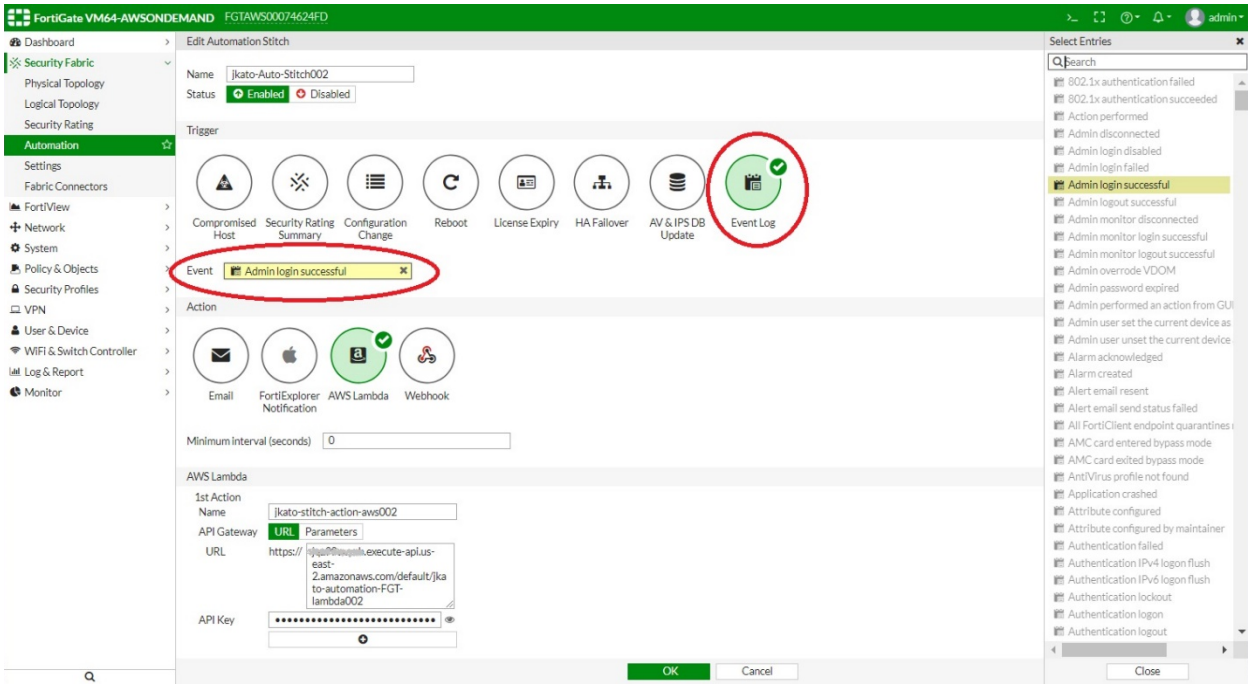
Let's try creating an example automation stitch with a simple pipeline. The example pipeline is as follows:

1. When an event log is created due to a successful login to the FortiGate,
2. Pick up one of the key-value pairs that the FortiGate sends to the API gateway
3. Invoke its AWS Lambda script, and, as an action, output the value on CloudWatch

Other actions you may want to configure include quarantining an EC2 instance by applying a different security group, renaming an EC2 tag, and so on. You can configure a variety of actions as fits your deployment scenario.

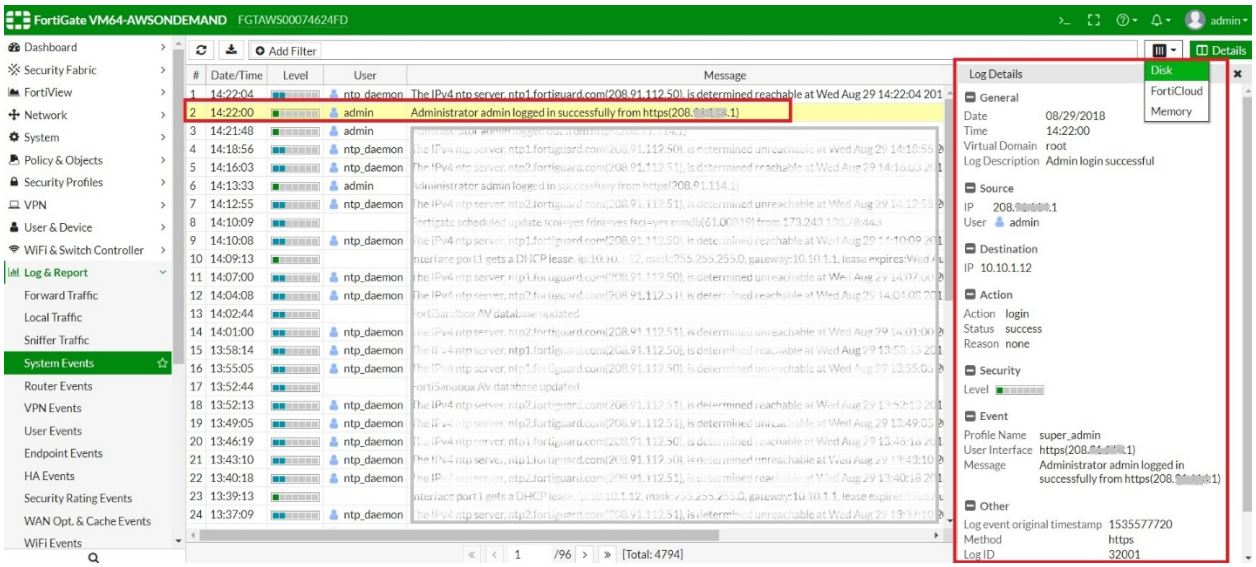
For this example, do the following:

1. Create an automation stitch by completing all steps in Creating an automation stitch.
2. Under *Trigger*, select *Event Log*.
3. In the *Event* dropdown list, select *Admin Login Successful*.



4. You will need to know what elements FortiGate sends with the event log and what to pick on the Lambda script. Now let's make the example event happen by logging into the FortiGate successfully as an admin user. Log out of the FortiGate, then log in again. You will see the corresponding event log.

5. Go to *Log & Report* > *System Events*. Find the desired event log.



6. Download the log as a file. You can filter logs as shown below.

7. Open the SystemEventLog-disk-<date/time/number>.log file in a text editor. It should look as below.

```
date=2018-08-29 time=15:56:13 logid="0100032001" type="event" subtype="system"
      level="information" vd="root" eventtime=1535583373 logdesc="Admin login successful"
      sn="15355xyz73" user="admin" ui="https(208.xx.yy.1)" method="https"
      srcip=208.xx.yy.1 dstip=192.168.1.15 action="login" status="success" reason="none"
      profile="super_admin" msg="Administrator admin logged in successfully from https
      (208.xx.yy.1)"
```

You have a rough idea about what elements can be picked. Raw JSON data will look as follows:

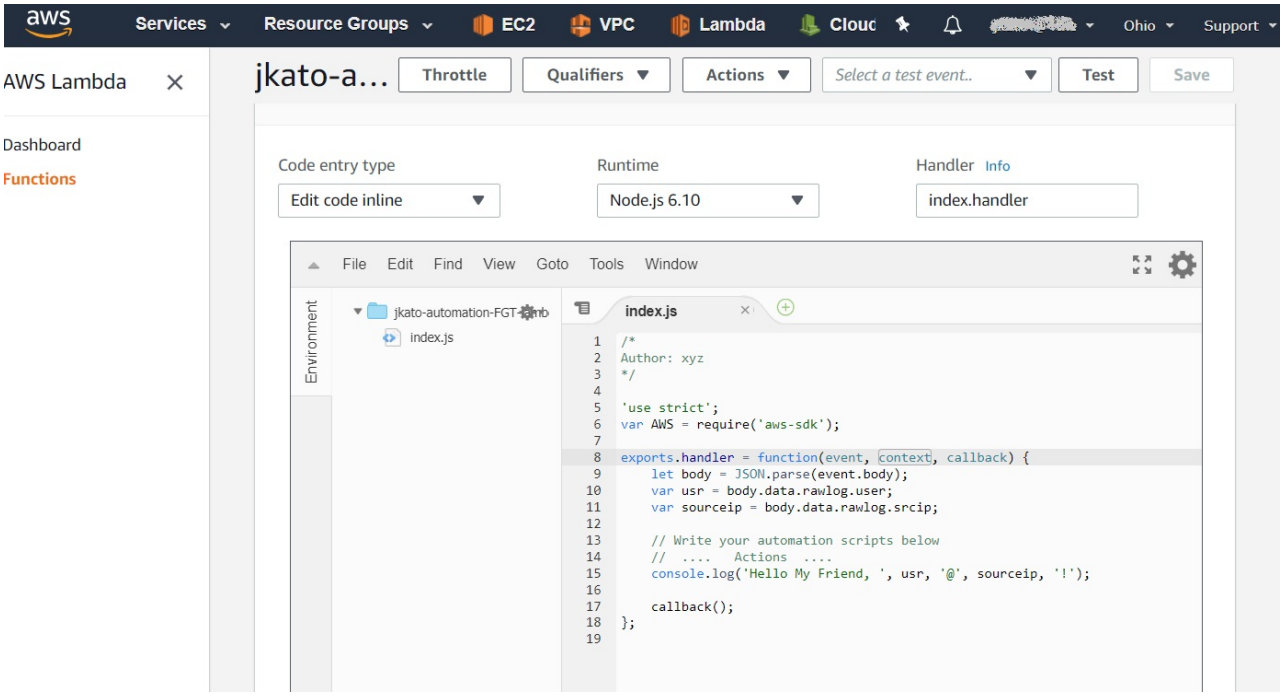
```
{ email: 'your_email@xyz.com',
   data:
      { stitch: 'Your Stitch Name',
      actions: [ [Object] ],
      eventtype: 'logid',
      sn: 'Serial Number of your FortiGate',
      time: 1535587464,
      rawlog:
         { date: '2018-08-29',
         time: '17:04:24',
         logid: '0100032001',
         type: 'event',
         subtype: 'system',
         level: 'information',
         vd: 'root',
         eventtime: '1535587464',
         logdesc: 'Admin login successful',
         sn: 'xyz',
         user: 'admin',
         ui: 'https(FortiGate IP address)',
         method: 'https',
         srcip: 'FortiGate IP address',
```

```
            dstip: '10.10.1.12',
            action: 'login',
            status: 'success',
            reason: 'none',
            profile: 'super_admin',
            msg: 'Administrator admin logged in successfully from https(FortiGate IP
                 address)'
        }
    }
}
```

