Censys Unified Cloud Connector

Censys, Inc.

Oct 06, 2023

CONTENTS

CHAPTER

ONE

WHAT'S NEW?

1.1 v3.2.1

- Improved handling of Azure resource type errors (#49)
- Improved handling of Azure subscription errors (#48)
- Fix invalid Docker deployment scan command (#48)
- Remove reference to invalid AWS profile in AWS ECS deployment (#48)

1.2 v3.2.0

1.2.1 Changelog

- (*Details*) Migration from Google's soon-to-be-deprecated Security Command Center Asset API to Google's Cloud Asset Inventory API. (#26)
- Remove GCP stale seeds on each scan (#26)
- Reset AWS STS creds between scanning for seeds and cloud assets. This will remedy the recursion errors that some customers have been seeing in their healthcheck logs. (#41)
- Combine seeds submission for resource types with multiple versions (ex: AWS API Gateway v1 and API Gateway v2) (#40)
- CI updates (#38)
- Update dependencies (#39)
- (*Details*) Optional environmental variable AZURE_REFRESH_ALL_REGIONS available to scan all Azure regions and clear out lingering stale seeds (#34)
- Updates to documentation (#42)

1.2.2 Details about GCP API migration

In response to Google's deprecation of the Security Command Center (SCC) Asset API, the cloud connector will now use the Cloud Asset Inventory (CAI) as its source of truth.

Currently, we use GCP's Security Command Center (SCC) API to list assets by asset type within an organization. SCC is deprecating functionality related to assets on June 26, 2024. Existing users of the SCC Asset API can continue using it until then, but new customers can no longer enable the API.

The Cloud Connector will migrate to using GCP's Cloud Asset Inventory (CAI) API as its source of truth. All customers will need to enable this API and upgrade their cloud connector instances to v3.2.0 by June 26, 2024.

Changes

API usage

SCC List Assets request –> CAI Search All Resources request

Permissions

Service accounts will need the Cloud Asset Viewer (roles/cloudasset.viewer) role.

Service accounts no longer need the roles Security Command Center Assets Discovery Runner (securitycenter.assetsDiscoveryRunner) and Security Command Center Assets Viewer (securitycenter.assetsViewer).

What do customers need to do?

Recommended

Run through the configuration CLI (`censys-cc config --provider gcp`) and select the same organization, project, and service account that you've been using. This will enable the CAI API and apply the new permissions to the service account.

Manual

Enable the CAI API:

gcloud CLI: `gcloud services enable cloudasset.googleapis.com --project {PROJECT_ID}`

Apply new permissions to service account:

gcloud CLI: `gcloud organizations add-iam-policy-binding {ORGANIZATION ID} --member 'serviceAccount:{SERVICE ACCOUNT EMAIL}' --role 'roles/cloudasset.viewer' --condition=None --quiet`

1.2.3 Details about Azure stale seeds workaround

The Azure cloud connector currently submits assets that it finds during each scan to the Censys seeds API. When set to true, the environmental variable `AZURE_REFRESH_ALL_REGIONS` will submit an empty list to the Censys seeds API for every possible label (subscription+region) where assets were not found. This may cause the scan to run more slowly, so it is not enabled by default. Users can opt in on a per-connector basis by setting the environmental variable to true in the connector's `.env`` file.

CHAPTER

TWO

OVERVIEW

The Censys Attack Surface Management platform already discovers on-premise and cloud assets through our best-in-class Internet-wide scanning and attribution methodologies. Censys Cloud Connectors offer users the ability to supercharge our ASM Platform with total, cross-cloud visibility. Continuously monitor storage buckets like S3, Google storage or Azure blobs, virtual instances, databases, and more using our easy-to-configure connectors.

2.1 Which resources does the Cloud Connector scan for?

The following providers and services are supported and will be used to import Seeds (IP Addresses, Domain Names, CIDRs, and ASNs) as well as Cloud Assets (Object Storage Buckets) into the Censys ASM platform.

2.1.1 Amazon Web Services

- Compute
  - Elastic Container Service (ECS)
  - Elastic Compute Cloud (EC2)
- Database
  - Relational Database Service (RDS)
- Network & Content Delivery
  - API Gateway
  - Elastic Load Balancing (ELB)
  - Route53
- Cloud Storage
  - Simple Storage Service (S3)

2.1.2 Azure Cloud

- Azure Networking
  - Azure DNS
- Azure Container Services
  - Container Instances
- Azure Databases
  - Azure SQL
- Azure Storage
  - Azure Blob Storage

2.1.3 Google Cloud Platform

- Google Cloud Compute
  - Compute Engine
- Google Cloud Containers
  - Kubernetes Engine
- Google Cloud Networking
  - Cloud DNS
- Google Cloud Databases
  - Cloud SQL
- Google Cloud Storage
  - Cloud Storage

CHAPTER

THREE

GETTING STARTED

It is important to note that this connector is a Python package. This allows you to run the connector from the command line as well as enables you to run the connector in as many different environments as you wish. We have provided a variety of deployment types and configuration options. We recommend that you install the package locally to take advantage of the configuration command line interface (*censys-cc*). After you have configured the connector, you can deploy it to your environment.

3.1 Prerequisites

- Python 3.9+
- Pip
- Poetry

Note

There may be additional requirements depending on the deployment method

3.2 Installation

Clone the repo

```
$ git clone https://github.com/censys/censys-cloud-connector.git
$ cd censys-cloud-connector
```

Ensure you have poetry installed (may require restarting shell)

```
$ pip install --upgrade poetry
```

Start a shell and activate the virtual environment (this is optional if you'd like to install dependencies globally)

```
$ poetry shell
```

Install the dependencies (a Makefile is provided for convenience in installation)

```
$ make install-all     # Install dependencies for all providers
$ make install-azure   # Azure only
$ make install-aws     # AWS only
$ make install-gcp     # GCP only
```

Copy .env.sample to .env

```
$ cp .env.sample .env
```

3.3 Environment Variables

The connector uses environment variables to configure the connector. The *CENSYS_API_KEY* environment variable is required to run the connector.

The following environment variables are available for use in the connector:

CENSYS_API_KEY

Your Censys ASM API key found in the ASM Integrations Page. (**Required**)

PROVIDERS_CONFIG_FILE

The path to *Provider Configuration*.

Default: `./providers.yml`

SECRETS_DIR

The path to the directory containing the secrets.

Default: `./secrets`

LOGGING_LEVEL

The logging level. Valid values are `DEBUG`, `INFO`, `WARN`, `ERROR`, and `CRITICAL`.

Default: `INFO`

DRY_RUN

If set to `true`, the connector will not write any data to the ASM platform.

Default: `false`

HEALTHCHECK_ENABLED

If set to `false`, the connector will not report its health to the ASM platform.

Default: `true`

AZURE_REFRESH_ALL_REGIONS

Azure-specific environmental variable. If set to `true`, the connector will clear stale seeds from regions no longer containing assets. This may take longer to run, but will ensure that stale seeds do not persist in the workspace. If set to `false`, the connector will submit seeds that are found as normal.

