CrowdStrike Falcon Proof of Value Guide for Amazon Elastic Kubernetes Service (EKS)

Introduction

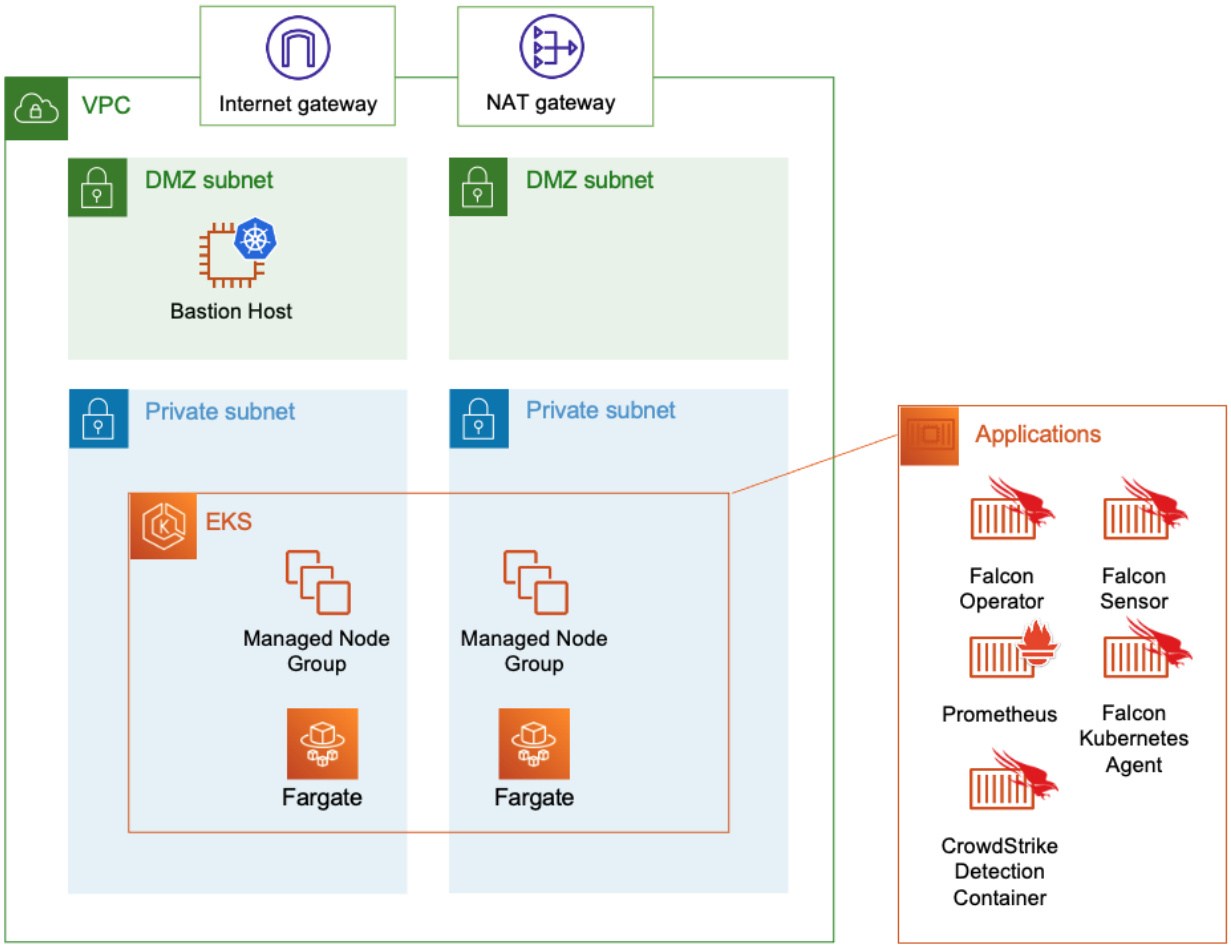
This Proof of Value (PoV) guide is designed to quickly deploy the core infrastructure needed to demonstrate the value of CrowdStrike Falcon in Amazon Elastic Kubernetes Service (EKS).

The solution allows the user to leverage AWS CloudFormation to easily deploy EKS on Amazon Elastic Compute Cloud (EC2) Managed Nodes or AWS Fargate with the Falcon Operator, Falcon Sensor and Kubernetes Protection Agent pre-installed as well as optional Detection Container to generate sample detections and Prometheus stack for monitoring.

How it Works

The EKS PoV infrastructure is deployed via a series of AWS CloudFormation templates. The user will reference a root template (entry.yaml) when creating the AWS CloudFormation Stack which will automatically launch other templates as nested stacks.

The following architectural diagram illustrates the resources deployed in an AWS account. Please note that some of the resources are optional. A full description about the architecture and resources deployed can be found in the Architecture Details and Appendix.



PoV Environment Architecture Overview

PoV Prerequisites

Amazon EC2 Key Pair

The optional Bastion Host deployment requires an existing EC2 Key Pair in the same AWS region where launching the CloudFormation Stacks. A key pair, consisting of a public key and a private key, is a set of security credentials that you use to prove your identity when connecting to an Amazon EC2 instance.

If you intend to utilize the Bastion Host, and a suitable EC2 Key Pair is not already available, please create a Key Pair prior to executing the CloudFormation Stack.

For more detailed instructions to create an EC2 Key Pair please see the official AWS Documentation at
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/create-key-pairs.html

Resource Requirements

The EKS Proof of Value Template will deploy the following resources:

- IAM Roles with various functions including CloudFormation Stack execution and EKS Access.
- CloudFormation Type Activations for both EKS Cluster and Helm
- VPC with 4 subnets, 3 route tables, IGW and NAT Gateway
- EKS Cluster and required Security Groups
- The option between an EKS Managed Node Group or Fargate Profiles.
- If EKS Managed Node Group is selected, instance details are as follows:
    - Operating System: Amazon Linux 2
    - Instance Type: m5.large
    - Group Size: 2 nodes
    - Volume Type and Size: GP3, 80gb
- Optional Prometheus Monitoring Stack installed via Helm
- Optional Bastion Host on EC2, instance details are as follows:
    - Operating System: Amazon Linux 2
    - Instance Type: t2.small
    - Volume Type and Size: GP2, 10gb

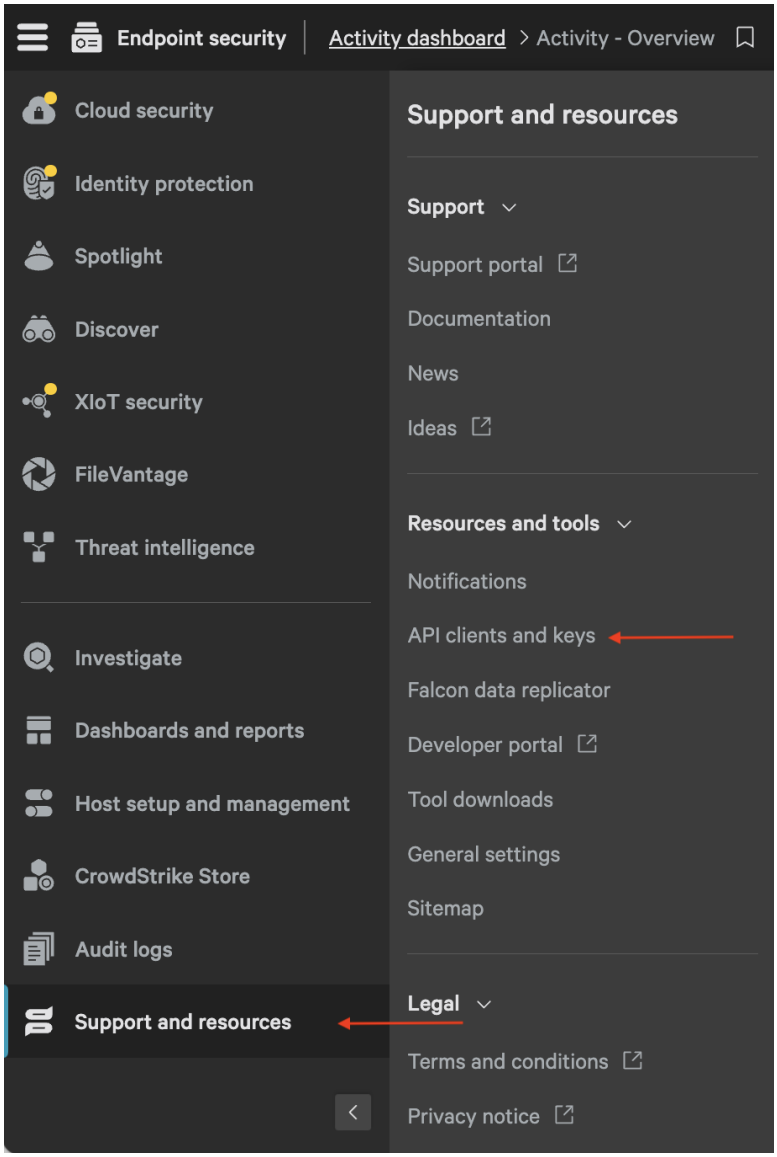
For more details about each resource deployed via the CloudFormation template, please see the Appendix.

CrowdStrike Falcon API Client

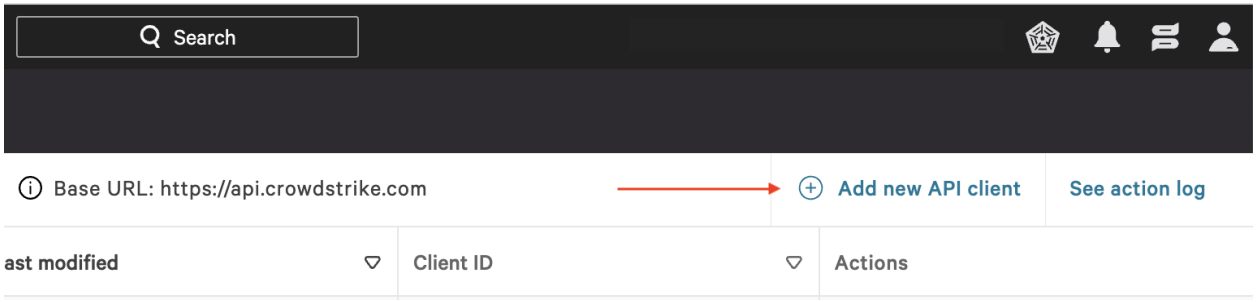
An API client is an identity mechanism that provides secure access to the CrowdStrike API. It contains credentials and scoped permissions to access specific API resources. You create an API client to generate your OAuth 2.0 client ID and secret credentials, which you exchange in the authentication flow for an access token that authorizes API requests.

Create API Client ID and Secret in Falcon Console

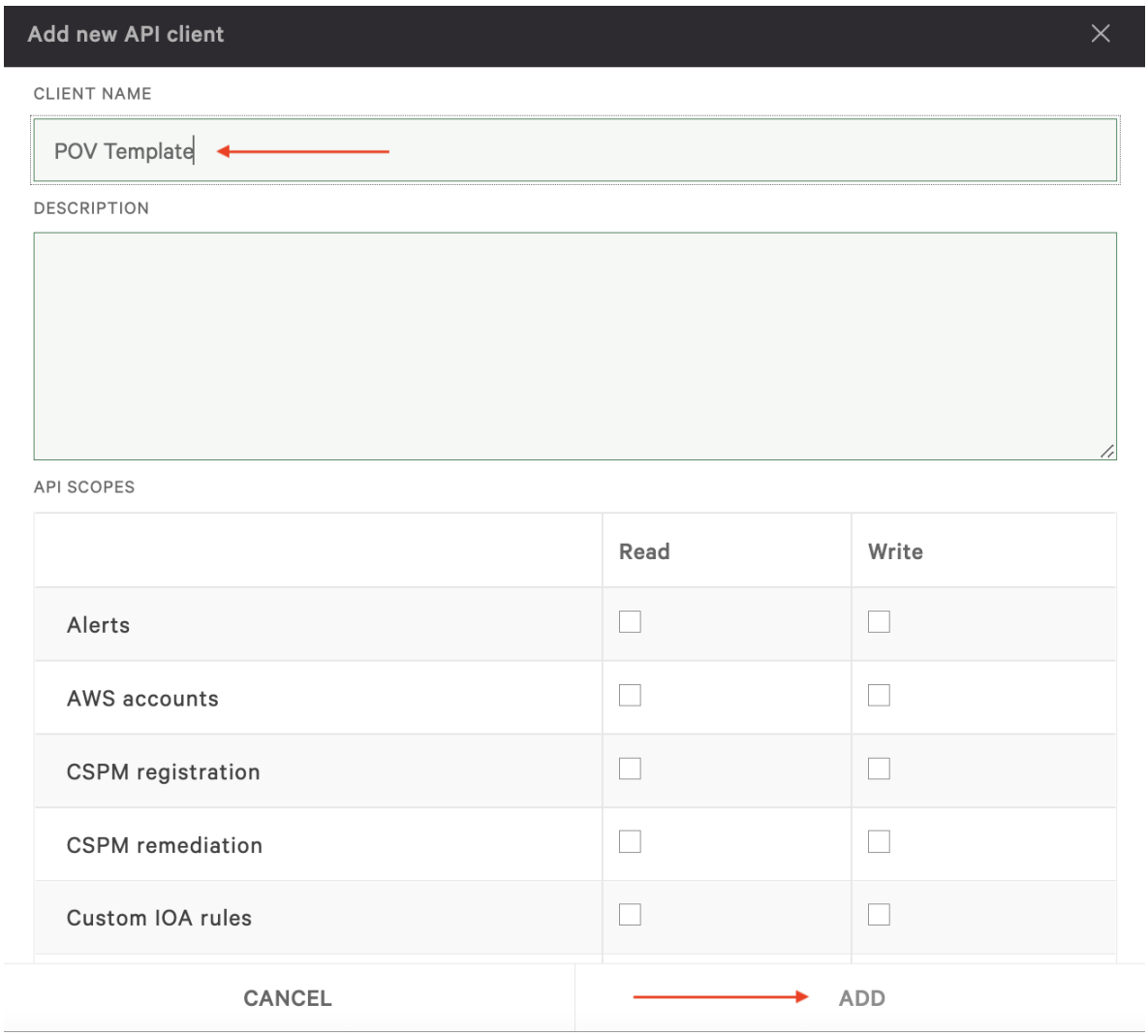
1. Go to Menu>Support and resources>API clients and keys



