

Secure Kubernetes Services in an EKS Cluster

BETA

Contact Information

Corporate Headquarters:

Palo Alto Networks

3000 Tannery Way

Santa Clara, CA 95054

www.paloaltonetworks.com/company/contact-support

About the Documentation

- To ensure you are viewing the most current version of this document, or to access related documentation, visit the Technical Documentation portal: docs.paloaltonetworks.com.
- To search for a specific topic, go to our search page: docs.paloaltonetworks.com/search.html.
- Have feedback or questions for us? Leave a comment on any page in the portal, or write to us at documentation@paloaltonetworks.com.

Copyright

Palo Alto Networks, Inc.

www.paloaltonetworks.com

© 2018-2019 Palo Alto Networks, Inc. Palo Alto Networks is a registered trademark of Palo Alto Networks. A list of our trademarks can be found at www.paloaltonetworks.com/company/trademarks.html. All other marks mentioned herein may be trademarks of their respective companies.

Last Revised

September 4, 2019

Table of Contents

Secure Kubernetes Services in an EKS Cluster.....	5
How Does the Panorama Plugin for Amazon Secure Elastic Kubernetes Services?.....	7
Requirements.....	7
System Architecture.....	8
Secure an EKS Cluster with VM-Series Firewall and AWS Plugin on Panorama.....	11
Set Up Your Panorama Configuration.....	11
Set Up Your AWS Bootstrap Bucket.....	12
Deploy the Firewall Template on AWS.....	13
Deploy the Cluster Stack.....	15
Set Up Kubectl and Configure Your Cluster.....	15
Add an EKS Cluster.....	17
Configure Inbound Protection and Outbound Monitoring.....	17
Configure the ALB.....	22
Test the Outbound Workflow.....	23
Known Issues.....	24
Get Help.....	25
Related Documentation.....	25
Requesting Support.....	25

Secure *Kubernetes Services* in an *EKS Cluster*

The AWS plugin enables you to secure North-South traffic destined to or originating from container services and workloads in Amazon Elastic Kubernetes Service (EKS) environments in which you have deployed VM-Series firewalls.

After you configure the plugin on Panorama to communicate with an EKS cluster, the plugin uses the Kubernetes SDK to retrieve information from each service that has an exposed IP address or FQDN. With this information you can create Security policy in Panorama. To secure inbound and outbound traffic to the cluster, push your configuration to managed VM-Series firewalls.

- > [How Does the Panorama Plugin for Amazon Secure Elastic Kubernetes Services?](#)
- > [Secure an EKS Cluster with VM-Series Firewall and AWS Plugin on Panorama](#)
- > [Known Issues](#)
- > [Get Help](#)

How Does the Panorama Plugin for Amazon Secure Elastic Kubernetes Services?

You can use VM-Series firewalls to secure inbound traffic for Amazon Elastic Kubernetes Service (EKS) clusters. The Panorama plugin for Amazon EKS secures inbound traffic to Kubernetes clusters, and provides outbound monitoring for traffic exiting the cluster. Outbound traffic can return through the VM-Series firewall, but firewall rules applied to outbound traffic must have a default allow all policy rule to permit Kubernetes orchestration traffic to function.

- The minimum Panorama software version is 9.0.3.
- You must use the community-supported [templates](#) to deploy your VM-Series firewall (or firewall set) in the same VPC as your EKS cluster. You can create up to 16 clusters in the same VPC and secure them with the same firewall or firewall set.

This chapter reviews different components that enable the AWS Plugin for Panorama to secure an EKS cluster.

- [Requirements](#)
- [System Architecture](#)
- [AWS Plugin on Panorama](#)

Requirements

This solution requires the following:

- Panorama—A licensed version of Panorama, version 9.0.3 and later.
 - Your Panorama version must be the same version or a later version than your VM-Series firewall PAN-OS version.
 - AWS Plugin on Panorama—version 2.0.0. See [AWS Plugin on Panorama](#).
- VM-Series firewalls—PAN-OS version 8.1 and later, or version 9.0.3 and later.

[VM-Series Firewall Licenses for Public Clouds](#) and you must know the auth code so that you can use it to bootstrap the firewall.

- Community-supported EKS template bundle, version 1.0. See [Templates](#).
- AWS account—In addition to your user name and password you must know your [AWS Access Key](#), which is comprised of the access key ID and the secret access key. If you have an account but do not know your secret access key, you can [create an access key](#) and save the .csv file in a secure place. The required policies and roles below will permit it.

Be sure to follow the [IAM best practices](#).

