CSA | Bangalore Chapter

Meetup - Nov 2024
In affiliation with

Flipkart

Cloud

.. bridging Azure, GCP, and Kubernetes

Synergies

Myntra Office, Kadubeesanahalli,
Bengaluru,

16 Nov 2024
09:00 AM - 4:30 PM

Meetup - Nov 2024
In affiliation with

Policy as Code for Cloud Security

Anjali S

Senior Security Engineer

Flipkart

Agenda

- Introduction
- Why do we need Policy as Code ?
- Implementing PaC in Cloud
- Azure Policy
- GCP Organisation Policy & Policy Analyser
- Kubernetes Kyverno

Introduction

- Policies are set of rules, instructions or guidelines set to run infrastructure in secure way. Using code to enforce or implement these policies is Policy as Code.
- We use yaml, json, rego(OPA) etc.

Image Ref: https://www.styra.com

Why to use PaC in Cloud Security

- × Manual approach
- ✓ Codified
- ✓ Automated and Shift left process.
- ✓ Fast track the process
- ✓ Version Controlled, helps in keeping track
- ✓ Increases visibility
- ✓ Misconfiguration are identified is early stages

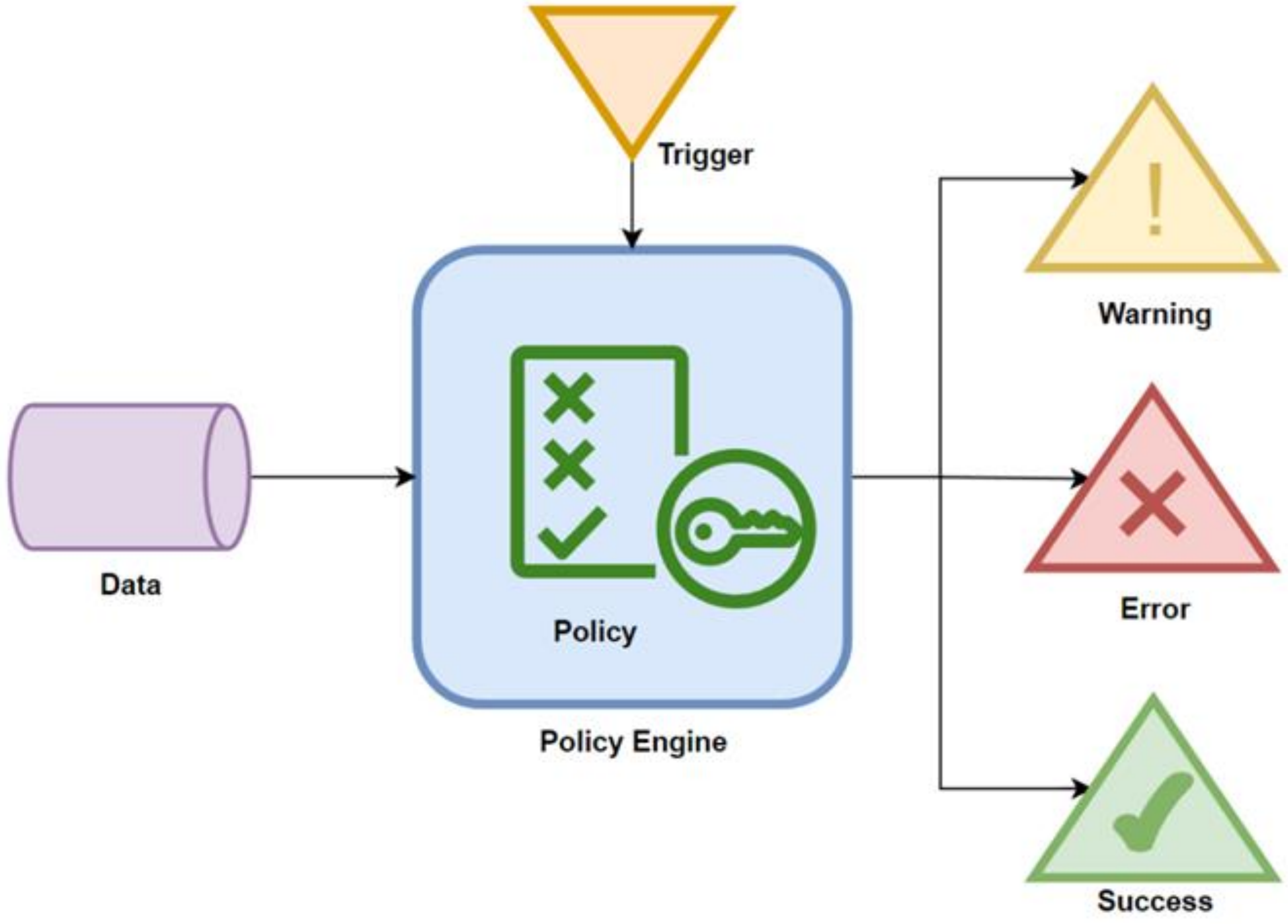
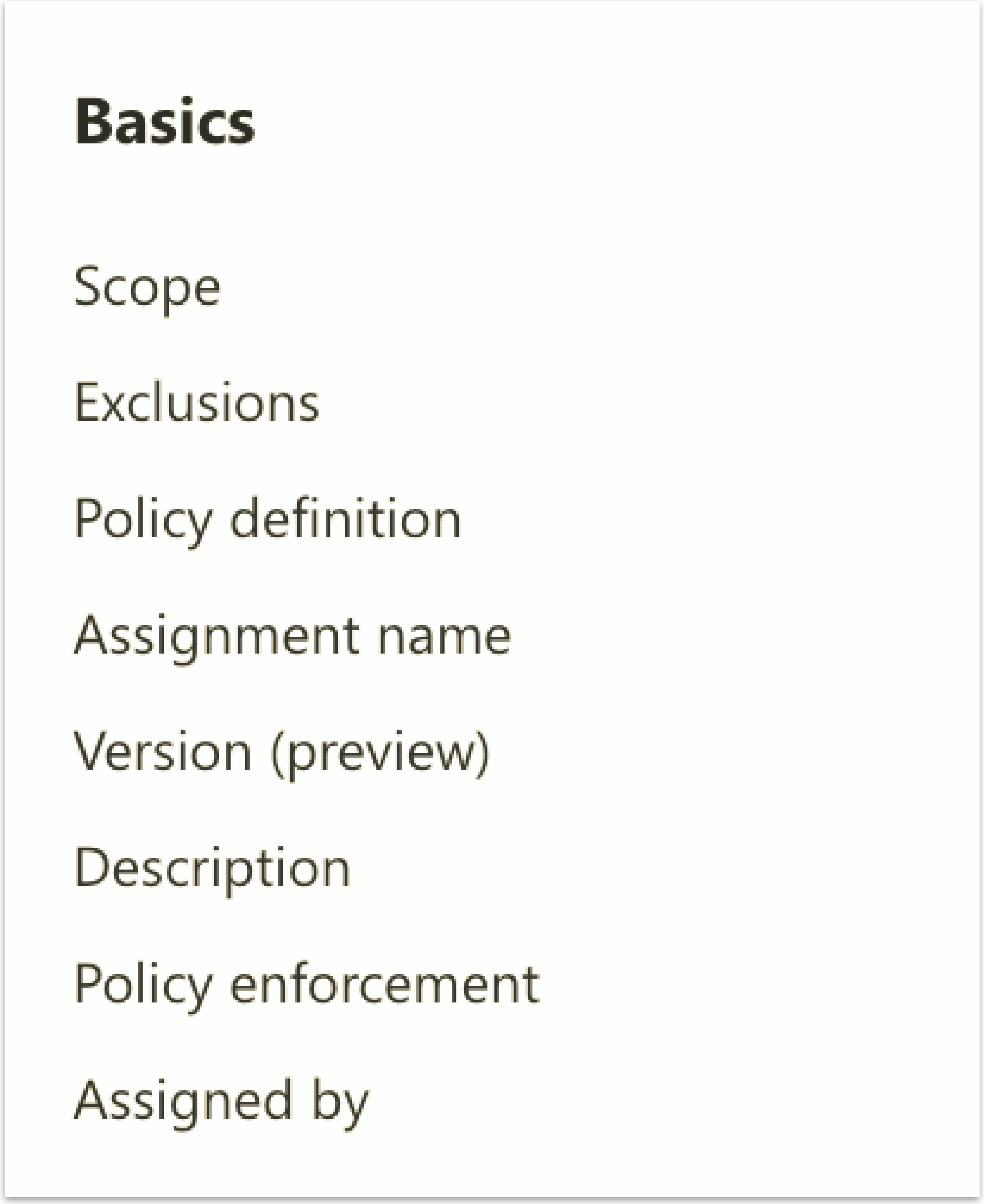


