

Deploy the ASA Virtual on AWS

You can deploy the ASA Virtual on the Amazon Web Services (AWS) cloud.

Important Beginning with 9.13(1), any ASA Virtual license now can be used on any supported ASA Virtual vCPU/memory configuration. This allows the ASA Virtual customers to run on a wide variety of VM resource footprints. This also increases the number of supported AWS instances types.

- Overview, on page 1
- Prerequisites, on page 7
- Guidelines and Limitations, on page 7
- Configuration Migration and SSH Authentication, on page 9
- Sample Network Topology, on page 9
- Deploy ASA Virtual, on page 10
- Integrating Amazon GuardDuty Service and Firewall Threat Defense Virtual, on page 14
- About Secure Firewall ASA Virtual and GuardDuty Integration, on page 14
- Supported Software Platforms, on page 17
- Guidelines and Limitations for Amazon GuardDuty and Secure Firewall ASA Virtual Integration, on page 17
- Integrate Amazon GuardDuty with ASA Virtual, on page 18
- Update Existing Solution Deployment Configuration, on page 29
- Performance Tuning, on page 30

Overview

The ASA Virtual runs the same software as physical ASAs to deliver proven security functionality in a virtual form factor. The ASA Virtual can be deployed in the public AWS cloud. It can then be configured to protect virtual and physical data center workloads that expand, contract, or shift their location over time.

The ASA Virtual support the following AWS instance types.

Table 1: AWS Supported Instance Types and ASA Virtual Versions

AWS Instance type	Attributes		Maximum Number of Interfaces	ASA Virtual Version
	vCPUs	Memory (GB)		
c3.large	2	3.75	3	9.12 and earlier
c3.xlarge	4	7.5	4	9.12 and earlier
c3.2xlarge	8	15	4	9.13 and later
c4.large	2	3.75	3	9.12 and earlier
c4.xlarge	4	7.5	4	9.12 and earlier
c4.2xlarge	8	15	4	9.13 and later
c5.large	2	4	3	9.13 and later
c5.xlarge	4	8	4	9.13 and later
c5.2xlarge	8	16	4	9.13 and later
c5.4xlarge	16	32	8	9.14 and later
c5a.large	2	4	3	9.17 and later
c5a.xlarge	4	8	4	9.17 and later
c5a.2xlarge	8	16	4	9.17 and later
c5a.4xlarge	16	32	8	9.17 and later
c5ad.large	2	4	3	9.17 and later
c5ad.xlarge	4	8	4	9.17 and later
c5ad.2xlarge	8	16	4	9.17 and later
c5ad.4xlarge	16	32	8	9.17 and later
c5d.large	2	4	3	9.17 and later
c5d.xlarge	4	8	4	9.17 and later
c5d.2xlarge	8	16	4	9.17 and later
c5d.4xlarge	16	32	8	9.17 and later
c5n.large	2	5.3	3	9.13 and later
c5n.xlarge	4	10.5	4	9.13 and later
c5n.2xlarge	8	21	4	9.13 and later
c5n.4xlarge	16	42	8	9.13 and later

AWS Instance type	Attributes		Maximum Number of Interfaces	ASA Virtual Version
	vCPUs	Memory (GB)		
m4.large	2	8	2	9.12 and later
m4.xlarge	4	16	4	9.12 and later
m4.2xlarge	8	32	4	9.13 and later
m5n.large	2	8	3	9.17 and later
m5n.xlarge	4	16	4	9.17 and later
m5n.2xlarge	8	32	4	9.17 and later
m5n.4xlarge	16	64	8	9.17 and later
m5zn.large	2	8	3	9.17 and later
m5zn.xlarge	4	16	4	9.17 and later
m5zn.2xlarge	8	32	4	9.17 and later
c6i.large	2	4	3	9.23 and later
c6i.xlarge	4	8	4	9.23 and later
c6i.2xlarge	8	16	4	9.23 and later
c6i.4xlarge	16	32	8	9.23 and later
c6a.large	2	4	3	9.23 and later
c6a.xlarge	4	8	4	9.23 and later
c6a.2xlarge	8	16	4	9.23 and later
c6a.4xlarge	16	32	8	9.23 and later
c6in.large	2	4	3	9.23 and later
c6in.xlarge	4	8	4	9.23 and later
c6in.2xlarge	8	16	4	9.23 and later
c6in.4xlarge	16	32	8	9.23 and later
c6id.large	2	4	3	9.24.10 and later
c6id.xlarge	4	8	4	9.24.10 and later
c6id.2xlarge	8	16	4	9.24.10 and later
c6id.4xlarge	16	32	8	9.24.10 and later
c7i.large	2	4	3	9.24.10 and later

AWS Instance type	Attributes		Maximum Number of Interfaces	ASA Virtual Version
	vCPUs	Memory (GB)		
c7i.xlarge	4	8	4	9.24.10 and later
c7i.2xlarge	8	16	4	9.24.10 and later
c7i.4xlarge	16	32	8	9.24.10 and later
c7a.large	2	4	3	9.24.10 and later
c7a.xlarge	4	8	4	9.24.10 and later
c7a.2xlarge	8	16	4	9.24.10 and later
c7a.4xlarge	16	32	8	9.24.10 and later
c7i-flex.large	2	4	3	9.24.10 and later
c7i-flex.xlarge	4	8	4	9.24.10 and later
c7i-flex.2xlarge	8	16	4	9.24.10 and later
c7i-flex.4xlarge	16	32	8	9.24.10 and later
m5.large	2	4	3	9.24.10 and later
m5.xlarge	4	8	4	9.24.10 and later
m5.2xlarge	8	16	4	9.24.10 and later
m5.4xlarge	16	32	8	9.24.10 and later
m6i.large	2	4	3	9.24.10 and later
m6i.xlarge	4	8	4	9.24.10 and later
m6i.2xlarge	8	16	4	9.24.10 and later
m6i.4xlarge	16	32	8	9.24.10 and later
m6in.large	2	4	3	9.24.10 and later
m6in.xlarge	4	8	4	9.24.10 and later
m6in.2xlarge	8	16	4	9.24.10 and later
m6in.4xlarge	16	32	8	9.24.10 and later
m6a.large	2	4	3	9.24.10 and later
m6a.xlarge	4	8	4	9.24.10 and later
m6a.2xlarge	8	16	4	9.24.10 and later
m6a.4xlarge	16	32	8	9.24.10 and later