Default: `false`

3.3.1 Sample `.env` File

`.env.sample` is a sample file that contains the above environment variables. Please use this file as a template to create your own `.env` file.

```
CENSYS_API_KEY=your-censys-api-key-here-xxxxxxxxxxx
SECRETS_DIR=./secrets
PROVIDERS_CONFIG_FILE=./providers.yml
LOGGING_LEVEL=INFO
DRY_RUN=false
HEALTHCHECK_ENABLED=true
AZURE_REFRESH_ALL_REGIONS=false
```

(continues on next page)

(continued from previous page)

```
# Censys API Settings
# CENSYS_ASM_API_BASE_URL=https://app.censys.io/api
# CENSYS_COOKIES={"key": "value"}
```

CHAPTER

FOUR

PROVIDER CONFIGURATION

To configure the connector, you can use the *Command Line Interface*. The configuration command is:

```
$ poetry run censys-cc config
```

The *censys-cc config* command will guide you through the configuration of supported cloud providers. This command will assist you in generating your `providers.yml` file. This file can contain multiple provider configurations.

Note

Before configuring the connector, make sure you are logged in to your cloud provider's CLI tool. See our *Provider Specific Setup* for more information.

4.1 Provider Specific Setup

4.1.1 AWS Provider Setup

StackSet Deployment

Add assets from all of your AWS accounts for the most up-to-date view of your cloud attack surface.

Ready to get started? Here's what you need:

- Your Censys ASM API key, located on the Integrations page of the app.
- Sufficient privileges in your Primary AWS account to run a CloudFormation StackSet across all of your AWS accounts (e.g., `admin`).
- Sufficient privileges in your Primary AWS account to run a CloudFormation StackSet to create roles and policies (e.g., `admin`).
- You may need to enable trusted access with AWS Organizations.

Getting Started

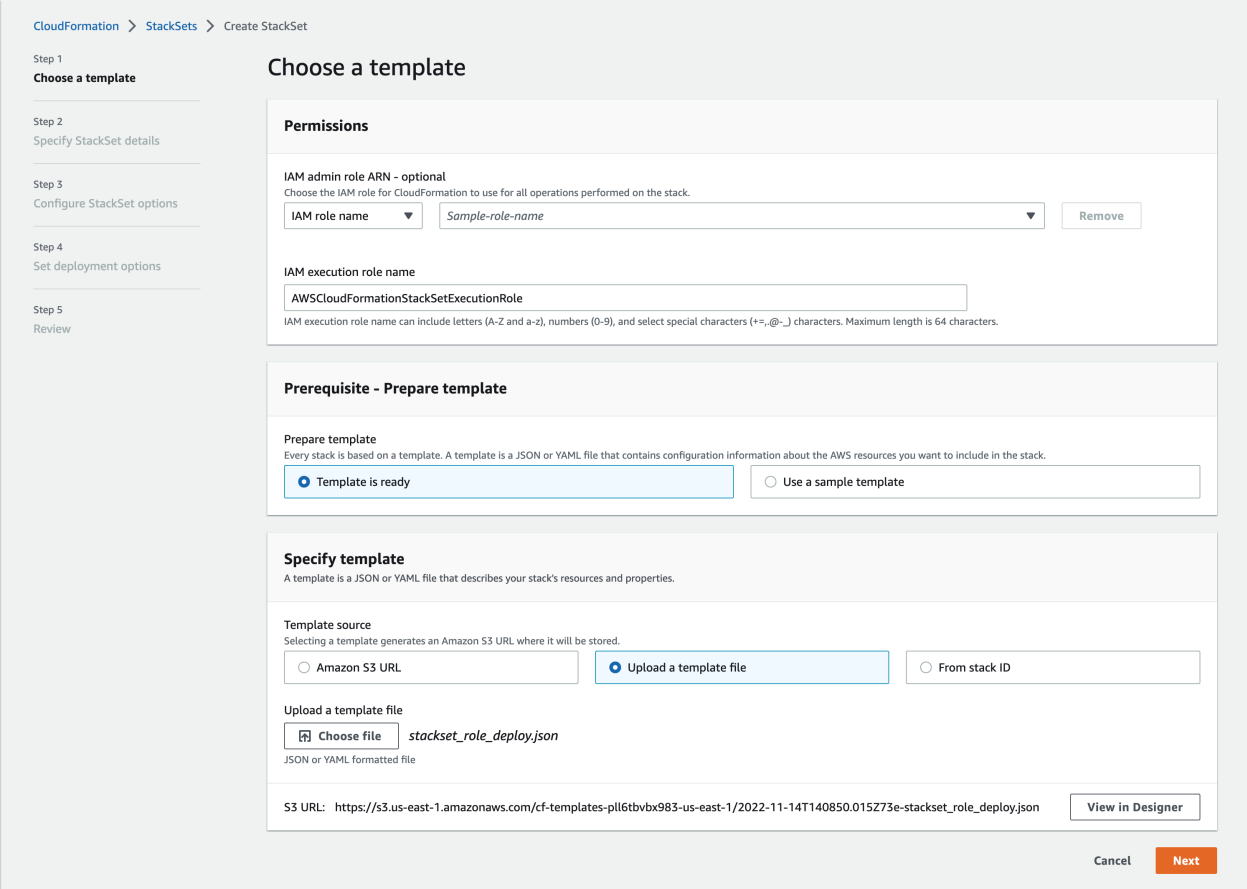
Log in to your Primary AWS account and navigate to Cloud Formation.

1: Create a Role via CloudFormation StackSets

Use the Censys-provided template to create a role in all of your accounts for cross-account access.

1. `Download` the StackSet template.
2. From the CloudFormation landing page, click **StackSets**.
3. Click the **Create StackSet** button.
4. In the **Prerequisite** section, select the "Template is ready" option.
5. In the **Specify template** section, select "Upload a template file".
6. Click **Choose file**.
7. Choose the template from Step 1.

Click **Next**.