- AWS policies and roles—Your AWS account must be able to access to the following service policies:

To manage the firewalls:

- AmazonEC2ContainerServiceforEC2Role
- AmazonEKS_CNI_Policy
- AmazonEKSClusterPolicy
- AmazonEKSServicePolicy
- AmazonEKSWorkerNodePolicy

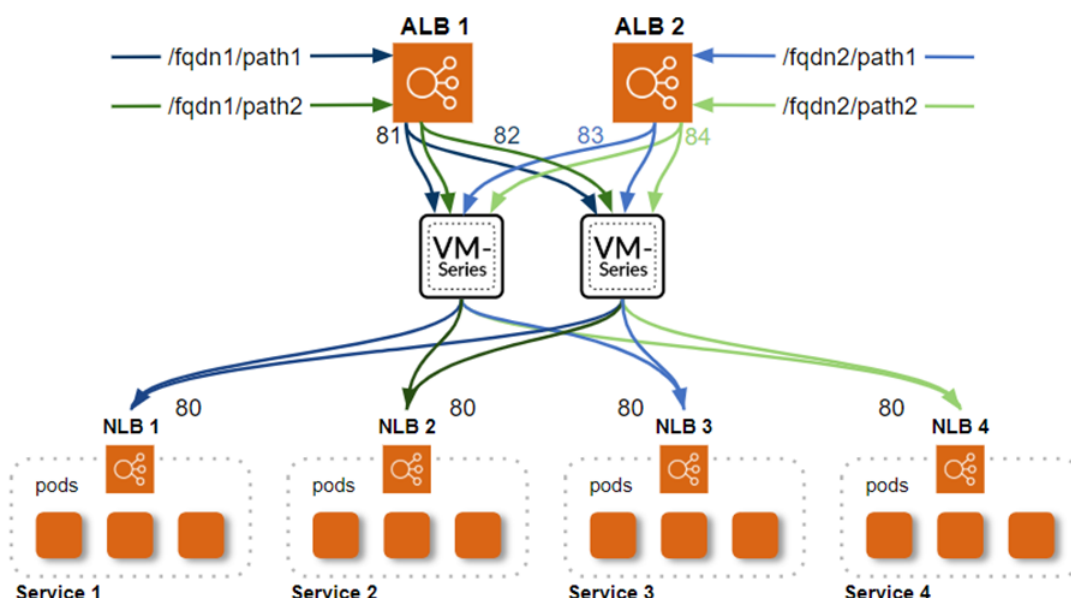
To manage EKS:

- AmazonEC2FullAccess
- AmazonLambdaFullAccess

- AmazonS3FullAccess
- AmazonSQSFullAccess
- AmazonVPCFullAccess
- AWSCloudFormationFullAccess
- AWSMarketplaceFullAccess
- ElasticLoadBalancingFullAccess
- IAMFullAccess
- AWS CLI—Most actions can be performed in the AWS console or the AWS CLI. If you prefer the CLI, [install or update the AWS CLI](#), ensuring that you have a supported version of Python.
- Kubernetes and kubectl—View the [available Amazon EKS versions](#) and [install kubectl](#) for your local OS. The version you install must be within one minor version of the EKS version (you will choose the Kubernetes version when you create the cluster).
- Amazon EC2 Key Pair—If you do not have a key pair, [create](#) one.

System Architecture

The following diagram illustrates a sample deployment that secures inbound traffic for Amazon EKS clusters—a load balancer sandwich.



The application load balancers (ALBs) face the internet. The VM-Series firewall set is sandwiched between the ALBs and the NLBs to provide inbound security to the cluster.

Inbound Security

To secure traffic without interrupting communication flows, the VM-Series firewall set is programmed with static routes that properly route traffic to the desired destination, and NAT rules to perform source and destination NATs on the inbound packets, ensuring that the initial traffic as well as the return traffic passes through the firewall set.

To register a service with the firewall, you must label your services with `panw-tg-port` and a port value. This label is applied when the service launches. You must also configure a target group for the ALB with the destination of the firewall set and a destination port matching the label on the service. When the traffic hits the firewall, the port that receives it tells the firewall which NAT rule to apply.

The applied NAT rule is a source and destination NAT. The source changes from the ALB to the firewall trust interface, ensuring that return traffic hits the firewall for inspection. The destination then changes from the firewall untrust interface to the service internal load balancer (ILB).

Outbound Security

To route the traffic from the trust to untrust interface, the template ensures the virtual router on the firewall has a default route pointed to untrust. Static routes are programmed for each cluster subnet so that traffic returning to the firewall is routed properly to its destination. To ensure return traffic passes through a single firewall, the outbound NAT rule does a source translation, redirecting the source from the Node IP address to the managed firewall's untrust interface. If you have a firewall set, the return traffic must go through only one of the firewalls in the set.

AWS Plugin on Panorama

The AWS Plugin on Panorama configures the firewall set according to the services deployed in a cluster. It creates inbound NAT rules for services, outbound NAT rules (one for each cluster subnet), and static routes for each cluster subnet.

The plugin uses the Kubernetes Python SDK to retrieve information related to services deployed in your cluster. The plugin queries for services that are labeled `panw-tg-port` and have been assigned a valid port value. The plugin uses the port to create an inbound NAT rule that is programmed on the VM-Series firewall. When traffic hits the firewall on that specified port, the firewall applies the inbound NAT rule for that port and routes the packet to its destination. For each service the plugin creates:

- An address object created with the FQDN of the service ILB.
- A service object created for the port specified in the label.
- An inbound NAT rule which creates source and destination NAT using the address object and service object just created.

