# Alert manager

# INSTALL ALERTMANAGER

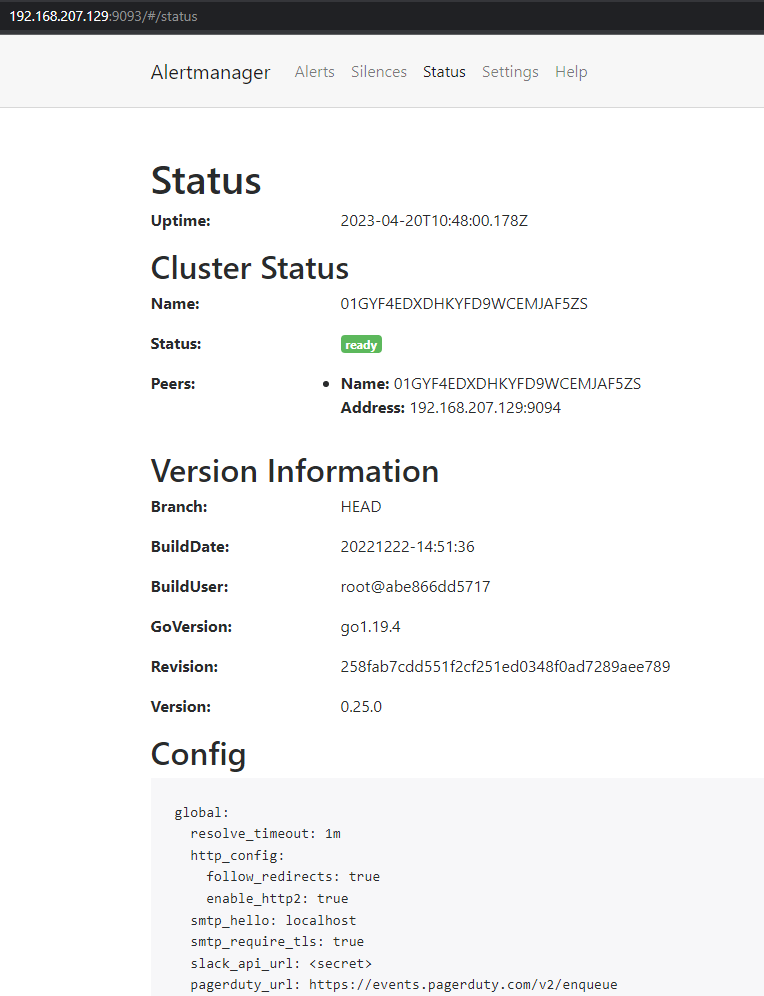
|  |
| --- |
| wget <https://github.com/prometheus/alertmanager/releases/download/v0.25.0/alertmanager-0.25.0.linux-amd64.tar.gz>    tar xvzf alertmanager-0.25.0.linux-amd64.tar.gz  cd alertmanager-0.25.0.linux-amd64/  ./alertmanager    cat alertmanager.yml  route:  group\_by: ['alertname']  group\_wait: 30s  group\_interval: 5m  repeat\_interval: 1h  receiver: 'web.hook'  receivers:  - name: 'web.hook'  webhook\_configs:  - url: '<http://127.0.0.1:5001/>'  inhibit\_rules:  - source\_match:  severity: 'critical'  target\_match:  severity: 'warning'  equal: ['alertname', 'dev', 'instance'] |

Reference:

<https://acloudguru.com/hands-on-labs/installing-prometheus-alertmanager>

Default Config

|  |
| --- |
| global:  resolve\_timeout: 5m  http\_config:  follow\_redirects: true  enable\_http2: true  smtp\_hello: localhost  smtp\_require\_tls: true  pagerduty\_url: <https://events.pagerduty.com/v2/enqueue>  opsgenie\_api\_url: <https://api.opsgenie.com/>  wechat\_api\_url: <https://qyapi.weixin.qq.com/cgi-bin/>  victorops\_api\_url: <https://alert.victorops.com/integrations/generic/20131114/alert/>  telegram\_api\_url: <https://api.telegram.org>  webex\_api\_url: <https://webexapis.com/v1/messages>  route:  receiver: web.hook  group\_by:  - alertname  continue: false  group\_wait: 30s  group\_interval: 5m  repeat\_interval: 1h  inhibit\_rules:  - source\_match:  severity: critical  target\_match:  severity: warning  equal:  - alertname  - dev  - instance  receivers:  - name: web.hook  webhook\_configs:  - send\_resolved: true  http\_config:  follow\_redirects: true  enable\_http2: true  url: <http://127.0.0.1:5001/>  max\_alerts: 0  templates: [] |