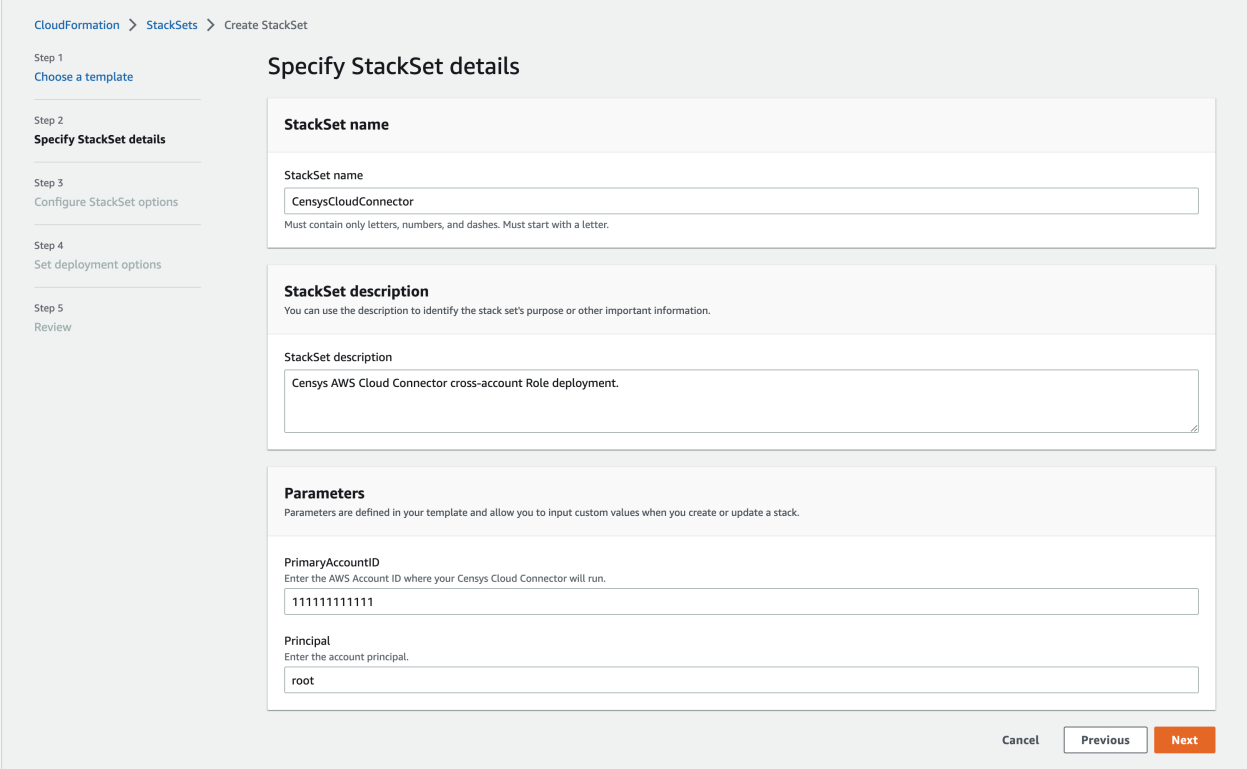


1a: Specify StackSet Details

On the second page:

1. Give the StackSet a name, which can include uppercase and lowercase letters, numbers, and dashes.
2. In the **Parameters** section, paste in your Primary AWS Account ID.

Click **Next**.

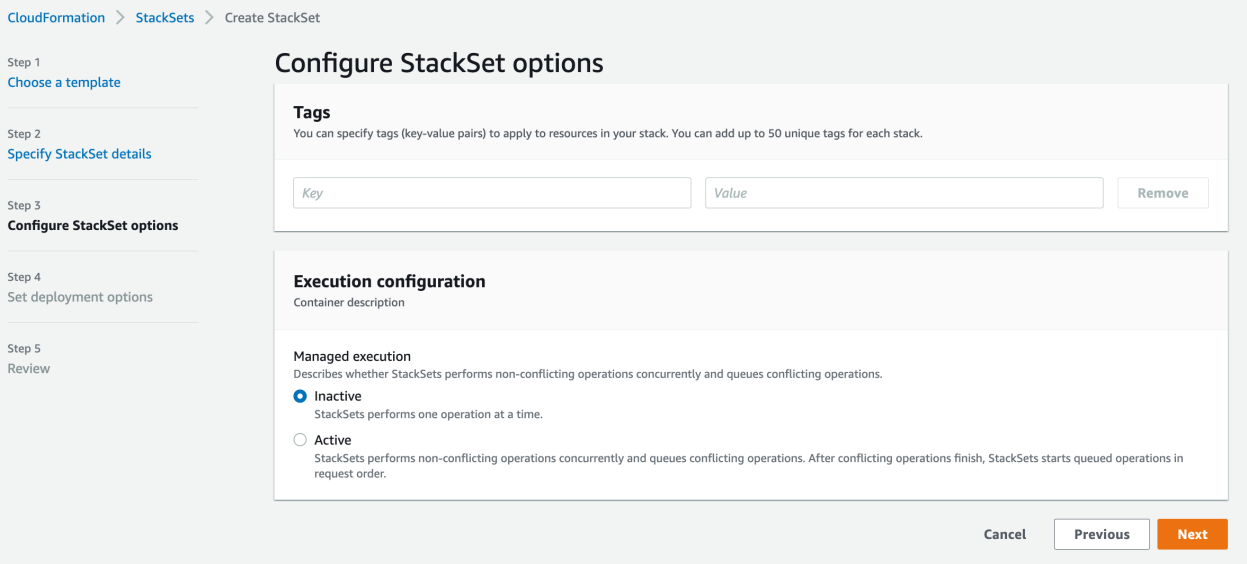


1b: Configure StackSet Options

On the third page, nothing needs to be specified, as this stack will use all of the default options.

You can optionally tag this stack with tags according to your organization's best practices.

Click **Next**.

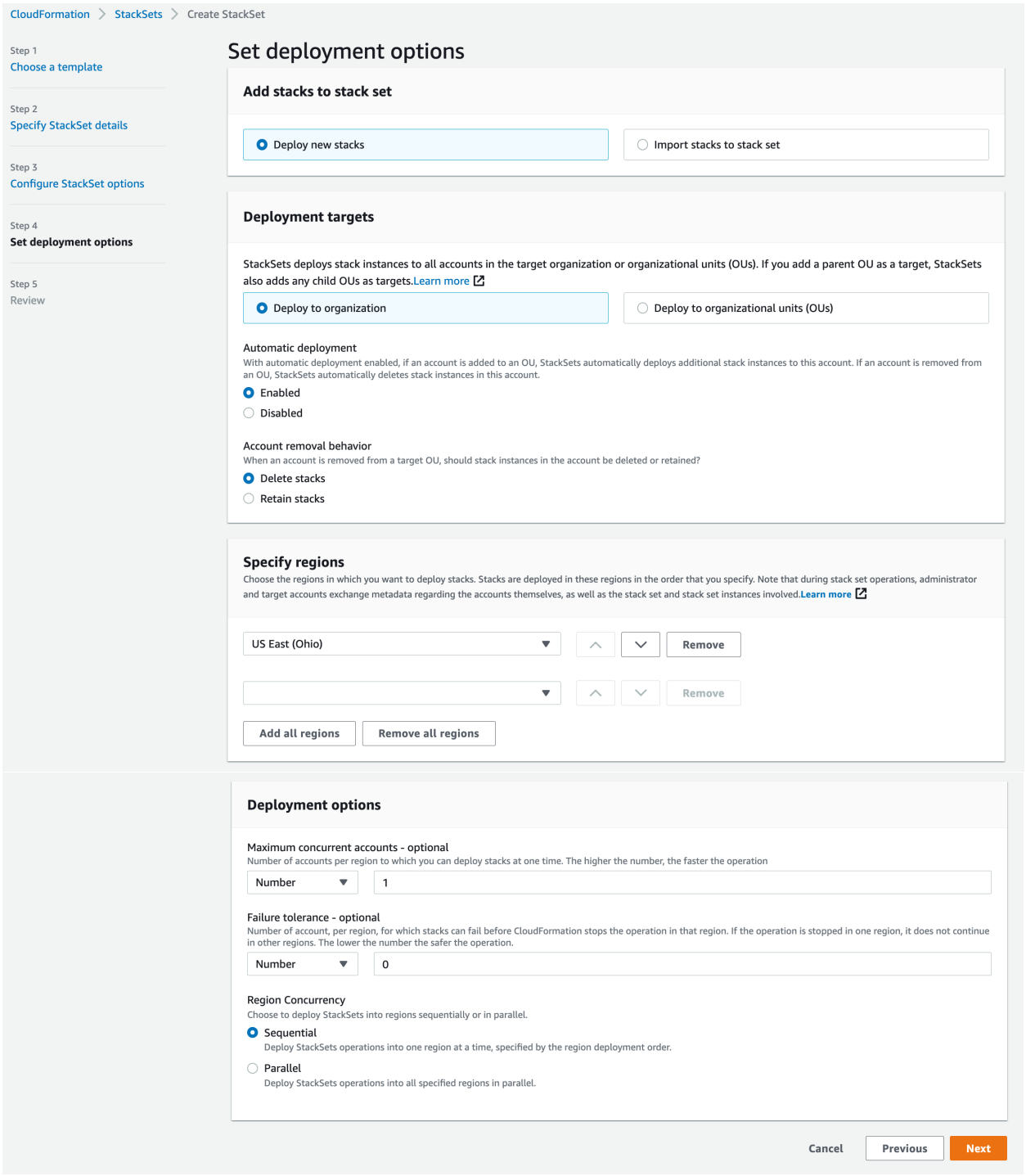


1c: StackSet Deployment Options

On the fourth page, you'll specify the StackSet deployment options. Censys suggests deploying the StackSet to your organization to ensure that all AWS Accounts are accounted for.