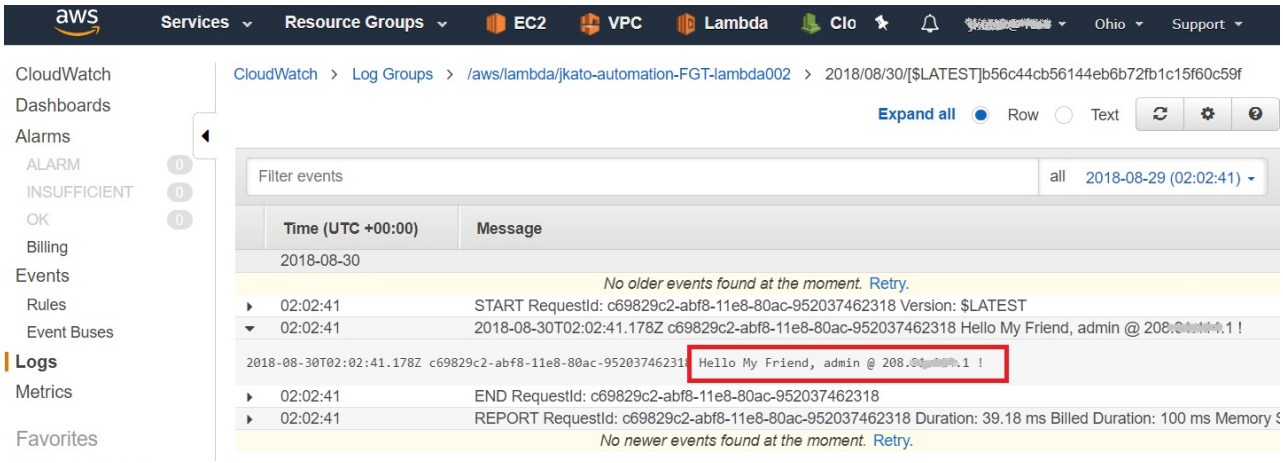
8. You can pick available key-value pairs in your AWS Lambda code. In this particular event log, useful keys include stitch / date /time / vd / logdesc / user / ui / method / srcip / dstip / action / status / profile / msg.
9. You can see all JSON logs sent by FortiGate on CloudWatch Log by entering the following line in the Lambda code: `console.log(JSON.parse(event.body));`
10. Now, as an example, let's pick `user: 'admin'` and `srcip: '208.xx.yy.1'`. Here is the Lambda script:

```
'use strict';
var AWS = require('aws-sdk');
exports.handler = function(event, context, callback) {
    let body = JSON.parse(event.body);
    var usr = body.data.rawlog.user;
    var sourceip = body.data.rawlog.srcip;
    // Write your automation scripts below
    // .... Actions ....
    console.log('Hello My Friend, ', usr, '@', sourceip, '!');
    callback();
};
```

This is what the Lambda script will look like:



11. Save the script.
12. Log out of the FortiGate, then log in again as an administrator. This triggers the event log. The Lambda code is invoked, and CloudWatch Log shows something like the following:



Configuring FortiGate-VM load balancer using dynamic address objects

FortiOS supports using dynamic firewall addresses in real servers under a virtual server load balancing configuration. Combined with support for the autoscaling group filter (see AWS SDN connector using certificates, this enables you to use the FortiGate as a load balancer in AWS for an autoscaling deployment. You do not need to manually change each server's IP address whenever a scale in/out action occurs, as FortiOS dynamically updates the IP addresses following each scale in/out action.

Consider a scenario where the FortiGate-VM is deployed on AWS and load balancing for three servers. The SDN connector configured in FortiOS dynamically loads the server IP addresses. If a scale in action occurs, the load balancer dynamically updates to load balance to the two remaining servers.

The following instructions assume the following:

1. An AWS SDN connector is configured and up.
2. An AWS dynamic firewall address with a filter is configured.

To configure a dynamic address object in a real server under virtual server load balance:

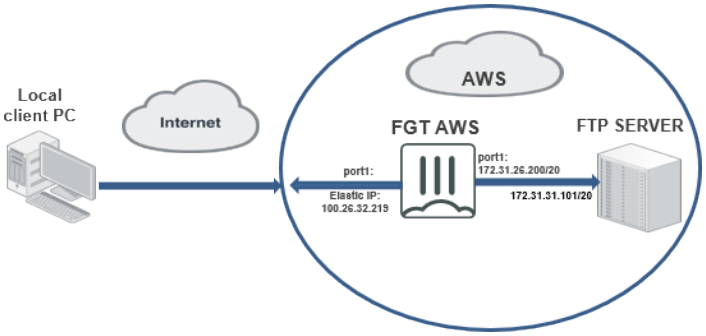
CLI commands introduced in FortiOS 6.4 are shown bolded below.

```
config firewall vip
   edit "0"
      set id 0
      set uuid 0949dfbe-7512-51ea-4671-d3a706b09657
      set comment ''
      set type server-load-balance
      set extip 0.0.0.0
      set extintf "port1"
      set arp-reply enable
      set server-type http
      set nat-source-vip disable
      set gratuitous-arp-interval 0
      set http-ip-header disable
      set color 0
      set ldb-method static
```

```
      set http-redirect disable
      set persistence none
      set extport 80
      config realservers
         edit 1
            set type address
            set address "aws addresses"
            set port 8080
            set status active
            set holddown-interval 300
            set healthcheck vip
            set max-connections 0
            unset client-ip
         next
      end
      set http-multiplex disable
      set max-embryonic-connections 1000
   next
end
```

Accessing a cloud server using an SDN connector via VPN

This guide provides a sample configuration that allows a local client PC to access an FTP server deployed inside the AWS cloud by using an AWS SDN connector via SSL VPN.



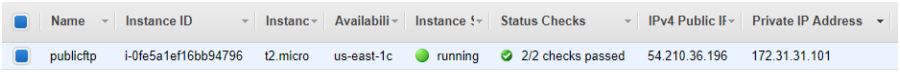
In this topology, a FortiGate-VM for AWS is deployed inside the AWS cloud. The FortiGate-VM can dynamically resolve the FTP server's private IP address in the AWS cloud through an AWS SDN connector. A local client PC with FortiClient installed can establish an SSL VPN tunnel to the FortiGate-VM inside the AWS cloud, then access the FTP server through the SSL VPN tunnel.

To configure the FortiGate-VM:

1. Configure the AWS SDN connector:
   a. In FortiOS, go to *Security Fabric* > *Fabric Connectors*.
   b. Click *Create New*.
   c. Select *Amazon Web Services (AWS)*.
   d. In the *AWS region name* field, enter *us-east-1*.
   e. Leave the *AWS VPC ID* field blank if no VPC ID is specified.
   f. Configure other fields as required. Click *OK*.

g. Go to *Security Fabric* > *Fabric Connectors*. Click the refresh icon for the configured connector. The green arrow means that the connector is connected.

2. Create an SDN connector firewall address to associate the configured SDN connector:
   a. Go to *Policy & Objects* > *Addresses*.
   b. Click *Create New*, then select *Address*.
   c. From the *Type* dropdown list, select *Fabric Connector Address*.
   d. From the *SDN Connector* dropdown list, select the connector created in step 1.
   e. For *SDN address type*, select *Private*.
   f. In the *Filter* field, enter *Tag.Name=publicftp*. This is the name of the FTP server in the AWS cloud.
   g. From the *Interface* dropdown list, select *any*.
   h. Click *OK*. The following shows the FTP server as seen in the AWS management console.



Name	Instance ID	Instanc	Availabili	Instance S	Status Checks	IPv4 Public IF	Private IP Address
publicftp	i-0fe5a1ef16bb94796	t2.micro	us-east-1c	running	2/2 checks passed	54.210.36.196	172.31.31.101

3. After the update interval (60 seconds by default), check the resolved firewall address:
   a. Go to *Policy & Objects* > *Addresses*.
   b. Hover over the address created in step 2. In this example, it shows the firewall address (172.31.31.101) that the configured SDN connector resolves to.