AWS Instance type	Attributes		Maximum Number of Interfaces	ASA Virtual Version
	vCPUs	Memory (GB)		
r6i.large	2	4	3	9.24.10 and later
r6i.xlarge	4	8	4	9.24.10 and later
r6i.2xlarge	8	16	4	9.24.10 and later
r6i.4xlarge	16	32	8	9.24.10 and later
r6id.large	2	4	3	9.24.10 and later
r6id.xlarge	4	8	4	9.24.10 and later
r6id.2xlarge	8	16	4	9.24.10 and later
r6id.4xlarge	16	32	8	9.24.10 and later
m7i.large	2	4	3	9.24.10 and later
m7i.xlarge	4	8	4	9.24.10 and later
m7i.2xlarge	8	16	4	9.24.10 and later
m7i.4xlarge	16	32	8	9.24.10 and later
m7a.large	2	4	3	9.24.10 and later
m7a.xlarge	4	8	4	9.24.10 and later
m7a.2xlarge	8	16	4	9.24.10 and later
m7a.4xlarge	16	32	8	9.24.10 and later
m7i-flex.large	2	4	3	9.24.10 and later
m7i-flex.xlarge	4	8	4	9.24.10 and later
m7i-flex.2xlarge	8	16	4	9.24.10 and later
m7i-flex.4xlarge	16	32	8	9.24.10 and later
r7i.large	2	4	3	9.24.10 and later
r7i.xlarge	4	8	4	9.24.10 and later
r7i.2xlarge	8	16	4	9.24.10 and later
r7i.4xlarge	16	32	8	9.24.10 and later
c6in.8xlarge (supported with ASAv-Unlimited only)	32	64	8	9.24.10 and later

AWS Instance type	Attributes		Maximum Number of Interfaces	ASA Virtual Version
	vCPUs	Memory (GB)		
c6in.16xlarge (supported with ASAv-Unlimited only)	64	128	8	9.24.10 and later

Note

- ASA virtual maximum vCPU limit: ASA virtual supports a maximum of 64 vCPUs on AWS. Instances exceeding this capacity are not supported.
- Use the ASAv Unlimited (ASAvU) license for 32/64 vCPU requirements.
- ASA virtual minimum vCPU requirement: ASA virtual requires a minimum of 2 vCPUs. Instances with fewer than 2 vCPUs are not supported.
- For optimal performance, we recommend using c5 instance types.
- By default, ASA Virtual instances are deployed with Secure Boot enabled.
- If the selected instance type supports only the BIOS mode, then the ASA Virtual instance will boot up with BIOS mode.

Tip If you are using M4 or C4 instance type, then we recommend that you migrate to M5 or C5 instance type that uses Nitro hypervisor and Elastic Network Adapter (ENA) interface drivers for improved performance.

Table 2: ASA Virtual Licensed Feature Limits Based on Entitlement

Performance Tier	Instance type (Core/RAM)	Rate Limit	RA VPN Session Limit
ASAv5	c5.large 2 core/4 GB	100 Mbps	50
ASAv10	c5.large 2 core/4 GB	1 Gbps	250
ASAv30	c5.xlarge 4 core/8 GB	2 Gbps	750
ASAv50	c5.2xlarge 8 core/16 GB	10 Gbps	10,000
ASAv100	c5n.4xlarge 16 core/42 GB	16 Gbps	20,000

You create an account on AWS, set up the ASA Virtual using the AWS Wizard, and chose an Amazon Machine Image (AMI). The AMI is a template that contains the software configuration needed to launch your instance.

Important The AMI images are not available for download outside of the AWS environment.

Prerequisites

- Create an account on aws.amazon.com.
- License the ASA Virtual. Until you license the ASA Virtual, it will run in degraded mode, which allows only 100 connections and throughput of 100 Kbps. See Licensing for the ASA Virtual.

Note All the default License entitlements offered by Cisco, previously for ASA Virtual, will have the IPv6 configuration support.

- Interface requirements:
  - Management interface
  - Inside and outside interfaces
  - (Optional) Additional subnet (DMZ)
- Communications paths:
  - Management interface—Used to connect the ASA Virtual to the ASDM; can't be used for through traffic.
  - Inside interface (required)—Used to connect the ASA Virtual to inside hosts.
  - Outside interface (required)—Used to connect the ASA Virtual to the public network.
  - DMZ interface (optional)—Used to connect the ASA Virtual to the DMZ network when using the c3.xlarge interface.
- For ASA Virtual system requirements, see Cisco Secure Firewall ASA Compatibility.

Guidelines and Limitations

Supported Features

The ASA Virtual on AWS supports the following features:

- Support for Amazon EC2 C5 instances, the next generation of the Amazon EC2 Compute Optimized instance family.
- Deployment in the Virtual Private Cloud (VPC)

- Enhanced networking (SR-IOV) where available
- Deployment from Amazon Marketplace
- User deployment of L3 networks
- Routed mode (default)
- IPv6
- Amazon CloudWatch
- Clustering

Unsupported Features

The ASA Virtual on AWS does not support the following:

- Console access (management is performed using SSH or ASDM over network interfaces)
- VLAN
- Promiscuous mode (no sniffing or transparent mode firewall support)
- Multiple context mode
- ASA Virtual native HA
- EtherChannel is only supported on direct physical interfaces
- VM import/export
- Hypervisor agnostic packaging
- VMware ESXi
- Broadcast/multicast messages

  These messages are not propagated within AWS so routing protocols that require broadcast/multicast do not function as expected in AWS. VXLAN can operate only with static peers.

- Gratuitous/unsolicited ARPs

  These ARPS are not accepted within AWS so NAT configurations that require gratuitous ARPs or unsolicited ARPs do not function as expected.

UEFI and Secure Boot Limitations

On AWS, UEFI firmware is supported only for greenfield (fresh) deployments and must be enabled during initial deployment.

Existing brownfield deployments can be upgraded to Version 9.24 without impact. Changing the boot mode or enabling UEFI Secure Boot after deployment is not supported.

Upgrade Restrictions and Limitations

Revert upgrade restrictions

Caution Reverting an upgrade is not supported.

Ensure that you take a backup before upgrading. After upgrading to ASA Virtual 9.24, you cannot roll back to a previous software version. To return to an earlier version, you must redeploy the Management Center.

Configuration Migration and SSH Authentication

Upgrade impact when using SSH public key authentication—Due to updates to SSH authentication, additional configuration is required to enable SSH public key authentication; as a result, existing SSH configurations using public key authentication no longer work after upgrading. Public key authentication is the default for the ASA Virtual on Amazon Web Services (AWS), so AWS users will see this issue. To avoid loss of SSH connectivity, you can update your configuration before you upgrade. Or you can use ASDM after you upgrade (if you enabled ASDM access) to fix the configuration.

The following is a sample original configuration for a username "admin":

```
username admin nopassword privilege 15
username admin attributes
  ssh authentication publickey 55:06:47:eb:13:75:fc:5c:a8:c1:2c:bb:
  07:80:3a:fc:d9:08:a9:1f:34:76:31:ed:ab:bd:3a:9e:03:14:1e:1b hashed
```

To use the **ssh authentication** command, before you upgrade, enter the following commands:

```
aaa authentication ssh console LOCAL
username admin password <password> privilege 15
```

We recommend setting a password for the username as opposed to keeping the **nopassword** keyword, if present. The **nopassword** keyword means that any password can be entered, not that no password can be entered. Prior to 9.6(2), the **aaa** command was not required for SSH public key authentication, so the **nopassword** keyword was not triggered. Now that the **aaa** command is required, it automatically also allows regular password authentication for a **username** if the **password** (or **nopassword**) keyword is present.

