Censys Unified Cloud Connector

Censys, Inc.

Nov 16, 2022

CONTENTS

CHAPTER

ONE

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.

1.1 Prerequisites

- Python 3.9+
- Pip
- Poetry

1.2 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
```

Install the dependencies

```
$ poetry install
```

Copy .env.sample to .env

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

1.3 Environment Variables

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

To learn more about the environment variables, see *Environment Variables*.

1.4 Configuration

Note: Before configuring the connector, make sure you are logged in to your cloud provider's CLI tool. See our *Supported Providers* for more information.

To configure the connector, you can use the command line interface. The base command is *censys-cc*. 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 *Provider Configuration*. This file can contain multiple provider configurations.

You have successfully configured your cloud connector if your *Provider Configuration* is populated with your credentials.

1.5 Running 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.

1.6 Deploying the Connector

The connector can be deployed to a variety of environments. We have provided several deployment methods. See *Deployment Methods* for more information.

1.7 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).

1.8 Additional options

- You can specify one or more providers in the command line with the flag `--provider`. The connector will only scan for assets from the specified providers.
- 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
```

CHAPTER

TWO

DEPLOYMENT METHODS

2.1 AWS Elastic Container Service (ECS) Task

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

2.1.1 Prerequisites

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

2.1.2 Login Instructions

Use the AWS CLI tool to configure a named profile. You can set the profile to use with the variable `aws_profile`. This can be defined using a Terraform variable definition file.

2.1.3 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
```

2.1.4 Cleanup

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

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

2.1.5 Requirements

Name	Version
terraform	>= 0.13
aws	~> 3.0

2.1.6 Providers

Name	Version
aws	3.75.2
random	3.3.2

2.1.7 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

2.1.8 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

2.1.9 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

2.1.10 Outputs

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

2.2 Google Cloud Scheduled Function

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

2.2.1 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
  ```

2.2.2 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
   ```

2.2.3 Cleanup

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

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

2.2.4 Requirements

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

2.2.5 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

2.2.6 Modules

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

2.2.7 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

2.2.8 Inputs

Name	Description	Type	Default	Required
bucket_force_destroy	When destroying 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)	[“.gitignore”]	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”, “securitycenter.googleapis.com”]	no
job_description	Addition 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({ name	null	no

2.2.9 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

2.3 Docker Deployment Methods

2.3.1 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-off)

```
$ 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.
```

2.3.2 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.

2.4 Kubernetes Deployment Method

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

2.4.1 Prerequisites

The following prerequisites are required to deploy using Kubernetes:

- A Kubernetes cluster
- Helm
- Kubectl

2.4.2 Getting Started

1. Install the Censys Cloud Connector Chart

```
helm install censys-cloud-connectors ./kubernetes/censys-cloud-connectors --namespace␣
↪YOUR_NAMESPACE
```

- To upgrade the Censys Cloud Connector Chart:

  ```
  helm upgrade censys-cloud-connectors ./kubernetes/censys-cloud-connectors --
  ↪namespace YOUR_NAMESPACE
  ```

2.5 Picking a Deployment Method

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
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.
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.

CHAPTER

THREE

ENVIRONMENT VARIABLES

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`

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

# Censys API Settings
```

(continues on next page)

(continued from previous page)

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

CHAPTER FOUR

PROVIDER CONFIGURATION

The `providers.yml` file contains the configuration for all cloud providers. The file is a YAML file and is structured as follows:

Note: You will need to have generated your `providers.yml` file using the *censys-cc config* command before you can run the connector.

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

(continues on next page)