Update alertmanager.yml

|  |
| --- |
| global:  resolve\_timeout: 1m  slack\_api\_url: '[https://hooks.slack.com/services/<ID>](https://hooks.slack.com/services/%3cID%3e)'    route:  receiver: 'slack-notifications'    receivers:  - name: 'slack-notifications'  slack\_configs:  - send\_resolved: true  channel: '#prometheus'  icon\_url: <https://avatars3.githubusercontent.com/u/3380462>  title: |-  [{{ .Status | toUpper }}{{ if eq .Status "firing" }}:{{ .Alerts.Firing | len }}{{ end }}] {{ .CommonLabels.alertname }} for {{ .CommonLabels.job }}  {{- if gt (len .CommonLabels) (len .GroupLabels) -}}  {{" "}}  {{- end }}  title\_link: '{{ template "slack.default.titlelink" . }}'  text: >-  {{ range .Alerts -}}  \*Alert:\* {{ .Annotations.summary }}{{ if .Labels.severity }} - `{{ .Labels.severity }}`{{ end }}  \*Description:\* {{ .Annotations.description }}  \*Details:\*  {{ range .Labels.SortedPairs }} • \*{{ .Name }}:\* `{{ .Value }}`  {{ end }}  {{ end }} |

This is a configuration file for Alertmanager, a component of the Prometheus monitoring system that manages alerts sent by Prometheus server. The configuration file defines how alerts are processed and routed to different receivers, such as email, Slack, PagerDuty, etc.

Here's a breakdown of the different sections and settings in this particular alertmanager.ymlfile:

The global section contains global settings for Alertmanager, including:

* + resolve\_timeout: The time to wait before resolving an alert if it has stopped firing (default is 5m).
  + slack\_api\_url: The URL for the Slack API webhook that Alertmanager uses to send notifications to Slack. This is specific to the Slack integration and must be configured with your own Slack webhook URL.

The route section defines the default receiver for alerts, which in this case is set to slack-notifications. This means that any alerts that are not specifically routed to another receiver will be sent to the slack-notificationsreceiver.

The receivers section defines the available receivers for alerts, which in this case only includes one receiver named slack-notifications. The settings for this receiver include:

* + name: The name of the receiver.
  + slack\_configs: The Slack-specific configuration for the receiver.

Under the slack\_configssetting, the following configuration options are defined:

* + send\_resolved: Whether or not to send notifications when an alert has been resolved.
  + channel: The Slack channel to send the notifications to (e.g. "#prometheus").
  + icon\_url: The URL for the icon that appears next to the notification in Slack.
  + title: The title of the notification, which is a template that includes the status of the alert, the name of the alert, and the job associated with the alert.
  + title\_link: A template for the URL to use for the title of the notification, which in this case is using a default template defined elsewhere in the configuration file.
  + text: The body of the notification, which includes the summary and description of the alert, as well as any additional label values associated with the alert. The range function is used to loop over all alerts in the group and create a list of them in the notification body.

Overall, this configuration file is defining how alerts should be sent to Slack and what information should be included in the notification message. It is only one example of how Alertmanager can be configured, and there are many other configuration options and integrations available.

The title field in the slack\_configssection of the alertmanager.ymlconfiguration file is a template for the title of the notification that is sent to Slack when an alert is fired. It contains the following elements:

* + |-\n: This is a YAML block scalar style that indicates a multi-line string literal. The |character indicates that the string should be preserved exactly as it is, including line breaks, and the -character indicates that any trailing whitespace on each line should be ignored.
  + {{ .Status | toUpper }}: This is a Go template expression that retrieves the Statusfield of the alert and converts it to uppercase using the toUpperfunction. The .character refers to the current context, which is the alert group that triggered the notification.
  + {{ if eq .Status "firing" }}:{{ .Alerts.Firing | len }}{{ end }}: This is another Go template expression that checks if the Statusfield of the alert is "firing", and if it is, adds a colon and the number of firing alerts to the title. The Alerts.Firingfield returns a list of all alerts that are currently firing, and the lenfunction returns the length of that list.
  + {{ .CommonLabels.alertname }} for {{ .CommonLabels.job }}: This retrieves the alertname and job fields from the CommonLabelssection of the alert group and formats them into a string that describes the alert.
  + {{- if gt (len .CommonLabels) (len .GroupLabels) -}} {{" "}} {{- end }}: This is another Go template expression that checks if the number of labels in the CommonLabels section is greater than the number of labels in the GroupLabelssection, and if it is, adds a space to the end of the title. This is used to ensure that the title is aligned properly when different alerts have different label sets.
  + title\_link: '{{ template "slack.default.titlelink" . }}': This is a reference to another template in the configuration file that specifies the URL to link the title to.
  + text: >- ...: This is a multi-line string literal that defines the body of the notification, which contains more details about the alerts that have fired. It uses another Go template expression to loop over all alerts in the group and generate a list of their details.

Reference:

(1) Notification template examples | Prometheus. <https://prometheus.io/docs/alerting/latest/notification_examples/>.