4. Configure SSL VPN to access the FTP server:
   a. Configure the user and user group:
      i. Go to *User & Device* > *User Definition*.
      ii. Create a new local user.
      iii. Go to *User & Device* > *User Groups*.
      iv. Create a group that includes the new local user.
   b. Configure SSL VPN settings:
      i. Go to *VPN* > *SSL-VPN Settings*.
      ii. In the *Listen on Interface* field, select the proper interface. This example selects port1.
      iii. In the *Listen on Port* field, enter 10443.
      iv. From the *Server Certificate* dropdown list, select the desired certificate.

Self-signed certificates are provided by default to simplify initial installation and testing. It is **HIGHLY** recommended that you acquire a signed certificate for your installation.

Continuing to use these certificates can result in your connection being compromised, allowing attackers to steal your information, such as credit card details.

For more information, please review Use a non-factory SSL certificate for the SSL VPN portal and learn how to Purchase and import a signed SSL certificate.

v. Under *Authentication/Port Mapping*, set the default full-access portal for *All Other Users/Groups*.
vi. Create a new authentication/portal mapping for the group created in step a, mapping to the full-access portal.

c. Configure the SSL VPN firewall policy:
i. Go to *Policy & Objects* > *IPv4 Policy*.
ii. From the *Incoming Interface* dropdown list, select the SSL VPN tunnel interface (ssl.root).
iii. From the *Outgoing Interface* dropdown list, select *port1*.
iv. In the *Source* field, select *all* and the group configured in step a.
v. In the *Destination* field, select the address created in step 2.
vi. From the *Schedule* dropdown list, select *always*.
vii. In the *Service* field, select *ALL*.
viii. For *Action*, select *Accept*.
ix. Click *OK*.

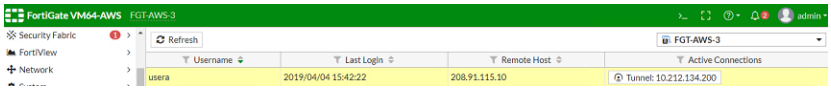
To establish an SSL VPN connection from the local client PC:

This example assumes that you are not using EMS to manage endpoints. If you are using EMS, use a licensed FortiClient endpoint for the following configuration, skipping the installation step.

1. Download VPN-only FortiClient from FortiClient.com. Install onto the local client PC.
2. In FortiClient, on the *Remote Access* tab, add a new connection.
3. For *VPN*, select *SSL-VPN*.
4. In the *Remote Gateway* field, enter the IP address of the listening FortiGate interface. In this example, it is 100.26.32.219, the FortiGate-VM port1 public IP address.
5. Select *Customize port*, then enter 10443.
6. Save the configuration.
7. Use the credentials configured in step 4a above to connect to the SSL VPN tunnel. After connection, traffic to the SDN connector resolved IP address (172.31.31.101) goes through the tunnel. Other traffic goes through the local gateway. The client PC side shows the routing entry for the SSL VPN tunnel:

```
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         172.16.200.1    0.0.0.0         UG    0      0        0 eth1
172.31.31.101   10.212.134.200  255.255.255.255 UGH   0      0        0 ppp0
```

The FortiGate-VM shows the logged in user and the assigned SSL VPN tunnel virtual IP address.




```
execute vpn sslvpn list
    SSL VPN Login Users:
     Index   User    Auth Type   Timeout        From      HTTP in/out    HTTPS
in/out
     0       usera   1(1)          284      208.91.115.10     0/0            0/0
    SSL VPN sessions:
     Index   User    Source IP     Duration   I/O Bytes    Tunnel/Dest IP
     0       usera   208.91.115.10   76      1883/1728     10.212.134.200
```

To run diagnose commands:

1. To show SDN connector status, run the `diagnose sys sdn status` command. The output should be as follows:

```
SDN Connector                       Type        Status
------------------------------------------------------------
aws1                                aws         connected
```

2. To debug the SDN connector to resolve the firewall address, run the `diagnose debug application awsd -1` command. The output should be as follows:

```
...
awsd checking firewall address object dynamic-aws, vd 0
 address change, new ip list:
  172.31.31.101
awsd sdn connector aws1 finish updating IP addresses
...
```

3. To restart the AWS SDN connector daemon, run the `diagnose test application awsd 99 command`.

VPN for FortiGate-VM on AWS

Connecting a local FortiGate to an AWS VPC VPN

This recipe provides sample configuration of a site-to-site VPN connection from a local FortiGate to an AWS VPC VPN via IPsec with static routing.

Instances that you launch into an Amazon VPC can communicate with your own remote network via a site-to-site VPN between your on-premise FortiGate and AWS VPC VPN. You can enable access to your remote network from your VPC by configuring a virtual private gateway (VPG) and customer gateway to the VPC, then configuring the site-to-site VPC VPN.

The following prerequisites must be met for this configuration:

- An AWS VPC with some configured subnets, routing tables, security group rules, and so on
- An on-premise FortiGate with an external IP address

This recipe consists of the following steps:

1. Create a VPG.
2. Create a customer gateway.
3. Create a site-to-site VPN connection on AWS.
4. Configure the on-premise FortiGate.

To create a VPG:

A VPG is the VPN concentrator on the Amazon side of the site-to-site VPN connection. You can create a VPG and attach it to the VPC from which you want to create the site-to-site VPN connection.

1. In the AWS management console, go to *Virtual Private Gateways*, then click *Create Virtual Private Gateway*.
2. In the *Name tag* field, enter the desired gateway name.
3. For static route configuration, the ASN is not important, as the ASN is for BGP routing. By default, the VPG is created with the default ASN, 64512. You cannot change the ASN once the VPG has been created.
4. After creating the VPG, select it from the list of VPGs, and click *Actions* > *Attach to VPC*.
5. On the *Attach to VPC* page, select the ID for the desired VPC from the *VPC* dropdown list.

To create a customer gateway:

In this example, the customer gateway refers to the on-premise FortiGate for the VPC VPN to connect to.

1. Go to *Customer Gateways*, then click *Create Customer Gateway*.
2. In the *Name* field, enter the desired gateway name.
3. For *Routing*, select *Static*.
4. In the *IP Address* field, enter the on-premise FortiGate's external address.

To create a site-to-site VPN connection on AWS:

AWS VPC VPN supports the following:

- Internet Key Exchange version 2 (IKEv2)
- NAT traversal
- Four-byte ASN (in addition to two-byte ASN)
- Reusable IP addresses for customer gateways
- Additional encryption options including AES 256-bit encryption, SHA-2 hashing, and additional Diffie-Hellman groups
- Configurable tunnel options
- Custom private ASN for the Amazon side of a BGP session

This example describes creating an IPsec site-to-site VPN.

1. Go to *VPN Connections*, then click *Create VPN Connection*.
2. In the *Name tag* field, enter the desired VPN connection name.
3. From the *Virtual Private Gateway* dropdown list, select the VPG ID for the VPG created earlier.
4. For *Routing Options*, select *Static*.
5. In the *IP Prefixes* field, enter the CIDR of the networks behind your on-premise FortiGate.
6. Leave the tunnel options blank. You will obtain this information from a configuration file download.

To configure the on-premise FortiGate:

1. After creating the VPN, select it in the VPN list, then click *Download Configuration*. This document contains information needed to configure the FortiGate correctly.
2. You can configure the FortiGate using this downloaded configuration file. The example FortiGate has port1 with an external IP address of 35.188.119.246 and an internal IP address of 10.6.30.2/24. Port2 has an internal IP address of 10.1.100.3/24. The downloaded configuration file resembles the following. The most important information here is the `remote-gw` value, which in this case is 3.95.86.157, and the `psksecret` value.

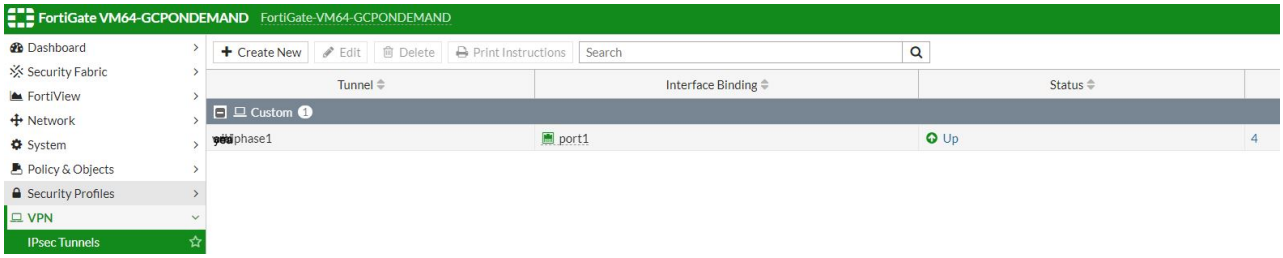
```
! ----------------------------------------------------------------------------
! IPSec Tunnel #1
! ----------------------------------------------------------------------------
! #1: Internet Key Exchange (IKE) Configuration
!
! A policy is established for the supported ISAKMP encryption,
! authentication, Diffie-Hellman, lifetime, and key parameters.
! Please note, these sample configurations are for the minimum requirement of AES128, SHA1, and DH Group 2.
! Category "VPN" connections in the GovCloud region have a minimum requirement of AES128, SHA2, and DH Group 14.
! You will need to modify these sample configuration files to take advantage of AES256, SHA256, or other DH groups like 2, 14-18, 22, 23, and 24.
! Higher parameters are only available for VPNs of category "VPN," and not for "VPN-Classic".
!
! The address of the external interface for your customer gateway must be a static address.
! Your customer gateway may reside behind a device performing network address translation (NAT).
! To ensure that NAT traversal (NAT-T) can function, you must adjust your firewall rules to unblock UDP port 4500. If not behind NAT, we recommend disabling NAT-T.
!
! Configuration begins in root VDOM.

config vpn ipsec phase1-interface
edit vpn-032037fa39969e15b-0 ! Name must be shorter than 15 chars, best if shorter than 12
  set interface "wan1"

! The IPSec Dead Peer Detection causes periodic messages to be
! sent to ensure a Security Association remains operational

  set dpd enable
  set local-gw 35.188.119.246
  set dhgrp 2
  set proposal aes128-sha1
  set keylife 28800
  set remote-gw 3.95.86.157
  set psksecret N1ITFTQJfiVuRWkQui_A5IjNT_41VTtP
  set dpd-retryinterval 10
 next
end
```