Image Ref: https://crowdstrike.com

Implementing PaC in Cloud

Cloud Provider	Policy Type
AZURE	• Azure Policy • Azure Blueprints
GCP	• Organization Policy • Policy Analyser
AWS	• Service Control Policy(SCP) • AWS Config Rule
Kubernetes	• Open Policy Agent (OPA) • Kyverno

Azure Policy: Environment Tag is enabled



Basics

Scope

Exclusions

Policy definition

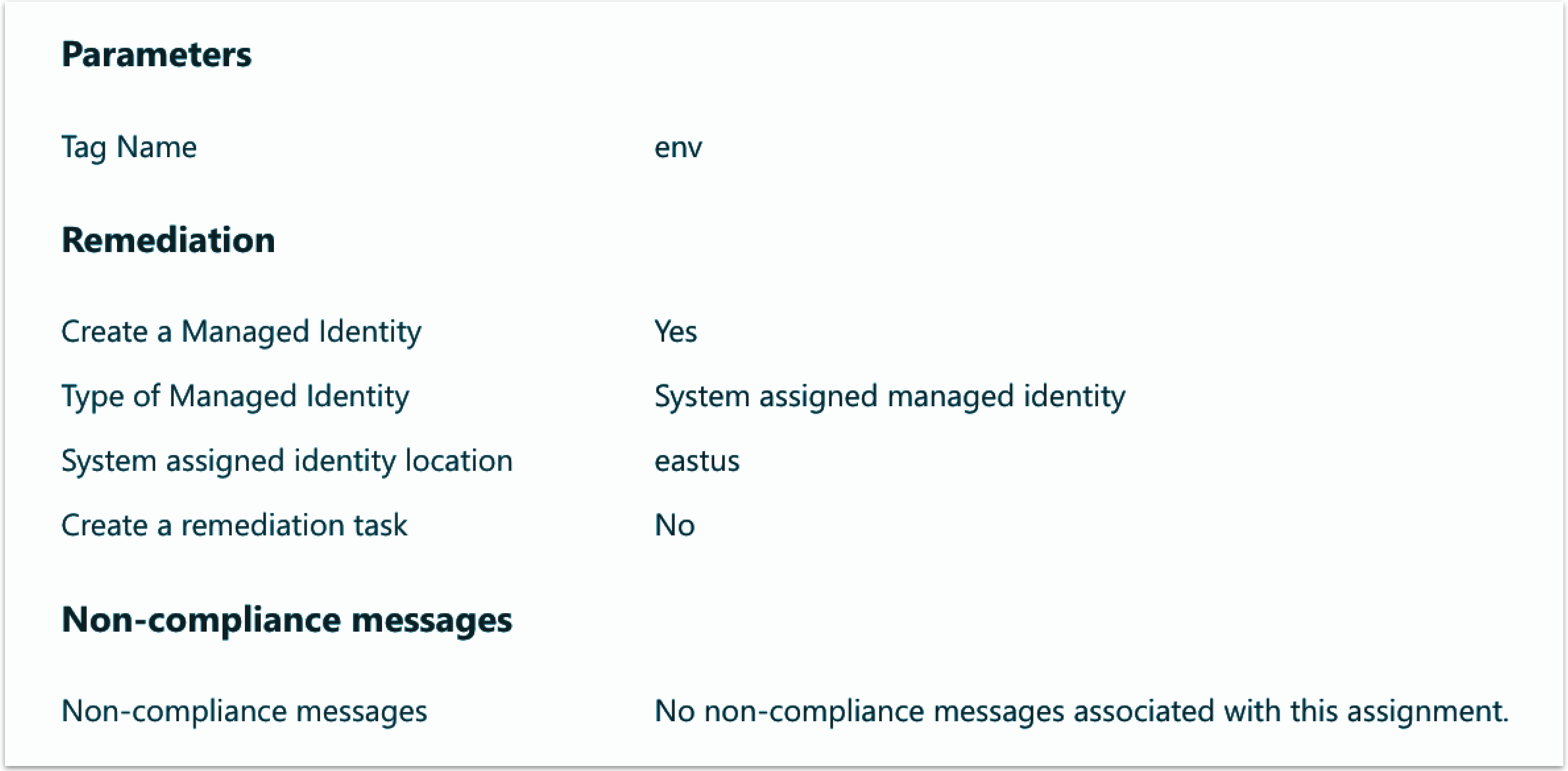
Assignment name

Version (preview)

Description

Policy enforcement

Assigned by



Parameters

Tag Name	env

Remediation

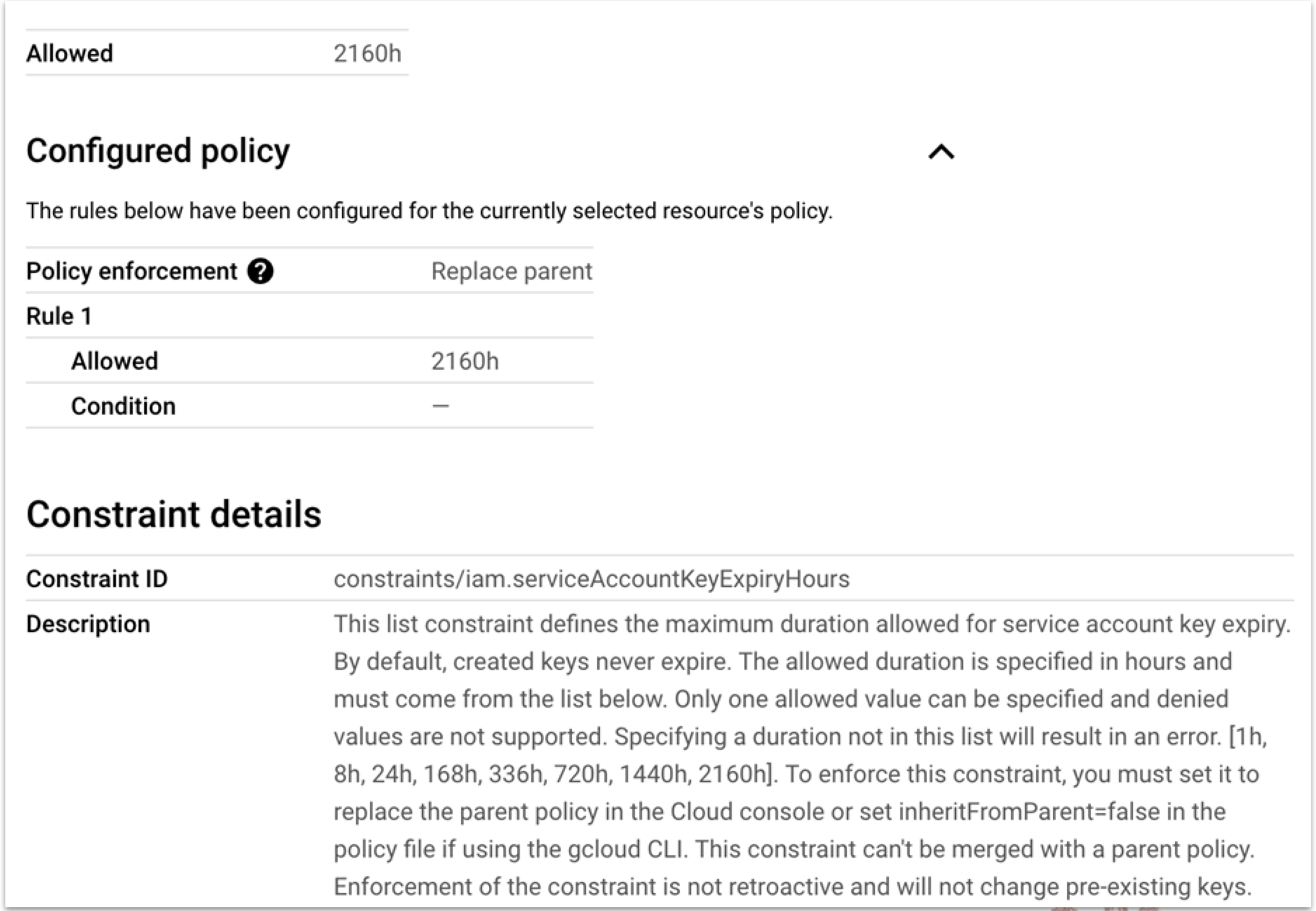
Create a Managed Identity	Yes
Type of Managed Identity	System assigned managed identity
System assigned identity location	eastus
Create a remediation task	No