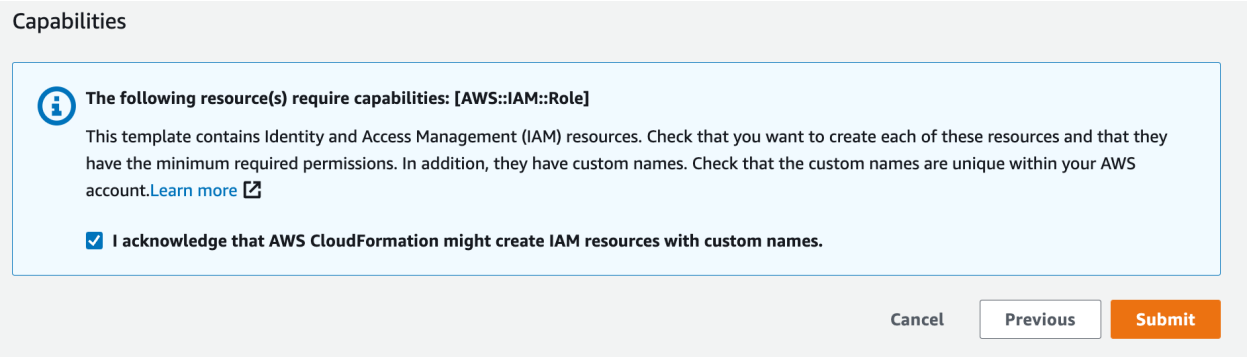
1. In the Deployment targets section, keep the default option of "Deploy to organization," or specify only certain organizational units.
2. In the Specify regions section, add your preferred region.

Click **Next**.


CloudFormation > StackSets > Create StackSet
Step 1
Choose a template
Step 2
Specify StackSet details
Step 3
Configure StackSet options
Step 4
Set deployment options
Step 5
Review
Set deployment options
Add stacks to stack set
Deploy new stacks
Import stacks to stack set
Deployment targets
StackSets deploys stack instances to all accounts in the target organization or organizational units (OUs). If you add a parent OU as a target, StackSets also adds any child OUs as targets.Learn more
Deploy to organization
Deploy to organizational units (OUs)
Automatic deployment
With automatic deployment enabled, if an account is added to an OU, StackSets automatically deploys additional stack instances to this account. If an account is removed from an OU, StackSets automatically deletes stack instances in this account.
Enabled
Disabled
Account removal behavior
When an account is removed from a target OU, should stack instances in the account be deleted or retained?
Delete stacks
Retain stacks
Specify regions
Choose the regions in which you want to deploy stacks. Stacks are deployed in these regions in the order that you specify. Note that during stack set operations, administrator and target accounts exchange metadata regarding the accounts themselves, as well as the stack set and stack set instances involved.Learn more
US East (Ohio)
Remove
Remove
Add all regions
Remove all regions
Deployment options
Maximum concurrent accounts - optional
Number of accounts per region to which you can deploy stacks at one time. The higher the number, the faster the operation
Number
1
Failure tolerance - optional
Number of account, per region, for which stacks can fail before CloudFormation stops the operation in that region. If the operation is stopped in one region, it does not continue in other regions. The lower the number the safer the operation.
Number
0
Region Concurrency
Choose to deploy StackSets into regions sequentially or in parallel.
Sequential
Deploy StackSets operations into one region at a time, specified by the region deployment order.
Parallel
Deploy StackSets operations into all specified regions in parallel.
Cancel
Previous
Next

1d: Review & Submit

On the review page, check all of the settings and confirm that you are aware that this stack will create a role with a custom name in order to run properly by checking the box next to the acknowledgment statement.



When this StackSet completes successfully, you'll have the required cross-account role set up to allow the Cloud Connector to read from all of your AWS accounts.

Finally, the StackSet must also be installed in the parent account. Otherwise, you will encounter permission denied errors.

Templates

StackSet Template

download

```
{
  "Parameters": {
    "PrimaryAccountID": {
      "AllowedPattern": "\\d{12}",
      "ConstraintDescription": "\"PrimaryAccountID\" must be a valid AWS Account ID (12␣
↪digits).",
      "Description": "Enter the AWS Account ID where your Censys Cloud Connector will␣
↪run.",
      "MaxLength": 12,
      "MinLength": 12,
      "Type": "String"
    },
    "Principal": {
      "AllowedPattern": "[a-zA-Z_0-9+=,.@\\-_/]+",
      "ConstraintDescription": "\"Principal\" must be a valid AWS IAM Principal name.",
      "Description": "Enter the account principal.",
      "MaxLength": 64,
      "MinLength": 1,
      "Type": "String",
      "Default": "root"
    }
  },
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Censys AWS Cloud Connector cross-account Role deployment.",
```

(continues on next page)

(continued from previous page)

```
  "Resources": {
    "CensysCloudConnectorSetup": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "AWS": {
                  "Fn::Sub": "arn:aws:iam::${PrimaryAccountID}:${Principal}"
                }
              },
              "Action": ["sts:AssumeRole"]
            }
          ]
        },
        "Description": "This role was created by the Censys Cloud Connector. The Censys␣
↪Cloud Connector utilizes this role to enumerate assets in this account.",
        "ManagedPolicyArns": ["arn:aws:iam::aws:policy/SecurityAudit"],
        "Policies": [
          {
            "PolicyName": "CensysAPIGatewayPolicy",
            "PolicyDocument": {
              "Version": "2012-10-17",
              "Statement": [
                {
                  "Sid": "CensysCloudConnectorPolicy",
                  "Effect": "Allow",
                  "Action": ["apigateway:GET"],
                  "Resource": "*"
                }
              ]
            }
          }
        ],
        "Path": "/",
        "RoleName": "CensysCloudConnectorRole"
      }
    }
  }
}
```

IAM Policies

Note

As a security best-practice, the connector also supports creation of temporary credentials via Secure Token Service (STS).

Recommended

In order to ease the burden of maintaining an evolving list of policies, it's possible to run the Censys Cloud Connector using a role with the following policies:

1. AWS arn:aws:iam::aws:policy/SecurityAudit
2. `censysCloudConnectorPolicy` (below)

`download`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "censysCloudConnectorPolicy",
      "Effect": "Allow",
      "Action": ["apigateway:GET"],
      "Resource": "*"
    }
  ]
}
```

Least Privilege

Use this policy to follow the AWS best-practice of least-privilege.

`download`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "censysLeastPrivilegeCloudConnector",
      "Effect": "Allow",
      "Action": [
        "apigateway:GET",
        "ec2:DescribeTags",
        "ec2:DescribeNetworkInterfaces",
        "ecs:ListContainerInstances",
        "ecs:ListClusters",
        "elasticloadbalancing:DescribeLoadBalancers",
        "rds:DescribeDBInstances",
        "route53:ListHostedZones",
```

(continues on next page)

(continued from previous page)

```
        "route53:ListResourceRecordSets",
        "route53domains:ListDomains",
        "s3:GetBucketLocation",
        "s3:ListAllMyBuckets",
        "s3:ListBucket"
      ],
      "Resource": "*"
    }
  ]
}
```

Installation

Install the AWS CLI.