The plugin is also responsible for adding configuration when a new cluster is added. The plugin uses the AWS API to retrieve cluster information, such as subnets, and VM-Series firewall information, such as the instance ID. The plugin uses the information to create one route per firewall per cluster subnet. For example, if there are two firewalls and three cluster subnets, the plugin creates six static routes.

Additionally, for every cluster subnet, the plugin creates an outbound NAT rule. The NAT rule is applied to any traffic originating from these subnets and it does a source NAT to change the source from the Node IP address to the firewall untrust interface.

In Panorama, the plugin provides visibility into discovered services and services that are currently protected.

Templates

You can download the templates from <https://github.com/PaloAltoNetworks/aws/eks>. The template files are as follows:

- `firewall-vpc-v1.0.template`
Creates a new VPC and deploys a VM-Series firewall set (greenfield deployment).
- `firewall-existing-vpc-v1.0.template`
Deploys a VM-Series firewall set in an existing VPC (brownfield deployment).
- `eks-cluster-v1.0.template`
 - Creates an EKS cluster.
 - Creates control plane security group.
 - Creates private cluster subnets.

-
- Creates route table associated with cluster subnets. The default route points to the internet gateway (IGW).
 - `eks-node-v1.0.template`
 - Adds node autoscaling group.
 - Adds node security group.
 - Adds entry to cluster control plane security group.

Secure an EKS Cluster with VM-Series Firewall and AWS Plugin on Panorama

To enable Panorama to monitor and secure Elastic Kubernetes Services, you must install the AWS 2.0.0 plugin and add your cluster service account credentials. You must also associate your cluster credentials with a Panorama device group and a template stack to which the firewall set protecting the cluster belongs.

- [Set Up Your Panorama Configuration](#)
- [Set Up Your AWS Bootstrap Bucket](#)
- [Deploy the Firewall Template on AWS](#)
- [Deploy the Cluster Stack](#)
- [Set Up Kubectl and Configure Your Cluster](#)
- [Add an EKS Cluster](#)
- [Configure Inbound Protection and Outbound Monitoring](#)
- [Configure the ALB](#)
- [Test the Outbound Workflow](#)

Set Up Your Panorama Configuration

Configure these Panorama elements before you use the templates to deploy firewalls.

STEP 1 | Add a template.

In Panorama, go to **Panorama > Templates** and **Add** a template.

STEP 2 | Add a stack.

Select **Panorama > Templates** and **Add Stack**. In the Templates pane, **Add** the template you created in [Step 1](#).

STEP 3 | Add a device group.

Select **Panorama > Device Groups** and **Add** a device group. You don't need to enter anything yet.

STEP 4 | Configure the DNS server to point to the AWS DNS server.

1. In the **Device** context, from the **Template** menu, select the template stack you created in [Step 2](#).
2. Select **Services** and click the Edit gear.
3. Under **Services** select **Servers** and add the IP address of the AWS DNS server: 169.254.169.253
4. Click **OK**.

STEP 5 | Configure untrust and trust interfaces, virtual routers, and zones to push to your managed firewalls.

1. Select **Network > Interfaces**, and from the **Template** menu, select the template you created in [Step 1](#) (not the template stack).
2. Select **Ethernet > Add Interface** to configure the untrust interface.
 1. **Slot**: Select Slot 1.
 2. **Interface Name**: Select ethernet1/1.
 3. **Interface Type**: Select Layer3.

4. To create the virtual router, select **Config** and under **Assign Interface To > Virtual Router** choose **New Virtual Router**. To name the router, prefix your template stack name with **VR-**. For example: **VR-<my-template-stack-name>**. The plugin searches for this specific router name.
Select **ECMP** and select **Enable**, then click **OK** to return to the **Config** tab.
5. Go to **Assign Interface > Security Zone**, choose **New Zone**, name the zone **untrust**, and click **OK**.
6. Select **IPV4 > DHCP Client**. Leave **Enable** and **Automatically create default route pointing to default gateway provided by server** checked. This sets the default route to point to the untrust interface.
7. Click **OK**.
3. Configure the trust interface.
 1. Select **Interfaces > Ethernet > Add Interface**.
 2. **Slot**: Select Slot 1.
 3. **Interface Name**: Select ethernet1/2.
 4. **Interface Type**: Select Layer3.
 5. Select **Config** and under **Assign Interface > Virtual Router** choose the router you just created (**VR-<template-stack-name>**).
 6. Select **Security Zone > New Zone**, name the zone **trust**, and click **OK**.
 7. Select **IPV4**, choose **DHCP Client**, and disable (uncheck) **Automatically create default route pointing to default gateway provided by server**.
 8. Click **OK**.
4. (Optional) Configure an allow all policy rule so you can test connectivity.
 1. Select **Policies** and from the **Device Group** menu, select the device group you made in step 3.
 2. Select **Security > Pre Rules** and **Add** a security policy rule with the following values.
 - **General**: Name the policy **allow-all**.
 - **Source**: Select **Any**.
 - **Destination**: Select **Any**.
 - **Service/URL Category**: Select **Any**.
 - Click **OK**.