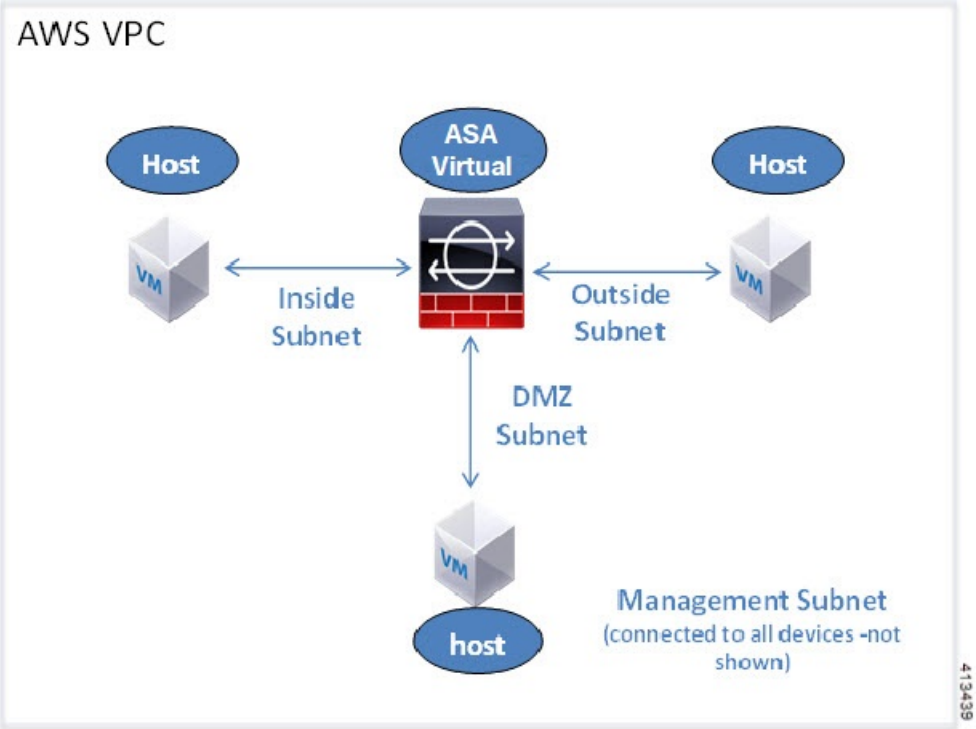
After you upgrade, the **username** command no longer requires the **password** or **nopassword** keyword; you can require that a user cannot enter a password. Therefore, to force public key authentication only, re-enter the **username** command:

```
username admin privilege 15
```

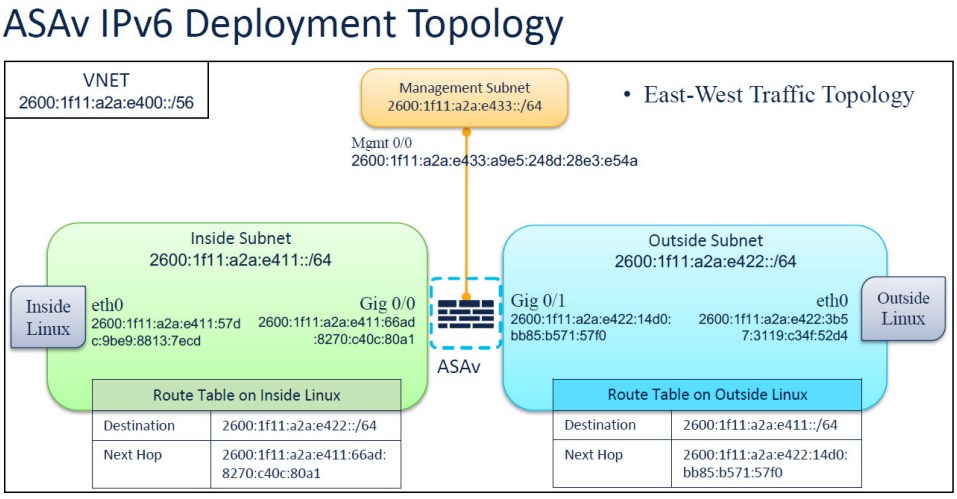
Sample Network Topology

The following figure shows the recommended topology for the ASA Virtual in Routed Firewall Mode with four subnets configured in AWS for the ASA Virtual (management, inside, outside, and DMZ).

Figure 1: Sample ASA Virtual on AWS Deployment



IPv6 Topology



Deploy ASA Virtual

The following procedure provides a top-level list of steps to set up AWS on ASA Virtual. For detailed steps, see Getting Started with AWS.

Procedure

Step 1 Log in to aws.amazon.com and choose your region.

Note AWS is divided into multiple regions that are isolated from each other. The regions are displayed on the upper-right corner of your page. Resources available in one region do not appear in another region. Check periodically to make sure you are in the intended region.

Step 2 Click **My Account** > **AWS Management Console**, and under **Networking**, click **VPC** > **Start VPC Wizard**, and create your VPC by choosing a single private subnet, and set up the following (use the default settings unless otherwise specified):

- Inside and Outside subnet—Enter a name for the VPC and the subnets.
- Internet Gateway—Enter the name of the Internet gateway. It enables direct connectivity over the internet.
- Outside table—Add an entry to enable outbound traffic to the internet (add 0.0.0.0/0 to the internet gateway).

Note Virtual Networks, Subnets, Interface, etc., cannot be created by using IPv6 alone. The IPv4 is used by default, and IPv6 can be enabled along with it. For more information on IPv6, see AWS IPv6 Overview and AWS VPC Migration.

Step 3 Click **My Account** > **AWS Management Console** > **EC2**, and then click **Create an Instance**.

- Select your AMI, for example, Ubuntu Server 14.04 LTS.

  Use the AMI identified in the your image delivery notification.
- Choose the instance type supported by ASA Virtual, for example, c3.large.
- Configure the instance (CPUs and memory are fixed).
- Expand the **Advanced Details** section, and in the optional **User data** field you can enter the Day 0 configuration, which is the text input containing the ASA Virtual configuration applied when the ASA Virtual is launched. For more information on Day 0 configuration with more information, such as Smart Licensing, see Prepare the Day 0 Configuration File.
  - **Management interface**: If you choose to provide the Day 0 configuration details, you *must* provide management interface details, which should be configured to use DHCP.
  - **Data interfaces**: IP addresses for the data interfaces will be assigned and configured only if you provide that information as part of the Day 0 configuration. Data interfaces can be configured to use DHCP, or if the network interfaces to be attached are already created and the IP addresses that are known, you can provide the IP address details in the Day 0 configuration.
  - **Without Day 0 Configuration**: If you deploy the ASA Virtual *without* providing the Day 0 configuration, ASA Virtual applies the default ASA Virtual configuration where it fetches the IP addresses of the attached interfaces from the AWS metadata server and allocates the IP addresses (the data interfaces get the IP addresses assigned but the ENIs will be down). The Management0/0 interface will be up and gets the IP address configured

with the DHCP address. See IP Addressing in your VPC for information about Amazon EC2 and Amazon VPC IP addressing.