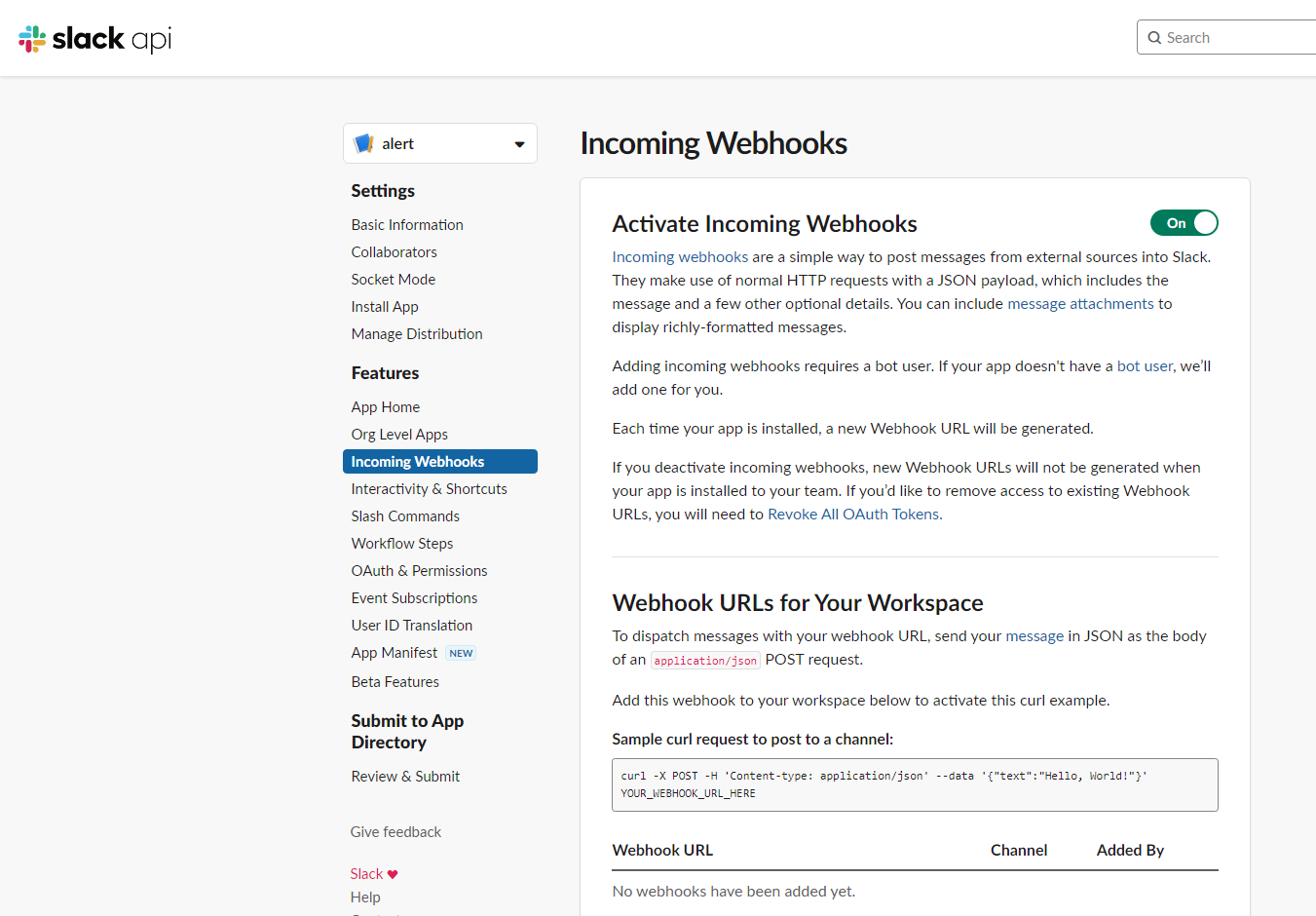
(2) Creating Awesome Alertmanager Templates for Slack - Hodovi. <https://hodovi.cc/blog/creating-awesome-alertmanager-templates-for-slack/>.

(3) Step-by-step guide to setting up Prometheus Alertmanager with Slack .... <https://grafana.com/blog/2020/02/25/step-by-step-guide-to-setting-up-prometheus-alertmanager-with-slack-pagerduty-and-gmail/>.