Non-compliance messages

Non-compliance messages	No non-compliance messages associated with this assignment.

GCP Organisation Policy: Service account key expiry duration in hours

- SA Json key does not have expiry date by default.
- Restrict it with Organisation policy
- On Enforcing this policy, SA key generated will automatically expired post 2160 hours.
- Can be automated with Terraform in CI/CD pipeline



Allowed	2160h

Configured policy

The rules below have been configured for the currently selected resource's policy.

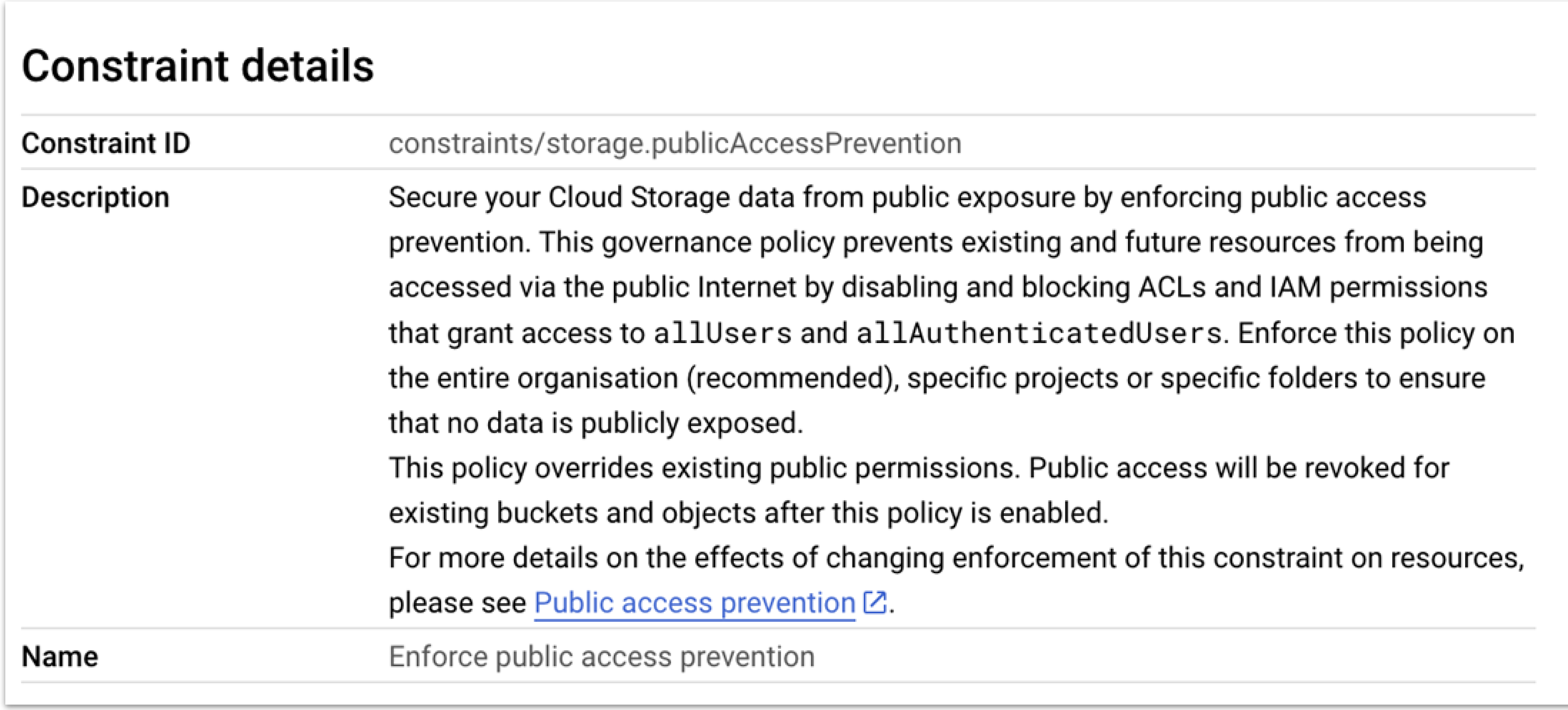
Policy enforcement ?	Replace parent
Rule 1	
Allowed	2160h
Condition	—