Sample Day 0 Configuration -

```
! ASA Version 9.x.1.200
!
interface management0/0
management-only
nameif management
security-level 100
ip address dhcp setroute
ipv6 enable
ipv6 address dhcp default
no shutdown
!
!
GWLB facing VTEP interface
interface TenGigabitEthernet0/0
nameif data-interface-in
security-level 100
ip address dhcp
no shut

!
Internet-facing outside interface
interface TenGigabitEthernet0/1
nameif data-interface-out
security-level 0
ip address dhcp
no shut

nve 1
encapsulation geneve
source-interface data-interface-in
interface vni1
proxy dual-arm
nameif vni-in
security-level 0
vtep-nve 1
! NAT for internet-bound traffic
nat (vni-in, data-interface-out) source dynamic any interface
!Default route to internet gateway= 10.1.200.1 (Outside gateway)
!Route East-West traffic (Application subnet CIDR) back to vni interface (U-turn)
route data-interface-out 0.0.0.0 0.0.0.0 10.1.200.1
route vni-in 192.168.1.0 255.255.255.0 10.1.100.1 1
!
mtu data-interface-in 1826
jumbo-frame reservation
same-security-traffic permit inter-interface
same-security-traffic permit intra-interface


crypto key generate rsa modulus 2048
ssh 0 0 management
ssh ::/0 management
ssh timeout 60
ssh version 2
username admin password Q1w2e3r4 privilege 15
username admin attributes
service-type admin
aaa authentication ssh console LOCAL
!
```

```
same-security-traffic permit inter-interface
same-security-traffic permit intra-interface
access-list allow-all extended permit ip any any
access-list allow-all extended permit ip any6 any6
access-group allow-all global
!
interface G0/0
nameif outside
ip address dhcp setroute
ipv6 enable
ipv6 address dhcp default
no shutdown
!
interface G0/1
nameif inside
ip address dhcp
ipv6 enable
ipv6 address dhcp default
no shutdown
!
```

- Storage: Retain the default values.
- Tag Instance: You can create a lot of tags to classify your devices. Giving a name to your devices helps you locate them easily.
- Security Group: Create a security group and name it. The security group is a virtual firewall for an instance to control inbound and outbound traffic.

  By default the Security Group is open to all addresses. Change the rules to only allow SSH in from addresses used to access your ASA Virtual.

  For information on how the security group controls the traffic, refer to AWS documentation - Control traffic to your AWS resources using security groups.
- Expand the **Advanced Details** section and in the **User data** field you can optionally enter a Day 0 configuration, which is text input that contains the ASA Virtual configuration applied when the ASA Virtual is launched. For more information on how to configure the Day 0 configuration with more information, such as Smart Licensing, see Prepare the Day 0 Configuration File.
  - **Management interface** - If you choose to provide a Day 0 configuration, you **must** provide management interface details, which should be configured to use DHCP.
  - **Data interfaces** - IP addresses for the data interfaces will be assigned and configured only if you provide that information as part of the Day 0 configuration. Data interfaces can be configured to use DHCP or, if the network interfaces to be attached are already created and the IP addresses are known, you can provide the IP details in the Day 0 configuration.
  - **Without Day 0 Configuration** - If you deploy the ASA Virtual **without** providing the Day 0 configuration, the ASA Virtual applies the default ASA Virtual configuration where it fetches the IPs of the attached interfaces from the AWS metadata server and allocates the IP addresses (the data interfaces will get the IPs assigned but the ENIs will be down). Management0/0 interface will be up and gets the IP configured with DHCP address. See IP Addressing in your VPC for information about Amazon EC2 and Amazon VPC IP addressing.
- Review your configuration and then click **Launch**.

Step 4 Create a Key Pair.

Caution Give the key pair a name you will recognize and download the key to a safe place; the key can never be downloaded again. If you lose the key pair, you must destroy your instances and redeploy them again.

Step 5 Click **Launch Instance** to deploy your ASA Virtual.

Step 6 Click **My Account** > **AWS Management Console** > **EC2** > **Launch an Instance** > **My AMIs**.

Step 7 Make sure that the Source/Destination Check is disabled per interface for the ASA Virtual.

AWS default settings only allow an instance to receive traffic for its IP address (IPv4 and IPv6) and only allow an instance to send traffic from its own IP address (IPv4 and IPv6) . To enable the ASA Virtual to act as a routed hop, you must disable the Source/Destination Check on each of the ASA Virtual's traffic interfaces (inside, outside, and DMZ).

Integrating Amazon GuardDuty Service and Firewall Threat Defense Virtual

Amazon GuardDuty is a monitoring service that processes data from various sources such as VPC logs, CloudTrail management event logs, CloudTrail S3 data event logs, DNS logs, and so on to identify potentially unauthorized and malicious activity in the AWS environment.

About Secure Firewall ASA Virtual and GuardDuty Integration

Cisco offers a solution to integrate the Amazon GuardDuty service with Secure Firewall ASA Virtual using CLI over SSH.

This solution use the threat analysis data or results from the Amazon GuardDuty (malicious IPs generating threats, attacks and so on) and feeds that information (malicious IP) to the Secure Firewall ASA Virtual to protect the underlying network and applications against future threats originating from these sources (malicious IP).

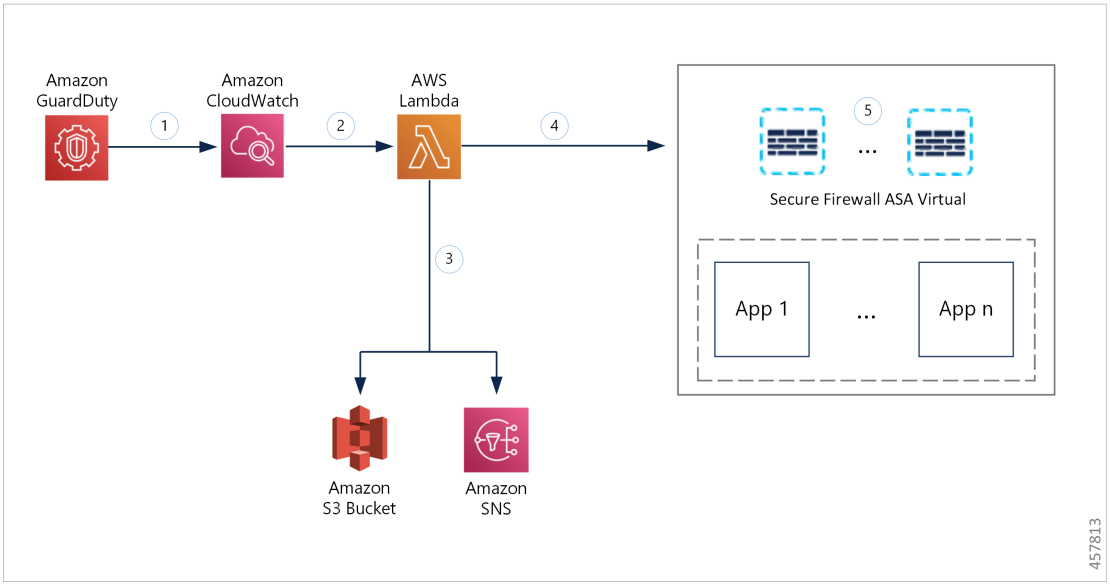
End-to-End Procedure

The following integration solutions with workflow illustrations help you understand the integration of Amazon GuardDuty with Secure Firewall Threat Defense Virtual.