STEP 6 | Commit your changes.

Set Up Your AWS Bootstrap Bucket

STEP 1 | Create an Amazon S3 bucket and [bootstrap package](#) as described in [Bootstrap the VM-Series Firewall on AWS](#).

STEP 2 | Download **eks.zip** from <https://github.com/PaloAltoNetworks/aws/eks>. In a local directory, extract the contents:

```
\cfg
  init-config.txt
\templates
panw-aws.zip
```

STEP 3 | Upload **panw-aws.zip** to your S3 bucket.

This file contains the AWS Lambda code for the templates.

STEP 4 | Edit the `init-config.txt` file to supply the values for `vm-auth-key`, `panorama-server`, `tplname`, and `dgname`. This sample configuration uses only one Panorama server, so `Panorama-server-2` remains undefined.

- `vm-auth-key`
 - If you have an auth-key, log on to your Panorama CLI and type:

```
request bootstrap vm-auth-key show
```

- If you don't have an auth-key, to [generate one](#) from the CLI, type:

```
request bootstrap vm-auth-key generate lifetime <1-8768>
```






- `panorama-server`: The IP address of Panorama server.
- `tplname`: The name of the [template stack](#) you created.
- `dgname`: The name of the [device group](#) you created.

Save the file.

STEP 5 | In your Amazon S3 bucket, add files to your bootstrap package as follows:

1. Upload the edited `init-config.txt` file to `\config`.
2. Upload `authcodes` to `\license`.

`authcodes` (no extension) is a text file you create that contains the VM auth code you received when you purchased your license. The `authcodes` file ensures bootstrapped firewalls are licensed.

Viewing 1 to 5				
<input type="checkbox"/>	Name ▾	Last modified ▾	Size ▾	Storage class ▾
<input type="checkbox"/>	 <code>config</code>	--	--	--
<input type="checkbox"/>	 <code>content</code>	--	--	--
<input type="checkbox"/>	 <code>license</code>	--	--	--
<input type="checkbox"/>	 <code>software</code>	--	--	--
<input type="checkbox"/>	 <code>panw-aws.zip</code>	Jul 31, 2019 4:49:58 PM GMT-0700	16.4 KB	Standard



You can leave the `\content` and `\software` directories empty.

Deploy the Firewall Template on AWS

This task uses the `firewall-new-vpc-v1.0.template` to configure a firewall stack.

STEP 1 | In AWS, ensure that you are working in a region that supports EKS. See the [region table](#).

STEP 2 | In AWS go to **AWS Services > Management & Governance > Cloud Formation > Stacks > Create stack**.

If you completed the steps in [Set Up Your AWS Bootstrap Bucket](#), your template is ready.

STEP 3 | Select template.

Select **Upload a template file** and upload `firewall-new-vpc-v1.0.template` from your local drive.
Click **Next**.

STEP 4 | Specify the stack name.

STEP 5 | Configure the VPC.

- **VPCName:** `panwVPC`
- **Number of AZs:** The number of available zones (AZs) in the region you chose for your S3 bucket (two, three, or four).
- **Select AZs:** From the list, select the available AZs for your region. Enter the number of AZs you specified in the previous step.
- **ELBType:** Choose either application or network. For this sample, choose application.

STEP 6 | Configure the VM-Series firewall instance.

- **AMIID of PANFW image:** Go to the [AMI list](#), copy the 9.0.1 AMI for the BYOL license, and paste it here.
- **Key pair:** Select an Amazon EC2 key pair.
- **SSH From:** Enter your public IP address. This address is added to the security group to allow SSH access. To find it, type [what's my IP](#) in a browser. If you are specifying a new VPC you must enter a valid CIDR range. For example, `x.x.x.x/x`.
- **NumberofFWs:** Minimum 2, maximum 6.

STEP 7 | Provide S3 Bucket details: Supply the name of your bucket from [Set Up Your AWS Bootstrap Bucket](#), which contains both firewall and Lambda code.

- **Bootstrap bucket for VM-Series firewalls:** Your [bucket](#) name.
- **S3 Bucket Name for Lambda Code:** Your [bucket](#) name.

STEP 8 | Specify the ELBName.

- Click **Next**.
- Click **Next**. Skip configuring stack options.
- Click **Next**.

STEP 9 | On the review page, scroll down and check **I acknowledge that AWS CloudFormation might create IAM resources** and click **Create stack**.

Creation can take up to ten minutes.

STEP 10 | In **CloudFormation > Stacks** confirm that the stack is active and the status is `CREATE_COMPLETE`.

STEP 11 | In Panorama, confirm the firewalls are up and connected to Panorama. This can take 20-30 minutes.

1. Select **Panorama > Device Groups**, and choose the device group you created. In the **Devices/Virtual System** column, verify that you have two IP addresses.
2. Select **Panorama > Templates**, select the template stack you created earlier and you also see the two IP addresses.