Constraint details

Constraint ID	constraints/iam.serviceAccountKeyExpiryHours
Description	This list constraint defines the maximum duration allowed for service account key expiry. By default, created keys never expire. The allowed duration is specified in hours and must come from the list below. Only one allowed value can be specified and denied values are not supported. Specifying a duration not in this list will result in an error. [1h, 8h, 24h, 168h, 336h, 720h, 1440h, 2160h]. To enforce this constraint, you must set it to replace the parent policy in the Cloud console or set inheritFromParent=false in the policy file if using the gcloud CLI. This constraint can't be merged with a parent policy. Enforcement of the constraint is not retroactive and will not change pre-existing keys.

GCP Organisation Policy: Enforce public access prevention

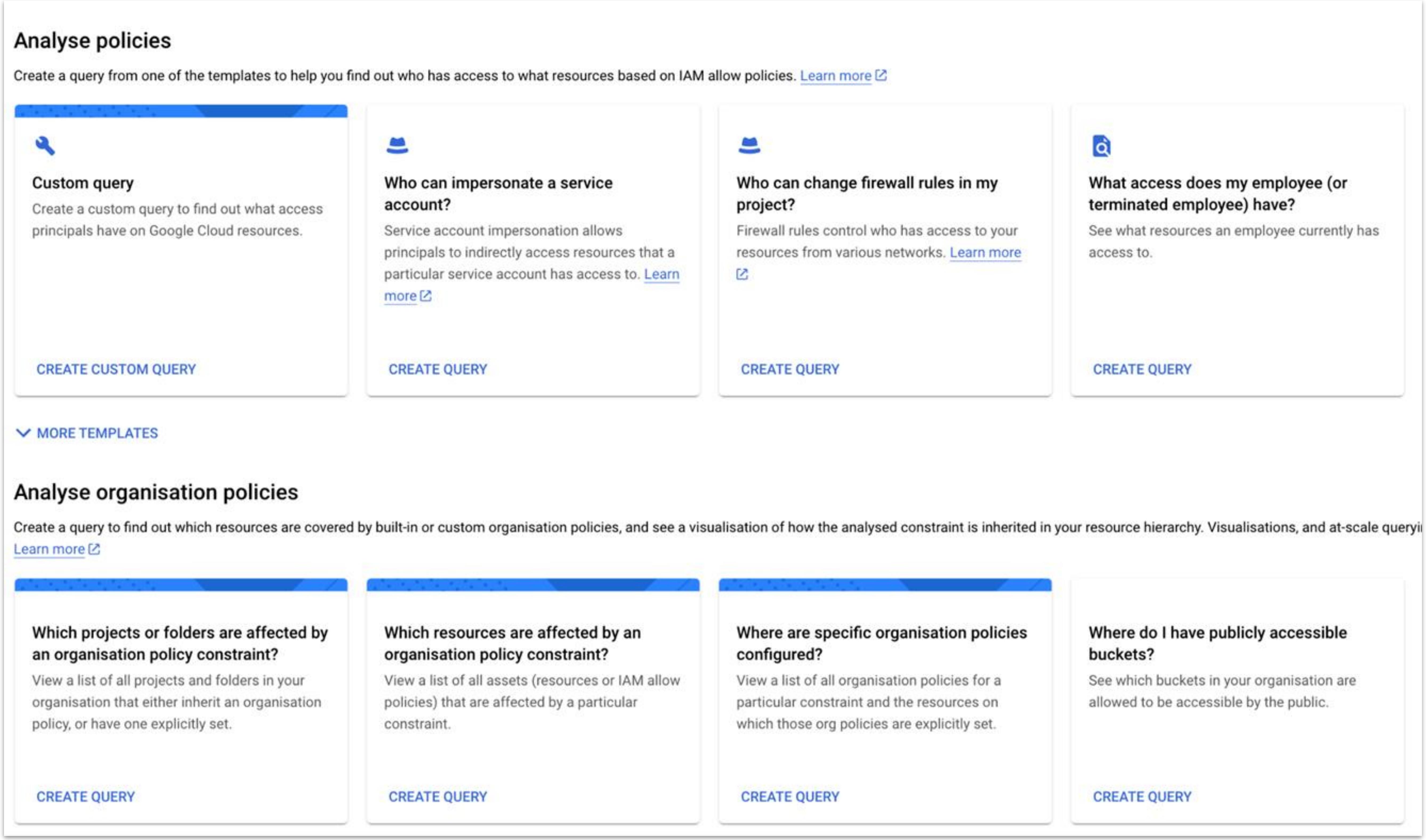
- Public access is restricted over GCS buckets.
- allUsers & allAuthenticatedUsers access is disabled
- Exception can be added
- Applied to existing and newer ones.