   Run the following commands in the FortiOS CLI to configure the FortiGate, using the `remote-gw` and `psksecret` values from the downloaded configuration file as shown below. When setting the destination for the static route, use the VPC's IPv4 CIDR:

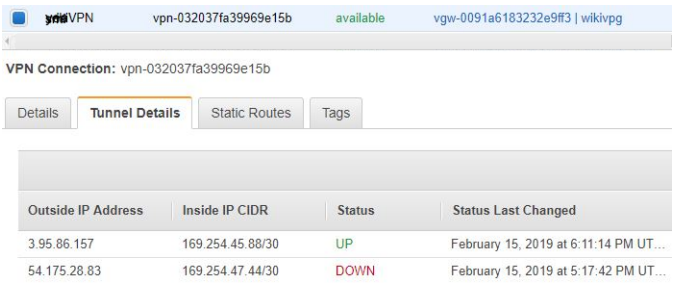
```
config vpn ipsec phase1-interface
   edit "examplephase1"
      set interface "port1"
      set keylife 28800
      set peertype any
```

```
      set proposal aes128-sha1
      set dhgrp 2
      set remote-gw 3.95.86.157
      set psksecret NlITFTQJfiVuRWkQui_A5IjNT_41VTtP
      set dpd-retryinterval 10
   next
end
config vpn ipsec phase2-interface
   edit "examplephase2"
      set phase1name "examplephase1"
      set proposal aes128-sha1
      set dhgrp 2
      set keylifeseconds 3600
   next
end
config router static
   edit 1
      set dst 10.0.0.0 255.255.0.0
      set device "examplephase1"
   next
end
config firewall policy
   edit 1
      set srcintf "examplephase1"
      set dstintf "port2"
      set srcaddr "all"
      set dstaddr "all"
      set action accept
      set schedule "always"
      set service "ALL"
   next
   edit 2
      set srcintf "port2"
      set dstintf "examplephase1"
      set srcaddr "all"
      set dstaddr "all"
      set action accept
      set schedule "always"
      set service "ALL"
   next
end
```

3. Run the `diagnose vpn tunnel up examplephase2` command if the tunnel is not up automatically already.
4. Check in the FortiOS GUI in *VPN* > *IPsec Tunnels* that the tunnel is up.



5. In the AWS management console, check that the tunnel is up:

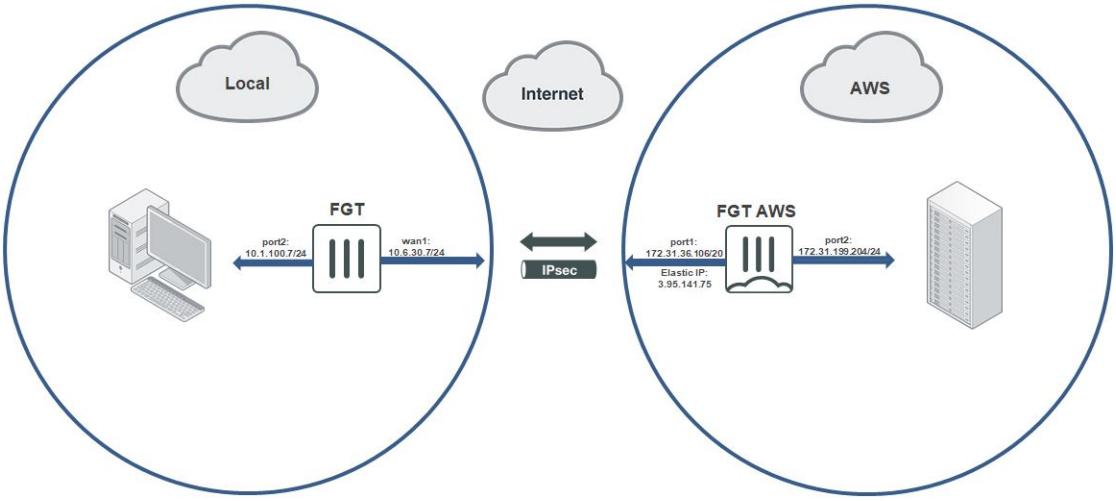


6. After the tunnel is up, you must edit a custom route table and security group rules to achieve connectivity between a resource behind the FortiGate to a resource on the AWS cloud.
7. On AWS, there are two tunnels for each created VPN. This example only shows connecting to one tunnel, but you can create the second tunnel in FortiOS as well. The second tunnel is for redundancy. If one tunnel goes down, the FortiGate can reach AWS resources using the other tunnel.

Connecting a local FortiGate to an AWS FortiGate via site-to-site VPN

This guide provides sample configuration of a site-to-site VPN connection from a local FortiGate to an AWS FortiGate via site-to-site IPsec VPN with static routing. You can access resources that are protected behind a FortiGate on AWS from your local environment by using a site-to-site VPN.

The following depicts the network topology for this sample deployment:



The following prerequisites must be met for this configuration:

- A FortiGate located on AWS with some resources behind it. In this example, the AWS FortiGate has port1 connected to WAN and port2 connected to local LAN.
- An on-premise FortiGate. For your local environment, determine if your FortiGate has a publicly accessible IP address or if it is behind NAT. In this example, the on-premise FortiGate is behind NAT.

This recipe consists of the following steps:

1. Create a VPN on the local FortiGate to the AWS FortiGate.

2. Create a VPN on the AWS FortiGate to the local FortiGate.
3. Establish a connection between the FortiGates.

To create a VPN on the local FortiGate to the AWS FortiGate:

1. In FortiOS on the local FortiGate, go to *VPN* > *IPsec Wizard*.
2. On the *VPN Setup* tab, configure the following:
   a. In the *Name* field, enter the desired name.
   b. For *Template Type*, select *Site to Site*.
   c. For *Remote Device Type*, select *FortiGate*.
   d. For *NAT Configuration*, select the appropriate option. In this example, since the local FortiGate is behind NAT, *This site is behind NAT* is selected. Click *Next*. For non-dialup situations where the local FortiGate has an external IP address, select *No NAT between sites*.
3. On the *Authentication* tab, configure the following:
   a. For *Remote Device*, select *IP Address*.
   b. In the *IP Address* field, enter the AWS FortiGate's elastic IP address. In this example, it is 3.95.141.75.
   c. For *Outgoing Interface*, allow FortiOS to detect the interface via routing lookup.
   d. For *Authentication Method*, select *Pre-shared Key*.
   e. In the *Pre-shared Key* field, enter the desired key. Click *Next*.
4. On the *Policy & Routing* tab, configure the following:
   a. For *Local Interface*, select the desired local interface. In this example, port2 is selected. The *Local Subnets* field should then auto-populate.
   b. In the *Remote Subnets* field, enter the remote subnet on the other side of the AWS FortiGate. In this example, it is 172.31.199.0/24.
   c. For *Internet Access*, select *None*.
5. Click *Create*. The IPsec Wizard creates the following:
   a. Firewall addresses for local and remote subnets
   b. Firewall address groups containing the above firewall addresses
   c. phase-1 and phase-2 interfaces
   d. Static route and blackhole route
   e. Two firewall policies: one for traffic to the tunnel interface and one for traffic from the tunnel interface

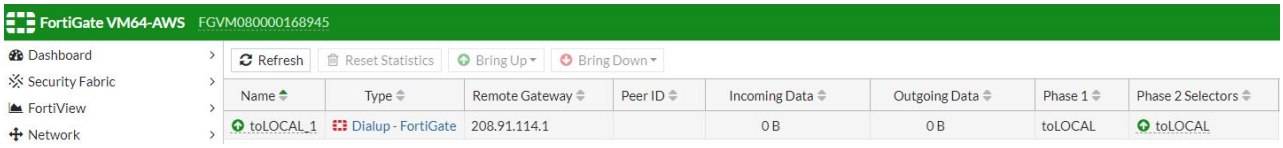
To create a VPN on the AWS FortiGate to the local FortiGate:

1. In FortiOS on the AWS FortiGate, go to *VPN* > *IPsec Wizard*.
2. On the *VPN Setup* tab, configure the following:
   a. In the *Name* field, enter the desired name.
   b. For *Template Type*, select *Site to Site*.
   c. For *Remote Device Type*, select *FortiGate*.
   d. For *NAT Configuration*, select *This site is behind NAT*. This is the correct configuration since the AWS FortiGate has an elastic IP address. Click *Next*.
3. On the *Authentication* tab, configure the following:
   a. For *Incoming Interface*, select the WAN-facing incoming interface. In this example, it is port1.
   b. For *Authentication Method*, select *Pre-shared Key*.
   c. In the *Pre-shared Key* field, enter the same key configured on the local FortiGate. Click *Next*.