<https://prometheus.io/docs/alerting/latest/configuration/#slack_config>

<https://prometheus.io/docs/alerting/latest/notification_examples/>

Next, create new webhook and copy like webhook url like tutorial before.



alert is requesting permission to access the 
devsecops Slack workspace 
Where should alert post? 
# alert requires a channel to post to as an app 
# random 
Cancel 
Allow 

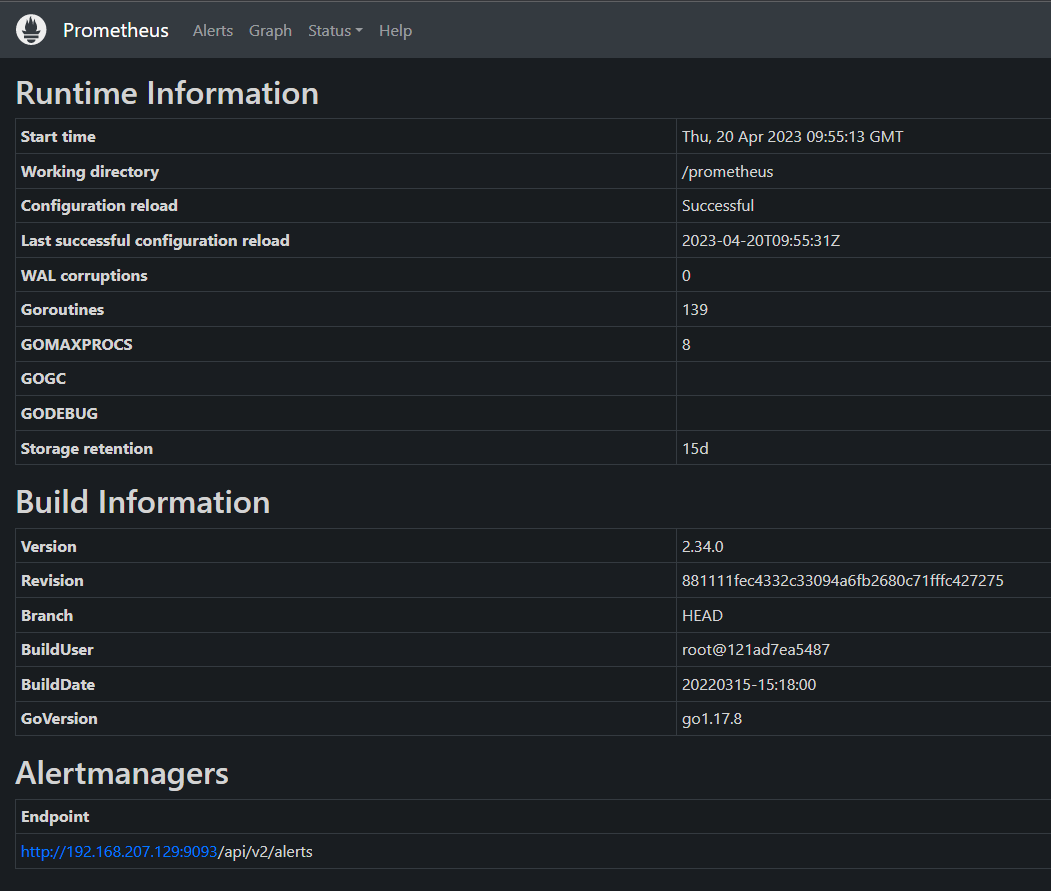
Add new webhook url into alertmanager.yml

|  |
| --- |
| global:  resolve\_timeout: 1m  slack\_api\_url: '[https://hooks.slack.com/services/<ID>](https://hooks.slack.com/services/%3cID%3e)' |

Use /-/reload to reload config

|  |
| --- |
| curl -X POST <IP>:9093/-/reload |

Go to promotheus dashboard and check runtime Information, we can see alert manager endpoint



# CREATE ALERT RULE

First, we go to prometheus config to know simple config.

|  |
| --- |
| <https://github.com/istio/istio/blob/master/samples/addons/prometheus.yaml>  data:  allow-snippet-annotations: "false"  alerting\_rules.yml: |  {}  alerts: |  {} |

This yaml file linked is a configuration file for Prometheus in istio. The data section of the file contains the following key-value pairs:

* + allow-snippet-annotations: This setting controls whether Prometheus will allow users to add custom annotations to Prometheus rules. If set to "false", users will only be able to use the pre-defined annotations.
  + alerting\_rules.yml: This file contains the YAML configuration for Prometheus alerting rules.
  + alerts: This file contains the YAML configuration for Prometheus alerts.

The alerting\_rules.yml and alerts files are both empty, which means that Prometheus will not be configured to send any alerts by default. If you want to configure Prometheus to send alerts, you will need to add your own rules and alerts to these files.