Deploy the Cluster Stack

This task uses `eks-cluster-v1.0.template` to set up the cluster subnets and the control plane.

STEP 1 | In AWS go to **AWS Services > Management & Governance > Cloud Formation > Stacks > Create stack**.

Your template is ready.

STEP 2 | In **Specify a template**, select **Upload a template file** and upload `eks-cluster-v1.0.template` from your local drive.

STEP 3 | Click **Next**.

STEP 4 | Name the cluster.

STEP 5 | Configure the cluster.

- **VPCID**: Select the VPC you just deployed.
- **Number of Cluster Subnets**: Choose at most one subnet per availability zone, based on your choice in the next step.
- **AZs for cluster subnets**: Two, three, or four, depending on the region.
- **Private Subnet IP Blocks**: Enter a CIDR for each cluster subnet. For example, 192.168.110.0/24, 192.168.111.0/24
- **Internet Gateway ID of VPC**: Enter the internet ID for the stack you just created.

To find the ID in AWS, go to **Services > VPC > Internet Gateways**, and copy the ID (igw-*) corresponding to the firewall stack you created when you [deployed the firewall templates](#).

- Click **Next**.
- Click **Next**.

STEP 6 | On the review page, scroll down and check **I acknowledge that AWS CloudFormation might create IAM resources** and click **Create**.

STEP 7 | In **CloudFormation > Stacks** confirm that the stack is active and the status is `CREATE_COMPLETE`.

STEP 8 | In the cluster you just deployed, note the API server endpoint and your subnets.

Set Up Kubectl and Configure Your Cluster

Set up Kubectl config file so you can use Kubectl commands locally to configure your cluster (when you do not have the AWS CLI installed).

If you prefer the AWS CLI, follow the instructions in [Configuring the AWS CLI](#).

STEP 1 | Set up your Kubectl configuration.

1. Go to [Create a kubeconfig for Amazon EKS](#) and follow the directions in “To create your kubeconfig file manually.”
 - Copy the sample `.config` file from “To use the AWS IAM Authenticator for Kubernetes.”
 - On the command line, open a text file.

```
vi ~/.kube/config-<YourClusterName>
```

2. Paste in the sample configuration.

3. Edit the sample config file.

- server: View your EKS Cluster and copy the API server endpoint (https://...) and paste it into your config file.
- certificate-authority-data: View your EKS Cluster and copy the certificate authority and paste it into your config file.
- args: Replace the cluster name variable with your cluster name.
- Save the file.

4. Set an environment variable for AWS authentication.

```
export AWS_ACCESS_KEY_ID=<your-access-key>
export AWS_SECRET_ACCESS_KEY_ID= <your-secret-access-key>
```

5. Apply the configuration.

```
export KUBECONFIG=$KUBECONFIG:~/.kube/config-<clusterName>
```

6. Print the current service.

```
kubectl get svc
```

STEP 2 | Create credentials.

- Create a service account for pan-user, and assign permissions.

```
kubectl create serviceaccount pan-user
```

- Create the cluster role cluster-read-all.

```
vi cluster_role.yaml
kubectl create -f cluster_role.yaml
```

- Associate the service account with pan-user.

```
vi cluster_role_binding.yaml
create -f cluster_role_binding.yaml
```

STEP 3 | Export service account credentials. The service account name <SA_NAME> is pan-user.

1. Get your service accounts:

```
MY_TOKEN=`kubectl get serviceaccounts <SA_NAME> -o
jsonpath='{.secrets[0].name}'`
```

The SA_NAME is typically pan-user.

2. Get your secret token:

```
kubectl get secret $MY_TOKEN -o json > <file_name.json>
```

Replace <file_name.json> with the name of your credential file.

Add an EKS Cluster

Add your configuration to the Panorama plugin for AWS. The configuration requires the access information from your account, which is typically governed by an IAM role. For each cluster you can either use an IAM role you [created](#) or [assume a role](#).

To perform this task you must know the access key ID and the secret access key.

STEP 1 | Select **Panorama > AWS > Setup > IAM Role**.

Supply values for Name, Access Key ID, Secret Access Key, and Confirm Secret Access Key.

STEP 2 | Select **Panorama > AWS > Setup > EKS Service Account** and click **Add**.

Enter your service account information:

- **Name:** Your choice. The plugin does not use the name.
- **Description:** Your choice.
- **API server address:** In EKS, this is the API server endpoint for your cluster.
- **EKS Credential:** Upload the JSON file you exported in step 3 of [Set Up Kubectl and Configure Your Cluster](#).

STEP 3 | Select **Panorama > EKS Clusters** and **add** a cluster.

Enter the following values:

- **Cluster Name:** The exact name of your EKS cluster.
- **(Optional) Description:** Your choice.
- **AWS FW Stack Name:** Name of CloudFormation stack in which you deployed your firewalls.
- **Region:** The region for your VPC and S3 bucket.
- **EKS Service Account:** Select the account you just configured.
- **IAM Role:** Choose the EKS role or the role you want to [assume](#).
- **Assume Role ARN:** Leave this field blank if you chose the EKS role. If you choose to assume a role, view the role, copy the Role ARN, and paste it here.
- **Device Group:** Choose the device group you created earlier.
- **Template Stack:** Choose the template stack you created earlier.
- **Enable:** Check this box.