Authentication

Configure the AWS CLI.

Configure Cloud Connector IAM

We recommend *deploying a StackSet*, but *alterative options* are available.

Configuration

The Censys Cloud Connector *provider setup CLI* will ask a series of questions that have opt-in defaults.

```
$ censys-cc config --provider aws
```

Note

Permissions required during provider setup are described *here*.

Example AWS Provider Setup: Basic Usage

Alternative AWS Configuration Options

Manually create an IAM role and attach the either the *Least Privilege* policy or the *Recommended* set of policies.

Supported Provider Configurations

The Censys Cloud Connector officially supports the following IAM configurations:

- *IAM User in Parent, Assume Role in Children*
- *IAM User in Parent, IAM Users in each children*
- *ECS Role in Parent, Assume Role in Children*

IAM User in Parent, Assume Role in Children

This is the recommended configuration if you are running the connector outside of ECS.

```
- provider: AWS
  account_number: '111111111111'
  access_key: example-access-key-1
  secret_key: example-secret-key-1
  regions:
  - us-east-1
  accounts:
  - account_number: '111111111112'
    role_name: example-role-2
  - account_number: '111111111113'
    role_name: example-role-3
```

IAM User in Parent, IAM Users in each children

```
- provider: AWS
  account_number: '111111111111'
  access_key: example-access-key-1
  secret_key: example-secret-key-1
  regions:
  - test-region
  accounts:
  - account_number: '111111111112'
    access_key: example-access-key-2
    secret_key: example-secret-key-2
```

ECS Role in Parent, Assume Role in Children

This configuration can be used in conjunction with the *AWS ECS* deployment.

```
- provider: AWS
  account_number: '111111111111'
  role_name: example-role-1
  role_session_name: censys-cloud-connector
  regions:
  - test-region
  accounts:
  - account_number: '111111111112'
```

(continues on next page)

(continued from previous page)

```
    role_name: example-role-2
    role_session_name: censys-cloud-connector
```

Provider Setup Permissions Overview

The permissions used are dependant on options chosen during setup.

Service	Action	Reason
STS	`GetCallerIdentity`	Used to find the primary account number
Organizations	`ListAccounts`	Allows finding accounts within an organization
CloudFormation	`ListStackInstances`	Allows finding accounts using a specific StackSet instance

Find Accounts Feature

Add assets from all of your AWS accounts for the most up-to-date view of your cloud attack surface.

Find Accounts by StackSet (recommended)

Censys provides a CloudFormation *StackSet template* available to create the `CensysCloudConnectorRole`. It also serves as a way to list your organization's account numbers with the CloudFormation Stack Instance API.

Example 1

Find Accounts by Organizations

Provider setup will use the Organizations List Accounts feature to find a list of accounts. You will then have the option to choose which accounts are saved into `providers.yml`.

Example 2

Asset Deny List

In certain situations it is desirable not to have assets sent to Censys. This can be accomplished by utilizing the cloud provider's tagging feature. At this time, only AWS ENI and EC2 tags are supported.

Usage:

- AWS supports `ignore_tags` at the provider and account levels in *providers.yml*.
- Tags named `censys-cloud-connector-ignore` are ignored.

4.1.2 Azure Provider Setup

Installation

Install the Azure CLI.

Authentication

Log in to Azure's CLI tool using the following command: `az login`.

Configuration

Use our *Command Line Interface* to step through the configuration process:

```
$ censys-cc config --provider azure
```

Roles and Permissions

Azure uses role-based access control. Ensure that your account's role has the following permission to create a service principal with a Reader role:

- `Microsoft.Authorization/roleAssignments/write` over scope `/subscriptions/uuid`

The following permissions will be used by this service principal:

- `Microsoft.ContainerInstance/containerGroups/read`
- `Microsoft.Network/dnszones/read`
- `Microsoft.Network/publicIPAddresses/read`
- `Microsoft.Sql/servers/read`
- `Microsoft.Storage/storageAccounts/read`

Example

4.1.3 GCP Provider Setup

Installation

Install GCP's gcloud CLI.

Authentication

Log in to GCP's CLI tool using the following command: `gcloud auth login`.

Configuration

Use our *Command Line Interface* to step through the configuration process:

```
$ censys-cc config --provider gcp
```

Roles and Permissions

During the configuration, you will notice after you have selected the GCP account, project, organization ID, and service account, the CLI will apply all required roles to the service account upon your confirmation. For your reference, these roles are listed below:

- Security Reviewer (roles/iam.securityReviewer)
- Folder Viewer (roles/resourcemanager.folderViewer)
- Organization Viewer (roles/resourcemanager.organizationViewer)
- Cloud Asset Viewer (roles/cloudasset.viewer)

Note

The linked documentation from GCP includes a list of permissions that come with each role.

Example

4.2 Verify Configuration (Optional)

At this point, you should be able to run the cloud connector. If you would like to run the connector once before moving onto deployment, you can run the following command:

Caution: This is a real-time scan of your cloud environment and may take a long time if you have a large cloud environment. You may adjust the environment variable `DRY_RUN` to `true` to opt out of submitting scan results to Censys.

```
$ poetry run censys-cc scan
```

4.3 Sample `providers.yml` File

The `providers.yml` file contains the configuration for all cloud providers.

The file is a YAML file and is structured as follows:

```
- provider: aws
  account_number: xxxxxxxxxxxx
  access_key: xxxxxxxxxxxxxxxxxxx
  secret_key: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
  regions:
    - xxxxxxxxx
  # ignore:
```

(continues on next page)

(continued from previous page)

```
  #   - AWS::ApiGateway
  #   - AWS::ECS
  #   - AWS::ElasticLoadBalancing
  #   - AWS::NetworkInterface
  #   - AWS::RDS
  #   - AWS::Route53
  #   - AWS::S3
- provider: azure
  tenant_id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
  client_id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
  client_secret: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
  subscription_id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
  # The subscription_id field takes one or more subscription IDs.
  # subscription_id:
  #   - xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
  #   - xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
  # The ignore field takes a list of Azure resource types to ignore during scanning.
  # ignore:
  #   - Microsoft.Network/publicIPAddresses
  #   - Microsoft.ContainerInstance/containerGroups
  #   - Microsoft.Sql/servers
  #   - Microsoft.Network/dnszones
  #   - Microsoft.Storage/storageAccounts
- provider: gcp
  organization_id: xxxxxxxx-xxxx-xxxx
  service_account_json_file: service_account.json
  service_account_email: censys-cloud-connector@project-id.iam.gserviceaccount.com
  # The ignore field takes a list of GCP resource types to ignore during scanning.
  # ignore:
  #   - google.compute.Instance
  #   - google.compute.Address
  #   - google.container.Cluster
  #   - google.cloud.sql.Instance
  #   - google.cloud.dns.ManagedZone
  #   - google.cloud.storage.Bucket
```

CHAPTER

FIVE

DEPLOYMENT METHODS

5.1 AWS Elastic Container Service (ECS) Task

This module allows Terraform to manage AWS ECS Service for the Censys Cloud Connector.

5.1.1 Prerequisites

- Install Poetry.
- Install Terraform.
- Install AWS CLI.
- Optional: AWS Terraform Authentication and Configuration

5.1.2 Login Instructions

Use the AWS CLI tool to configure a named profile. The AWS Terraform provider uses standard configuration and credential precedence.