For more information on configuring Prometheus, please see the Prometheus documentation: <https://prometheus.io/docs/prometheus/latest/configuration/>

Now we update alerting target and rule into prometheus.

|  |
| --- |
| sudo kubectl -n istio-system get configmap  NAME DATA AGE  prometheus 5 9d    sudo kubectl -n istio-system edit configmap prometheus    prometheus.yml: |  global: # This section defines global settings for Prometheus  evaluation\_interval: 1m # Evaluate rules every 1 minute  scrape\_interval: 15s # Scrape targets every 15 seconds  scrape\_timeout: 10s # Timeout for scraping targets  alerting: # This section defines alerting settings for Prometheus  alertmanagers: # This section specifies the alertmanager instances to send alerts to  - static\_configs: # This section specifies the static list of alertmanager targets  - targets: <http://192.168.207.129:9093> # This is the address of the alertmanager instance      apiVersion: v1 # This is the API version for Kubernetes resources  data: # This section contains the data for the configmap resource  alerting\_rules.yml: | # This is the name of the file that contains the alerting rules  {  "groups": [ # This is a list of groups of rules  {  "name": "Rules", # This is the name of the group  "rules": [ # This is a list of rules in the group  {  "alert": "InstanceDown", # This is the name of the alert  "expr": "up == 0", # This is the expression that triggers the alert  "for": "0m", # This is the duration that the expression must be true before firing the alert  "annotations": { # This section contains additional information for the alert  "title": "Instance {{ $labels.instance }} down", # This is the title of the alert, using label templating  "description": "{{ $labels.instance }} of job {{ $labels.job }} has been down for more than 1 minute." # This is the description of the alert, using label templating  },  "labels": { # This section contains additional labels for the alert  "severity": "critical" # This is a label that indicates the severity of the alert  }  },  {  "alert": "KubernetesPodClientError", # This is another alert name  "expr": "istio\_requests\_total{reporter=\"destination\", response\_code=\"403\"} > 10", # This is another expression that triggers the alert, using metric and label filtering  "labels": { # This section contains additional labels for this alert  "severity": "warning" # This is another label that indicates the severity of this alert  },  "annotations": { # This section contains additional information for this alert  "summary": "Kubernetes pod Client Error (instance {{ $labels.instance }})", # This is another title of this alert, using label templating  "description": "Pod {{ $labels.instance }} of job {{ $labels.job }} reported client specific issues" # This is another description of this alert, using label templating  }  }  ]  }  ]  } |

This file is a configuration contains the following settings:

evaluation\_interval: This setting controls how often Prometheus will evaluate its rules. The default value is 1 minute.

scrape\_interval: This setting controls how often Prometheus will scrape metrics from its targets. The default value is 15 seconds.

scrape\_timeout: This setting controls how long Prometheus will wait for a response from a target before giving up. The default value is 10 seconds.

alertmanagers: This section configures Prometheus to send alerts to an Alertmanager. The Alertmanager is a separate service that is responsible for handling alerts and notifying users.

It contains the following rules:

InstanceDown: This rule alerts if an instance has been down for more than 1 minute.

KubernetesPodClientError: This rule alerts if a Kubernetes pod has reported more than 10 client errors.

Each rule has the following settings:

alert: This is the name of the alert.

expr: This is the expression that Prometheus will use to evaluate the rule.

for: This is the duration for which Prometheus will keep an alert in the firing state.

annotations: This is a map of annotations that will be added to alerts that fire.

labels: This is a map of labels that will be added to alerts that fire.

we can add larger rule based on rule file below and using converter yaml to json:

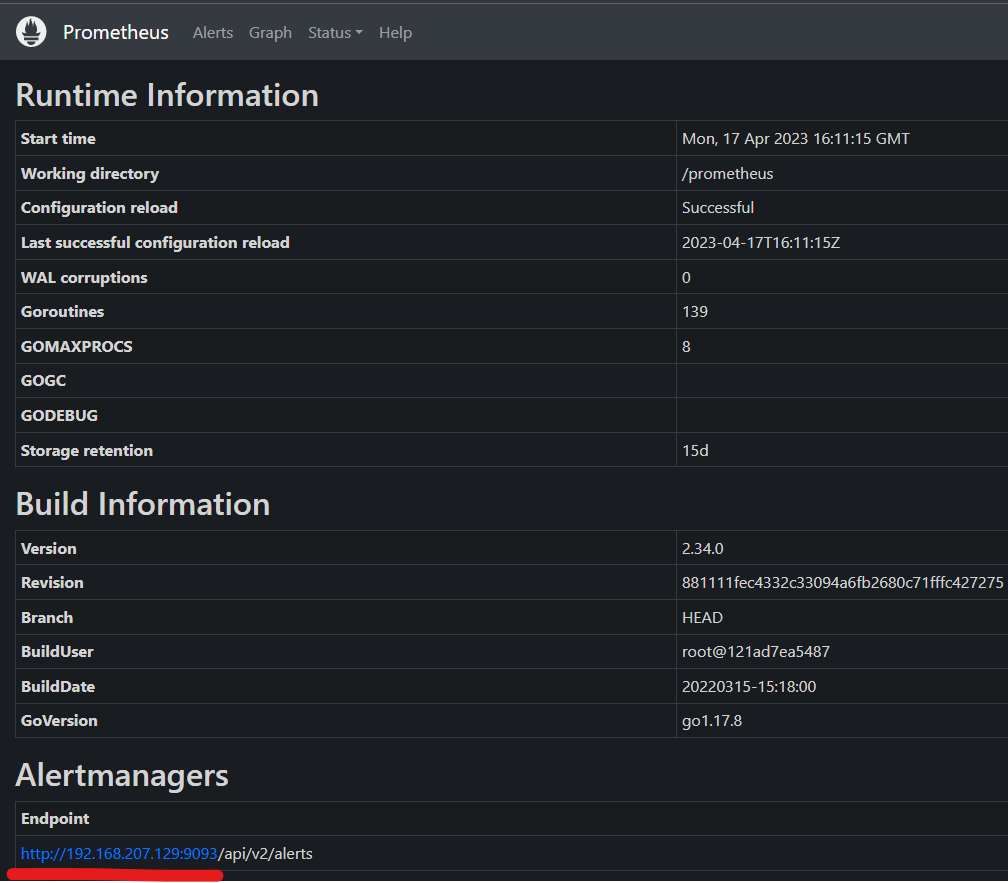
<https://github.com/samber/awesome-prometheus-alerts/blob/master/dist/rules/istio/embedded-exporter.yml>