The following workflow diagram shows the Amazon GuardDuty integration solution with ASA Virtual.

Integration with Secure Firewall Device Manager using Network Object Group

The following workflow diagram shows the Amazon GuardDuty integration solution with Secure Firewall Device Manager using the network object group.



1	The GuardDuty service sends threat findings to CloudWatch when it detects a malicious activity.
2	The CloudWatch event activates the AWS Lambda function.
3	The Lambda function updates the malicious host in the report file in the S3 bucket and sends a notification via SNS.
4	The Lambda function configures or updates the network object group with the malicious host IP address in Secure Firewall Device Manager.
5	The Secure Firewall Device Manager access control policy directs the managed device to handle the traffic based on the configured actions, for example, block traffic from the malicious hosts reported by GuardDuty. This access control policy uses the network object group with the malicious IP address provided by the Lambda function.

Key Components of This Integration

Component	Description
Amazon GuardDuty	An Amazon service responsible for generating threat findings for the various AWS resources in a specific region, such as EC2, S3, IAM, and so on.

Amazon Simple Storage Service (S3)	An Amazon service used for storing various artifacts associated with the solution: • Lambda function zip file • Lambda layer zip file • ASA Virtual configuration input file(.ini) • Output report file (.txt) containing a list of malicious IP addresses reported by the Lambda function
Amazon CloudWatch	An Amazon service used for: • Monitoring the GuardDuty service for any reported findings and triggering the Lambda function to process the finding. • Logging the Lambda function-related activities in the CloudWatch log group.
Amazon Simple Notification Service (SNS)	An Amazon service used to push email notifications. These email notifications contain: • The details of the GuardDuty finding that was successfully processed by the Lambda function. • The details of the updates performed on the Secure Firewall managers by the Lambda function. • Any significant errors encountered by the Lambda function.
AWS Lambda Function	An AWS serverless compute service that runs your code in response to events and automatically manages the underlying compute resources for you. The Lambda function is triggered by the CloudWatch event rule based on GuardDuty findings. In this integration, the Lambda function is responsible for: • Processing the GuardDuty findings to verify that all the required criteria are met, such as severity, connection direction, presence of malicious IP address, and so on. • (Depending on the configuration) Updating the network object group on the Secure Firewall managers with the malicious IP address. • Updating the malicious IP address in the report file on the S3 bucket. • Notifying the Secure Firewall administrator about various manager updates and any errors.

CloudFormation Template	Used to deploy various resources required for the integration in AWS. The CloudFormation template contains the following resources: • **AWS::SNS::Topic**: The SNS Topic for pushing email notifications. • **AWS::Lambda::Function, AWS::Lambda::LayerVersion** : The Lambda function and layer files • **AWS::Events::Rule**: The CloudWatch event rule to trigger the Lambda function based on the GuardDuty findings event. • **AWS::Lambda::Permission**: Permission for the CloudWatch event rule to trigger the Lambda function. • **AWS::IAM::Role, AWS::IAM::Policy**: The IAM role and policy resources to allow various access permissions to the Lambda function for various AWS resources. This template accepts user input parameters to customize the deployment.

Supported Software Platforms

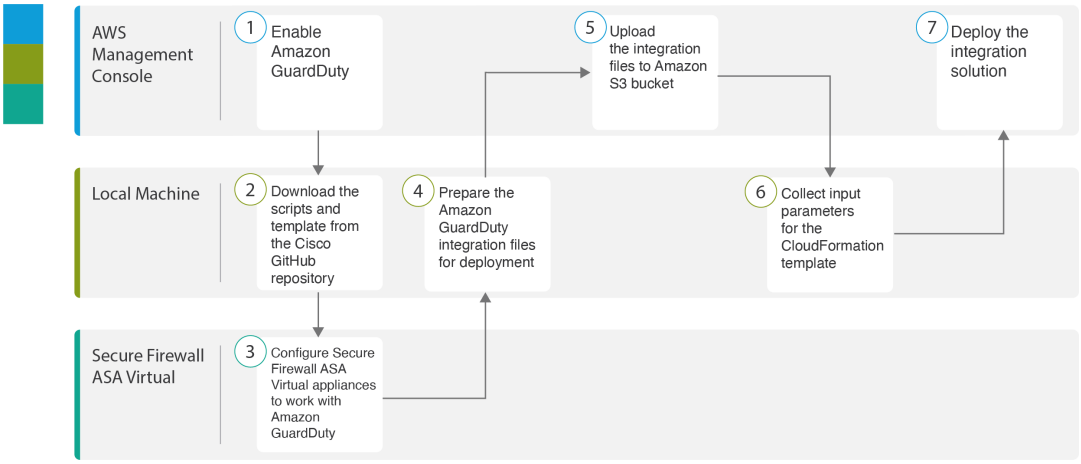
- The GuardDuty integration solution is applicable to Secure Firewall ASA Virtual managed using CLI over SSH.
- The Lambda function can update the network object groups in the Secure Firewall ASA Virtual. Ensure that the Lambda function can connect to Secure Firewall ASA Virtual using public IP addresses.

Guidelines and Limitations for Amazon GuardDuty and Secure Firewall ASA Virtual Integration

- The Lambda function is responsible only for updating the network objects groups with the malicious IP addresses. Depending on your requirement, create access rules and access policies to block any traffic that is not required.
- The AWS services used in this integration are region-specific. If you want to use GuardDuty findings from different regions, you must deploy region-specific instances.
- You can configure ASA Virtual updates using CLI over SSH. ASDM, CSM, and Security Cloud Control, are not supported.
- You can use only password-based login. No other authentication methods are supported.
- If you are using encrypted passwords in the input file, keep in mind that:
  - Only encryption using the symmetric KMS keys is supported.
  - All the passwords must be encrypted using a single KMS key accessible to the Lambda function.