5.1.3 Setup

1. Ensure you are in the root directory of the project.
2. Source your environment variables.

   ```
   $ source .env
   ```

3. Run `poetry install` to install the dependencies.
4. Ensure your `providers.yml` file contains your cloud provider credentials.

   If you have not already done so, you can create a `providers.yml` file by running the following command:

   ```
   $ poetry run censys-cc config
   ```

5. Change the working directory to the `aws-ecs-task` directory with the following command:

   ```
   $ cd ./terraform/aws-ecs-task
   ```

6. Copy `terraform.tfvars.example` to `terraform.tfvars` and update the values to match your environment.

   ```
   $ cp terraform.tfvars.example terraform.tfvars
   ```

7. Initialize the project with the following command:

```
$ terraform init
```

8. To see what resources will be created or updated, run the following command:

```
$ terraform plan -var-file terraform.tfvars -out=censys-tfplan -input=false
```

9. To create or update the resources, run the following command:

```
$ terraform apply -input=false censys-tfplan
```

5.1.4 Cleanup

To clean up the resources created by this module, run the following command:

```
$ terraform destroy -var-file terraform.tfvars
```

5.1.5 Requirements

Name	Version
terraform	>= 0.13.1
aws	>= 4.7

5.1.6 Providers

Name	Version
aws	4.51.0
random	3.4.3

5.1.7 Modules

Name	Source	Version
ecs	terraform-aws-modules/ecs/aws	~> 3.0
eventbridge	terraform-aws-modules/eventbridge/aws	n/a
vpc	terraform-aws-modules/vpc/aws	n/a

5.1.8 Resources

Name	Type
aws_cloudwatch_log_group.cloud_connector	resource
aws_ecs_task_definition.cloud_connector	resource
aws_iam_policy.cross_account	resource
aws_iam_policy.get_secret	resource
aws_iam_role.cc_task_exec_role	resource
aws_iam_role.cc_task_role	resource
aws_secretsmanager_secret.censys_api_key	resource
aws_secretsmanager_secret.providers	resource
aws_secretsmanager_secret_version.censys_api_key	resource
aws_secretsmanager_secret_version.providers	resource
random_pet.censys	resource

5.1.9 Inputs

Name	Description	Type	Default	Required
aws_availability_zone	The AWS availability zones to use.	`string`	`"us-east-1a"`	no
aws_region	The AWS region to use.	`string`	`"us-east-1"`	no
censys_api_key	The Censys ASM API key	`string`	n/a	yes
image_tag	The tag of the Docker image to use for ECS.	`string`	`"latest"`	no
image_uri	The URI of the Docker image to use for ECS.	`string`	`"gcr.io/censys-io/censys-cloud-connector"`	no
logging_level	The logging level	`string`	`"INFO"`	no
providers_config	The path to the providers config file	`string`	`"../../providers.yml"`	no
role_name	The cross-account AWS IAM Role name.	`string`	`"CensysCloudConnectorRole"`	no
schedule_expression	Cloud Connector scan frequency.	`string`	`"rate(4 hours)"`	no
secrets_dir	The path to the secrets directory	`string`	`"../../secrets"`	no
task_cpu	The number of CPU units to allocate to the ECS task.	`number`	`1024`	no
task_memory	The amount of memory to allocate to the ECS task.	`number`	`2048`	no

5.1.10 Outputs

Name	Description
eventbridge_bus_arn	The EventBridge Bus ARN
eventbridge_rule_arns	The EventBridge Rule ARNs
eventbridge_rule_ids	The EventBridge Rule IDs

5.2 Google Cloud Scheduled Function

This module allows Terraform to manage Google Cloud Scheduled Functions for the Censys Cloud Connector.

5.2.1 Prerequisites

- Install Poetry.
- Install Terraform.
- Install the Cloud SDK for your operating system.

  If you are running from your local machine, you also need Default Application Credentials:

  ```
  $ gcloud auth application-default login
  ```

5.2.2 Setup

1. Ensure you are in the root directory of the project.
2. Source your environment variables.

   ```
   $ source .env
   ```
3. Install the dependencies.

   ```
   $ poetry install
   ```
4. Ensure your `providers.yml` file contains your cloud provider credentials.

   If you have not already done so, you can create a `providers.yml` file by running the following command:

   ```
   $ poetry run censys-cc config
   ```
5. Change the working directory to the `google-scheduled-function` directory with the following command:

   ```
   $ cd ./terraform/google-scheduled-function
   ```
6. Copy `terraform.tfvars.example` to `terraform.tfvars` and update the values to match your environment.

   ```
   $ cp terraform.tfvars.example terraform.tfvars
   ```
7. Initialize the project with the following command:

   ```
   $ terraform init
   ```
8. To see what resources will be created or updated, run the following command:

   ```
   $ terraform plan -var-file terraform.tfvars -out=censys-tfplan -input=false
   ```
9. To create or update the resources, run the following command:

   ```
   $ terraform apply -input=false censys-tfplan
   ```

5.2.3 Cleanup

To clean up the resources created by this module, run the following command:

```
$ terraform destroy -var-file terraform.tfvars
```

5.2.4 Requirements

Name	Version
terraform	>= 0.13
google	>= 3.53, < 5.0

5.2.5 Providers

Name	Version
archive	2.2.0
external	2.2.2
google	4.17.0
local	2.2.2
null	3.1.1
random	3.1.2

5.2.6 Modules

Name	Source	Version
pubsub_topic	terraform-google-modules/pubsub/google	~> 1.0

5.2.7 Resources

Name	Type
google_cloud_scheduler_job.job	resource
google_cloudfunctions_function.main	resource
google_project_service.gcp_services	resource
google_secret_manager_secret.censys_api_key	resource
google_secret_manager_secret.providers	resource
google_secret_manager_secret_iam_member.api_key_member	resource
google_secret_manager_secret_iam_member.providers_member	resource
google_secret_manager_secret_version.censys_api_key	resource
google_secret_manager_secret_version.providers	resource
google_secret_manager_secret_version.providers_config	resource
google_storage_bucket.main	resource
google_storage_bucket_object.main	resource
local_file.requirements_txt	resource
null_resource.copy_build	resource
random_id.suffix	resource
archive_file.main	data source
external_external.poetry_build	data source
google_project.project	data source
google_secret_manager_secret_version.censys_api_key	data source