2. Select **Add new API client**

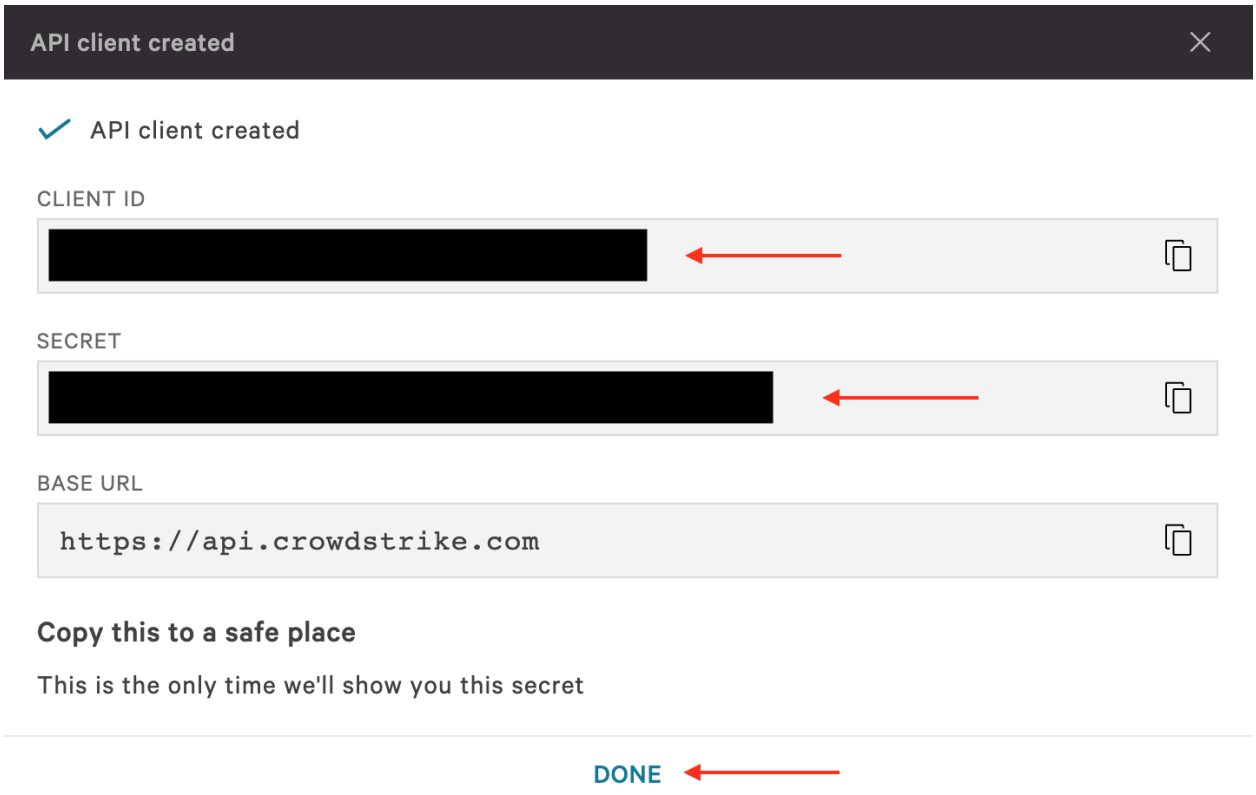


3. A new window will open. Name the new API Client. Use the table below to assign Permissions and then select ADD.



API Scope	Read	Write
Falcon Container Image	✅	✅
Falcon Images Download	✅	✅
Kubernetes Protection Agent	-	✅
Kubernetes Protection	✅	✅
Sensor Download	✅	-

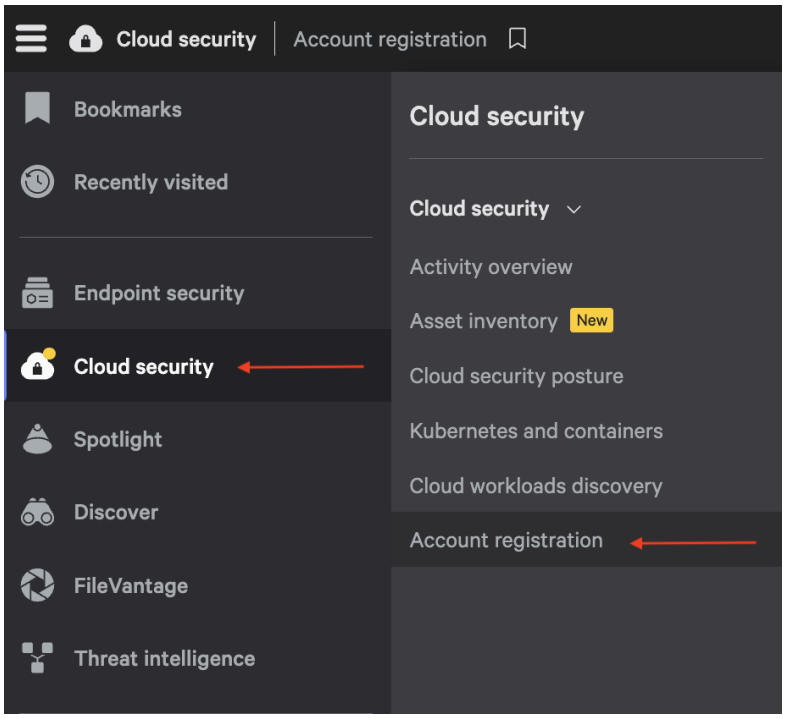
4. Record **Client ID** and **Secret** for later use then Select Done. Secret will only be displayed once and should be stored securely.



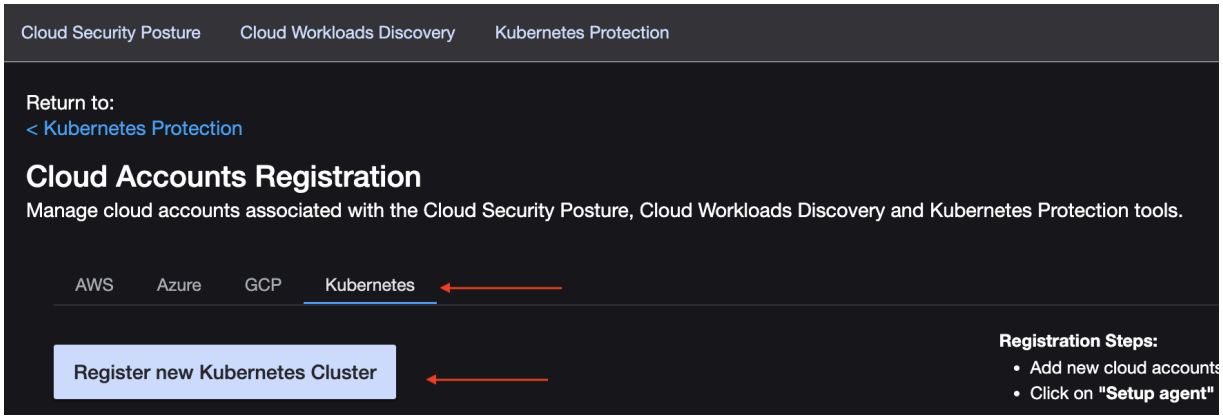
Kubernetes Protection Agent Configuration

The Kubernetes Protection Agent (KPA) configuration consists of specific settings to allow the KPA to be installed and to be registered with Falcon. For the EKS PoV especially the DockerAPIToken is required to download (pull) the KPA container image from the CrowdStrike container registry. The KPA configuration includes the DockerAPIToken, Falcon cloud region and Falcon CID and can be obtained from Falcon Console.

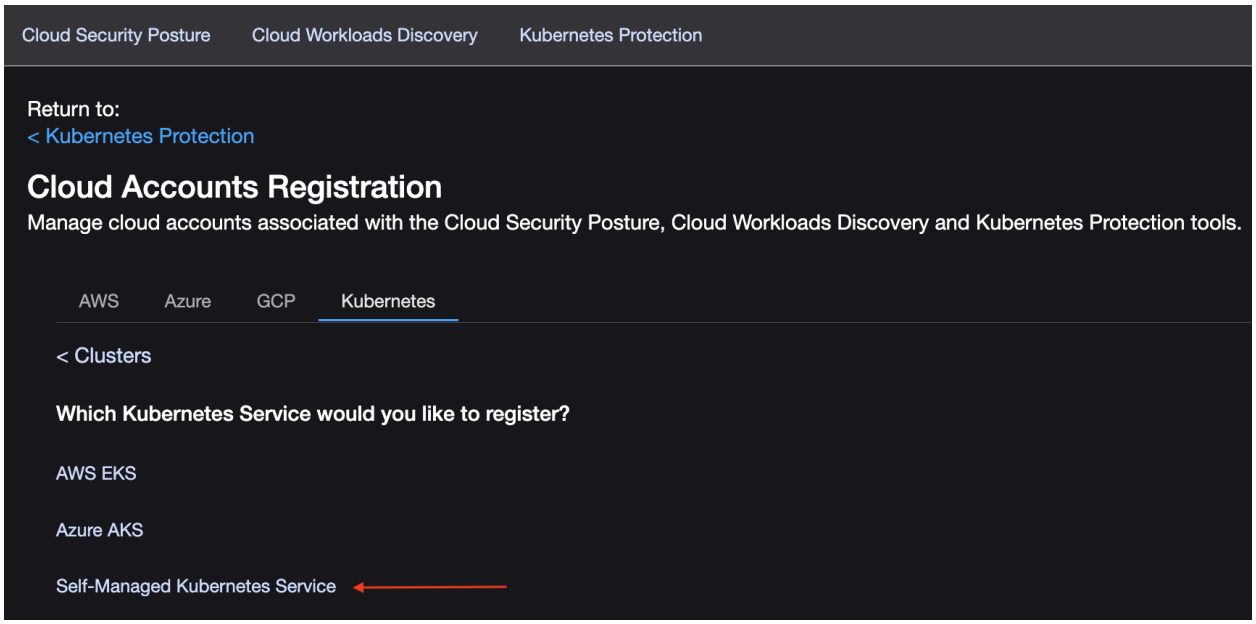
1. Go to Menu>Cloud Security>Account Registration



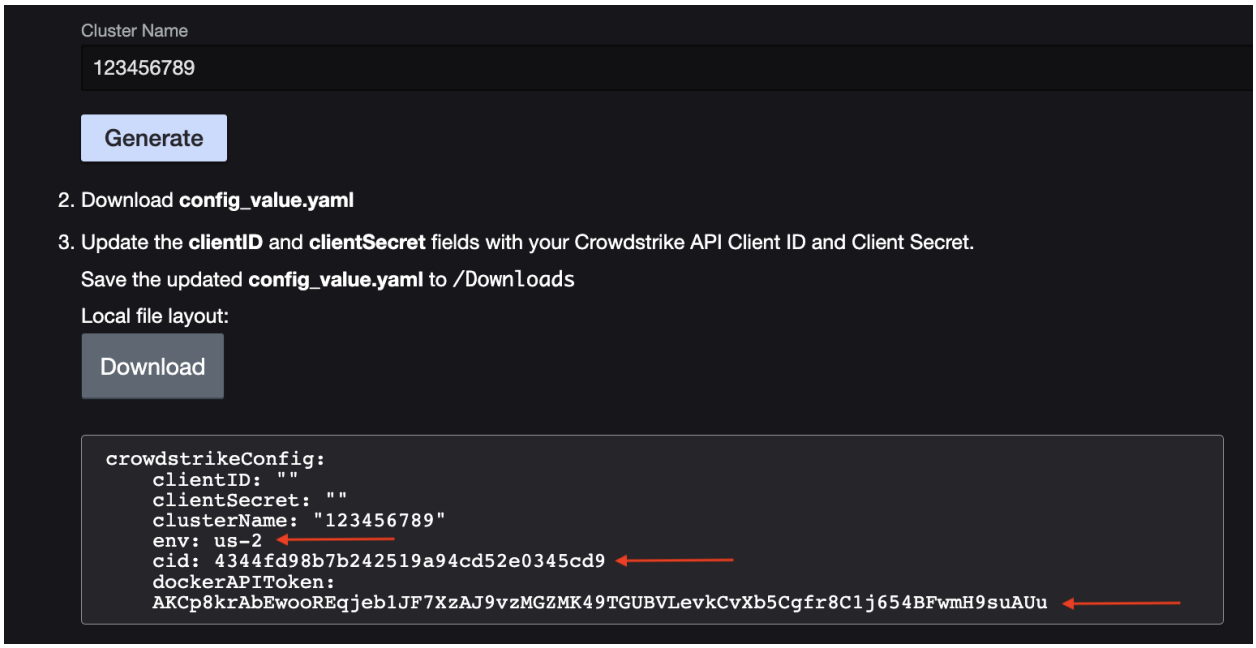
2. Select **Kubernetes** and then **Register new Kubernetes Cluster**