Commit your changes.

STEP 4 | After you add the EKS cluster definition, verify plugin actions.

When you add a new cluster, the plugin creates a NAT rule for every cluster subnet that you created, and configures a static route for each firewall to tell it how to access each subnet and the cluster.

In this case there are two outbound NAT rules under in the device group. Select **Policies > Device Group > <your Device Group> > NAT** and view two outbound NAT rules static route for each firewall.



It may take up to two minutes for the result to populate.

Given one firewall and two subnets, there are 4 static routes.

Configure Inbound Protection and Outbound Monitoring

- [Configure Outbound Monitoring](#)
- [Deploy a Node Stack](#)

-
- [Associate the Nodes with the Cluster](#)
 - [Use the Guestbook Application to Verify the Deployment](#)
 - [View the Cluster in Panorama](#)

Configure Outbound Monitoring

STEP 1 | Add a public IP address to eth0 on the outbound firewall.

1. Go to **AWS > EC2 > Instances** and search for firewalls you deployed with the templates. If you used the template naming conventions, search for your VPC name.
2. Select one firewall to be the outbound firewall and attach a tag.
 - Select the tags group and click Add/Edit Tags.
 - (Optional) Edit the name to append **-outbound**. This is a convenience; the plugin does not look for it.
3. Select ENI eth0 and attach a public IP address.
 1. Copy the ENI ID and choose **Network & Security > Elastic IPs**.
 2. Select an available IP address and choose **Actions > Associate Address**.
 - Select Network Interface and paste the ENI that you copied.
 - From the drop menu, select the public IP address.
 - Click **Associate** and choose the network interface.
 - Return to **Instances**. The outbound instance has an IPv4 public IP address. View eth0.

STEP 2 | Change the default route of cluster subnets to point to the trust interface, in this case eth2.

1. Copy the ENI from the outbound firewall you tagged in step 1, go to **Amazon Container Services > Amazon EKS > Clusters**, and choose the cluster the template created.

Under **Networking**, select one of the subnets to open **Virtual Private Cloud > Subnets**. (There are two subnets and they both share the same routing table.)
2. Click the Route Table tab, and click the route table link to modify the route table.
3. Click **Routes** to see the default route 0.0.0.0/0 pointed to igw, causing all outbound traffic to go to the internet.
4. Click Edit routes and change the target to from the igw to the ENI of the trust interface of your outbound firewall (see the previous [step](#)).

Save the routes.

Deploy a Node Stack

STEP 1 | Go to CloudFormation > Stacks. Click Create Stack.

STEP 2 | Select Choose a template > Upload a template to Amazon S3.

1. Choose `eks-node-v1.0.template` and click Open, then Next.
2. Specify Details.
 - **Stack Name:** The exact name of the cluster stack you deployed.
 - Enter cluster information:
 - **Cluster Name:** Must match the cluster name exactly or it will not associate correctly.
 - **Cluster Stack Name:** Your choice.
 - **VPC ID:** Select your VPC.
 - Configure the node.
 - **Node Group Name:** Your choice.

- **SSH Key:** Select an SSH key (so that you can log into the nodes).
- **Node Image ID:** You need to specify the Amazon Machine Image when the node boots up and runs a bootstrap script to associate with the cluster.

Go to <https://docs.aws.amazon.com/eks/latest/userguide/getting-started-console.html>, find `NodeImageId`, and locate the AMI table.

Choose a Kubernetes version (we use 1.11).

Copy the Amazon EKS-optimized AMI for your region, then paste it into the **Node Image ID** field.

- **Node Instance Type:** t2.medium
- **Max Number of Nodes:** 1
- **Min Number of Nodes:** 1
- **Node Subnets:**

Return to **CloudFormation** and select the stack where you deployed your cluster. On the Outputs tab, choose the IDs for all subnets and copy them, one at a time, into the Node Subnets field.

- Click **Next**.
- Click **Create**.

On the Stacks page you see `CREATE_IN_PROGRESS` in yellow, then `CREATE_COMPLETE` in green.

- When your stack has finished creating, select it in the console and choose the Outputs tab.

Record the **NodeInstanceRole** for the node group that was created. You need this when you configure your Amazon EKS worker nodes.

Associate the Nodes with the Cluster

After the nodes come up, apply a config map telling the cluster the nodes are active and must be associated with the cluster.

STEP 1 | Return to <https://docs.aws.amazon.com/eks/latest/userguide/getting-started-console.html> and find “enable worker nodes to join your cluster”.