4. On the *Policy & Routing* tab, configure the following:
   a. For *Local Interface*, select the desired local interface. In this example, port2 is selected. The *Local Subnets* field should then auto-populate.
   b. In the *Remote Subnets* field, enter the remote subnet on the other side of the local FortiGate. In this example, it is 10.1.100.0/24.
   c. For *Internet Access*, select *None*.
5. Click *Create*. The IPsec Wizard creates the following:
   a. Firewall addresses for local and remote subnets
   b. Firewall address groups containing the above firewall addresses
   c. phase-1 and phase-2 interfaces
   d. Static route and blackhole route
   e. Two firewall policies: one for traffic to the tunnel interface and one for traffic from the tunnel interface

To establish a connection between the FortiGates:

1. The tunnels are down until you initiate a connection from the local FortiGate to the AWS FortiGate. In FortiOS on the local FortiGate, go to *Monitor* > *IPsec Monitor*.
2. Right-click the phase-2 interface, and select *Bring Up*.
3. In FortiOS on the AWS FortiGate, go to *Monitor* > *IPsec Monitor* and verify that the connection is up.



The elastic IP address can be considered as one to one to the FortiGate's IP address, even though the port IP address may be an internal IP address.

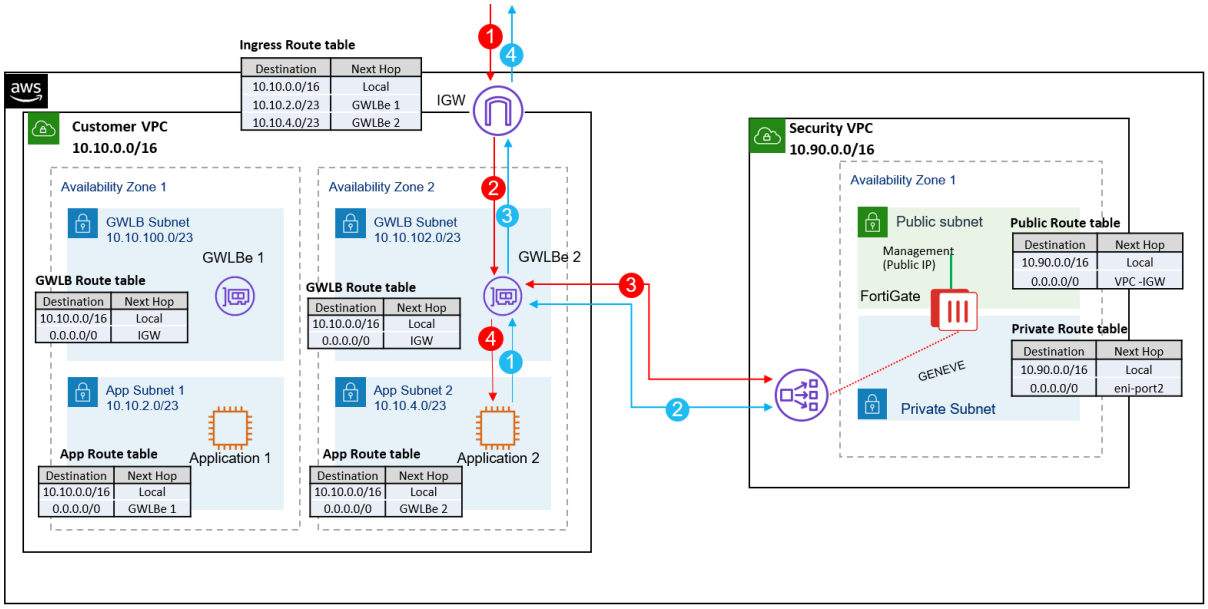
SD-WAN cloud on-ramp

See SD-WAN cloud on-ramp.

Security inspection with Gateway Load Balancer integration

The following deployment scenarios describe configuring security inspection with AWS Gateway Load Balancer (GWLB):

North-south security inspection to customer VPC



This guide assumes that the following are already created and in place as the diagram shows:

- Customer VPC
- Security VPC
- FortiGate with at least one management network interface and elastic IP address assigned
- Application instances

The guide describes configuring additional network interfaces to handle data traffic. The following describes the two VPCs in this deployment:

VPC	Description
Customer	Where customer workloads are deployed. The customer VPC has four subnets (two in each availability zone (AZ)). Each AZ has an application-purposed subnet and a GWLB endpoint subnet: • **Application-purposed subnet:** deploy application workloads where the

VPC	Description
	FortiGate must inspect the traffic. • **GWLB endpoint subnet:** deploy the GWLB endpoint so that traffic is redirected to the GWLB, which then redirects the traffic to the FortiGate for inspection.
Security	Where the FortiGate is deployed. You create the GWLB in this VPC.

The following describes the traffic flow in this deployment:

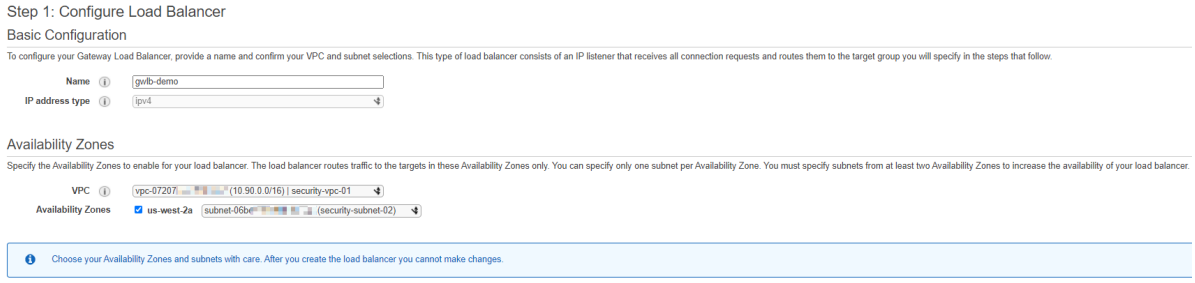
Traffic flow	Description
Inbound traffic	With this configuration, the FortiGate inspects traffic that is destined for the application instances. The Internet gateway in the customer VPC is associated with an ingress route table. The route table directs the traffic for the application subnets through the GWLB endpoints (GWLBe) in its dedicated subnets. The traffic then goes through the GWLB in the security VPC, where it is encapsulated with Geneve protocol and sent to the FortiGate. The FortiGate inspects the traffic and redirects it to the application instances.
Outbound traffic	The route tables that the application subnets are associated with have a default route through the GWLB endpoints in their AZ. The traffic originating from the application instances is forwarded to the FortiGate through the GWLB. After inspection, the FortiGate sends the traffic to the Internet. You set static routes for all of these traffic redirects after deployment. See Post-deployment configuration on page 189.

Creating the GWLB and registering targets

To create the GWLB and register targets:

1. Go to *Compute* > *EC2 Dashboard* > *Load Balancing* > *Load Balancers*.
2. Click *Create Load Balancer*, then *Gateway Load Balancer*.
3. Configure the GWLB:
   a. From the *IP address type* dropdown list, select *ipv4*.
   b. From the *VPC* dropdown list, select the security VPC, where the FortiGate is deployed.
   c. From the *Availability Zones* dropdown list, select the AZ and subnet where the FortiGate is deployed. This example selects the private subnet where the FortiGate port2 is mapped to. In this example, you can enable multiple VDOMs (only available on BYOL instances) or split-task VDOMs (available on BYOL and on-demand instances), and port2 is mapped to the traffic-handling VDOM. You then create the Geneve interface on port2

to handle the traffic that has been redirected via the GWLB. See Post-deployment configuration on page 189



4. Configure routing:
   a. From the *Target group* dropdown list, create a new target group with the desired name.
   b. For *Target type*, select *IP*.
   c. Ensure that *Protocol:Port* displays as *GENEVE: 6081*.
   d. From the *Protocol* dropdown list, select *HTTPS*.
   e. From the *Port* dropdown list, select the desired port. This example uses port 443. Ensure that your security group configuration allows traffic on that port.
5. Register the targets:
   a. In the *IP* field, enter the FortiGate IP address. In this example, you would enter the FortiGate port2 IP address.
   b. Click *Add to list*, then *Next*.
   c. Click *Review and Create*.
6. Ensure that cross-zone LB is enabled:
   a. Go to *Compute* > *EC2 Dashboard* > *Load Balancing* > *Load Balancers*.
   b. Select the newly created LB.
   c. On the *Description* tab, ensure that cross-zone LB is enabled.