Integrate Amazon GuardDuty with ASA Virtual

Perform the following tasks to integrate Amazon GuardDuty with ASA Virtual



	Workspace	Steps
1	AWS Management Console	Enable Amazon GuardDuty Service on AWS, on page 18
2	Local Machine	Download the Secure Firewall ASA Virtual and Amazon GuardDuty Solution Template, on page 19
3	ASA Virtual	Configure your Managed Devices to Work with Amazon GuardDuty, on page 19
4	Local Machine	Prepare Amazon GuardDuty Resource Files for Deployment, on page 22
5	AWS Management Console	Upload Files to Amazon Simple Storage Service, on page 25
6	Local Machine	Collect Input Parameters for CloudFormation Template, on page 25
7	AWS Management Console	Deploy the Stack, on page 27

Enable Amazon GuardDuty Service on AWS

This section describes how to enable Amazon GuardDuty service on AWS.

Before you begin

Ensure that all the AWS resources are in the same region.

Procedure

Step 1 Go to https://aws.amazon.com/marketplace (Amazon Marketplace) and sign in.

Step 2 Choose **Services** > **GuardDuty**.

Step 3 Click **Get Started** in the **GuardDuty** page.

Step 4 Click **Enable GuardDuty** to enable the Amazon GuardDuty service.

For more information on enabling GuardDuty, see Getting started with GuardDuty in AWS Documentation.

What to do next

Download the Amazon GuardDuty solution files (templates and scripts) from the Cisco GitHub repository. See .Download the Secure Firewall ASA Virtual and Amazon GuardDuty Solution Template, on page 19

Download the Secure Firewall ASA Virtual and Amazon GuardDuty Solution Template

Download the files required for the Amazon GuardDuty solution. The deployment scripts and templates for your Secure Firewall ASA Virtual version are available from the Cisco GitHub repository at:

https://github.com/CiscoDevNet/cisco-asav

The following is a list of resources in the Cisco GitHub repository:

Files	Description
READ.MD	ReadMe file
configuration/	Secure Firewall ASA Virtual Configuration file template.
images/	It contains the Secure Firewall ASA Virtual and Amazon GuardDuty integration solution illustrations.
lambda/	Lambda function Python files.
templates/	CloudFormation template for deployment.

Configure your Managed Devices to Work with Amazon GuardDuty

The Lambda function processes the Amazon GuardDuty findings and identifies the malicious IP address that triggered the CloudWatch Event. Then, the Lambda function updates the network object group in the ASAv with the malicious IP address. You can then configure an access control policy that uses this network object group to handle the traffic.

Create Network Object Group

In the ASA Virtual, you must configure or create a network object group for the Lambda function to update the malicious IP address detected by the Amazon GuardDuty.

If you do not configure a network object group with the Lambda function, then a network object group with the default name **aws-gd-suspicious-hosts** is created by the Lambda function to update the malicious IP address.

Create Network Object Group in Secure Firewall ASA Virtual

In Secure Firewall ASA Virtual, you must create a network object group for the Lambda function to update the malicious IP address detected by the Amazon GuardDuty.

If you do not configure a network object group with the Lambda function, then a network object group with the default name *aws-gd-suspicious-hosts* is created by the Lambda function to update the malicious IP address.

Initially, to use the network object group in an ACL rule, you may have to create the object group with a dummy IP address. You can create multiple network object groups on a single ASAv.

For more information about network object group and access policy, see Cisco ASA Series Firewall CLI Configuration Guide.

To create the network object group, perform the following steps:

Procedure

Step 1 Log in to Secure Firewall ASA Virtual.

Step 2 Create a network object group with a description. In this example, a dummy host IP address 12.12.12.12 is added to the network object group created.

Example:

```
hostname(config)# object-group network aws-gd-suspicious-hosts
hostname(config)# description Malicious Hosts reported by AWS GuardDuty
hostname(config)# network-object host 12.12.12.12
```

Step 3 Create or update the Access Policy or Access Control Rule to handle the traffic using the network object group. \

Tip You can also create or update the Access Control Policy or Access Control Rule after verifying that the Lambda function is updating the network object group with the malicious IP address.

Example:

```
hostname(config)# access-list out-iface-access line 1 extended deny ip object-group
aws-gd-suspicious-hosts any
```

Creating User Accounts in ASAv for Lambda Function access

The Lambda function requires a dedicated user on ASAv to handle configuration updates. A privilege level of 15 ensures that the user has all privileges.

For more information about creating user, see Cisco ASA Series Firewall CLI Configuration Guide.

Procedure

Step 1 Create a user.

username *name* [**password** *password*] **privilege** *level*

Example:

```
hostname(config)# username aws-gd password MyPassword@2021 privilege 15
```

Step 2 Configure username attributes.

username *username* **attributes**

Example:

```
hostname(config)# username aws-gd attributes
```

Step 3 Provide the user with admin level access to all services.

service-type admin

Example:

```
hostname(config)# service-type admin
```

(Optional) Encrypt Passwords

If required, you can provide encrypted passwords in the input configuration file. You can also provide passwords in plain text format.

Encrypt all the passwords using a single KMS key that is accessible to the Lambda function. Use the **aws kms encrypt --key-id** <*KMS-ARN*> **--plaintext** <*password*> command to generate the encrypted password. You have to install and configure AWS CLI to run this command.

Note Ensure that passwords are encrypted using symmetric KMS keys.

For more information on AWS CLI, see AWS Command Line Interface. For more information on master keys and encryption, see the AWS document Creating keys and the AWS CLI Command Reference about password encryption and KMS.

Example:

```
$ aws kms encrypt --key-id <KMS-ARN> --plaintext <password>
```

```
{
    "KeyId": "KMS-ARN",
    "CiphertextBlob":
"AQICAHgcQFAGtz/hvaxMtJvY/x/rfHnKI3clFPpSXUU7HQRnCAFwfXhXHJAHL8tcVmDqurALAAAAajBoBgkqhki
G9w0BBwagWzBZAgEAMFQGCSqGSIb3DQEHATAeBglghkgBZQMEAS4wEQQM45AIkTqjSekX2mniAgEQgCcOav6Hhol
+wxpWKtXY4y1Z1d0z1P4fx0jTdosfCbPnUExmNJ4zdx8="
}
$
```

The value of *CiphertextBlob* key should be used as a password.

Prepare Amazon GuardDuty Resource Files for Deployment

The Amazon GuardDuty solution deployment resource files are available on the Cisco GitHub repository.

Before deploying the Amazon GuardDuty solution on AWS, you must prepare the following files:

- Secure Firewall ASA Virtual manager configuration input file
- Lambda function zip file
- Lambda layer zip file

Prepare Configuration Input file

In the configuration template, you must define the details of the ASAv you are integrating with the Amazon GuardDuty solution.

Before you begin

- Ensure to authenticate and verify the user account of the device manager before you provide the user account details in the configuration file.
- Ensure that you configure only one ASAv in the configuration file. If you configure multiple ASAvs, then the Lambda function may simultaneously update all the ASAvs configured in the file resulting in race conditions and non deterministic behavior.
- You must have noted the IP address and name of the ASAv.
- You must have created a user account having admin privileges for the Lambda function to access and update these network object groups in the ASAv.