STEP 2 | Get the sample YAML file from AWS.

```
curl -o aws-auth-cm.yaml https://amazon-eks.s3-us-west-2.amazonaws.com/cloudformation/2019-02-11/aws-auth-cm.yaml
```

View the file with a text editor:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: aws-auth
  namespace: kube-system
data:
  mapRoles: |
    arn:aws:iam::#####:role/<nodeName>-NodeInstanceRole-CEMFVNZGL5XL
  - rolearn: <ARN of instance role (not instance profile)>
    username: system:node:{{EC2PrivateDNSName}}
  groups:
    - system:bootstrappers
    - system:nodes
```

STEP 3 | Return to CloudFormation > Stacks and choose the node you deployed. On the Outputs tab, the ARN Value is in the center column.

Copy the ARN Value.

STEP 4 | Return to `aws-auth-cm.yaml`, paste in the ARN, save, and close.

STEP 5 | Run the following Kubectl commands.

```
kubectl apply -f aws-auth-cm.yaml
```

You see a confirmation that the file is created:

```
configmap/aws-auth created
```

STEP 6 | View the progress as the node comes up.

```
kubectl get nodes --watch
```

The node starts to come up and the STATUS is NotReady. Eventually it switches to Ready. After the node is deployed, you can deploy a service to this node.

Use the Guestbook Application to Verify the Deployment

This task is optional. In it you adapt and deploy the Kubernetes tutorial [Create a Guestbook with Redis and PHP](#). The tutorial has five objectives, but you only need the first four:

1. [Set up a Redis master](#).
2. [Set up redis workers](#).
3. [Set up the guestbook web frontend](#).
4. [Visit the guestbook website](#).

Follow the [tutorial](#) to configure your environment and download the configuration files. The following task highlight exceptions or alternatives for AWS.

STEP 1 | Before you begin.

Follow the [Create a Guestbook with Redis and PHP](#) tutorial to configure your environment and download the configuration files.

- Ignore any gcloud instructions. You can use Kubectl or the AWS console.
You should already have Kubectl if you installed the [Requirements](#).
- Billing is beyond the scope of this task. This tutorial deploys a load balancer, which requires an external IP address. See the targetPort property, as described in [Set up the guestbook web frontend](#).
- [Download the configuration files](#) as directed.
- Instead of creating a GKE cluster, use the [EKS cluster](#) you created earlier.

STEP 2 | Follow the instructions in [Set up a Redis master](#) and [Set up Redis workers](#).

STEP 3 | [Set up the guestbook web frontend](#).

Follow the instructions up to [Expose frontend on an external IP address](#).

STEP 4 | Use a text editor to modify `frontend-service.yaml` as follows:

- service.beta.kubernetes/aws-load-balancer-type must be: `nlb`.

alb is not supported.

- service.beta.kubernetes.io/aws-load-balancer-internal must be: 0.0.0.0/0
- The type must be: **LoadBalancer**
- Add the label **panw-tg-port** and specify a port number—for example, 82. When traffic hits port 82, your firewall applies a NAT rule to forward the traffic to this service.

```
apiVersion: v1
kind: Service
metadata:
  name: frontend
  annotations:
    service.beta.kubernetes.io/aws-load-balancer-type: "nlb"
    service.beta.kubernetes.io/aws-load-balancer-internal: 0.0.0.0/0
  labels:
    app: guestbook
    tier: frontend
    panw-tg-port: "82"
spec:
  type: LoadBalancer
  ports:
    - port: 80
  selector:
    app: guestbook
    tier: frontend
```

STEP 5 | Deploy the service.

```
kubectl create -f frontend-service.yaml
```

You see a message when the service is created:

```
service/frontend created
```

STEP 6 | View the FQDN for all services.

```
kubectly get svc
```

View the Cluster in Panorama

Return to Panorama and select **Panorama > EKS Clusters**.

- Select the cluster you just deployed and in the **Action** column, select **Show Port Mapping**. For the frontend service, the protected column should show **True**.
- Under **Policies** look at the NAT rule. Choose your device group and select **NAT > Pre Rules**. The rule is fronted-82-inbound.
- To test that you can reach the service through the firewall, use:

```
curl http://<Firewall-untrust-IP-URL>:82
```

If the HTML prints, you are successful.

- Log in to the firewall CLI and type:

```
show session all
```

Look for "web-browsing" in the application field.

Configure the ALB

Send traffic to ALB then forward it to firewalls and services deployed in the cluster.

STEP 1 | Create a target group for every service that you are securing with managed firewalls. Every service for which you create a NAT rule for must have its own target group.

1. Create a target group.

Select **EC2 > Load Balancing > Target Groups > Create target group**.

Fill out the form as follows:

- **Target group name:** Enter a name.
- **Target type:** Instance
- **Protocol:** HTTP
- **Port:** Enter the port on the firewall that will receive traffic when this Target Group is applied.
- **VPC:** Select the VPC you just created.

2. Click **Create**.

STEP 2 | Edit the firewall auto scaling group.

Select **EC2 > Auto Scaling Group**.

- Select the auto scaling group you deployed previously.
- Under Target Groups, choose the target group you created in the previous step.
- Click **Save**. Wait a minute before continuing.

STEP 3 | Verify the targets are registered.

- Return to **Load Balancing > Target Groups**.
- Select your service, and on the **Targets** tab below, verify the targets are registered.

STEP 4 | Verify load balancing.

- Go to **EC2 > Load Balancing > Load Balancers**.
- Choose your load balancer (check your Cloud Formation template for the name you supplied).
- **(Optional)** Click + to add a rule and click **Insert Rule**. Add a condition and an action (forward to).
- Click the pencil to edit the default rule (if no rules match, choose forward to). Requests otherwise not routed Forward to frontend-demo-service. click UPDATE. If traffic hitting the ALB on port 80 does not meet any rules, it forwards traffic to front end-demo-service which should forward traffic to firewall on port 82. From there, it should go to the service
- View the Load Balancer description to get the DNS name for the ALB.

Issue a curl command to ping the DNS name.