Constraint details

Constraint ID	constraints/storage.publicAccessPrevention
Description	Secure your Cloud Storage data from public exposure by enforcing public access prevention. This governance policy prevents existing and future resources from being accessed via the public Internet by disabling and blocking ACLs and IAM permissions that grant access to `allUsers` and `allAuthenticatedUsers`. Enforce this policy on the entire organisation (recommended), specific projects or specific folders to ensure that no data is publicly exposed. This policy overrides existing public permissions. Public access will be revoked for existing buckets and objects after this policy is enabled. For more details on the effects of changing enforcement of this constraint on resources, please see Public access prevention.
Name	Enforce public access prevention

GCP Policy Analyser



Analyse policies

Create a query from one of the templates to help you find out who has access to what resources based on IAM allow policies. Learn more

Custom query

Create a custom query to find out what access principals have on Google Cloud resources.

CREATE CUSTOM QUERY

Who can impersonate a service account?

Service account impersonation allows principals to indirectly access resources that a particular service account has access to. Learn more

CREATE QUERY

Who can change firewall rules in my project?

Firewall rules control who has access to your resources from various networks. Learn more

CREATE QUERY

What access does my employee (or terminated employee) have?

See what resources an employee currently has access to.

CREATE QUERY

MORE TEMPLATES

Analyse organisation policies

Create a query to find out which resources are covered by built-in or custom organisation policies, and see a visualisation of how the analysed constraint is inherited in your resource hierarchy. Visualisations, and at-scale queryi
Learn more

Which projects or folders are affected by an organisation policy constraint?

View a list of all projects and folders in your organisation that either inherit an organisation policy, or have one explicitly set.

CREATE QUERY

Which resources are affected by an organisation policy constraint?

View a list of all assets (resources or IAM allow policies) that are affected by a particular constraint.

CREATE QUERY

Where are specific organisation policies configured?

View a list of all organisation policies for a particular constraint and the resources on which those org policies are explicitly set.

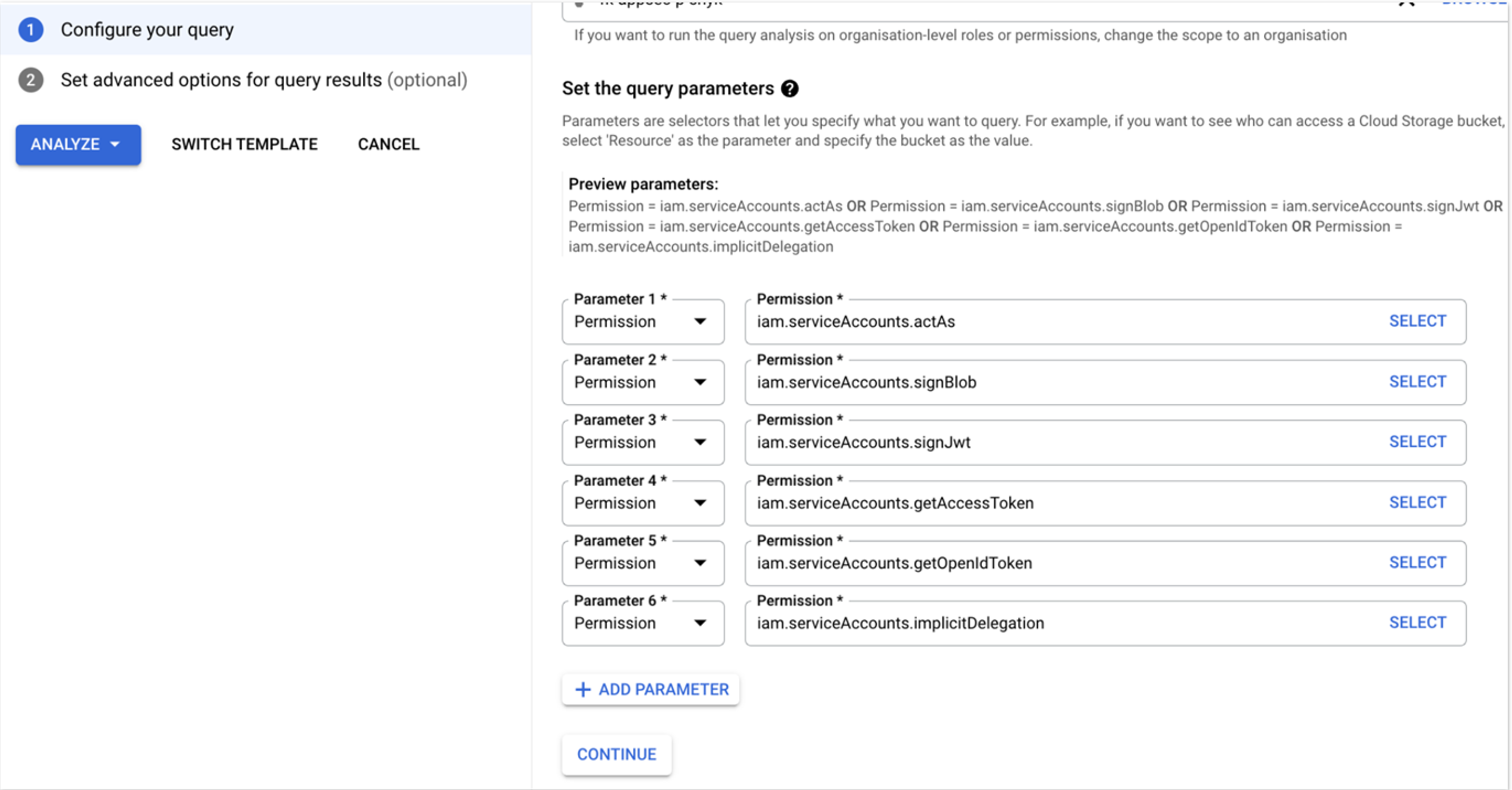
CREATE QUERY

Where do I have publicly accessible buckets?