Procedure

Step 1 Log in to the local machine where you have downloaded the Amazon GuardDuty resource files.

Step 2 Browse to the **asav-template** > **configuration** folder.

Step 3 Open the `asav-manager-config-input.ini` file in a text editor tool. In this file, you must enter the details of the ASAv on which you plan to integrate and deploy the Amazon GuardDuty solution.

Step 4 Enter the following ASAv parameters:

Parameters	Description
[asav-1]	Section name: Unique ASAv identifier within the file

Parameters	Description
public-ip	Public IP address of the ASAv
user name	User name to log in to ASAv.
password	Password to log in to ASAv. The password can be in plain text format or an encrypted string that has been encrypted using KMS.
enable-password	Enable password of the ASAv. The password can be in plain text format or an encrypted string that has been encrypted using KMS.
object-group-name	Name of the network object groups name that is updated with malicious host IP by the Lambda function. If you are entering multiple network object groups name, ensure that they are comma separated values.

Step 5 Save and close the `asav-manager-config-input.ini`file.

What to do next

Create the Lambda Function archive file.

Preparing Lambda Function Archive File

This section describes how to archive the Lambda function files in a Linux environment.

Note The archiving process may differ depending on the operating system of the local machine where you are archiving the files.

Before you begin

Ensure that your Linux host is running Ubuntu version 18.04 with Python version 3.6 or later.

Procedure

Step 1 Open the CLI console on the local machine on which you have downloaded the Amazon GuardDuty resources.

Step 2 Navigate to the /lambda folder and archive the files. The following is a sample transcript from a Linux host.

```
$ cd lambda
$ zip asav-gd-lambda.zip *.py
adding: aws.py (deflated 71%)
adding: asav.py (deflated 79%)
adding: main.py (deflated 73%)
adding: utils.py (deflated 65%)
$
```

The zip file `asav-gd-lambda.zip` is created.

Step 3 Exit and close the CLI console.

What to do next

Create the Lambda layer zip file using the zip file `asav-gd-lambda.zip` file.

Prepare Lambda Layer File

This section describes how to archive the Lambda layer file in a Linux environment.

Note The archiving process may differ depending on the operating system of the local machine where you are archiving the file.

Procedure

Step 1 Open the CLI console on the local machine where you have downloaded the Amazon GuardDuty resources.

Step 2 Perform the following actions in your CLI console.

The following is a sample transcript from a Linux host such as Ubuntu 22.04 with Python 3.9 installed.

```
$ mkdir -p layer
$ virtualenv -p /usr/bin/python3.9 ./layer/
$ source ./layer/bin/activate
$ pip3.9 install cffi==1.15.0
$ pip3.9 install cryptography==37.0.2
$ pip3.9 install paramiko==2.7.1
$ mkdir -p ./python/.libs_cffi_backend/
$ cp -r ./layer/lib/python3.9/site-packages/* ./python/
$ zip -r asav-gd-lambda-layer.zip ./python
```

The zip file `asav-gd-lambda-layer.zip` is created.

Note that you must install Python 3.9 and its dependencies for creating the Lambda layer.

The following is the sample transcript for installing Python 3.9 on a Linux host such as Ubuntu 22.04.

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

Step 3 Exit and close the CLI console.

What to do next

In Amazon S3 bucket, you must upload the Secure Firewall ASA Virtual configuration file, the Lambda function zip file, and the Lambda layer zip file. See Upload Files to Amazon Simple Storage Service, on page 25

Upload Files to Amazon Simple Storage Service

After you prepare all the Amazon GuardDuty solution artifacts, you must upload the files to an Amazon Simple Storage Service (S3) bucket folder in the AWS portal.

Procedure

Step 1 Go to https://aws.amazon.com/marketplace (Amazon Marketplace) and sign in.

Step 2 Open the Amazon S3 console.

Step 3 Create an Amazon S3 Bucket for uploading the Amazon GuardDuty artifacts. See Creating Amazon S3.

Step 4 Upload the following Amazon GuardDuty artifacts to the Amazon S3 bucket.

- Secure Firewall ASA Virtual configuration file: `asav-config-input.ini`

Note This file is not required to be uploaded when you are using Security Intelligence Network Feed method for deploying the Amazon GuardDuty solution in management centers.

- Lambda layer zip file: `asav-gd-lambda-layer.zip`
- Lambda function zip file: `asav-gd-lambda.zip`

What to do next

Prepare the CloudFormation template that is used for deploying Amazon GuardDuty resources. See Collect Input Parameters for CloudFormation Template, on page 25.

Collect Input Parameters for CloudFormation Template

Cisco provides the CloudFormation template that is used to deploy resources required by Amazon GuardDuty solution in AWS. Collect values for the following template parameters before deployment.

Procedure

Template Parameters

Parameter	Description	Example
Deployment name*	The name you enter in this parameter is used as prefix for all the resources created by the Cloud Formation template.	**cisco-asav-gd**
Minimum severity level of GD finding*	Minimum severity level Amazon GuardDuty findings to be considered for processing must be in the range between **1.0** to **8.9**. Any finding reported with a lesser severity than the minimum range is ignored. Severity classification is as follows: • Low: 1.0 to 3.9 Medium: 4.0 to 6.9 High: 7.0 to 8.9.	**4.0**
Administrator email ID*	Administrator email address to receive notifications on Secure Firewall ASA Virtual about the updates done by Lambda function in the Secure Firewall ASA Virtual.	**abc@xyz.com**
S3 Bucket name*	Name of the Amazon S3 bucket containing Amazon GuardDuty artifacts files (Lambda function zip, Lambda layer zip, and Secure Firewall ASA Virtual configuration manager files).	For example: **asav-gd-bucket**
S3 Bucket folder/path prefix	Amazon S3 bucket path or folder name where the configuration files are stored. If there is no folder, leave this field empty.	For example: "" or "**cisco/asav-gd/**"
Lambda layer zip file name*	Lambda layer zip file name.	For example: `asav-gd-lambda-layer.zip`
Lambda function zip file name*	Lambda function zip file name.	For example:`asav-gd-lambda.zip`
Secure Firewall ASA Virtual manager configuration file name	The ***.ini** file containing the manager configuration details of the Secure Firewall ASA Virtual. (Public IP, username, password, device-type, network object group names and so on.)	For example: `asav-config-input.ini`
ARN of KMS key used for password encryption	ARN of an existing KMS (AWS KMS key used for password encryption). You can leave this parameter empty in case	For example:**arn:aws:kms:<region>:<account-id>:key/<key-id>**

Parameter	Description	Example
	plain text passwords are provided in the Secure Firewall ASA Virtual configuration input file. If you specify, all the passwords mentioned in the Secure Firewall ASA Virtual configuration input file must be encrypted. The passwords must be encrypted using only the specified ARN. Generating encrypted passwords: aws kms encrypt --key-id <KMS ARN> --plaintext <password>	
Enable/Disable debug logs*	Enable or Disable the Lambda function debug logs in the CloudWatch.	For example: **enable** or **disable**

*: Mandatory field

What to do next

Deploy the stack using the CloudFormation template. See Deploy the Stack, on page 27

Deploy the Stack

After all the pre-requisite processes for Amazon GuardDuty solution deployment are completed, create the AWS CloudFormation stack. Use the template file in the target directory: `templates/cisco-asav-gd-integration.yaml`, and provide the parameters collected in Collect Input Parameters for CloudFormation Template.