(continued from previous page)

```
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

SUPPORTED PROVIDERS

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.

5.1 AWS Provider Setup

5.1.1 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

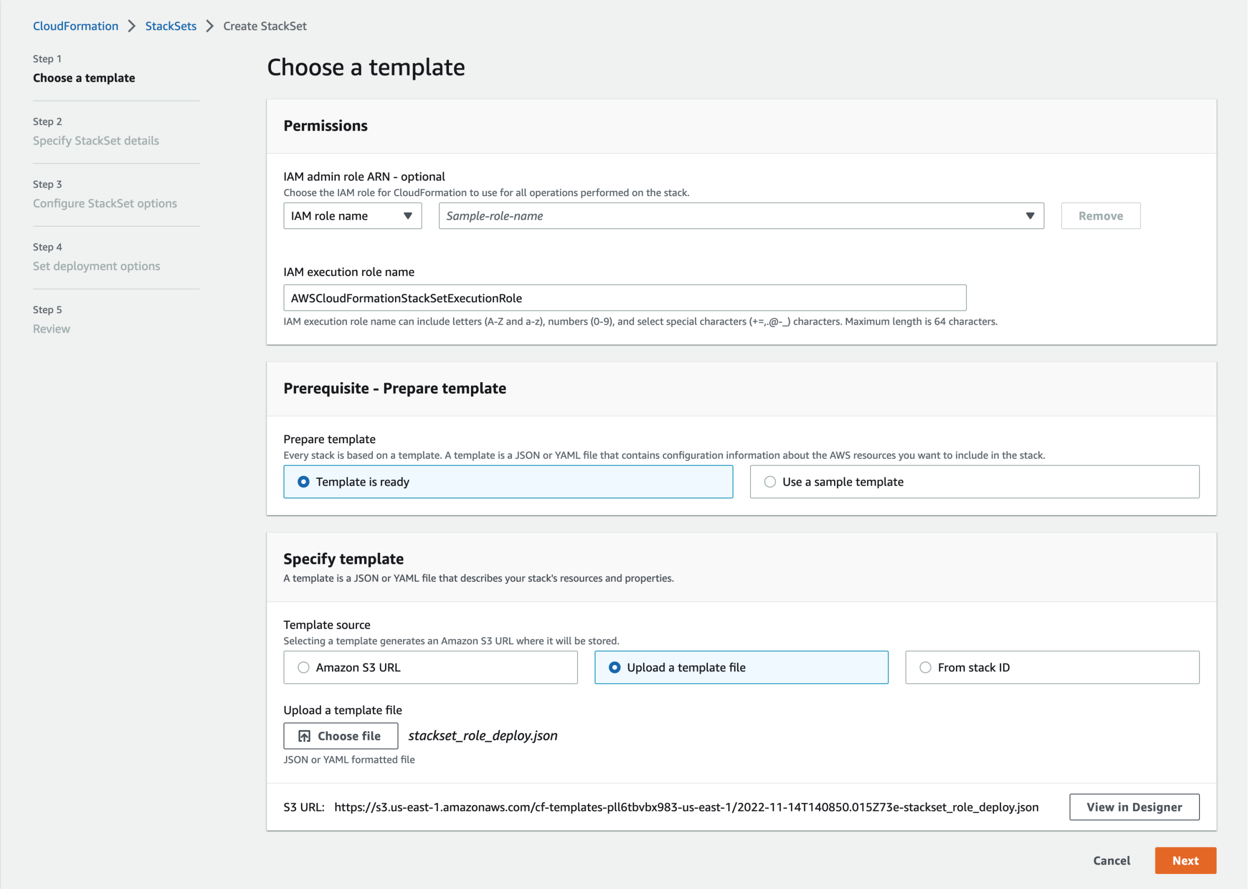
Log in to your Primary AWS account and navigate to Cloud Formation.

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 in *step 1*.

Click **Next**.



Choose a template

Permissions

IAM admin role ARN - optional
Choose the IAM role for CloudFormation to use for all operations performed on the stack.
IAM role name ▼ | *Sample-role-name* ▼ | Remove

IAM execution role name
AWSCloudFormationStackSetExecutionRole
IAM execution role name can include letters (A-Z and a-z), numbers (0-9), and select special characters (+=,.@-_) characters. Maximum length is 64 characters.

Prerequisite - Prepare template

Prepare template
Every stack is based on a template. A template is a JSON or YAML file that contains configuration information about the AWS resources you want to include in the stack.
● Template is ready ○ Use a sample template

Specify template