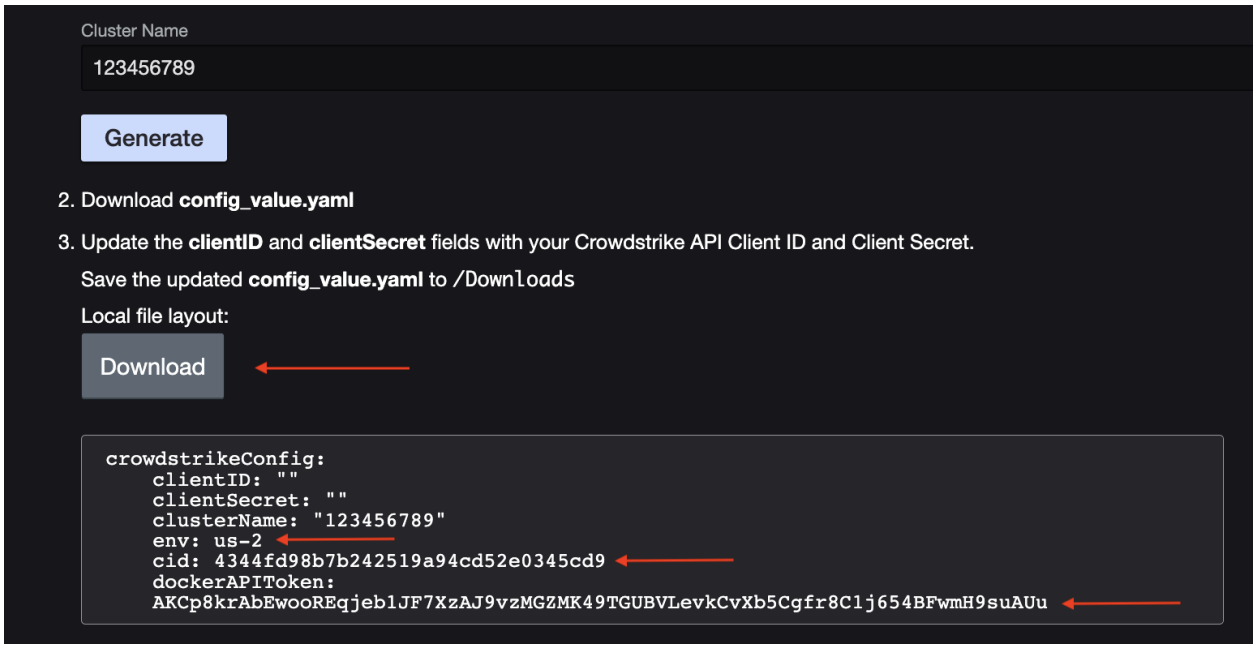
3. Select **Self-Managed Kubernetes Service**



4. Enter a **cluster name**. The cluster name is not important at this point and can be anything. Select **Generate** to generate the KPA configuration file.



5. Select **Download** to download the KPA configuration as .yaml file in order to save the Falcon cloud region (env), Falcon CID (cid) and DockerAPIToken (dockerAPIToken) for later use.



AWS IAM Permissions

The following AWS IAM Permissions are required to complete prerequisite actions, launch and delete the CloudFormation Stacks successfully.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "stackPermissions",
            "Effect": "Allow",
            "Action": [
                "cloudformation:ActivateType",
                "cloudformation:CreateChangeSet",
                "cloudformation:CreateStack",
                "cloudformation:DeactivateType",
                "cloudformation:DeleteChangeSet",
                "cloudformation:DeleteStack",
                "cloudformation:DeregisterType",
                "cloudformation:DescribeType",
                "cloudformation:ExecuteChangeSet",
                "cloudformation:RollbackStack",
                "cloudformation:UpdateStack",
```