5.2.8 Inputs

Name	Description	Type	Default	Required
bucket_force_destroy	When deleting the GCS bucket containing the cloud function, delete all objects in the bucket first.	bool	true	no
bucket_labels	A set of key/value label pairs to assign to the bucket.	map(string)	{}	no
bucket_name	The name to apply to the bucket. Will default to a string of censys-cloud-connector-bucket-XXXX with XXXX being random characters.	string	""	no
censys_api_key	The Censys ASM API key	string	n/a	yes
create_bucket	Whether to create a new bucket or use an existing one. If false, bucket_name should reference the name of the alternate bucket to use.	bool	true	no
files_to_exclude_in_source_dir	Specify files to ignore when reading the source_dir	list(string)	[".gcloudignore"]	no
function_available_memory_mb	The amount of memory in megabytes allotted for the function to use.	number	256	no
function_description	The description of the function.	string	"Cloud Function to run the Censys Cloud Connector."	no
function_labels	A set of key/value label pairs to assign to the function.	map(string)	{}	no
function_name	The name to apply to the function. Will default to a string of censys-cloud-connector-function-XXXX with XXXX being random characters.	string	""	no
function_source_dir	The directory containing the source code for the function.	string	"function_source"	no
function_timeout_s	The amount of time in seconds allotted for the execution of the function. (Can be up to 540 seconds)	number	540	no
gcp_service_list	The list of apis necessary for the project	list(string)	["cloudbuild.googleapis.com", "cloudfunctions.googleapis.com", "cloudresourcemanager.googleapis.com", "cloudscheduler.googleapis.com", "pubsub.googleapis.com", "secretmanager.googleapis.com", "cloudasset.googleapis.com"]	no
job_description	Additional text to describe the job	string	"Scheduled time to run the Censys Cloud Connector function"	no
job_name	The name of the scheduled job to run	string	"censys-cloud-connector-job"	no
job_schedule	The cron schedule for triggering the cloud function	string	"0 */4 * * *"	no
logging_level	The logging level	string	"INFO"	no
message_data	The data to send in the topic message.	string	"c3RhcnQtY2Vuc3lzLWNjLXNjYW4="	no
project_id	The project ID to host the cloud function in	string	n/a	yes
providers_config	The path to the providers config file	string	"../../providers.yml"	no
region	The region the project is in	string	"us-central1"	no
scheduler_job	An existing Cloud Scheduler job instance	object({ full name		no

5.2.9 Outputs

Name	Description
api_secret_version	The secret version of the API key
bucket_name	The name of the bucket created
function_name	The name of the function created
function_region	The region the function is in
job_name	The name of the scheduled job to run
project_id	The project ID
providers_secrets_versions	The secret versions of the providers config
topic_name	The name of the topic created

5.3 Docker Deployment Methods

5.3.1 Docker Standalone

This method assumes you have Docker installed and running on your server.

1. Ensure you are in the root directory of the project.
2. Pull the Docker image

```
$ docker pull gcr.io/censys-io/censys-cloud-connector:latest
```

Note

If your environment does not allow you to pull the Docker image, you can build it from the Dockerfile using the following command. You can then push the image to a Docker registry.

```
$ docker build -t gcr.io/censys-io/censys-cloud-connector:latest .
```

---

3. Run the Docker container

The following command will run the Docker container. The container also requires the `providers.yml` file. The `-v` flag will mount the `providers.yml` file as a volume. If your `providers.yml` references additional secret files, you can mount it as a volume as well. The `-d` flag is used to run the container in the background. We also include the `--rm` flag to ensure the container is removed after it has finished.

- Run the Docker container (Once)

```
$ docker run -d --rm      --env-file .env      -v $(pwd)/providers.yml:/app/providers.yml      -v $
```

- Run the Docker container (Scheduled)

```
$ docker run -d --rm      --env-file .env      -v $(pwd)/providers.yml:/app/providers.yml      -v $
```

Note

The *–daemon* flag will run the connector in the background. The number specifies the number of hours between each scan.

- Run the Docker container (Without secrets mounted)

```
$ docker run -d --rm      --env-file .env      -v $(pwd)/providers.yml:/app/providers.yml      gcr.
```

5.3.2 Docker Compose

This method assumes you have Docker and Docker Compose installed and running on your server.

1. Run the Docker Compose file

   ```
   $ docker-compose up -d
   ```

2. (Optional) Run your connector on a scheduled interval

   Uncomment the line `# command:  scan --daemon 4` in `docker-compose.yml`.

   Note

   Learn more about the available options for the *scan* command.

5.4 Kubernetes Deployment Method

This guide describes how to deploy the Censys Cloud Connector using Kubernetes.

5.4.1 Prerequisites

The following prerequisites are required to deploy using Kubernetes:

- A Kubernetes cluster
- Helm
- Kubectl
- A valid `providers.yml` file

5.4.2 Getting Started

Note

The following steps assume that you have already cloned the Censys Cloud Connector repository and are in the root directory.

1. If you haven't already, create a namespace for the Censys Cloud Connector

```
$ kubectl create namespace censys-cloud-connectors
```

Please note that the the above namespace is used in the following steps. If you choose to use a different namespace, please update the commands accordingly.

2. Set the current namespace to the Censys Cloud Connector namespace

```
$ kubectl config set-context --current --namespace=censys-cloud-connectors
```

3. Create a Kubernetes secret for the Environment Variables from the `.env` file

```
$ kubectl create secret generic censys-cloud-connectors-env    --from-env-file=.env    --dry-run=client
```

4. Create a Kubernetes secret for the Censys Cloud Connector `providers.yml` file

The chart will look for a secret named `censys-cloud-connectors-providers` in the `censys-cloud-connectors` namespace. The secret should contain a file named `providers.yml` with the contents of your `providers.yml` file.

```
$ kubectl create secret generic censys-cloud-connectors-providers    --from-file=providers.yml    --dry
```

5. (Optional) Create a Kubernetes secret for the Censys Cloud Connector `secrets` directory

   Note

   This step is required if you are scanning Google Cloud Platform.

   If you choose to use this method, you will need to uncomment the `credentialsSecretName` value in the `values.yaml` file which should be set to `censys-cloud-connectors-secrets`.

```
$ kubectl create secret generic censys-cloud-connectors-secrets    --from-file=secrets    --dry-run=cli
```

6. (Optional) Modify the `values.yaml` file to customize the deployment

This is the place to customize the schedule of the Censys Cloud Connector, the default is to run every 4 hours. We recommend that you do not run the Censys Cloud Connector more frequently than every hour. For assistance with writing the cron schedule, please see the Crontab Guru website.

See the *Configuration* section for more information on the available configuration options.

7. Install the Censys Cloud Connector Chart

```
$ helm upgrade --install censys-cloud-connectors ./kubernetes/censys-cloud-connectors
```

8. Run the Censys Cloud Connector Manually

```
$ kubectl create job --from=cronjob/censys-cloud-connectors censys-cloud-connectors-manual --dry-run=cli
```

9. Check the logs of the Censys Cloud Connector Job

```
$ kubectl logs job.batch/censys-cloud-connectors-manual --follow
```

5.4.3 Configuration

The following table describes the available configuration options for the Censys Cloud Connector Chart.

Key	Description
`envSecretName`	The name of the secret containing the .env file.
`providersSecretName`	The name of the secret containing the providers.yml file.
`nameOverride`	(Optional) The override for the name of the chart.
`fullnameOverride`	(Optional) The override for the fullname (including release name) of the chart.
`imagePullSecrets`	(Optional) The authorization token to use when accessing the docker registry.
`image.repository`	The image for the censys-cloud-connector container.
`image.pullPolicy`	(Optional) Overrides the image pull policy.
`image.tag`	(Optional) Overrides the image tag whose default is latest.
`cronjob.schedule`	(Optional) The interval at which the censys-cloud-connector container will run (in cron format). Defaults to every 4 hours.
`cronjob.concurrencyPolicy`	(Optional) The concurrency policy for the cronjob.
`podAnnotations`	(Optional) The annotations to add to the pod.
`podSecurityContext`	(Optional) The security context to add to the pod.
`securityContext`	(Optional) The security context to add to the container.
`resources`	(Optional) The resources to allocate to the container.
`nodeSelector`	(Optional) The node selector to use when scheduling the pod.
`tolerations`	(Optional) The tolerations to use when scheduling the pod.
`affinity`	(Optional) The affinity to use when scheduling the pod.