A template is a JSON or YAML file that describes your stack's resources and properties.

Template source
Selecting a template generates an Amazon S3 URL where it will be stored.
○ Amazon S3 URL ● Upload a template file ○ From stack ID

Upload a template file
Choose file *stackset_role_deploy.json*
JSON or YAML formatted file

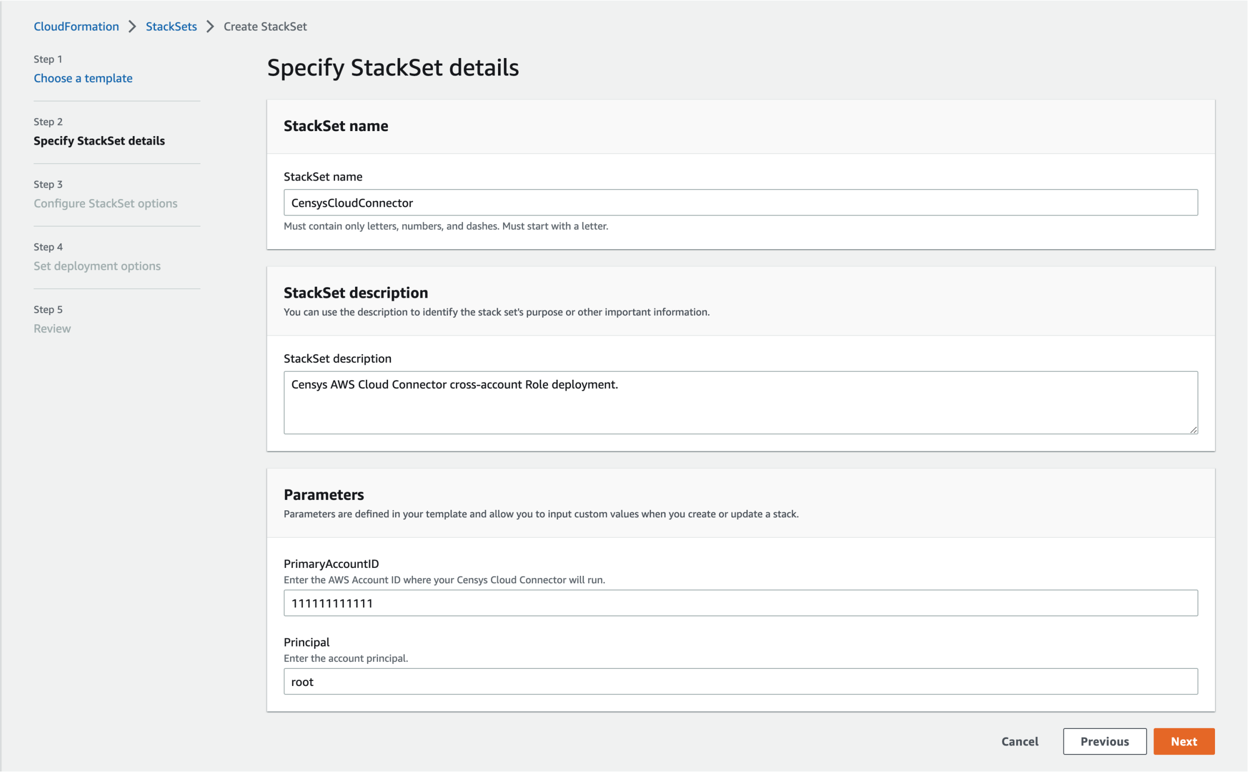
S3 URL: https://s3.us-east-1.amazonaws.com/cf-templates-pll6tbvbx983-us-east-1/2022-11-14T140850.015Z73e-stackset_role_deploy.json

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**.



Specify StackSet details

StackSet name

StackSet name

CensysCloudConnector

Must contain only letters, numbers, and dashes. Must start with a letter.

StackSet description

You can use the description to identify the stack set's purpose or other important information.

StackSet description

Censys AWS Cloud Connector cross-account Role deployment.

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

PrimaryAccountID

Enter the AWS Account ID where your Censys Cloud Connector will run.

111111111111

Principal

Enter the account principal.

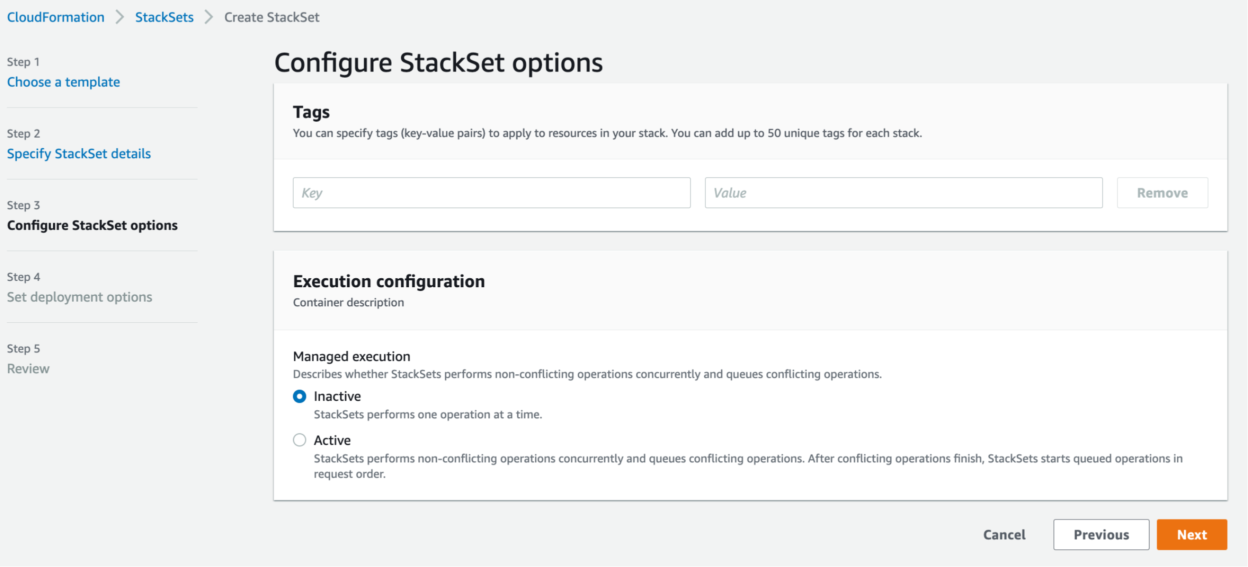
root