```
                "ec2:CreateKeyPair",
                "ec2:DeleteKeyPair",
                "ec2:DescribeKeyPairs",
                "ec2:AllocateAddress",
                "ec2:AssociateAddress",
                "ec2:AssociateRouteTable",
                "ec2:AttachInternetGateway",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:CreateInternetGateway",
                "ec2:CreateLaunchTemplate",
                "ec2:CreateNatGateway",
                "ec2:CreateRoute",
                "ec2:CreateRouteTable",
                "ec2:CreateSecurityGroup",
                "ec2:CreateSubnet",
                "ec2:CreateTags",
                "ec2:CreateVpc",
                "ec2:DeleteInternetGateway",
                "ec2:DeleteLaunchTemplate",
                "ec2:DeleteNatGateway",
                "ec2:DeleteNetworkInterface",
                "ec2:DeleteRoute",
                "ec2:DeleteRouteTable",
                "ec2:DeleteSecurityGroup",
                "ec2:DeleteSubnet",
                "ec2:DeleteTags",
                "ec2:DeleteVpc",
                "ec2:DescribeAccountAttributes",
                "ec2:DescribeAddresses",
                "ec2:DescribeAvailabilityZones",
                "ec2:DescribeImages",
                "ec2:DescribeInstances",
                "ec2:DescribeInternetGateways",
                "ec2:DescribeLaunchTemplates",
                "ec2:DescribeLaunchTemplateVersions",
                "ec2:DescribeNatGateways",
                "ec2:DescribeRouteTables",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "ec2:DetachInternetGateway",
                "ec2:DisassociateAddress",
                "ec2:DisassociateRouteTable",
                "ec2:ModifySubnetAttribute",
                "ec2:ModifyVpcAttribute",
                "ec2:ReleaseAddress",
                "ec2:RevokeSecurityGroupIngress",
                "ec2:RunInstances",
                "ec2:TerminateInstances",
                "eks:CreateNodegroup",
                "eks:DeleteNodegroup",
```

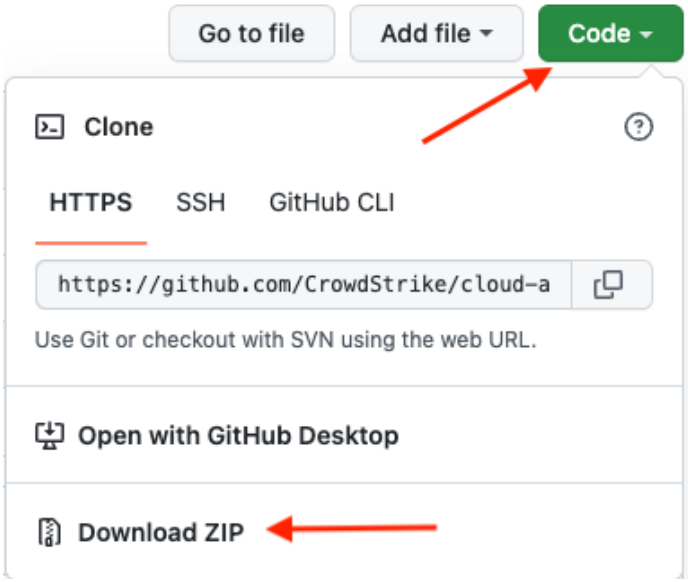
```
                "eks:DescribeCluster",
                "eks:DescribeFargateProfile",
                "eks:DescribeNodegroup",
                "eks:TagResource",
                "eks:UntagResource",
                "iam:AddRoleToInstanceProfile",
                "iam:AttachRolePolicy",
                "iam:CreateInstanceProfile",
                "iam:CreateOpenIDConnectProvider",
                "iam:CreatePolicy",
                "iam:CreateRole",
                "iam:CreateServiceLinkedRole",
                "iam:DeleteInstanceProfile",
                "iam:DeleteOpenIDConnectProvider",
                "iam:DeletePolicy",
                "iam:DeleteRole",
                "iam:DeleteRolePolicy",
                "iam:DetachRolePolicy",
                "iam:GetOpenIDConnectProvider",
                "iam:GetPolicy",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:ListAttachedRolePolicies",
                "iam:PassRole",
                "iam:PutRolePermissionsBoundary",
                "iam:PutRolePolicy",
                "iam:RemoveRoleFromInstanceProfile",
                "s3:CreateBucket",
                "s3:DeleteBucket",
                "s3:DeleteObject",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:PutObject",
                "sts:AssumeRole"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ssmParameter",
            "Effect": "Allow",
            "Action": [
                "ssm:PutParameter",
                "ssm:DeleteParameter",
                "ssm:GetParameters"
            ],
            "Resource": "arn:aws:ssm:*:*:parameter/*"
        }
    ]
}
```

How to Deploy

This chapter describes the steps for deploying the PoV infrastructure in an AWS account. All artifacts needed are stored in a public GitHub repository available at https://github.com/CrowdStrike/cloud-pov/tree/main/aws-eks-pov

Upload PoV Templates to an Amazon S3 Bucket

1) **Download PoV Templates from GitHub**
    a) The PoV templates are stored in a public GitHub repository available at https://github.com/CrowdStrike/cloud-pov/tree/main/aws-eks-pov
    b) Select the green **Code** button to download the templates to a local directory.
        i) Select **Download ZIP**



    c) Extract the files to a local folder

2) **Upload content extracted from the Zip file to an S3 bucket in your AWS Account**
    a) Access the AWS Account where the PoV environment will be deployed
        i) A user with the following permission is required to deploy the environment
            (1) See AWS IAM Permissions for permission requirements
    b) Upload files to an S3 bucket
        i) Locate the S3 service in AWS
        ii) Create or use an existing S3 bucket to store the files downloaded from GitHub
            (1) **Option 1** – Create a new S3 bucket - Name of S3 bucket is the only required parameter
                (a) Enter name
                (b) Select region

(c) Select “ACLs disabled” (default)
(d) Block all public access (default)
(e) Bucket Versioning Disabled(default)
(f) Tags (optional)
(g) Default encryption Disabled (default)
(h) Advanced setting
(i) Object Lock Disabled (default)

(2) **Option 2** – Use an existing S3 bucket

iii) Select the S3 bucket to upload the files.

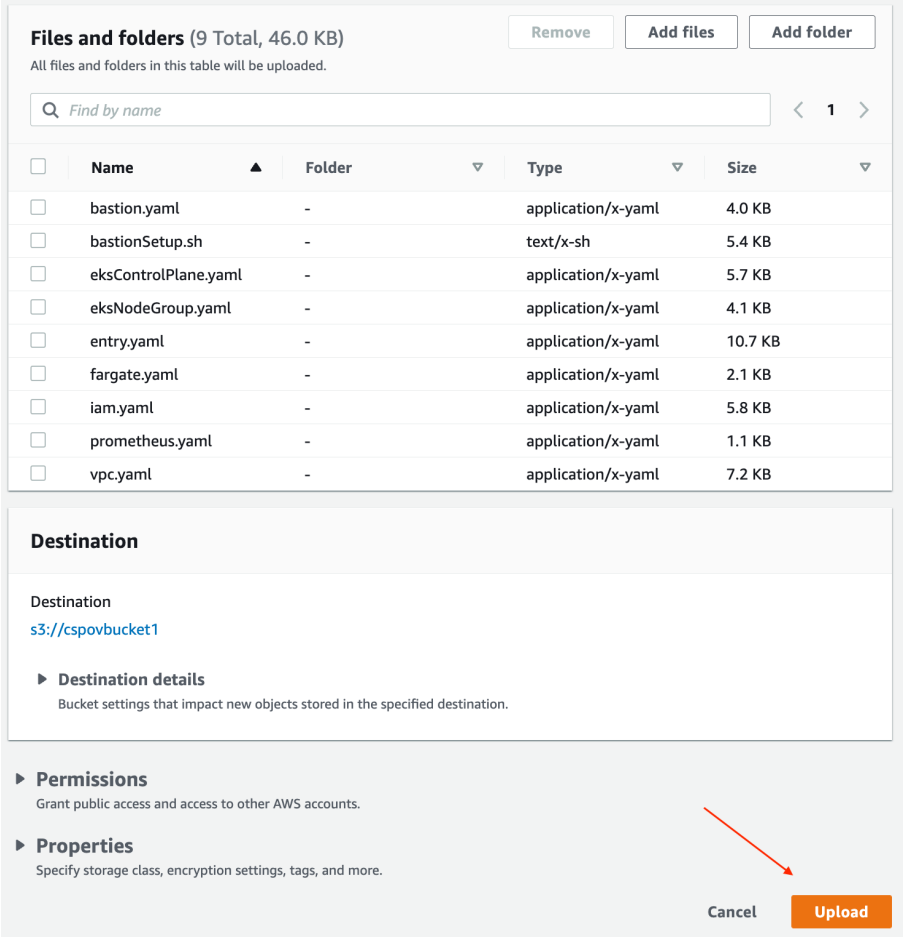
iv) Select “upload”

v) Select “Add Files”

vi) Select all files in the /templates folder of the package you downloaded from GitHub. Files should be uploaded to the root of the bucket.

(1) Select Open

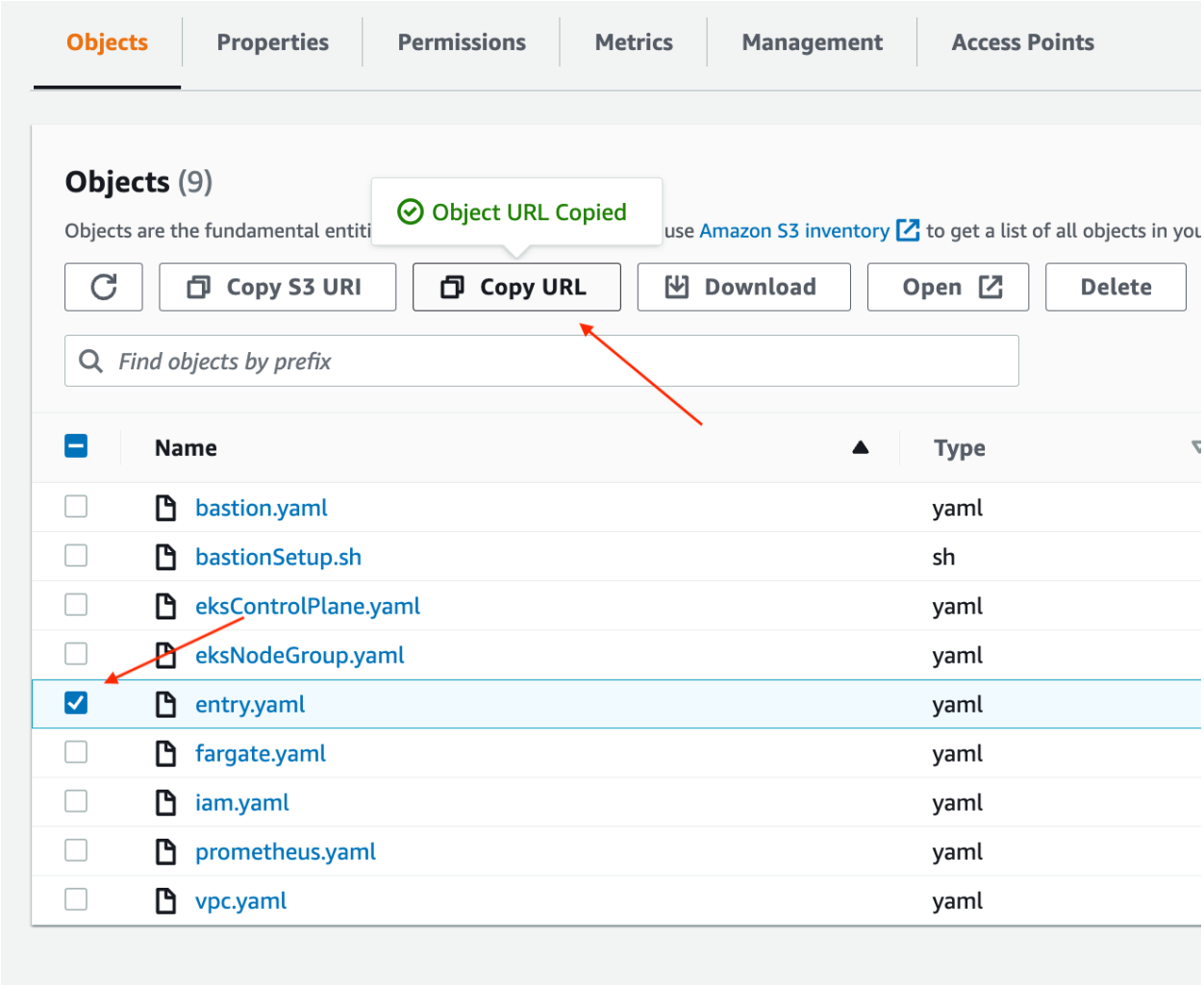
vii) Select Upload



viii) Select “Close” after the upload is complete returning to the bucket which now includes the folder with the templates

Create AWS CloudFormation Stack to build PoV Environment

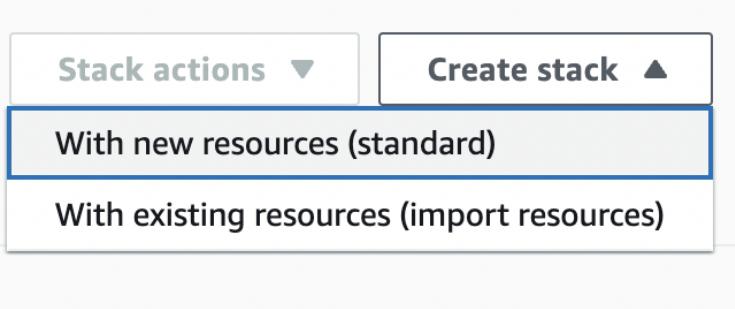
1) **Building the PoV Environment**