See which buckets in your organisation are allowed to be accessible by the public.

CREATE QUERY

GCP Policy Analyser: Impersonate a Service Account



If you want to run the query analysis on organisation-level roles or permissions, change the scope to an organisation

Set the query parameters

Parameters are selectors that let you specify what you want to query. For example, if you want to see who can access a Cloud Storage bucket, select 'Resource' as the parameter and specify the bucket as the value.

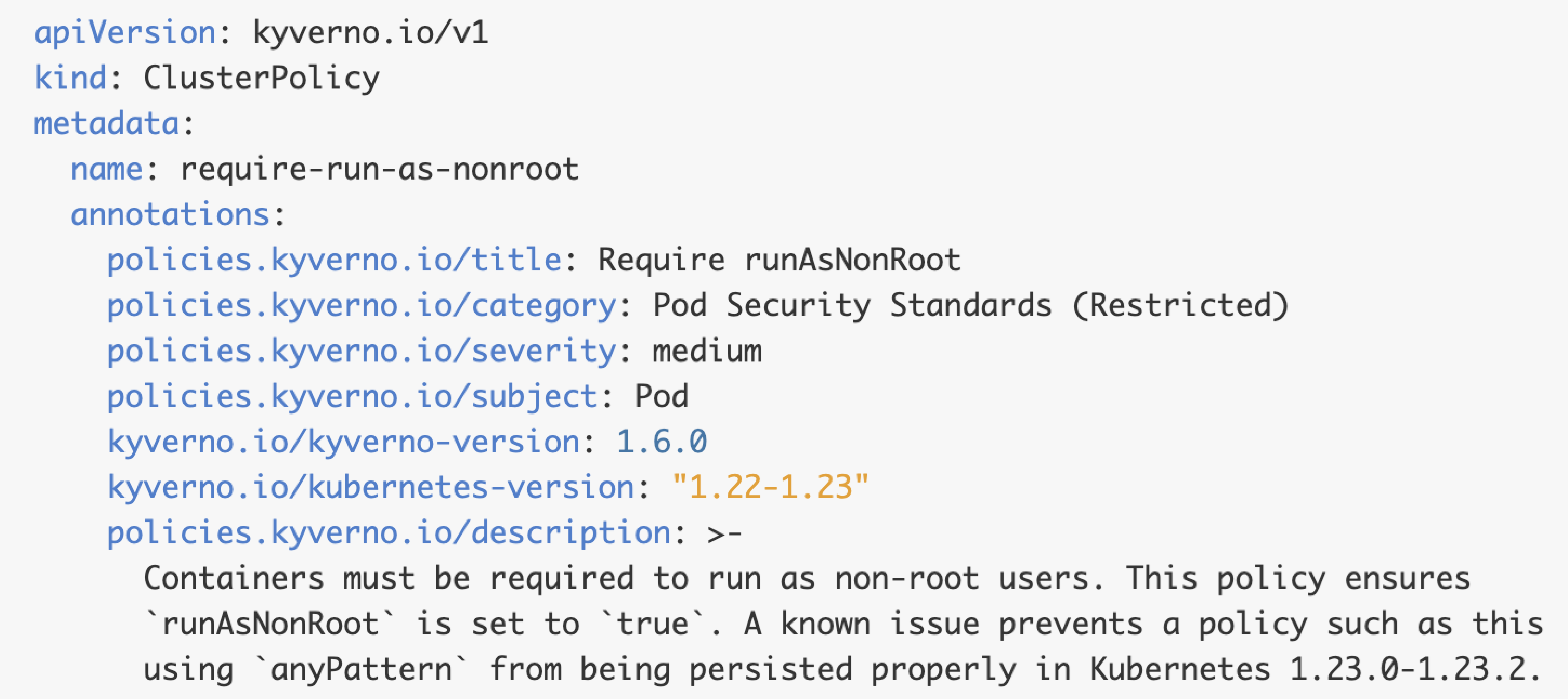
Preview parameters:
Permission = iam.serviceAccounts.actAs **OR** Permission = iam.serviceAccounts.signBlob **OR** Permission = iam.serviceAccounts.signJwt **OR** Permission = iam.serviceAccounts.getAccessToken **OR** Permission = iam.serviceAccounts.getOpenIdToken **OR** Permission = iam.serviceAccounts.implicitDelegation

Parameter 1 *: Permission	Permission *: iam.serviceAccounts.actAs
Parameter 2 *: Permission	Permission *: iam.serviceAccounts.signBlob
Parameter 3 *: Permission	Permission *: iam.serviceAccounts.signJwt
Parameter 4 *: Permission	Permission *: iam.serviceAccounts.getAccessToken
Parameter 5 *: Permission	Permission *: iam.serviceAccounts.getOpenIdToken
Parameter 6 *: Permission	Permission *: iam.serviceAccounts.implicitDelegation

GCP Policy Analyser: Impersonate a Service Account

- Principals/Service Accounts who has impersonation access in selected project/organisation
- IAM role granted that permission.
- Name of permissions
  - ex: iam.serviceAccounts.actAs, iam.serviceAccounts.signJwt
- Access is inherited or given directly by IAM role.