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**.



Configure StackSet options

Tags

You can specify tags (key-value pairs) to apply to resources in your stack. You can add up to 50 unique tags for each stack.

Execution configuration

Container description

Managed execution

Describes whether StackSets performs non-conflicting operations concurrently and queues conflicting operations.

● Inactive
StackSets performs one operation at a time.

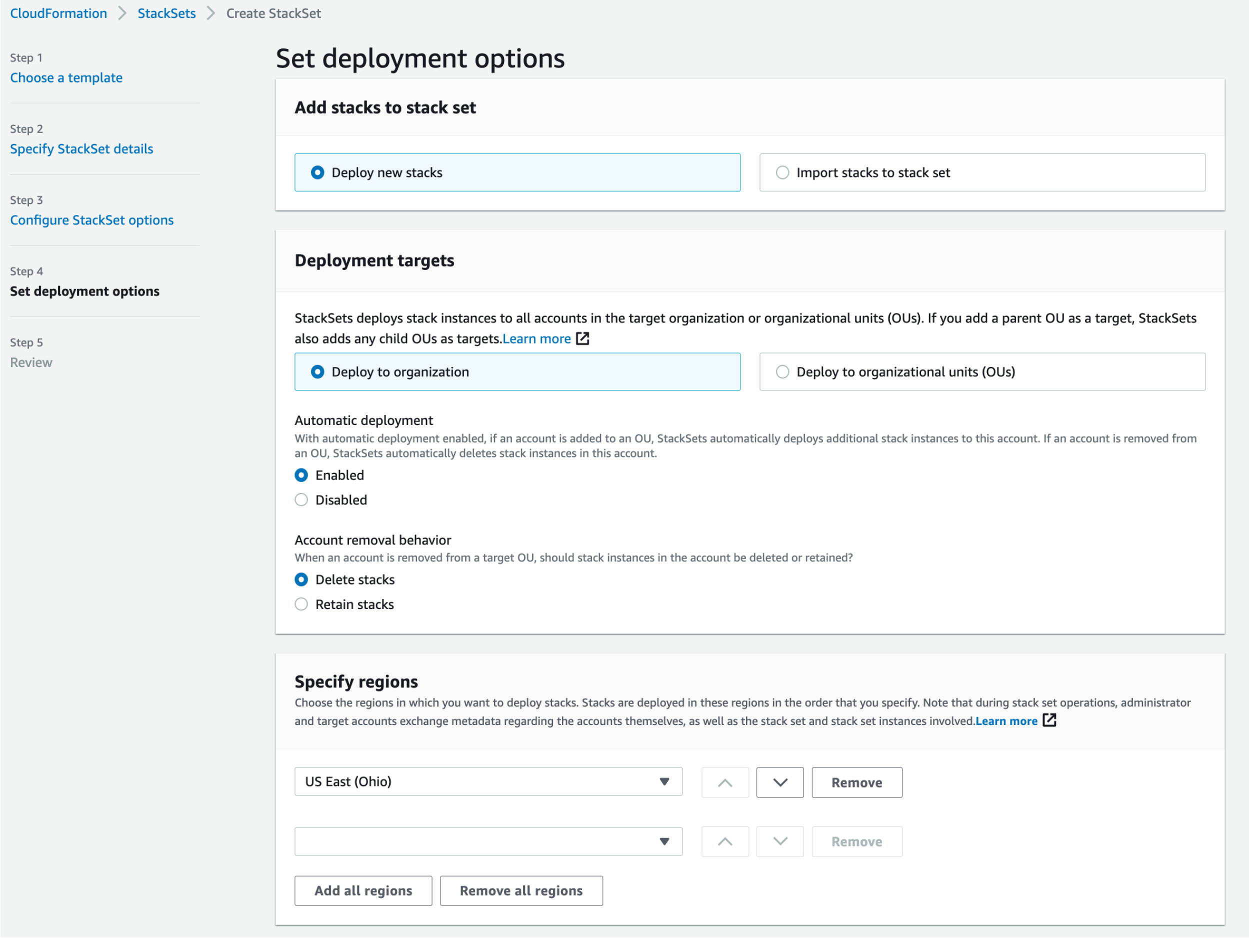
○ Active
StackSets performs non-conflicting operations concurrently and queues conflicting operations. After conflicting operations finish, StackSets starts queued operations in request order.

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**.



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

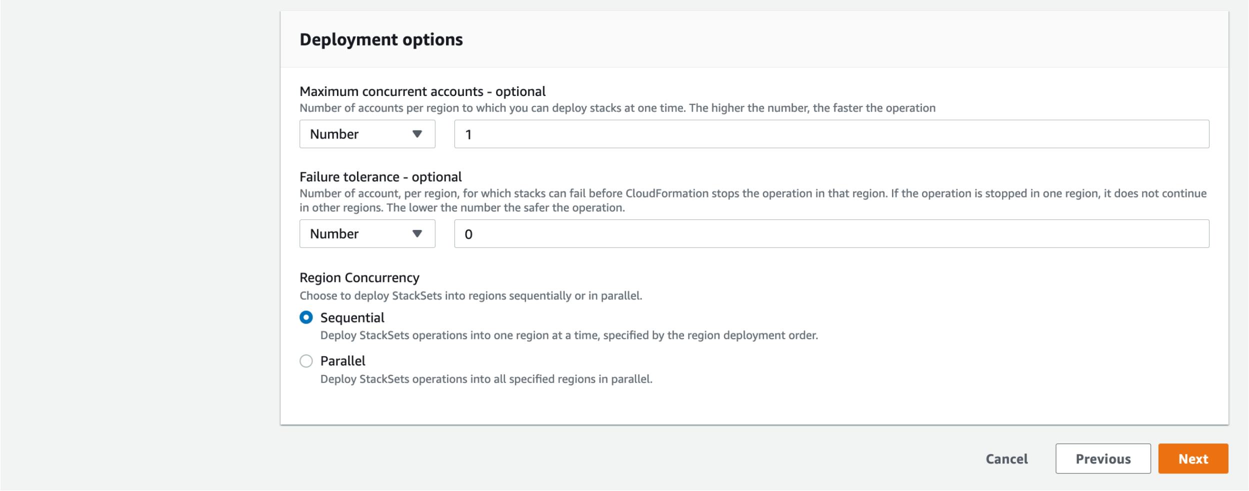
Account removal behavior
When an account is removed from a target OU, should stack instances in the account be deleted or retained?

● 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)



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

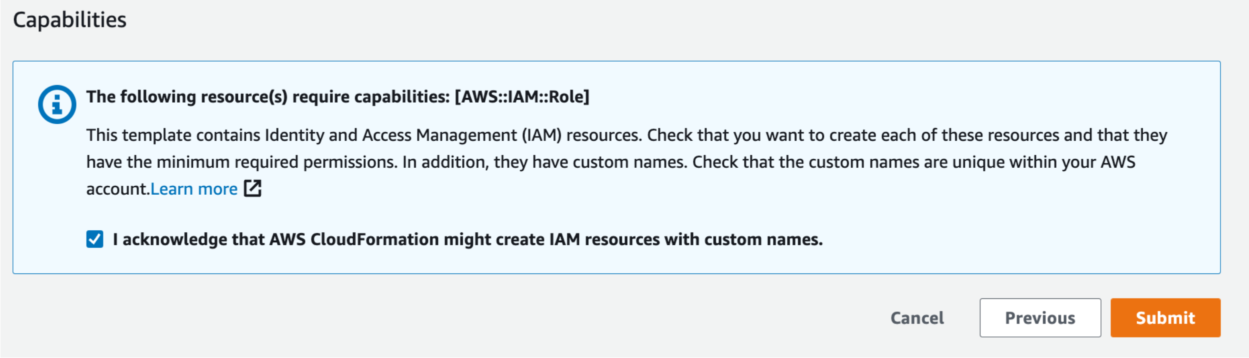
Choose to deploy StackSets into regions sequentially or in parallel.

● 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.

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.



Capabilities

The following resource(s) require capabilities: [AWS::IAM::Role]

This template contains Identity and Access Management (IAM) resources. Check that you want to create each of these resources and that they have the minimum required permissions. In addition, they have custom names. Check that the custom names are unique within your AWS account.Learn more

☑ **I acknowledge that AWS CloudFormation might create IAM resources with custom names.**

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.

5.1.2 Templates

IAM

To dynamically find accounts by StackSet, `cloudformation:ListStackInstances` is required.

Provider Setup Policy

This policy contains roles that might be used during provider setup.

download

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "censysCloudConnectorProviderSetup",
      "Effect": "Allow",
      "Action": [
        "sts:GetCallerIdentity",
        "organizations:ListAccounts",
        "cloudformation:ListStackInstances"
      ],
      "Resource": "*"
    }
  ]
}
```