    a) Browse to the S3 Service in AWS if not currently there
    b) Find the bucket where the PoV templates are stored.
        i) All previously uploaded files should be visible at the root of the S3 bucket
    c) Sort the files by alphabetical and locate the "**entry.yaml"**
        i) Check the box next to the entry.yaml file
        ii) Select Copy URL



2) Create AWS CloudFormation Stack that deploys the PoV environment

a) Browse to the CloudFormation Service in AWS

i) Select **Create Stack - With new resources (standard)**



ii) Create stack

(1) Prerequisites - Prepare templates

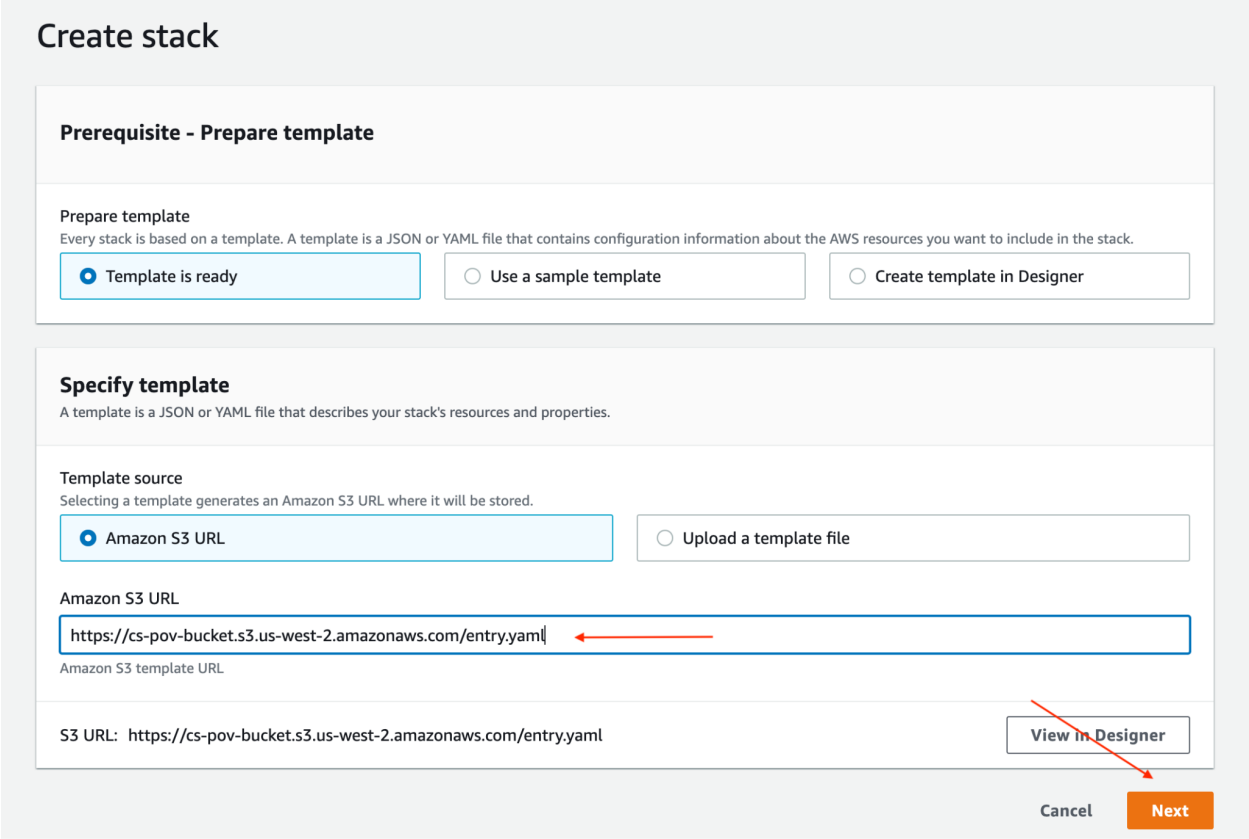
(a) Leave Prepare template as default - Template is ready

(2) Specify template

(a) Leave Amazon S3 URL selected

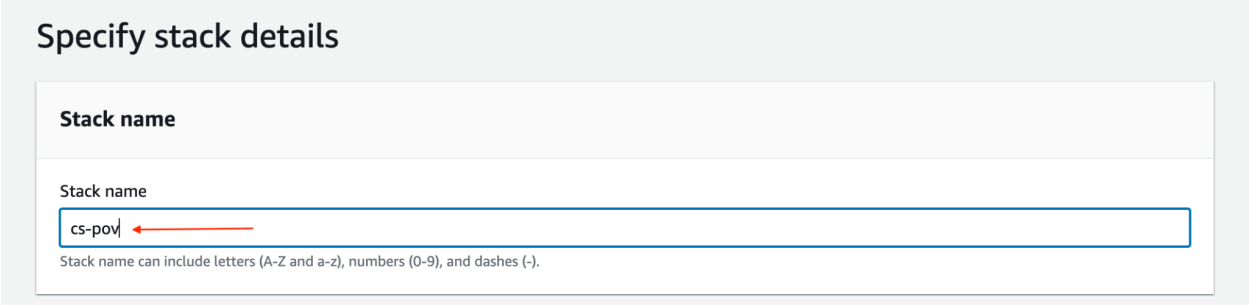
(b) Amazon S3 URL - paste in entry.yaml URL copied in previous section

(3) Select Next



(a) Specify stack details

(i) Enter **Stack name**: blank (default)



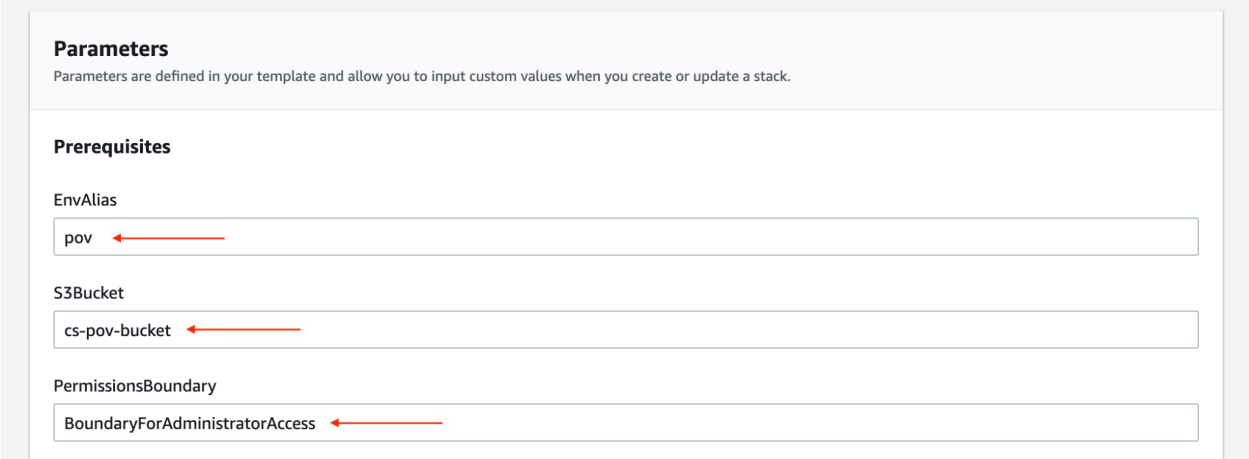
iii) Parameters - *click on parameters to get detailed information about each*

(1) Prerequisites

(a) **EnvAlias**: pov (default) - Update to preferred value

(b) **S3 Bucket**: blank (default)- Update to S3 bucket location where uploaded files where placed

(c) **PermissionsBoundary**: blank (default) - A permissions boundary is an advanced feature in AWS IAM for using a managed policy to set the maximum permissions that an identity-based policy can grant to an IAM entity. If a Permissions Boundary is in place, it must be included for IAM Roles to successfully launch. For additional information - link



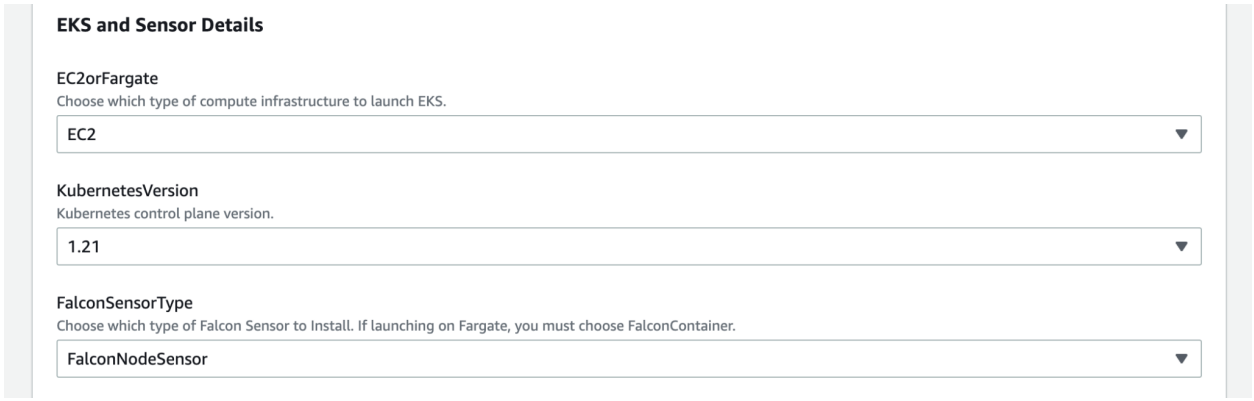
iv) EKS and Sensor Details

(1) EKS and Sensor Details

(a) **EC2orFargate**: EC2 (default)

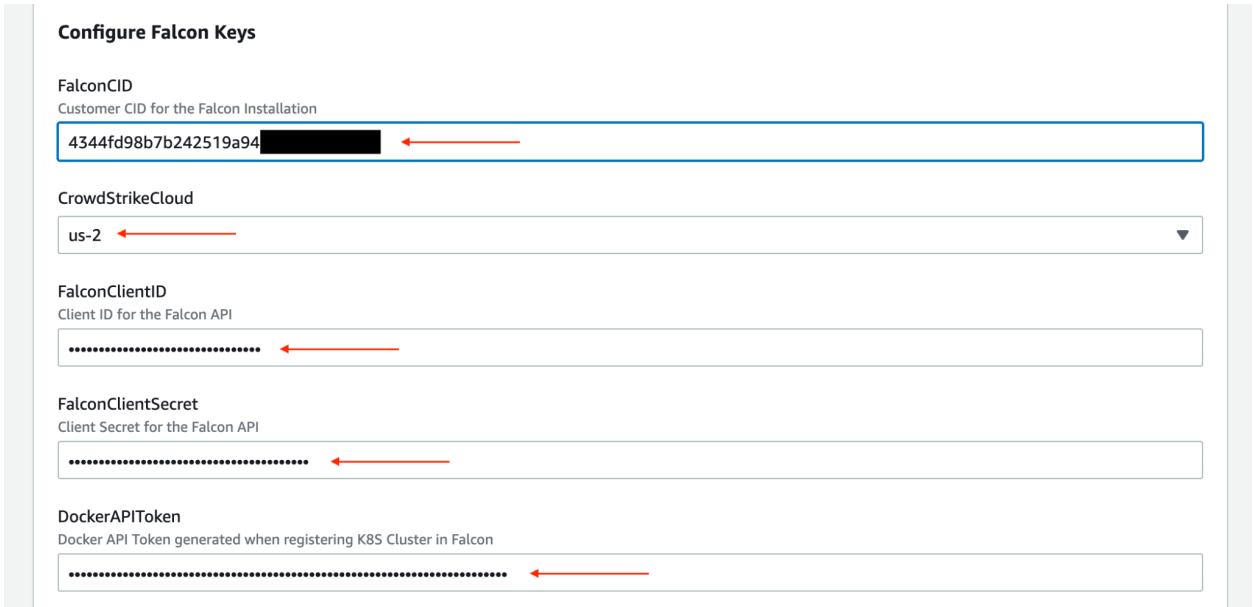
(b) **KubernetesVersion**: 1.21 (default)

(c) **FalconSensorType**: FalconNodeSensor (default)



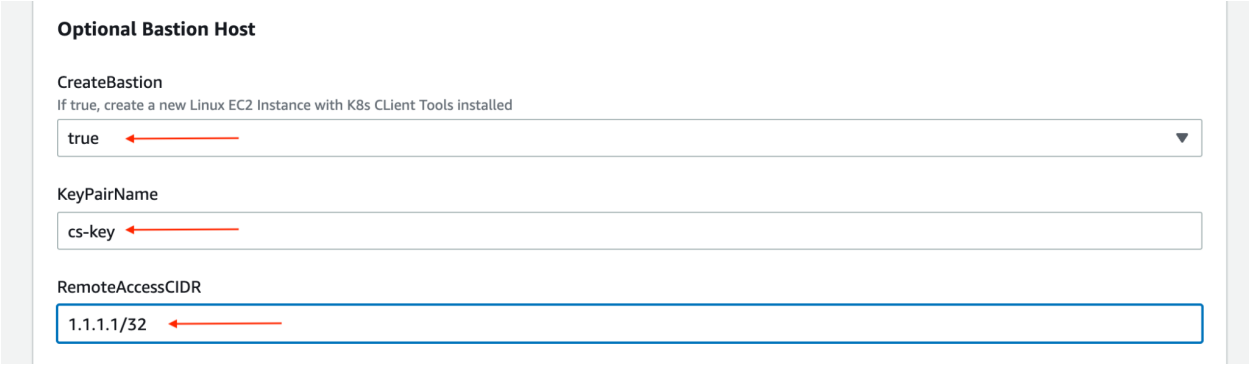
(2) Create New VPC
   (a) **NewVPCCIDR**: 10.1.0.0/24 (default)

(3) Configure Falcon Keys - See prerequisites
   (a) **FalconCID**: Enter FalconCID - All lower without Checksum
   (b) **CrowdStrikeCloud**: us-1 (default)
   (c) **FalconClientID**: Enter API CID
   (d) **FalconClientSecret**: Enter Client Secret associated with the Falcon API CID - Enter CID
   (e) **DockerAPIToken**: Enter DockerAPIToken



(4) Optional Monitoring Stack - Update as required
    (a) **InstallPrometheus**: false (default)

(5) Optional Bastion Host - Update as required
    (a) **CreateBastion**: true (default)
    (b) **KeyPairName**: keyname (default) - Use the Amazon EC2 key pair name created earlier (see Amazon EC2 key pair (Bastion Host)
    (c) **RemoteAccessCIDR**: 1.1.1.1/32 (default) - Change to External Public IP Address of the host running the PoV and keep the 32-bit mask to allow SSH access from a single IP address



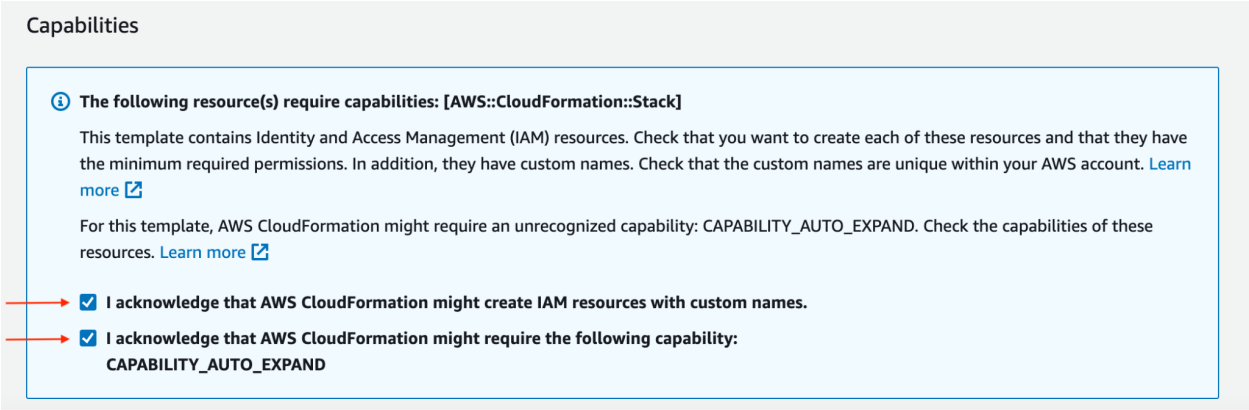
(6) Optional Detection Container - Update as required
    (a) **InstallDectectionContainer**: False (default)

(7) Select Next

v) Configure stack options - Following options can all be left at default settings
    (1) **Tags** - blank (default)
    (2) **Permissions** - blank (default)
    (3) **Stack failure options** - blank (default)

(4) Select Next

vi) Review and check boxes in Capabilities Box



vii) Select Create stack to start the PoV environment deployment

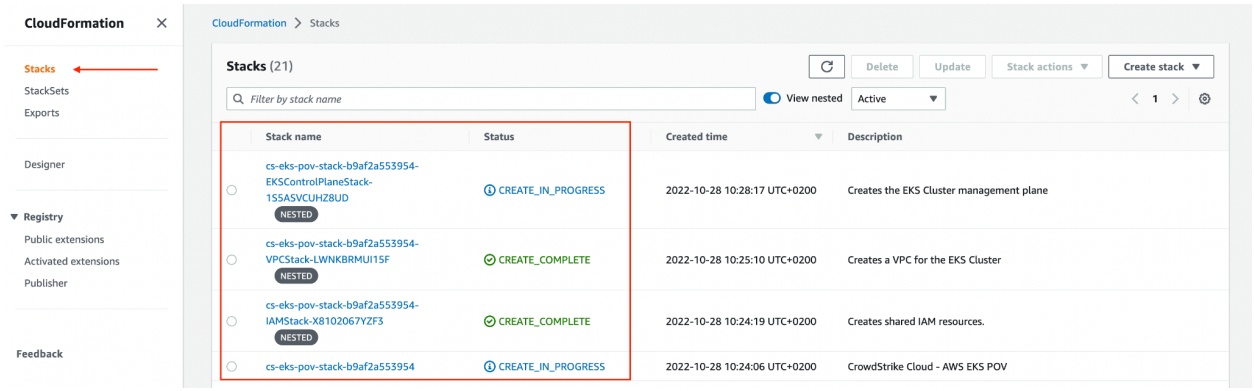
Please note: deployment of the PoV environment will take 20-30 minutes to complete, depending on the features selected. Please see the next section for steps to validate a successful deployment.

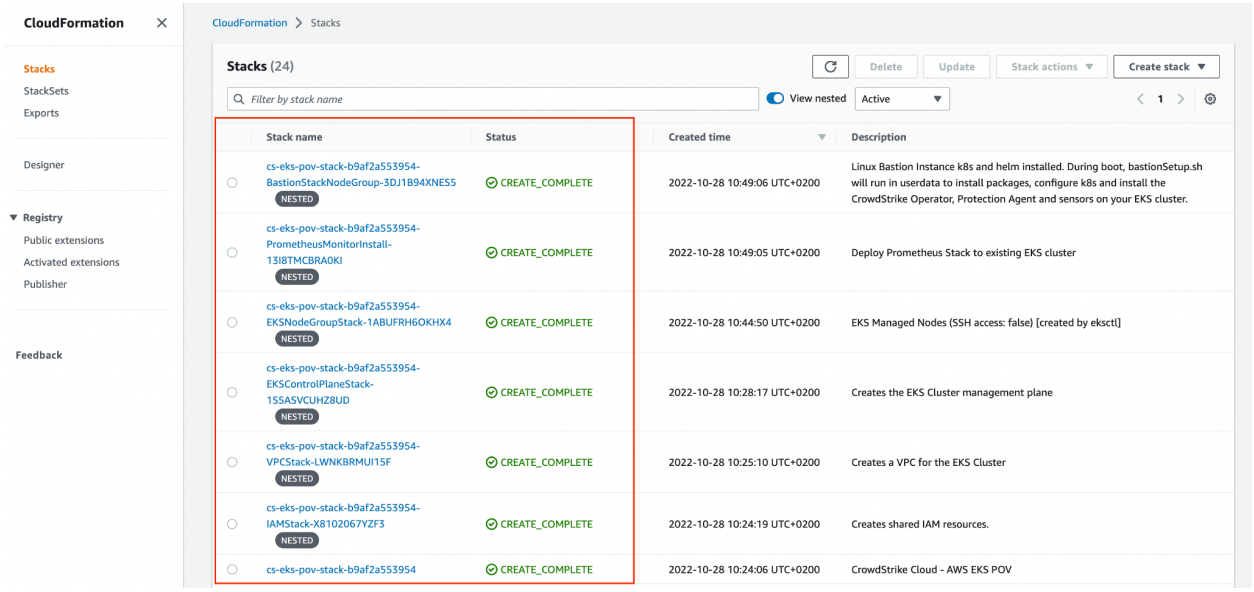
Validate Successful PoV Deployment

The successful deployment of the PoV environment can be tracked using the AWS and Falcon Console. The following shows the validation steps for an EKS EC2 deployment.

Validate AWS Deployment and Falcon Integration

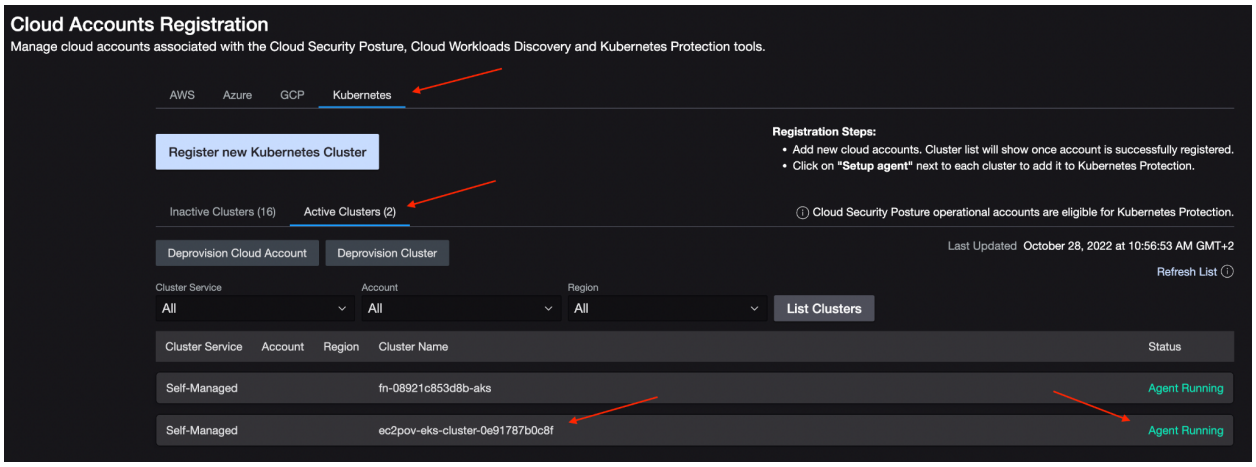
1. To validate the successful deployment of the AWS CloudFormation templates, open the AWS Console and ensure the right AWS account and location is selected. Open the CloudFormation management page, select **Stacks** on the left side and check the status of the CloudFormation Stacks, as shown in the following screenshots. Please note that the Stacks can look different depending on the features selected for deployment.



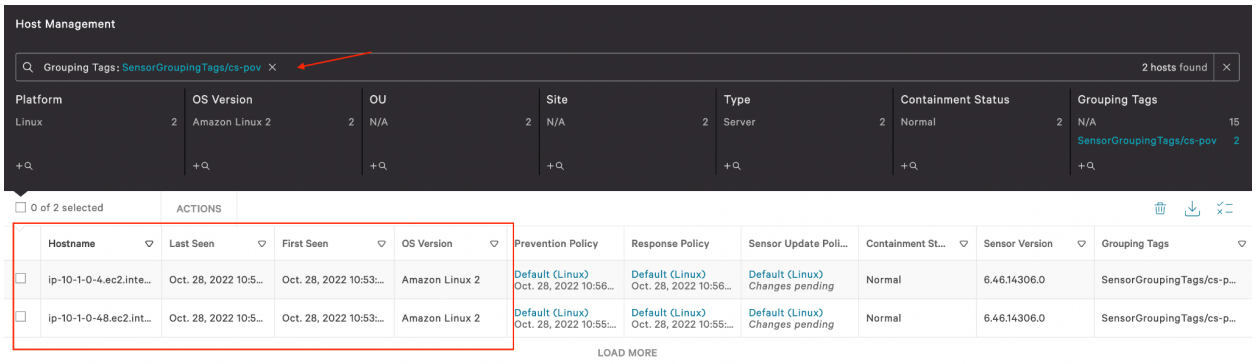


2. To validate the successful deployment of Falcon components and integration into Falcon, open the Falcon Console.

    a. Kubernetes Protection Agent (KPA) deployment can be validated by navigating to Menu → Cloud security → Account registration in Falcon Console. On this page select **Kubernetes**, select **Active Clusters** and validate that the EKS cluster is visible and has the Status of *Agent Running*.



b. To validate deployment of the Falcon Sensor deployment navigate to Menu → Host setup and management → Host management. This list contains all Falcon managed hosts in the environment. To filter for the PoV environment hosts the easiest way is to filter for the grouping tag ***SensorGroupingTags/cs-pov.***



Connect to EKS Cluster

Connect via Local Machine

Please Note: If you choose to launch the Prometheus Monitoring Stack, this option is required to connect to Grafana and query EKS Cluster performance data from Prometheus.

Prerequisites

- Install AWS CLI on your machine. For more information please see https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-getting-started.html
- Create and download your IAM User Access Keys. For more information see https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_access-keys.html

3. **Configure the AWS CLI**
    a. For Mac users, open Terminal on your machine.

i. For Windows users, open Command Prompt.

b. Configure AWS CLI with your access key.