5.4.4 Upgrading

To upgrade the Censys Cloud Connector Chart, ensure that you have the latest version of the chart and run the following command:

```
$ helm upgrade --install censys-cloud-connectors ./kubernetes/censys-cloud-connectors
```

5.4.5 Uninstalling

To uninstall the Censys Cloud Connector Chart, run the following command:

```
$ helm uninstall censys-cloud-connectors
```

You can also delete the Censys Cloud Connector namespace:

```
$ kubectl delete namespace censys-cloud-connectors
```

5.4.6 Troubleshooting

The Censys Cloud Connector is not running

If the Censys Cloud Connector is not running, you can check the logs of the Censys Cloud Connector Job to see if there are any errors.

```
$ kubectl logs job.batch/censys-cloud-connectors-manual --follow
```

The Censys Cloud Connector is not able to access the `.env` file

If you see an error similar to the following, it means that the Censys Cloud Connector is not able to access the `.env` file.

```
$ ERROR:censys_cloud_connectors: n validation error for Settings
$ ...
```

This means that the `envSecretName` value in the `values.yaml` file is either incorrect or the secret does not contain the `.env` file. You may also be provided with a more specific error message indicating which environment variable is missing or invalid.

The Censys Cloud Connector is not able to access the `providers.yml` file

If you see an error similar to the following, it means that the Censys Cloud Connector is not able to access the `providers.yml` file.

```
 Error: [Errno 2] No such file or directory: '/providers/providers.yml'
```

This means that the `providersSecretName` value in the `values.yaml` file is or the secret does not contain the `providers.yml` file.

The Censys Cloud Connector is not able to access the `secrets` directory

If you see an error similar to the following, it means that the Censys Cloud Connector is not able to access the `secrets` directory.

```
Error: [Errno 2] No such file or directory: 'secrets/<file>'
```

This means that the `secretsSecretName` value in the `values.yaml` file is either incorrect or the secrets directory does not contain the required files.

My issue is not listed here

If your issue is not listed here, please contact Censys Support.

5.5 Local Deployment

5.5.1 Run the Connector

To run the connector, you can use the command line interface. The scan command is:

```
$ poetry run censys-cc scan
```

The *censys-cc scan* command will enumerate the configured cloud providers and scan the resources. The scan command will submit the public cloud assets to Censys ASM as Seeds and Cloud Assets.

5.5.2 Additional Options

You can set a scheduled interval for the connector to run on with the flag `--daemon`. This option takes in a time interval in hours. If you do not specify an interval, the default will be set to 1 hour.

```
$ censys-cc scan --daemon         # Run every 1 hour
$ censys-cc scan --daemon 1.5     # Run every 1.5 hours
```

5.6 Picking a Deployment Method

After successfully completing *Provider Setup*, choose a deployment method to run the cloud connector on a schedule.

The Censys Unified Cloud Connector can be deployed in a variety of ways. The following table provides a high-level overview of the different deployment methods available.

Deployment Method	Description	Pros	Cons
AWS ECS Task	Run the connector in an AWS ECS Task.	- Easy to deploy and maintain. - Leverage the power of AWS ECS. - Can be deployed to AWS.	- Requires an AWS account. - Requires the `providers.yml` file and the `secrets` directory to be stored in AWS Secrets Manager.
Google Scheduled Function	Run the connector in a Google Scheduled Function.	- Easy to deploy and maintain. - Leverage the power of Google Cloud Functions. - Can be deployed to Google Cloud.	- Requires a Google Cloud account. - Requires the `providers.yml` file and the `secrets` directory to be stored in Google Secret Manager.
Docker	Run the connector in a Docker container.	- Easily deployable on any server with Docker installed.	- Requires Docker to be installed on the server. - Requires the `providers.yml` file and the `secrets` directory to be mounted as volumes.
Kubernetes	Run the connector in a Kubernetes cluster.	- Leverage the power of Kubernetes CronJobs. - Can be deployed to a variety of cloud providers.	- Requires a Kubernetes cluster to be deployed.
Local Deployment	Run the connector in a local environment.	- Good for testing. - Doesn't require external infrastructure.	- Not scalable. - Doesn't make use of IaaS best practices.

5.7 Confirm Results

Visit the Seed Data Page and the Storage Buckets Page to confirm that you're seeing seeds and storage buckets from your cloud provider(s).

CHAPTER

SIX

COMMAND LINE INTERFACE

6.1 censys-cc

```
usage: censys-cc [-h] [-v] {config,scan} ...
```

-h, --help
show this help message and exit

-v, --version
display version

6.1.1 censys-cc config

Configure Censys Cloud Connectors

```
usage: censys-cc config [-h] [-p [PROVIDER]]
```

-h, --help
show this help message and exit

-p {aws,azure,gcp}, **--provider** {aws,azure,gcp}
specify a cloud service provider: [‘aws’, ‘azure’, ‘gcp’]

6.1.2 censys-cc scan

Scan with Censys Cloud Connectors

```
usage: censys-cc scan [-h] [-p PROVIDER [PROVIDER ...]] [-d [SCAN_INTERVAL]]
```

-h, --help
show this help message and exit

-p {aws,azure,gcp}, **--provider** {aws,azure,gcp}
specify one or more cloud service provider(s): [‘aws’, ‘azure’, ‘gcp’]

-d <scan_interval>, **--daemon** <scan_interval>
run on a scheduled interval (must be greater than or equal to 1 hour)

CHAPTER

SEVEN

FAQ

7.1 General

7.1.1 My Python Version is Not Compatible

It is highly recommended that a Python version shim like pyenv is used. Once installed, Poetry will make a virtualenv using the correct version of Python automatically.

7.2 AWS

7.2.1 AWS Policy Actions

The following permissions are required to scan:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "censysLeastPrivilegeCloudConnector",
      "Effect": "Allow",
      "Action": [
        "apigateway:GET",
        "ec2:DescribeTags",
        "ec2:DescribeNetworkInterfaces",
        "ecs:ListContainerInstances",
        "ecs:ListClusters",
        "elasticloadbalancing:DescribeLoadBalancers",
        "rds:DescribeDBInstances",
        "route53:ListHostedZones",
        "route53:ListResourceRecordSets",
        "route53domains:ListDomains",
        "s3:GetBucketLocation",
        "s3:ListAllMyBuckets",
        "s3:ListBucket"
      ],
      "Resource": "*"
    }
  ]
}
```

7.2.2 Can I use a Session Role Name?

Yes, this can be set during the provider setup and will be defined in `providers.yml`.

7.2.3 Do you support Named Profiles?

Yes.

7.2.4 Can I use SSO?

AWS CLI supports Single Sign-On via IAM Identity Center. You can use the `aws sso login` command to authenticate before running provider setup.

7.3 Azure

7.3.1 Azure Roles

Read about Azure roles and permissions *here*.

If you see the following error message, check that you are logged into an account with the correct permissions:

```
The client 'user@example.com' with object id 'uuid' does not have authorization to␣
↪perform action 'Microsoft.Authorization/roleAssignments/write' over scope '/
↪subscriptions/uuid' or the scope is invalid. If access was recently granted, please␣
↪refresh your credentials.
```

7.4 GCP

7.4.1 GCP Service Account Keys

If you encounter the following error while configuring your GCP Cloud Connector, a likely cause is that your service account has reached its maximum quota of keys.

```
Failed to enable service account. ERROR: (gcloud.iam.service-accounts.keys.create)␣
↪FAILED_PRECONDITION: Precondition check failed.
```

Go to https://console.cloud.google.com/iam-admin/serviceaccounts to manage your service account keys.

INDEX

Symbols

C

E