Least Privilege Policy

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

Recommended Policy

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. Additional policy

download

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

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-Z0-9]{1,64}",
      "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.",
  "Resources": {
    "CensysCloudConnectorSetup": {
      "Type": "AWS::IAM::Role",
```

(continues on next page)

(continued from previous page)

```
   "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"],
    "Path": "/",
    "RoleName": "CensysCloudConnectorRole"
   }
  }
 }
}
```

5.1.3 Prerequisites

- Install the AWS CLI
- Configure the AWS CLI
- *Configure* Cloud Connector IAM
- Optional: Define a named profile

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

5.1.4 Overview

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

- Select a credential profile allows you to choose which named profile to use during provider setup.
  - You can optionally save the profile's credentials to `providers.yml`
- Define a role name to use STS Assume Role. This enables running the connector without defining an access or secret key.
  - When using a role, AWS recommends using a Session Role Name. Typically, you pass the name or identifier that is associated with the user who is using your application. That way, the temporary security credentials that your application will use are associated with that user.
- If your organization has multiple accounts, provider setup will give an option to find and load these accounts into `providers.yml`. The find accounts feature has two ways to look up accounts:

  - Find accounts with a CloudFormation StackSet Instance
  - Find accounts using Organization List Accounts

5.1.5 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

5.1.6 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 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`.

Find Accounts by StackSet

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.

Account Specific Roles

If you are utilizing multiple accounts in `providers.yml`, it's possible to configure roles that are unique to each account.

```yaml
- provider: aws
  account_number: 111 # <- primary account
  role_name: SharedRole
  accounts:
  - account_number: 222
  - account_number: 333
    role_name: Role333
  - account_number: 444
    role_name: Role444
```

In this example, account 222 will inherit the role `SharedRole`. Account 333 will overwrite the parent role with `Role333`.

5.1.7 Configure Cloud Connector IAM

The Censys Cloud Connector has a set of *minimum required permissions*. These permissions can be applied through standard IAM configuration. As a security best-practice, the connector also supports creation of temporary credentials via Secure Token Service (STS).

Censys also maintains a CloudFormation *StackSet template* that will deploy a `CensysCloudConnectorRole` role to all of your AWS accounts. The StackSet can also be used to list all of your accounts.

StackSet Deployment

See *StackSet Deployment* for a walk-through of how to install the Censys Cloud Connector StackSet in your account.

5.1.8 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.

5.2 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)

5.3 Azure Cloud

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

5.4 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

5.5 Authenticating

Log in to your cloud provider's CLI tool using the following commands:

- AWS CLI: Credentials are stored on your machine, making authentication unnecessary. See *AWS Provider Setup* for more information.
- Azure CLI: `az login`
- Google's gcloud CLI: `gcloud auth login`

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 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 Policy Actions

The following permissions are required to scan:

```json
{
  "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": "*"
    }
  ]
}
```

7.3 Azure Roles

Ensure the account's Access control (IAM) 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 with this service principal:

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

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 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