```
$ aws configure
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]:bPxRfiCYEXAMPLEKEY
Default region name [None]: [region-code]
Default output format [None]: (leave blank and hit enter to finish)
```

4. **Update Kube Config**
    a. The IAM user who launched the Template has direct access to the EKS Cluster without switching roles. To set up kubeconfig run the following command.

```
aws eks update-kubeconfig --region [region-code] --name
[cluster-name]
```

    b. Other IAM users may switch to the EKSAccessRole to access the EKS Cluster. To set up kubeconfig as this type of user, just add the --role-arn argument

```
aws eks update-kubeconfig --region [region-code] --name
[cluster-name] --role-arn [role-arn]
```

Note: to get the role-arn please visit the AWS Console, navigate to IAM and locate the EKSAccessRole

5. **Test Access and Verify Deployment**
    a. Once kubeconfig is configured on your local machine, execute the following two kubectl commands to validate deployment of the components.

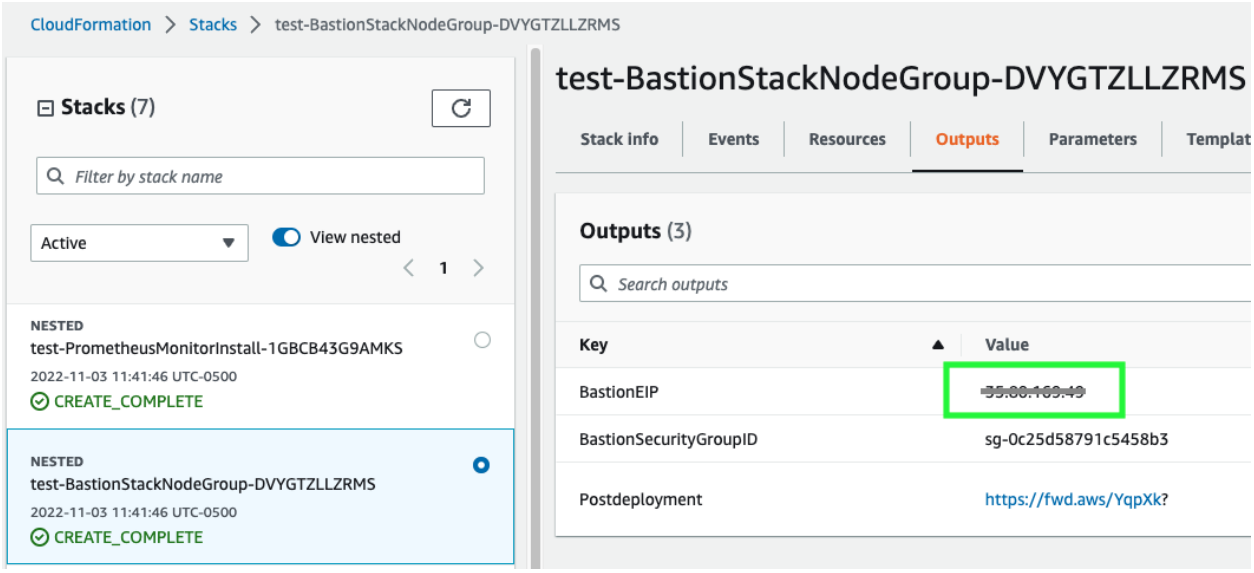
```
kubectl get nodes
Kubectl get pods --all-namespaces
```

```
rpayne@ML-X9QQDJ561G ~ % kubectl get nodes
NAME                                           STATUS   ROLES    AGE     VERSION
ip-10-1-0-17.us-west-2.compute.internal   Ready    <none>   6m10s   v1.21.12-eks-5308cf7
ip-10-1-0-50.us-west-2.compute.internal   Ready    <none>   6m10s   v1.21.12-eks-5308cf7
rpayne@ML-X9QQDJ561G ~ % kubectl get pods -A
NAMESPACE                       NAME                                                     READY   STATUS    RESTARTS   AGE
falcon-kubernetes-protection    kpagent-cs-k8s-protection-agent-7ff897d475-lw76h         1/1     Running   0          56s
falcon-operator                 falcon-operator-controller-manager-64b9f7c89-wqjn8       1/1     Running   0          62s
falcon-system                   falcon-node-sensor-jbwxl                                 1/1     Running   0          40s
falcon-system                   falcon-node-sensor-m5wb4                                 1/1     Running   0          40s
kube-system                     aws-node-6pt9r                                           1/1     Running   0          6m20s
kube-system                     aws-node-nm965                                           1/1     Running   0          6m20s
kube-system                     coredns-85d5b4454c-25dhw                                 1/1     Running   0          17m
kube-system                     coredns-85d5b4454c-xx7rp                                 1/1     Running   0          17m
kube-system                     kube-proxy-29lb4                                         1/1     Running   0          6m20s
kube-system                     kube-proxy-fbxf9                                         1/1     Running   0          6m20s
prometheus                      alertmanager-prometheus-kube-prometheus-alertmanager-0   2/2     Running   1          3m44s
prometheus                      prometheus-grafana-798cf79d7c-6g22d                      3/3     Running   0          3m50s
prometheus                      prometheus-kube-prometheus-operator-c5cb6797d-f2dht      1/1     Running   0          3m50s
prometheus                      prometheus-kube-state-metrics-5646db858d-m8klm           1/1     Running   0          3m50s
prometheus                      prometheus-prometheus-kube-prometheus-prometheus-0       2/2     Running   0          3m44s
prometheus                      prometheus-prometheus-node-exporter-pkg6c                1/1     Running   0          3m50s
prometheus                      prometheus-prometheus-node-exporter-sn5ml                1/1     Running   0          3m50s
rpayne@ML-X9QQDJ561G ~ %
```

Connect via Bastion Host

Once all CloudFormation Stacks have completed, and the Bastion Host is running, you are ready to connect to the EKS Cluster and verify the deployments.

1. Locat Bastion Host EIP
    a. Navigate to the Bastion Host CloudFormation Stack Outputs, and retrieve the value for **BastionEIP**



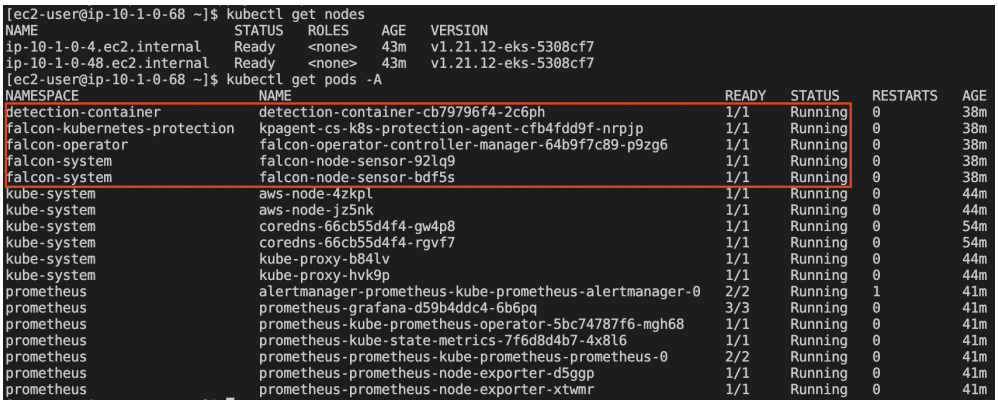
2. Connect to the Bastion Host
    a. SSH to the Bastion Host using the private key generated earlier and where user is *ec2-user* and hostname is the public IP or hostname of the Bastion Host, e.g.

```
ssh -i my-ec2-keypair.pem ec2-user@12.34.56.78
```

3. Verify the Kubernetes Deployments
    a. Once connected via SSH to the Bastion Host, execute the following two kubectl commands to validate deployment of the components.