Creating the LB endpoint

The LB endpoint is a listener that forwards traffic from the customer VPC to the GWLB and subsequently to the target group that you created in Creating the GWLB and registering targets on page 187. Before you create the LB endpoint, you must deploy an endpoint service in the region where your endpoint will be.

To create an endpoint service:

1. Go to *VPC Dashboard* > *Virtual Private Cloud* > *Endpoint services*.
2. Click *Create Endpoint Service*.
3. For *Associate Load Balancers*, select the GWLB that you created in Creating the GWLB and registering targets on page 187.
4. Enable endpoint acceptance if desired. This example does not require it.
5. Click *Create service*.

To create the LB endpoint:

1. Go to *VPC Dashboard* > *Virtual Private Cloud* > *Endpoint Services*.
2. Select the newly created endpoint service.
3. Copy the service name of the service on the *Details* tab.
4. Go to *VPC Dashboard* > *Virtual Private Cloud* > *Endpoints*.

5. Create the endpoint:
   a. Click *Create Endpoint*.
   b. For *Service category*, select *Find service by name*.
   c. In the *Service Name* field, paste the service name that you copied in step 1.
   d. Click *Verify*.
   e. From the *VPC* dropdown list, select the VPC where you need to deploy the endpoint.
   f. From the *Subnets* dropdown list, select the subnet where you need to deploy the endpoint.
   g. Click *Create Endpoint*.

VPC route tables

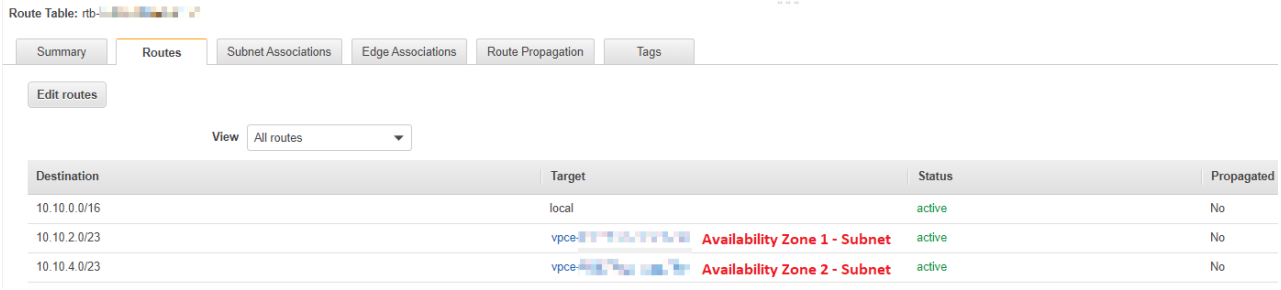
This example has two VPCs and multiple subnets within each VPC:

- **Customer VPC (10.10.0.0/16):** place protected resources whose traffic must be analyzed.
- **Security VPC (10.90.0.0/16):** place FortiGates here.

Application subnets are placed in different AZs.

Configure the ingress route table as follows:

- Subnet 1 (10.10.2.0/23) is mapped to the GWLB endpoint placed in AZ 1 subnet.
- Subnet 2 (10.10.4.0/23) is mapped to the GWLB endpoint placed in AZ 2 subnet.



- The Internet gateway is assigned on the route table *Edge Associations* tab. This allows traffic to flow into the VPC and then be redirected into their respective subnets via the routes that you created above.

Post-deployment configuration

You must create a Geneve interface on the FortiGate to handle traffic between the FortiGate and GWLB.

To create the Geneve interface:

1. Go to *EC2 Dashboard* > *Network & Security* > *Network Interfaces*. Copy the *Primary private IPv4 address* value for the GWLB interface created in the security VPC.
2. This example creates separate VDOMs via the split VDOM feature to handle traffic from the application VPC. Enable probe response on port 2 on both FortiGate instances. This allows LB health check to function:

```
config system global
   config system interface
      edit "port2"
         set vdom "FG-traffic"
         set alias private
         set mode dhcp
```

```
            set allowaccess ping https ssh fgfm probe-response
            set defaultgw disable
        next
    end
end
```

3. Create Geneve interfaces:

```
config vdom
   edit "FG-traffic"
      config system geneve
         edit "awsgeneve"
            set interface "port2"
            set type ppp
            set remote-ip <GWLB_interface_ip (from step 1)>
         next
      end
   next
end
```

4. Setting a higher priority on static routes for Geneve interfaces is recommended to avoid unintended functionality:

```
config router static
   edit 2
      set priority 100
      set device "awsgeneve"
   next
end
```

5. In a scenario where the load balancer is in a different subnet than the FortiGate interface, configure the following static route to avoid health check failures:

```
config router static
   edit 3
      set device port2
      set dst <loadbal_subnet>
      set gateway <local_gateway>
   next
end
```

To configure egress routes:

If the current VDOM has multiple interfaces, you must add egress routes to ensure that traffic entering through the Geneve interfaces egress through the same interface.

```
config router policy
   edit 1
      set input-device "awsgeneve"
      set src "0.0.0.0/0.0.0.0"
      set dst "10.10.2.0/255.255.254.0"
      set output-device "awsgeneve"
   next
end
```

Validating the configuration

Since traffic between the Internet and the application EC2 instance flows through the FortiGate Geneve interface, this example creates a FortiOS firewall policy that allows communication from the Geneve interface to the Geneve interface. The following shows an example policy.

This policy facilitates easy debugging. You should not configure this policy in a production environment.

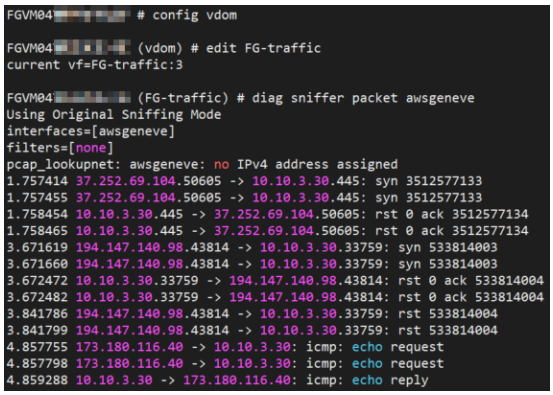
To configure the policy:

```
config firewall policy
   edit 1
      set name "test_policy"
      set srcintf "az2"
      set dstintf "az2"
      set srcaddr "all"
      set dstaddr "all"
      set action accept
      set schedule "always"
      set service "ALL"
   next
end
```

To run a packet sniffer on the Geneve interface created to handle GWLB traffic:

In this example, the VDOM name is FG-traffic. When multiple VDOM mode (available only on BYOL instances) is enabled, substitute the name of your VDOM here for FG-traffic.

1. Run a packet sniffer:
   ```
   config vdom
      edit FG-traffic
         diagnose sniffer packet awsgeneve
   ```
2. While the packet capture is running, attempt to access/ping a resource in the application subnet (protected subnet). The ping should succeed. The following shows the FortiGate packet capture for this access attempt:



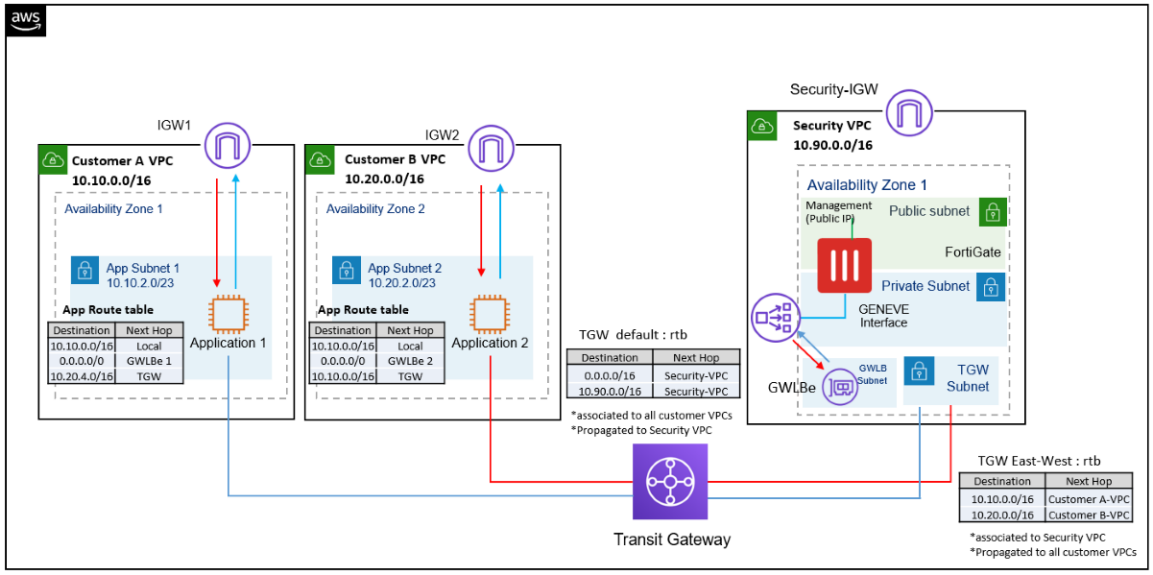
3. While the packet capture is running, attempt to access/ping an Internet resource from the protected resource. The