After update config, we restart pod prometheus:

|  |
| --- |
| sudo kubectl -n istio-system describe pod -l=app=prometheus  sudo kubectl -n istio-system delete pod -l=app=prometheus  sudo kubectl -n istio-system get pod -l=app=prometheus  sudo kubectl -n istio-system logs prometheus-7cc75b7c8c-nmbmw --container prometheus-server |

Now, we go to prometheus to check rule again

0 
Prometheus 
Rules 
Alerts 
Graph 
Status 
Help 
Rules 
Rule 
alert: 
expr: 
labels: 
Error 
InstanceDown 
up = 
severity: critical 
annotations: 
description: Slabels.instance l} of job $labels.job has been down for more than 1 minute. 
title: Instance {{ $labels.instance down 
45.886s ago 
Last Evaluation 
45.888s ago 
45.888s ago 
0.520ms 
Evaluation Time 
0.354ms 
0.155ms 
alert: 
expr: 
labels: 
KubernetesPodClientError 
istio_requests_total{reporter= 'Idestination" 
"403") > 10 
severity: warning 
annotations: 
description: Pod {{ $labels.instance of job {{ $labels.job l} reported client specific issues 
summary: Kubernetes pod Client Error (instance {{ $labels.instance }}) 



Disable peer authentication to run test rule.

|  |
| --- |
| sudo kubectl edit pa -n istio-system  spec:  mtls:  mode: DISABLE |

Create nginx service:

|  |
| --- |
| kubectl -n prod run nginx --image nginx  kubectl -n prod expose pod nginx --port 80  kubectl -n prod get svc |

Use curl to test connection:

|  |
| --- |
| curl <IP nginx svc>:80  or  kubectl -n prod exec -it nginx -- curl 10.32.0.28:80 |

Next, we go to nginx and delete index.html.

|  |
| --- |
| kubectl -n prod exec -it nginx -- bash  rm /usr/share/nginx/html/index.html |

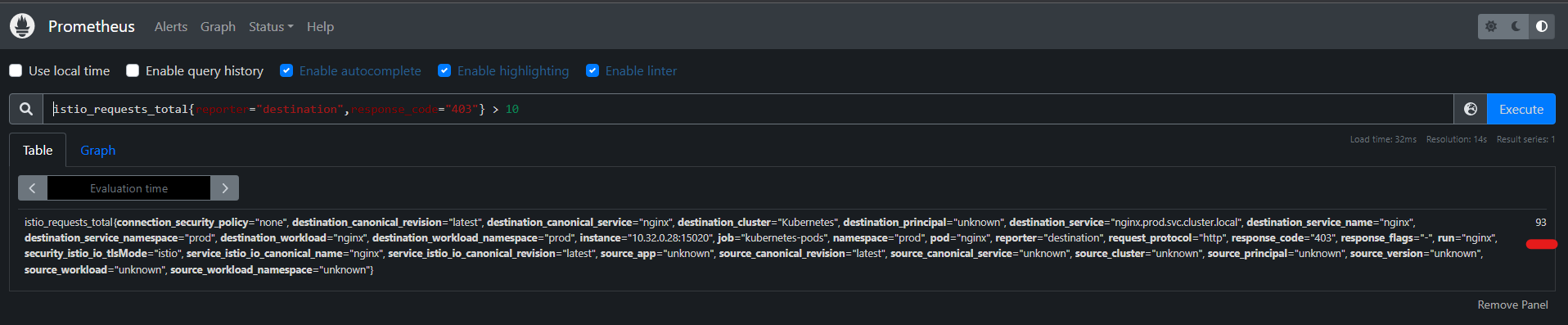
We check the connection again by using the following bash script and get a 403 error.