```
kubectl get nodes
Kubectl get pods --all-namespaces
```

An example output is shown in the following screenshot.

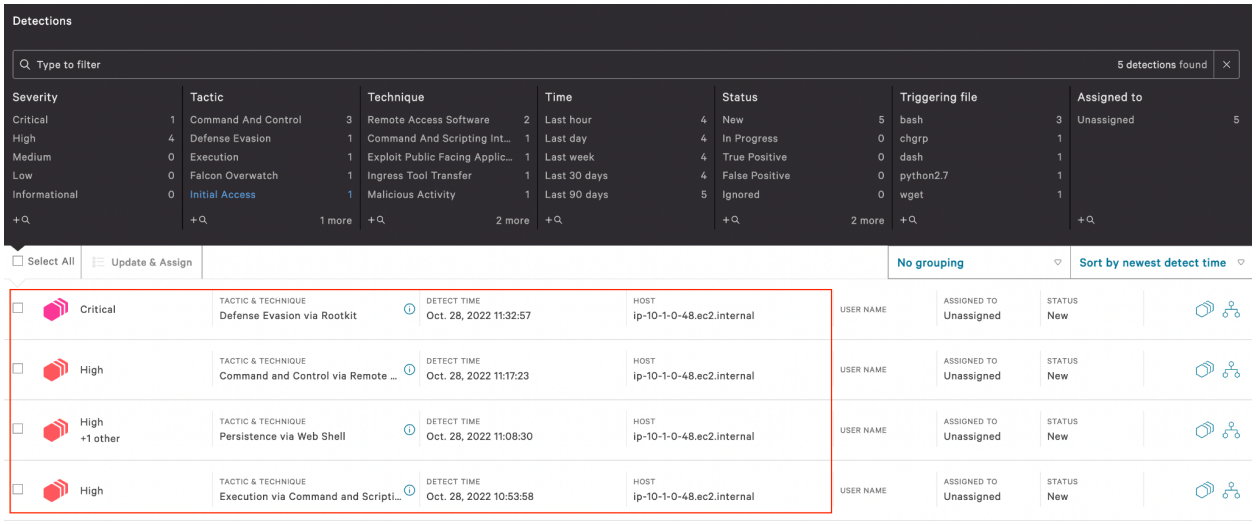