Kubernetes Kyverno Policy: runAsNonRoot



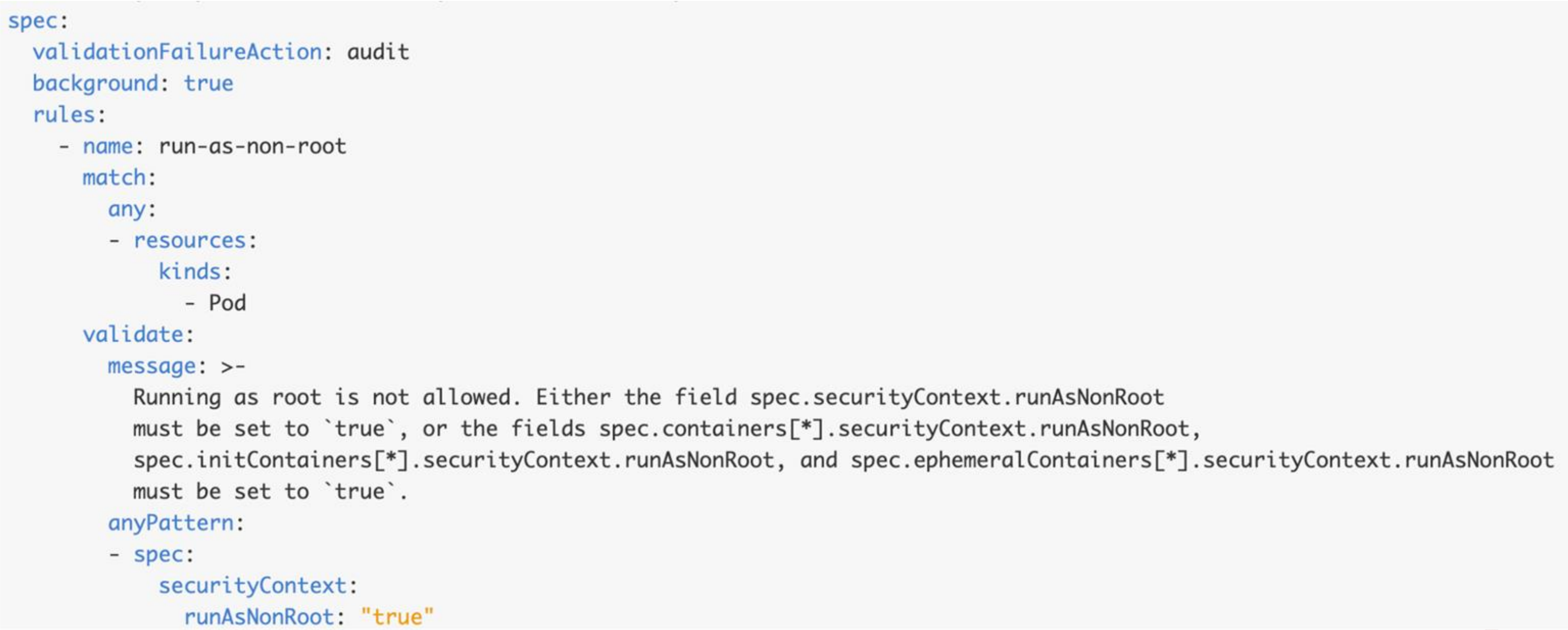
```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: require-run-as-nonroot
  annotations:
    policies.kyverno.io/title: Require runAsNonRoot
    policies.kyverno.io/category: Pod Security Standards (Restricted)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    kyverno.io/kyverno-version: 1.6.0
    kyverno.io/kubernetes-version: "1.22-1.23"
    policies.kyverno.io/description: >-
      Containers must be required to run as non-root users. This policy ensures
      `runAsNonRoot` is set to `true`. A known issue prevents a policy such as this
      using `anyPattern` from being persisted properly in Kubernetes 1.23.0-1.23.2.
```

Ref: https://kyverno.io/policies/pod-security/restricted/require-run-as-nonroot/require-run-as-nonroot/

Any Questions ?

Thank You

Kubernetes Kyverno Policy: runAsNonRoot



```
spec:
  validationFailureAction: audit
  background: true
  rules:
    - name: run-as-non-root
      match:
        any:
        - resources:
            kinds:
              - Pod
      validate:
        message: >-
          Running as root is not allowed. Either the field spec.securityContext.runAsNonRoot
          must be set to `true`, or the fields spec.containers[*].securityContext.runAsNonRoot,
          spec.initContainers[*].securityContext.runAsNonRoot, and spec.ephemeralContainers[*].securityContext.runAsNonRoot
          must be set to `true`.
        anyPattern:
        - spec:
            securityContext:
              runAsNonRoot: "true"
```

Ref: https://kyverno.io/policies/pod-security/restricted/require-run-as-nonroot/require-run-as-nonroot/