ping should succeed. The following shows the FortiGate packet capture for this access attempt:

```
FGVM04[redacted] (vdom) # edit FG-traffic
current vf=FG-traffic:3

FGVM04[redacted] (FG-traffic) # diag sniffer packet awsgeneve
Using Original Sniffing Mode
interfaces=[awsgeneve]
filters=[none]
pcap_lookupnet: awsgeneve: no IPv4 address assigned
6.246479 173.180.116.40.56015 -> 10.10.3.30.22: psh 3989997602 ack 3873790126
6.246504 173.180.116.40.56015 -> 10.10.3.30.22: psh 3989997602 ack 3873790126
6.247445 10.10.3.30.22 -> 173.180.116.40.56015: ack 3989997650
6.247458 10.10.3.30.22 -> 173.180.116.40.56015: ack 3989997650
6.247463 10.10.3.30.22 -> 173.180.116.40.56015: psh 3873790126 ack 3989997650
6.247465 10.10.3.30.22 -> 173.180.116.40.56015: psh 3873790126 ack 3989997650
6.315274 173.180.116.40.56015 -> 10.10.3.30.22: ack 3873790174
6.315303 173.180.116.40.56015 -> 10.10.3.30.22: ack 3873790174
6.372304 173.180.116.40.56015 -> 10.10.3.30.22: psh 3989997650 ack 3873790174
6.372314 173.180.116.40.56015 -> 10.10.3.30.22: psh 3989997650 ack 3873790174
```

East-west security inspection between two customer VPCs

The following shows the topology for this deployment, which uses GWLB for east-west security inspection between two customer VPCs:



This guide assumes that the following are already created and in place as the diagram shows:

- Customer A and B VPCs
- Security VPC
- FortiGate with at least one management network interface and elastic IP address assigned
- Application instances

The guide describes configuring additional network interfaces to handle data traffic. The following describes the two VPC types in this deployment:

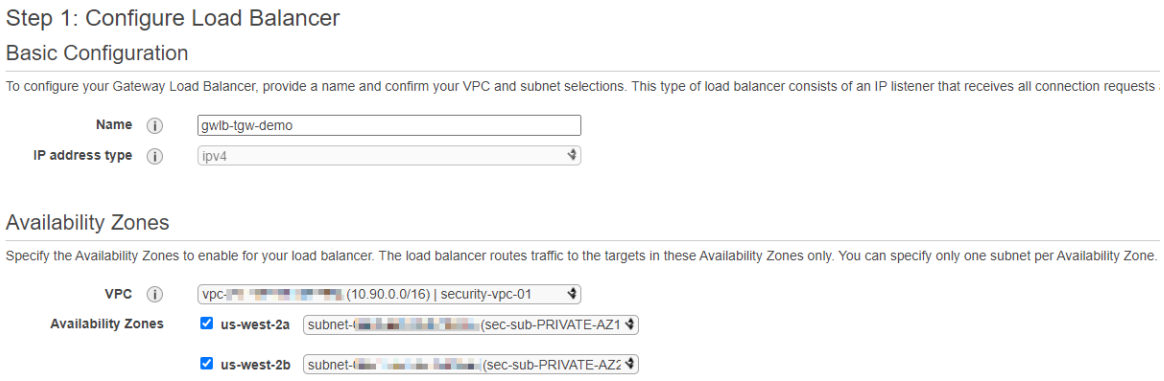
VPC	Description
Customer	Where customer workloads are deployed. The customer VPCs each have one AZ with an application-purposed subnet where you deploy application workloads where the FortiGate must inspect the traffic.
Security	Where the FortiGate is deployed. You create the GWLB in this VPC. The security VPC AZ also includes the following subnets: • **GWLB endpoint subnet**: deploy the GWLB endpoint so that traffic is redirected to the GWLB, which then redirects the traffic to the FortiGate for inspection. • **TGW subnet**: deploy the transit gateway (TGW) and associated resources, which allows connection of the customer VPCs to the security VPC.

Creating the GWLB and registering targets

For this deployment, you create the GWLB in the security subnet.

To create the GWLB and register targets:

1. Go to *Compute* > *EC2 Dashboard* > *Load Balancing* > *Load Balancers*.
2. Click *Create Load Balancer*, then *Gateway Load Balancer*.
3. Configure the GWLB:
   a. From the *IP address type* dropdown list, select *ipv4*.
   b. From the *VPC* dropdown list, select the security VPC, where the FortiGate is deployed.
   c. From the *Availability Zones* dropdown list, select the AZ and subnet where the FortiGate is deployed. This example selects the private subnets for the respective AZs where the FortiGate port2 is mapped to. In this example, you can enable multiple VDOMs (only available on BYOL instances) or split-task VDOMs (available on BYOL and on-demand instances), and port2 is mapped to the traffic-handling VDOM. You then create the Geneve interface on port2 to handle the traffic that has been redirected via the GWLB. See Post-deployment configuration on page 196.



4. Configure routing:
   a. From the *Target group* dropdown list, create a new target group with the desired name.
   b. For *Target type*, select *IP*.
   c. Ensure that *Protocol:Port* displays as *GENEVE: 6081*.
   d. From the *Protocol* dropdown list, select *HTTPS*.

   e. From the *Port* dropdown list, select the desired port. This example uses port 443. Ensure that your security group configuration allows traffic on that port.
5. Register the targets:
   a. In the *IP* field, enter the FortiGate IP address. In this example, you would enter the FortiGate port2 IP address.
   b. Click *Add to list*, then *Next*.
   c. Click *Review and Create*.
6. Ensure that cross-zone LB is enabled:
   a. Go to *Compute* > *EC2 Dashboard* > *Load Balancing* > *Load Balancers*.
   b. Select the newly created LB.
   c. On the *Description* tab, ensure that cross-zone LB is enabled.

Creating the LB endpoint

The LB endpoint is a listener that forwards traffic from the customer VPC to the GWLB and subsequently to the target group that you created in Creating the GWLB and registering targets on page 193. You must create an endpoint for each AZ. Before you create the LB endpoint, you must deploy an endpoint service in the region where your endpoint will be.

To create an endpoint service:

1. Go to *VPC Dashboard* > *Virtual Private Cloud* > *Endpoint services*.
2. Click *Create Endpoint Service*.
3. For *Associate Load Balancers*, select the GWLB that you created in Creating the GWLB and registering targets on page 193.
4. Enable endpoint acceptance if desired. This example does not require it.
5. Click *Create service*.

To create the LB endpoint:

1. Go to *VPC Dashboard* > *Virtual Private Cloud* > *Endpoint Services*.
2. Select the newly created endpoint service.
3. Copy the service name of the service on the *Details* tab.
4. Create the endpoint for the first AZ:
   a. Go to *VPC Dashboard* > *Virtual Private Cloud* > *Endpoints*.
   b. Click *Create Endpoint*.
   c. For *Service category*, select *Find service by name*.
   d. In the *Service Name* field, paste the service name that you copied in step 1.
   e. Click *Verify*.
   f. From the *VPC* dropdown list, select the VPC where you need to deploy the endpoint.
   g. From the *Subnets* dropdown list, select the subnet where you need to deploy the endpoint. This example selects the GWLB endpoint subnet created in each AZ in the security subnet.
   h. Click *Create Endpoint*.
5. Repeat the process to create the endpoint for the second AZ.

Creating the transit gateway

A transit gateway (TGW) is a transit hub used to connect two VPCs or a VPC to an on-premise network. This example connects the application VPC to the security VPC via a TGW. This ensures that any access to and from the application VPC is routed via the security VPC, where the FortiGates can inspect it.

To create the TGW:

1. Go to *VPC Dashboard* > *Transit Gateways* > *Transit Gateways*.
2. Click *Create Transit Gateway*.
3. Configure the TGW as needed. Creating a TGW creates a TGW default route table. The table is used as the default association and propagation route table for this gateway. You can access this table in *VPC Dashboard* > *Transit Gateways* > *Transit Gateway Route Tables*.

To create the TGW attachment:

You can create a gateway attachment to link separate VPCs and subnets to the newly created TGW. The two resources can be in the same or different AWS accounts. This example assumes that both VPCs are in the same AWS account.

1. Go to *VPC Dashboard* > *Transit Gateways* > *Transit Gateway Attachments*.
2. Click *Create Transit Gateway Attachment*.
3. From the *Transit Gateway ID* dropdown list, select the TGW that you created.
4. From the *Attachment type* dropdown list, select *VPC*.
5. From the *VPC ID* dropdown list, select the VPC that you want to attach to the TGW.
6. Under *Subnet IDs*, select the required subnet in the desired AZ.
7. Configure other fields as desired.
8. Click *Create Attachment*.
9. Repeat the process for the remaining two VPC attachments. The security VPC is attached to the TGW, with only the TGW subnets in each AZ selected. This ensures that traffic can be routed seamlessly to and from the GWLB endpoint. You must attach each subnet/AZ to the TGW separately.

Route tables

This example has three VPCs and multiple subnets within each VPC:

- **Customer A VPC (10.10.0.0/16) and Customer B VPC (10.20.0.0/16):** place protected resources whose traffic must be analyzed here.
- **Security VPC (10.90.0.0/16):** place FortiGates here.