```
[ec2-user@ip-10-1-0-68 ~]$ kubectl get nodes
NAME                         STATUS   ROLES    AGE   VERSION
ip-10-1-0-4.ec2.internal     Ready    <none>   43m   v1.21.12-eks-5308cf7
ip-10-1-0-48.ec2.internal    Ready    <none>   43m   v1.21.12-eks-5308cf7
[ec2-user@ip-10-1-0-68 ~]$ kubectl get pods -A
NAMESPACE                    NAME                                                     READY   STATUS    RESTARTS   AGE
detection-container          detection-container-cb79796f4-2c6ph                      1/1     Running   0          38m
falcon-kubernetes-protection kpagent-cs-k8s-protection-agent-cfb4fdd9f-nrpjp          1/1     Running   0          38m
falcon-operator              falcon-operator-controller-manager-64b9f7c89-p9zg6       1/1     Running   0          38m
falcon-system                falcon-node-sensor-92lq9                                 1/1     Running   0          38m
falcon-system                falcon-node-sensor-bdf5s                                 1/1     Running   0          38m
kube-system                  aws-node-4zkpl                                           1/1     Running   0          44m
kube-system                  aws-node-jz5nk                                           1/1     Running   0          44m
kube-system                  coredns-66cb55d4f4-gw4p8                                 1/1     Running   0          54m
kube-system                  coredns-66cb55d4f4-rgvf7                                 1/1     Running   0          54m
kube-system                  kube-proxy-b84lv                                         1/1     Running   0          44m
kube-system                  kube-proxy-hvk9p                                         1/1     Running   0          44m
prometheus                   alertmanager-prometheus-kube-prometheus-alertmanager-0   2/2     Running   1          41m
prometheus                   prometheus-grafana-d59b4ddc4-6b6pq                       3/3     Running   0          41m
prometheus                   prometheus-kube-prometheus-operator-5bc74787f6-mgh68     1/1     Running   0          41m
prometheus                   prometheus-kube-state-metrics-7f6d8d4b7-4x8l6            1/1     Running   0          41m
prometheus                   prometheus-prometheus-kube-prometheus-prometheus-0       2/2     Running   0          41m
prometheus                   prometheus-prometheus-node-exporter-d5ggp                1/1     Running   0          41m
prometheus                   prometheus-prometheus-node-exporter-xtwmr                1/1     Running   0          41m
```

Review Detections in Falcon

If the optional Detection Container has been deployed, the detection-container pod (see above) will randomly create detections. Between each detection the container will pause for a randomized amount of time ranging from 100 to 1800 seconds (roughly 1.5 - 30 minutes). This pause ensures events trigger unique detections in the Falcon console that are not grouped together.

1. Open the Falcon Console and navigate to Menu → Endpoint security → Endpoint detections to view the detections generated by the detection-container pod.



Monitoring with Prometheus and Grafana

Prometheus is an open-source systems monitoring and alerting toolkit based on a multi-dimensional data model with time series data identified by metric name and key/value pairs. Grafana is a multi-platform open source analytics and interactive visualization web application. It provides charts, graphs, and alerts for the web when connected to supported data sources like Prometheus.

Prometheus and Grafana are optional deployments in this PoV to help validate if CrowdStrike CWP meets your performance needs.

Please note that the following commands only work if the PoV EKS cluster is accessed locally via kubectl, not via the Bastion Hosts. For instructions on how to connect via your local machine, please review the earlier section “Connect via Local Machine”

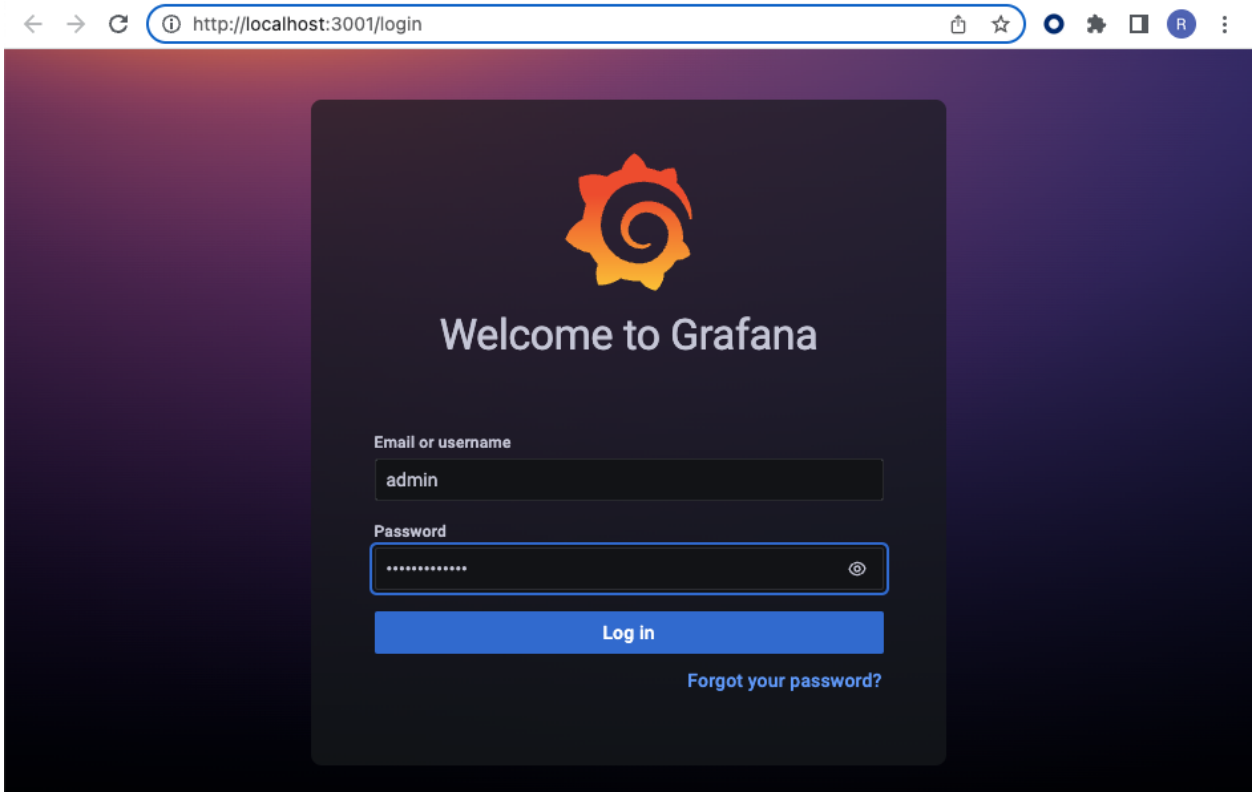
Connecting to Grafana

To connect to Grafana and visualize performance metrics, port-forwarding can be used. Execute the following command to create a port-forwarding rule to forward requests on localhost port 3001 to port 80 on the prometheus-grafana instance.

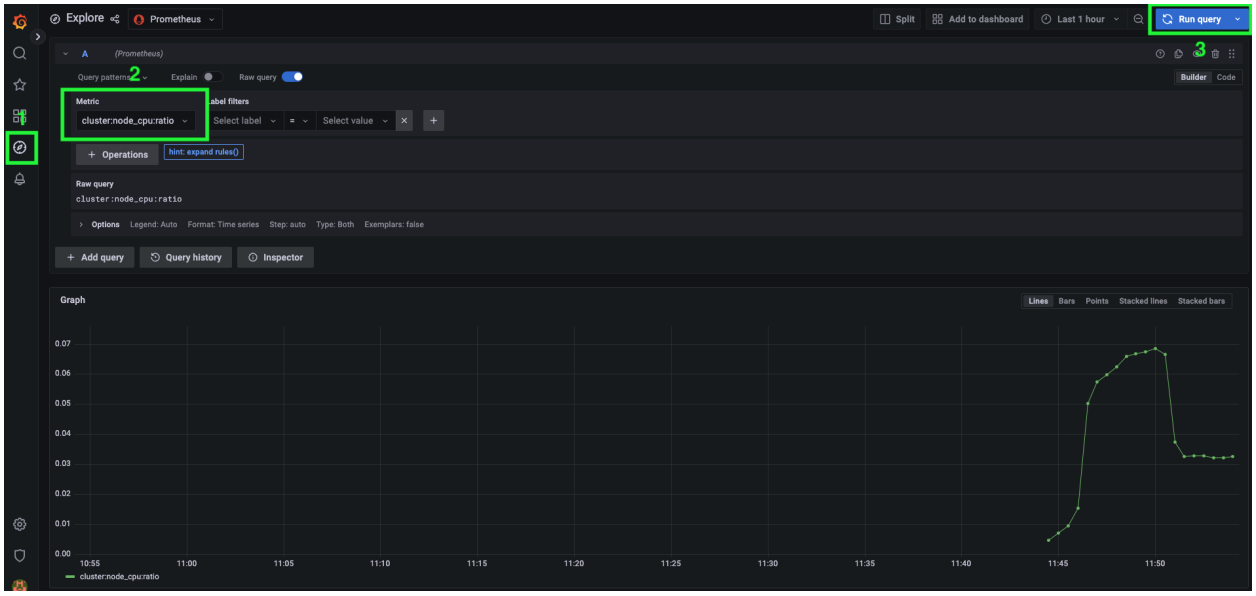
```
kubectl port-forward service/prometheus-grafana 3001:80 -n prometheus
```

This process will continue running in your terminal window, and must continue running to maintain access to the Grafana console. When you need to stop the service, press [ctrl]+C.

Open the URL http://localhost:3001 in a browser and login to Grafana using the username *admin* with the password *prom-operator*.



Once logged into Grafana, you may explore creating queries and dashboards to monitor the performance of your EKS Cluster. For example, a simple query can be created on the **Explore** page. To start, click the **compass icon** on the left, then select a **metric** from the drop down menu and finally click **Run Query** to view results.



For more information on how to build queries and dashboards, please see official documentation here https://prometheus.io/docs/visualization/grafana/

PoV Template Parameters Reference

Prerequisites

- **EnvAlias**: this will be appended to most created resources for identification
- **S3Bucket**: this is the S3 Bucket name where you uploaded the templates
- **PermissionBoundary**: this is the permission boundary name if required by IAM. For example, when launching in a CloudShare account, this value should be BoundaryForAdministratorAccess

EKS and Sensor Details

- **EC2orFargate**: choose whether to launch EKS Nodes on EC2 Managed Nodes or Fargate
- **KubernetesVersion**: leave as default (n-1), but can change when required
- **FalconSensorType**: NodeSensor or ContainerSensor.
    - Note: ContainerSensor is required if EC2orFargate = Fargate

Create New VPC

- **NewVPCCIDR**: Leave as default, if changed make sure it is still a /24

Configure Falcon Keys

- **FalconCID**: must be lower case and do not include the last three chars (-xx). This can be generated when retrieving the DockerAPIToken.
- **CrowdStrikeCloud:** us-1, us-2 or us-gov-1
- **FalconClientID:**
- **FalconClientSecret:**
- **DockerAPIToken**: Docker API Token generated for the Falcon CID to register EKS

Monitoring Stack (Optional)

- **InstallPrometheus**: Helm chart to run Prometheus on EKS for performance monitoring

Bastion Host (Optional)

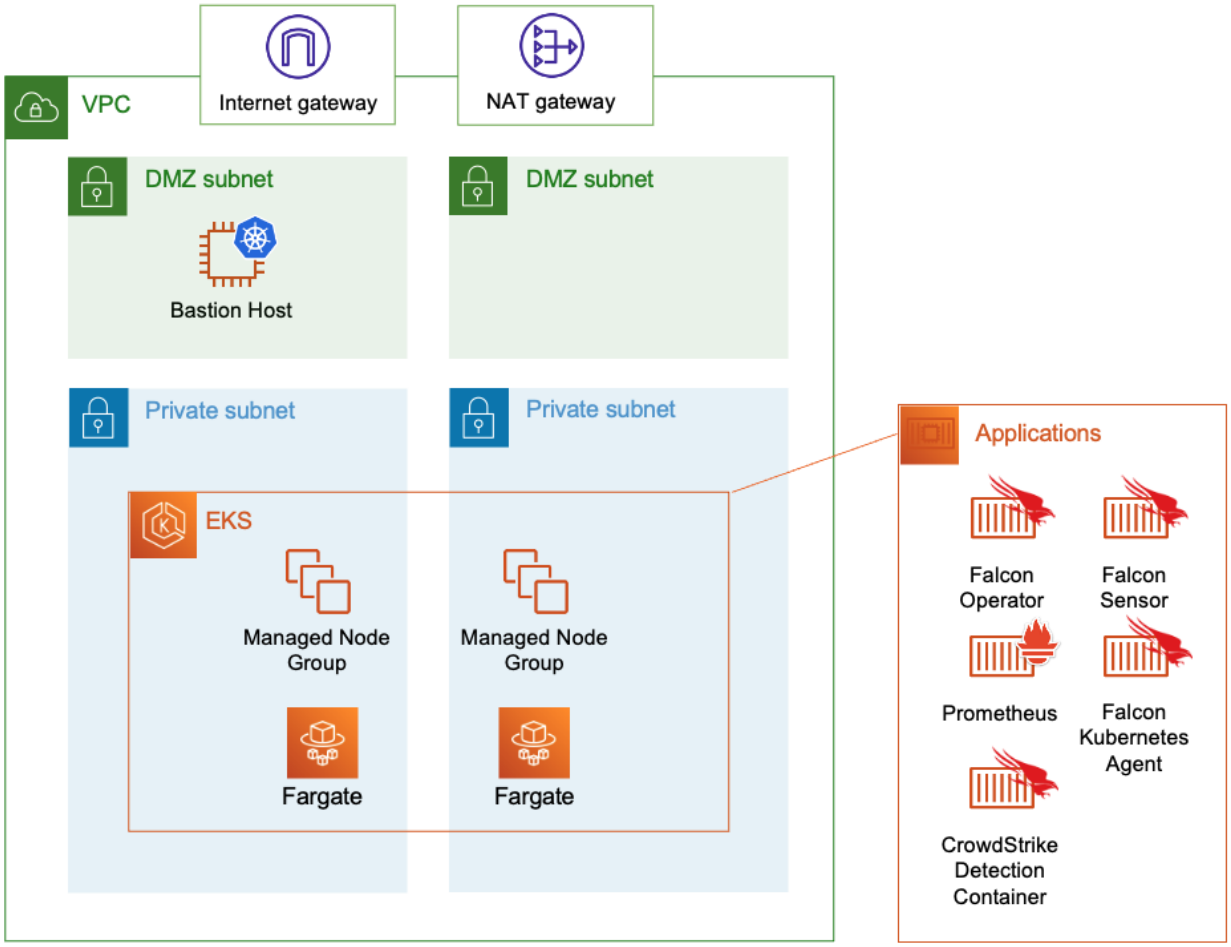
- **CreateBastion**: Change to true to get a preconfigured Bastion with access to kubectl
- **KeyPairName**: if CreateBastion=true then please provide valid existing Key Pair for connections to the Bastion Host
- **RemoteAccessCIDR**: if CreateBastion=true then please provide valid IP for SSH access (External Public IP Address of the host running the PoV from with a 32-bit mask to allow SSH access from a single IP address)

Detection Container (Optional)

- **InstallDetectionContainer**: Install detection-container on EKS Cluster to automatically produce endpoint detections.

Architecture Details

Architecture Overview



PoV Environment Architecture Overview

Required Resources

IAM

- Execution Role
- EKS Access Role
- EKS Service Role

CloudFormation

- Type Activation: Kubernetes Resource
- Type Activation: Helm
- Type Activation: EKS Cluster

EKS

- Cluster
- Cluster Shared Node Security Group
- Control Plane Security Group
- Cluster OIDC Provider

Kubernetes

- Falcon Operator
- Falcon Sensor
    - Container - Required if launching on Fargate Profile
    - Daemonset - Recommended if launching on EC2 Managed Nodes
- Kubernetes Protection Agent

VPC

- Size: /24
- Subnets: 4 /27s
- Internet Gateway
- NAT Gateway
- EIP for Nat Gateway
- 2 Route Tables for Private Subnets
    - Routes to NAT Gateway
- 1 Shared Route Table for DMZ Subnets
    - Route to Internet Gateway

Optional Resources

EKS Managed Node Group

- Launch Template
    - Volume: GP3, 80gb
- Node Group
    - Size: 2
    - Instance Type: m5.large
- Node Instance Role

Fargate

- Fargate Execution Role
- Fargate Profile
    - 2 Subnets
    - Up to 5 Selectors

            - falcon-operator
            - falcon-kubernetes-protection
            - prometheus
            - falcon-detection-container
            - Detection-container
- CoreDNS Fargate Profile
    - 2 Subnets
    - 1 Selector
        - Kube-system

Prometheus Monitoring

- Prometheus Kubernetes stack via Helm
- For details see: https://prometheus-community.github.io/helm-charts/

CrowdStrike Detection Container

- Detection Container via Kubernetes Resource
- For details see: https://github.com/CrowdStrike/detection-container

Bastion Host

- EC2 Instance
    - Instance Type: t2.small
    - Image: /aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2
    - Volume: GP2, 10gb
- EIP
- Security Group
- Instance Profile
    - Role: EKSAccessRole

PoV Cost Estimate

Cost for the PoV infrastructure varies based on how you choose to deploy EKS.

Deployment Method	Estimated Monthly Cost*
EKS on EC2 Managed Nodes	$165.95
EKS on Fargate	$147.93

Optional Bastion Host: Estimated $5.00 per month

*The estimated costs were determined using the AWS Pricing Calculator. AWS Pricing Calculator provides only an estimate of your AWS fees and doesn't include any taxes that might apply. Your actual fees depend on a variety of factors, including your actual usage of AWS services.

See a more detailed estimate below.

CrowdStrike EKS POV Cost Estimate

How to use this estimate:	When launching the EKS POV in your AWS Account, you will have the opportunity to choose whether to run EKS on EC2 Managed Nodes, Fargate, or both. This cost estimate provides the monthly cost for each option.
	You will also be able to launch a small linux bastion host with kubectl configured for easy management of the EKS Cluster, the monthly cost of this option is also listed below.

Group hierarchy	Region	Description	Service	Upfron	Monthl	Monthly Total	Currenc	Configuration summary
POV Estimate > EKS on EC2	US West (Oregon)	POV VPC	Network Address Translation (NAT) Gateway	0	33.3		USD	Number of NAT Gateways (1)
POV Estimate > EKS on EC2	US West (Oregon)	POV EKS	Amazon EKS	0	73		USD	Number of EKS Clusters (1)
POV Estimate > EKS on EC2	US West (Oregon)	EKS Managed Nodes	Amazon EC2	0	59.65		USD	Operating system (Linux), Quantity (2), Pricing strategy (On-Demand Instances), Storage amount (80 GB), Instance type (m5.large), General Purpose SSD (gp3) - IOPS (3000), General Purpose SSD (gp3) - Throughput (125 MBps)
						165.95	**USD**	
POV Estimate > EKS on Fargate	US West (Oregon)	POV VPC	Network Address Translation (NAT) Gateway	0	33.3		USD	Number of NAT Gateways (1)
POV Estimate > EKS on Fargate	US West (Oregon)	POV EKS	Amazon EKS	0	73		USD	Number of EKS Clusters (1)
POV Estimate > EKS on Fargate	US West (Oregon)	Optional Fargate	AWS Fargate	0	41.63		USD	Operating system (Linux), CPU Architecture (x86), Average duration (8 hours), Number of tasks or pods (9 per day), Amount of ephemeral storage allocated for Amazon ECS (20 GB)
						147.93	**USD**	
POV Estimate > EKS on Both	US West (Oregon)	POV VPC	Network Address Translation (NAT) Gateway	0	33.3		USD	Number of NAT Gateways (1)
POV Estimate > EKS on Both	US West (Oregon)	POV EKS	Amazon EKS	0	73		USD	Number of EKS Clusters (1)
POV Estimate > EKS on Both	US West (Oregon)	EKS Managed Nodes	Amazon EC2	0	59.65		USD	Operating system (Linux), Quantity (2), Pricing strategy (On-Demand Instances), Storage amount (80 GB), Instance type (m5.large), General Purpose SSD (gp3) - IOPS (3000), General Purpose SSD (gp3) - Throughput (125 MBps)
POV Estimate > EKS on Both	US West (Oregon)	Optional Fargate	AWS Fargate	0	41.63		USD	Operating system (Linux), CPU Architecture (x86), Average duration (8 hours), Number of tasks or pods (9 per day), Amount of ephemeral storage allocated for Amazon ECS (20 GB)
						207.58	**USD**	
POV Estimate > Optional Bastion	US West (Oregon)	Bastion Host	Amazon EC2	0	5		USD	Operating system (Linux), Quantity (1), Pricing strategy (On-Demand Instances), Storage amount (10 GB), Instance type (t2.small)
						5	**USD**	

Acknowledgement

* AWS Pricing Calculator provides only an estimate of your AWS fees and doesn't include any taxes that might apply. Your actual fees depend on a variety of factors, including your actual usage of AWS services.

Appendix

Resource Requirements

CloudFormation Stack	CloudFormation Logical ID	Resource Type	Required?	Details
IAMStack	ExecutionRole	AWS::IAM::Role	Yes	
IAMStack	EKSAccessRole	AWS::IAM::Role	Yes	
IAMStack	HelmResourceActivation	AWS::CloudFormation::TypeActivation	Yes	
IAMStack	ClusterResourceActivation	AWS::CloudFormation::TypeActivation	Yes	
VPCStack	EnvironmentID	AWS::SSM::Parameter	Yes	
VPCStack	VPC	AWS::EC2::VPC	Yes	
VPCStack	PrivateSubnet1	AWS::EC2::Subnet	Yes	
VPCStack	PrivateSubnet2	AWS::EC2::Subnet	Yes	
VPCStack	NATGateway	AWS::EC2::NatGateway	Yes	
VPCStack	NATIP	AWS::EC2::EIP	Yes	
VPCStack	PrivateRouteTable1	AWS::EC2::RouteTable	Yes	
VPCStack	PrivateRouteTable2	AWS::EC2::RouteTable	Yes	
VPCStack	PrivateSubnet1Route	AWS::EC2::Route	Yes	
VPCStack	PrivateSubnet2Route	AWS::EC2::Route	Yes	
VPCStack	DMZSubnet1	AWS::EC2::Subnet	Yes	
VPCStack	DMZSubnet2	AWS::EC2::Subnet	Yes	
VPCStack	InternetGateway	AWS::EC2::InternetGateway	Yes	
VPCStack	DMZRouteTable	AWS::EC2::RouteTable	Yes	
VPCStack	DMZSubnetIGWRoute	AWS::EC2::Route	Yes	

EKSControlPlaneStack	ClusterSharedNodeSecurityGroup	AWS::EC2::SecurityGroup	Yes	
EKSControlPlaneStack	serviceRole	AWS::IAM::Role	Yes	
EKSControlPlaneStack	Cluster	AWSQS::EKS::ClusterPOV	Yes	
EKSControlPlaneStack	ClusterOIDCProvider	AWS::IAM::OIDCProvider	Yes	
EKSControlPlaneStack	ControlPlaneSecurityGroup	AWS::EC2::SecurityGroup	Yes	
EKSControlPlaneStack	IngressDefaultClusterToNodeSG	AWS::EC2::SecurityGroupIngress	Yes	
EKSControlPlaneStack	IngressInterNodeGroupSG	AWS::EC2::SecurityGroupIngress	Yes	
EKSControlPlaneStack	IngressNodeToDefaultClusterSG	AWS::EC2::SecurityGroupIngress	Yes	
EKSControlPlaneStack	PolicyCloudWatchMetrics	AWS::IAM::Policy	Yes	
EKSControlPlaneStack	PolicyELBPermissions	AWS::IAM::Policy	Yes	
EKSNodeGroupStack	LaunchTemplate	AWS::EC2::LaunchTemplate	Optional: If Fargate is chosen this stack is omitted	
EKSNodeGroupStack	ManagedNodeGroup	AWS::EKS::Nodegroup	Optional: If Fargate is chosen this stack is omitted	
EKSNodeGroupStack	NodeInstanceRole	AWS::IAM::Role	Optional: If Fargate is chosen this stack is omitted	
FargateStack	FargateRole	AWS::IAM::Role	Optional: If Managed Nodes are chosen this stack is omitted	

FargateStack	FargateProfile	AWS::EKS::FargateProfile	Optional: If Managed Nodes are chosen this stack is omitted	
FargateStack	CoreDNSFargateProfile	AWS::EKS::FargateProfile	Optional: If Managed Nodes are chosen this stack is omitted	
FargateStack	PrometheusProfile	AWS::EKS::FargateProfile	Optional: If Managed Nodes are chosen this stack is omitted	
PrometheusMonitorInstall	InstallPrometheus	AWSQS::Kubernetes::HelmPOV	Optional	
BastionStack	BastionEIP	AWS::EC2::EIP	Optional	
BastionStack	BastionSG	AWS::EC2::SecurityGroup	Optional	
BastionStack	RemoteToBastion	AWS::EC2::SecurityGroupIngress	Optional	
BastionStack	BastionToAPIServerAccess	AWS::EC2::SecurityGroupIngress	Optional	
BastionStack	BastionHostProfile	AWS::IAM::InstanceProfile	Optional	
BastionStack	BastionHost	AWS::EC2::Instance	Optional	