Procedure

Step 1 Log in to AWS console.

Step 2 Go to Services > CloudFormation > Stacks > Create stack (with new resources) > Prepare template (The template is provided in the folder) > Specify template > Template source (Upload the template file from the target directory: `templates/cisco-asav-gd-integration.yaml`) > Create Stack

For more information on deploying a stack on AWS, see AWS Documentation.

What to do next

Validate your deployment. See Validate Your Deployment, on page 28.

Also, subscribe to receive an email notifications on threat detection updates reported by Amazon GuardDuty. See Subscribe to the Email Notifications, on page 28.

Subscribe to the Email Notifications

In the CloudFormation template, an email ID is configured to receive notification about GuardDuty finding updates done by the Lambda function. After deploying the CloudFormation template on AWS, an email notification is sent to this email ID via Amazon Simple Notification Service (SNS) service requesting you to subcribe for notification updates.

Procedure

Step 1 Open the email notification.

Step 2 Click the **Subscription** link available in the email notification.

What to do next

Validate your deployment. See Validate Your Deployment, on page 28.

Validate Your Deployment

In AWS, you have options to verify the Amazon GuardDuty solution as described in this section. You can follow these deployment validation instructions after the CloudFormation deployment is complete.

Before you begin

Ensure that you have installed and configured AWS Command Line Interface (CLI) to run commands for validating the deployment. For information on AWS CLI documentation, see AWS Command Line Interface.

Procedure

Step 1 Log in to AWS Management console.

Step 2 Go to **Services** > **GuardDuty** > **Settings** > **About GuardDuty** > **Detector ID** to note the detector ID.

This detector ID is required for generating sample Amazon GuardDuty findings.

Step 3 Open the AWS CLI console to generate the sample Amazon GuardDuty finding by running the following commands:

```
aws guardduty create-sample-findings --detector-id <detector-id> --finding-types
UnauthorizedAccess:EC2/MaliciousIPCaller.Custom
```

```
aws guardduty create-sample-findings --detector-id <detector-id> --finding-types
UnauthorizedAccess:EC2/MaliciousIPCaller.Custom
```

Step 4 Check for the sample finding in the findings list on Amazon GuardDuty console.

The sample findings contains the prefix **[sample]**. You can check the sample finding details viewing the attributes such as connection direction, remote IP address and so on.

Step 5 Wait for the Lambda function to run.

After the Lambda function is triggered, verify the following:

- An email notification with the details regarding Amazon GuardDuty finding received and Secure Firewall ASA Virtual updates done by the Lambda function.
- Verify whether the report file is generated in the Amazon S3 bucket. It contains the malicious IP address reported by the sample Amazon GuardDuty finding. You can identify the report file name in the format: `<deployment-name>-report.txt`.
- Verify that the network object groups are updated on the configured managers (Secure Firewall ASA Virtual) with the malicious IP address updated from the sample finding.

Step 6 Go to **AWS Console > Services > CloudWatch > Logs > Log groups >** *select the log group* to verify the Lambda logs in the CloudWatch console. You can identify the CloudWatch log group name in the format: `<deployment-name>-lambda`.

Step 7 After validating the deployment, we recommend that you can clean the data generated by the sample finding as follows:

a) Go to **AWS Console > Services > GuardDuty > Findings > Select the finding > Actions > Archive** to view the sample finding data.

b) Delete the malicious IP addresses added in the network object group to clear cached data from the Secure Firewall ASA Virtual.

c) Clean up the report file in Amazon S3 bucket. You may update the file by removing the malicious IP addresses reported by the sample finding.

Update Existing Solution Deployment Configuration

We recommend that you do not update the S3 bucket or the S3 bucket folder and path prefix values after deployment. However, if there is a requirement to update the configuration for a solution that has been deployed, use the **`Update Stack`** option on the CloudFormation page in the AWS console.

You can update any of the parameters given below.

Parameter	Description
Secure Firewall ASA Virtual manager configuration file name	Add or update the configuration file in Amazon S3 bucket. You are allowed to update the file with same name as previous one. If the configuration file name is modified, then you can update this parameter by using **Update stack** option in the AWS console.
Minimum severity level of GD finding*	Use the **Update stack** option in AWS console to update the parameter value.
Administrator email ID*	Update the email ID parameter value using the **Update Stack** option in AWS console. You can also add or update email subscriptions via SNS service console.
S3 Bucket name*	Update the zip file in the Amazon S3 bucket with a new name and then update the parameter by using the **Update Stack** option in AWS console.
Lambda layer zip file name*	Update the Lambda layer zip file name in the Amazon S3 bucket with a new name and then update this

Parameter	Description
	parameter value by using the **Update stack** option in AWS console.
Lambda function zip file name*	Update the Lambda function zip file in the Amazon S3 bucket with a new name and then update this parameter value by using the **Update stack** option in AWS console.
ARN of KMS key used for password encryption	Use the **Update stack** option in AWS console to update the parameter value.
Enable/Disable debug logs*	Use the **Update stack** option in AWS console to update the parameter value.

Procedure

Step 1 Go to the AWS management console.

Step 2 If required, create the new bucket and folder.

Step 3 Ensure that the artifacts given below are copied from the old bucket to the new bucket.

- Secure Firewall ASA Virtual configuration file: `asav-config-input.ini`
- Lambda layer zip file: `asav-gd-lambda-layer.zip`
- Lambda function zip file: `asav-gd-lambda.zip`
- Output report file: `<deployment-name>-report.txt`

Step 4 To update the parameter values, go to **Services** > **CloudFormation** > **Stacks** > > **Update (Update Stack)** > **Prepare template** > **Use current template** > **Next** > <*update parameters*>> **Update Stack**.

Performance Tuning

VPN Optimization

The AWS c5 instances offer much higher performance than the older c3, c4, and m4 instances. The approximate RA VPN throughput (DTLS using 450B TCP traffic with AES-CBC encryption) on the c5 instance family should be:

- 0.5Gbps on c5.large
- 1Gbps on c5.xlarge
- 2Gbps on c5.2xlarge

Console Logging Consideration

Enabling console logging at informational or debugging levels in AWS environments can increase CPU load on the vCPU that handles serial interrupts. This additional overhead may lead to reduced throughput or, in some cases, overall system instability. For this reason, it is recommended to enable such logging only during active troubleshooting and during a scheduled maintenance window.