Application subnets are placed in different AZs.

East-west egress route table

Ensure that the east-west egress route table is configured as follows:

- Link the customer application subnets on VPC A and B to the security VPC via the TGW.
- Route traffic from customer A VPC so that it must go through the security VPC to reach the customer B VPC.

The following shows the customer A VPC route table. You must configure the customer B VPC route table similarly:

Destination	Target	Status	Propagated
10.10.0.0/16	local	active	No
0.0.0.0/0	igw- Internet Access via IGW	active	No
10.20.0.0/16	tgw- CustomerB VPC access via Transit Gateway	active	No

Ensure that the TGW is attached to the designated TGW subnet in each AZ of the security VPC. Designated TGW VPCs allow you to configure forward and reverse routes to and from the FortiGate without causing routing loops. The following shows the TGW subnet route table for the forward route:

Destination	Target	Status	Propagated
10.90.0.0/16	local	active	No
10.10.0.0/16 Customer A VPC	vpce- GWLB endpoint - Security VPC	active	No
10.20.0.0/16 Customer B VPC	vpce-	active	No

An ideal configuration would have multiple GWLB endpoints in each AZ and selectively route traffic for high availability. Due to the way routes are configured on AWS, you can configure a single GWLB endpoint for multiple FortiGates.

The following shows the GWLB endpoint subnet route table for the reverse route:

Destination	Target	Status	Propagated
10.90.0.0/16	local	active	No
10.10.0.0/16 Customer A VPC	tgw- Transit Gateway	active	No
10.20.0.0/16 Customer B VPC	tgw-	active	No

Configuring TGW route tables

Since traffic from customer VPC A and customer VPC B must be routed via the security subnet and cannot be forward directly, you must configure the following on the TGW route table for east-west traffic.

To configure TGW route tables:

1. Go to *VPC Dashboard* > *Transit Gateways* > *Transit Gateway Route Tables*.
2. Delete the automatically generated route table and its associations. You will create two new TGW route tables.
3. Create the TGW default route table:
   a. On the *Associations* tab, associate the route table with Customer A and Customer B VPCs.
   b. On the *Propagations* tab, propagate the route table to the security VPC.
   c. On the *Routes* tab, add a default route to send all traffic to the security VPC.
4. Create the east-west route table:
   a. On the *Associations* tab, associate the route table with the security VPC.
   b. On the *Propagations* tab, propagate the route table to Customer A and Customer B VPCs.
   c. On the *Routes* tab, define customer A and B VPC routes.

Post-deployment configuration

You must create a Geneve interface on the FortiGate to handle traffic between the FortiGate and GWLB.

To create the Geneve interface:

1. Go to *EC2 Dashboard* > *Network & Security* > *Network Interfaces*. Copy the *Primary private IPv4 address* value for the GWLB interface created in the security VPC.
2. This example creates separate VDOMs via the split VDOM feature to handle traffic from the application VPC. Enable probe response on port 2 on both FortiGate instances. This allows LB health check to function:
```
config system global
   config system interface
      edit "port2"
         set vdom "FG-traffic"
         set alias private
         set mode dhcp
         set allowaccess ping https ssh fgfm probe-response
         set defaultgw disable
      next
   end
end
```
3. Create Geneve interfaces:
```
config vdom
   edit "FG-traffic"
      config system geneve
         edit "awsgeneve"
            set interface "port2"
            set type ppp
            set remote-ip <GWLB_interface_ip (from step 1)>
         next
      end
   next
end
```
4. Setting a higher priority on static routes for Geneve interfaces is recommended to avoid unintended functionality:
```
config router static
   edit 2
      set priority 100
      set device "awsgeneve"
   next
end
```
5. In a scenario where the load balancer is in a different subnet than the FortiGate interface, configure the following static route to avoid health check failures:
```
config router static
   edit 3
      set device port2
      set dst <loadbal_subnet>
      set gateway <local_gateway>
   next
end
```

To configure egress routes:

If the current VDOM has multiple interfaces, you must add egress routes to ensure that traffic entering through the Geneve interfaces egress through the same interface.

```
config router policy
   edit 1
      set input-device "awsgeneve"
      set src "0.0.0.0/0.0.0.0"
```

```
      set dst "10.10.2.0/255.255.254.0"
      set output-device "awsgeneve"
   next
end
```

Validating the configuration

Since traffic between the Internet and the application EC2 instance flows through the FortiGate Geneve interface, this example creates a FortiOS firewall policy that allows communication from the Geneve interface to the Geneve interface. The following shows an example policy.

This policy facilitates easy debugging. You should not configure this policy in a production environment.

To configure the policy:

```
config firewall policy
   edit 1
      set name "test_policy"
      set srcintf "az2"
      set dstintf "az2"
      set srcaddr "all"
      set dstaddr "all"
      set action accept
      set schedule "always"
      set service "ALL"
   next
end
```

To run a packet sniffer on the Geneve interface created to handle GWLB traffic:

In this example, the VDOM name is FG-traffic. When multiple VDOM mode (available only on BYOL instances) is enabled, substitute the name of your VDOM here for FG-traffic.

1. Run a packet sniffer:
   ```
   config vdom
      edit FG-traffic
         diagnose sniffer packet awsgeneve
   ```
2. While the packet capture is running, attempt to access/ping a resource in customer B VPC from a resource in customer A VPC. The ping should succeed. The following shows the FortiGate packet capture for this access attempt:
   ```
   FGTAWS [redacted] (FG-traffic) # diag sniffer packet awsgeneve
   Using Original Sniffing Mode
   interfaces=[awsgeneve]
   filters=[none]
   pcap_lookupnet: awsgeneve: no IPv4 address assigned
   9.216818 10.10.2.114 -> 10.20.2.147: icmp: echo request
   9.216871 10.10.2.114 -> 10.20.2.147: icmp: echo request
   9.218211 10.20.2.147 -> 10.10.2.114: icmp: echo reply
   9.218220 10.20.2.147 -> 10.10.2.114: icmp: echo reply
   10.217660 10.10.2.114 -> 10.20.2.147: icmp: echo request
   10.217672 10.10.2.114 -> 10.20.2.147: icmp: echo request
   ```

Change log

Date	Change Description
2020-03-31	Initial release.
2020-04-03	Updated Creating an address using the GUI and Creating an address using the CLI.
2020-04-06	Added Configuring FortiGate-VM load balancer using dynamic address objects on page 174.
2020-05-05	Updated Creating a support account on page 12.
2020-05-13	Added Migrating a FortiGate-VM instance between license types on page 14. Updated Order types on page 11 and Creating a support account on page 12.
2020-05-15	Updated Order types on page 11.
2020-05-21	Updated Launching the instance with shared FortiGate-VM AMI on page 32.
2020-06-18	Added SD-WAN cloud on-ramp on page 185. Updated Deploying FortiGate-VM active-passive HA AWS between multiple zones manually with Transit Gateway integration on page 139 subtopics.
2020-07-02	Updated Deploying auto scaling on AWS on page 39.
2020-07-09	Added To configure a VDOM exception: on page 138.
2020-08-04	Added Deploying FortiGate-VM using Terraform on page 155.
2020-09-21	Updated Deploying auto scaling on AWS on page 39. Added Hybrid licensing support for deployments with Transit Gateway integration.
2020-10-09	Updated diagrams in Deploying auto scaling on AWS on page 39.
2020-10-30	Updated To configure the FortiGate-VM: on page 175.
2020-12-04	Updated Deploying auto scaling on AWS on page 39. Added support for FortiOS 6.4.3.
2020-12-08	Updated Order types on page 11.
2020-12-11	Updated Creating a support account on page 12.
2021-02-04	Updated Deploying auto scaling on AWS on page 39. Added support for FortiOS 6.4.4. FortiAnalyzer can now be integrated into the deployment.
2021-02-17	Updated Deploying auto scaling on AWS on page 39. For details, refer to the FortiGate Autoscale for AWS document history on page 95.
2021-02-18	Updated SDN connector integration with AWS on page 156.
2021-02-22	Added Security inspection with Gateway Load Balancer integration on page 186.
2021-03-18	Updated Post-deployment configuration on page 189 and Post-deployment configuration on page 196.

Date	Change Description
2021-07-20	Updated Deploying auto scaling on AWS on page 39. Added Requirements when using an existing VPC on page 42
2021-07-26	Updated Deploying auto scaling on AWS on page 39. Added support for FortiOS 6.2.5, FortiOS 6.4.6, FortiOS 7.0.0, and FortiOS 7.0.1. Removed support for FortiOS 6.2.3 and FortiOS 6.4.4. Added support for FortiAnalyzer 6.4.6. Removed support for FortiAnalyzer 6.2.5 and FortiAnalyzer 6.4.4.
2021-09-08	Updated the CFT parameters on page 50
2022-03-23	Updated: • Creating VPCs and subnets on page 140 • Creating a Transit Gateway and related resources on page 141 • Creating VPC route tables on page 147 • Configuring the FortiGate-VMs on page 151 • Updating the route table and adding an IAM policy on page 152
2022-04-08	Added Upgrading the deployment on page 91.