while true; do curl -s 10.101.73.244:80; sleep 1; done

Now let's go back to the dashboard to check if the rule is working

0 
Prometheus 
Alerts Graph 
Pending (O) 
/etc/config/alerting_rules.yml > Rules 
> InstanceDown (O active) 
v KubernetesPodClientError (1 active) 
KubernetesPodCLientError 
name : 
Status 
Help 
Q 
"4B"} > la 
Show annotations 
Filter by name or labels 
istio_requests_total{reporter="destination" , response code= 
expr: 
labels : 
severity: warning 
annotations : 
description: Pod {{ 
sumary: Kubernetes 
Labels 
$1abe1s. instance 
pod Client Error 
of job {{ 
(instance {{ 
$1abe1s.job reported client specific issues 
$1abe1s. instance 
Active Since 
2023-04- 
Value 
93 

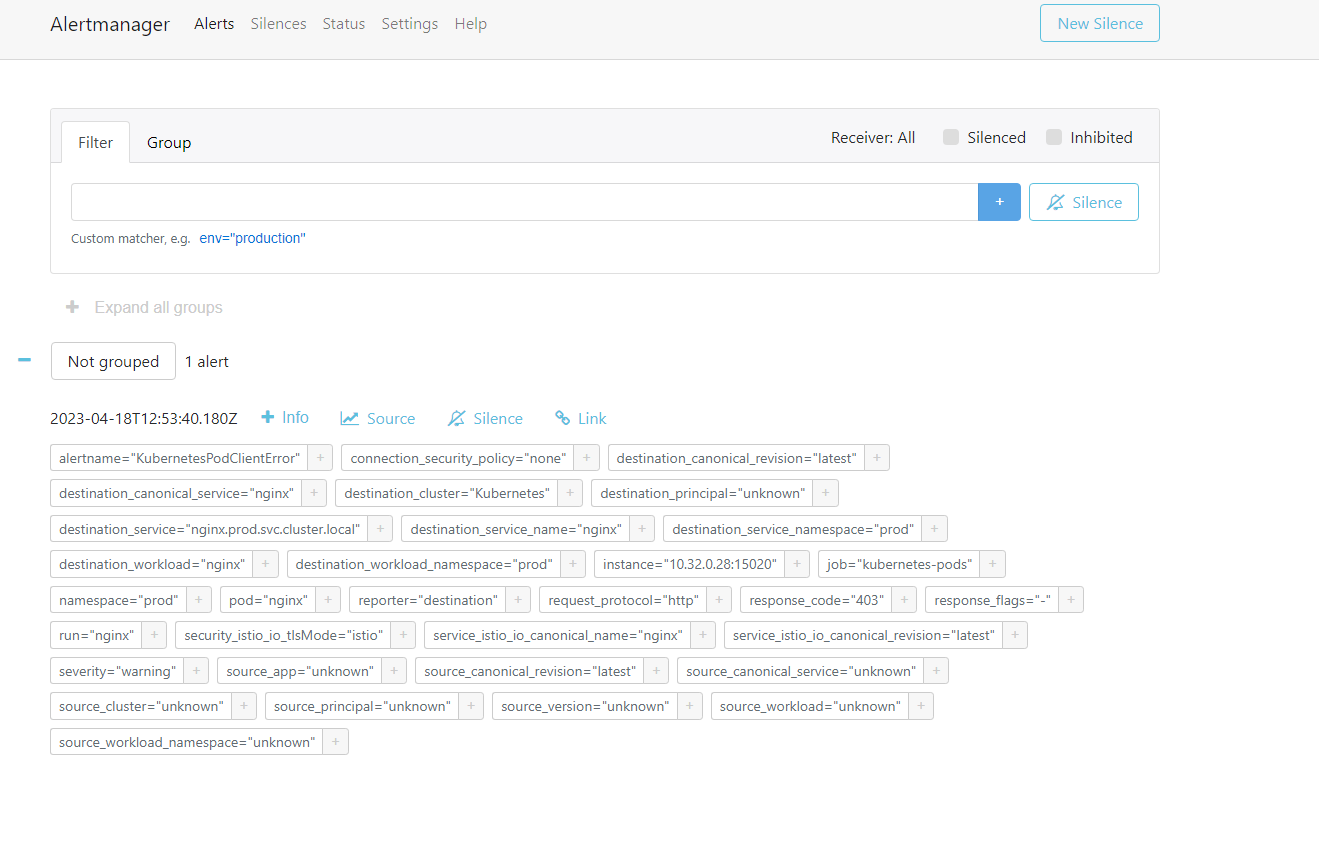
we can click on expr to execute the query