```
curl http://rhVPC2-1219937001.us-west-2.elb-amazonws.com
```

You receive a response from the guestbook demo application, meaning the traffic entered successfully.

STEP 5 | Log in to the firewall CLI to confirm traffic is directed to the correct port.

```
show session all
```

View web-browsing traffic originating from the untrust network and directed to port 80 on the firewall.

You can also go to **Panorama > Monitor** and switch to the device context.

Test the Outbound Workflow

STEP 1 | To configure outbound traffic, change the cluster subnet default route to point to the trust interface on one of the firewalls in the firewall set. On that same firewall, add the public IP address to the untrust interface.

STEP 2 | Log in to the outbound firewall, and from the CLI, show session all.

You should see SSL traffic originating from the cluster subnets. View the node IP address, and notice that it sends outbound traffic to communicate with the master node.

STEP 3 | Deploy a pod that you can log into.

1. Deploy a pod.

```
kubectl create -f shell-demo.yaml
```

2. Log in to the demo.

```
kubectl exec -it shell-demo - /bin/bash
```

You are logged in.

STEP 4 | Use apt-get to test the session.

1. From the OS, type:

```
apt-get update
```

2. From the firewall, type:

```
show session all
```

On the bash shell you can see the apt-get update goes to the firewall and apt-get requests are registered.

STEP 5 | You can also curl something from the internet to demonstrate traffic is going in and out.

1. From the OS, type:

```
apt-get install curl  
curl http://www.google.com
```

2. From the firewall, type:

```
show session all
```

You see a request to google-base originating from your node IP address.

Known Issues

Review the following known issues with using the Panorama plugin for Amazon to secure EKS clusters:

Issue ID	Description
PLUG-2246	<p>When the following command is issued from the Panorama CLI, the output does not show <code>plugin_aws.log</code>:</p> <pre>tail follow yes mp-log plugin_api_server.log plugin_azure.log</pre>
PLUG-2253	<p>Delete node stack fails due to dependency on network interfaces. You must delete the stack elements manually. You must delete services before deleting the node stack.</p>
PLUG-2302	<p>After you enter cluster configuration information in the EKS Clusters tab and commit your changes, the plugin should display a status within 2 minutes.</p> <p>If the status is not shown, view <code>plugin_aws_eks_proc.log</code> and search for the following error message.</p> <pre>File "/2.0/python-lib/pan/plugin_api/error_handler.py", line 9, in handle_error raise Plugin_Server_Error(data_json['status'], data_json['reason'])</pre> <p>If you see this error, generate a new service account credential JSON file, upload it to Panorama, and commit your changes.</p> <p>In rare cases you might have to repeat this process.</p>

Get Help

The following topics provide information on where to find more about this release and how to request support:

- [Related Documentation](#)
- [Requesting Support](#)

Related Documentation

Refer to the following documentation on <https://docs.paloaltonetworks.com/> or search the documentation for more information on our products:

- **Panorama Administrator's Guide**—Provides the basic framework to quickly set up the Panorama™ virtual appliance or an M-Series appliance running version [8.1](#) or [9.0](#) for centralized administration of the Palo Alto Networks firewalls.
- **PAN-OS Administrator's Guide**—Provides the concepts and solutions to get the most out of your Palo Alto Networks next-generation firewalls. This includes taking you through the initial configuration and basic set up on your Palo Alto Networks firewalls for PAN-OS [8.1](#) or [9.0](#).

Requesting Support

For contacting support, for information on support programs, to manage your account or devices, or to open a support case, go to <https://support.paloaltonetworks.com>.

To provide feedback on the documentation, please write to us at: documentation@paloaltonetworks.com.

Contact Information

Corporate Headquarters:

Palo Alto Networks

3000 Tannery Way

Santa Clara, CA 95054

<https://www.paloaltonetworks.com/company/contact-support>

[Palo Alto Networks, Inc.](#)

www.paloaltonetworks.com

© 2019 Palo Alto Networks, Inc. Palo Alto Networks is a registered trademark of Palo Alto Networks. A list of our trademarks can be found at <https://www.paloaltonetworks.com/company/trademarks.html>. All other marks mentioned herein may be trademarks of their respective companies.