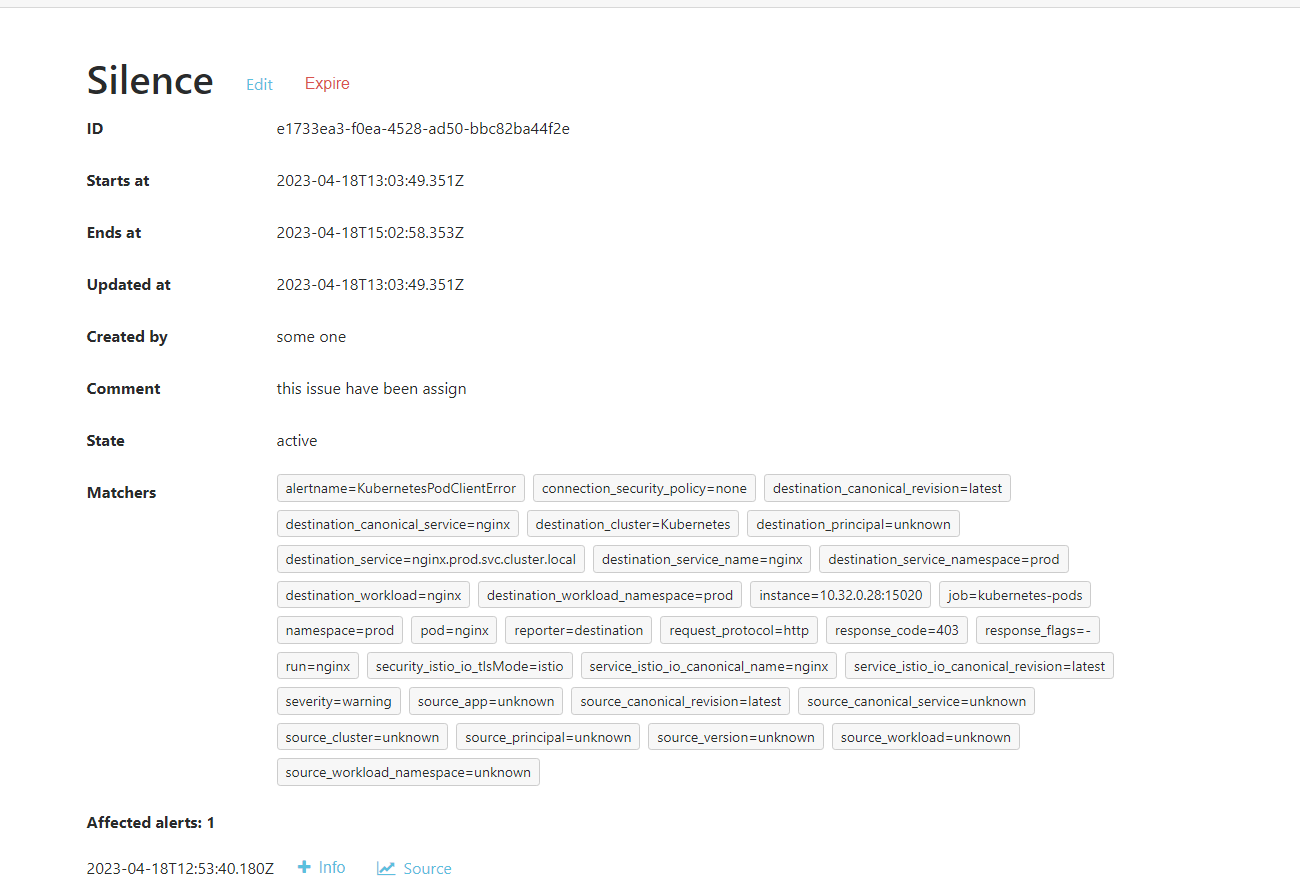
and alert is also sent to slack

alert APP 7:52 PM 
[FIRING:I] KubernetesPodClientError for kubernetes-pods 
Alert: Kubernetes pod Client Error (instance 10.32.028:15020) - 
Description: Pod 10.32.0.28:15020 of job kubernetes-pods reported client specific 
Issues 
Details: 
• alertname: KubernetesPodC1ientError 
• connection_security_policy. mutual_tls 
Show more 

similar to alertmanager, we can select silence.



We can enter the creator and comment information to assign the person in charge of handling this case



similarly we continue to test the rule instance down by deleting pod nginx:

|  |
| --- |
| kubectl -n prod delete pod nginx |

Go to dashboard to run query

0 
Prometheus 
Alerts Graph Status • Help 
LJse local time Enable query history Enable autocomplete 
Q up = 
e 
Enab'e highlighting 
Enable linter 
Table 
Graph 
Evaluation time 
job="kubernetes-pods", run="nginx", 

And we receive slack alert

alert APP 8:36 PM 
[FIRING:I] InstanceDown for kubernetes-pods 
Alert: -I criticall 
Description: 10.32.0.28:15020 of job kubernetes-pods has been down for more than 1 
minute. 
Details: 
• alertname: InstanceDown 
• instance: 110.32. a.28:15aæel 
Show more 