## Robusta Monitor Kubernetes Cluster

- 1. Robusta 的安装
  - a. 安装 robusta-cli

## pip install -U robusta-cli --no-cache

b. 通过 robusta-cli 创建 values.yaml 文件, 输入以下命令得到 generated values.yaml

## robusta gen-config

```
lizhu@master:-$ robusta gen-config

Robusta reports its findings to external destinations (we call them "sinks").

Me'll define some of them now.

Configure Slack integration? [y/N]: y

Please insert your MsTeams integration? [y/N]: y

Please insert your MsTeams webhook url. See https://docs.robusta.dev/master/automation/sinks/ms-teams.html: https://lenovobeijing.webhook.office.com/webhookb2/fd3eo194-6997-4484-af41-5e115e28cf3b@5c7d0b28-bdf8-410c-aa93-4df372b16203/IncomingWebhook/a4475d341d9e4f6c89d67f013bfad32d/49e91e2a-a59f-4202-8c20-5e74

04d5bedb

Configure Robusta UI sink? This is HIGHLY recommended. [Y/n]: n

Robusta can use Prometheus as an alert source.

If you haven't installed it yet, Robusta can install a pre-configured Prometheus.

Mould you like to do so? [y/N]: n

Mould you like to enable two-way interactivity (e.g. fix-it buttons in Slack) via Robusta's cloud? [y/N]: n

Last question! Mould you like to help us improve Robusta by sending exception reports? [y/N]: n

Saved configuration to ./generated values.yaml - save this fille for future use!

Finish installing with Helm (see the Robusta docs). By the way, you're missing out on the UI! See https://home.robusta.dev/ul/
```

c. 方法一: 获取 Robusta Helm chart, 然后安装到 K8S 集群

helm repo add robusta https://robusta-charts.storage.googleapis.com && helm repo update helm install robusta robusta/robusta -f ./generated\_values.yaml --set clusterName=lizhutest --namespace robusta

方法二: 或者使用 github repo:

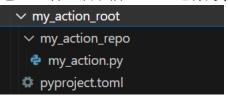
需要修改./helm/robusta/Chart.yaml 内的 version 和 appVersion (0.10.17)

需要修改/helm/robusta/values.yaml 内的 runner image 的 version 0.10.17

git clone https://github.com/robusta-dev/robusta.git

helm install robusta ./robusta/helm/robusta -f ./generated\_values.yaml --set clusterName=lizhutest --namespace robusta

- 2. Robusta 自定义 playbooks 配置
  - a. 创建自定义 action 的 playbooks,该 action 是当 node 状态变为 unknown 的时候发送 alert 并且展示该 node 上运行的 pod



b. 为自定义 playbooks 创建 persistent volume

```
accessModes:
  - ReadWriteOnce
  capacity:
   storage: 4Gi
  local:
   path: /home/lizhu/data1
  nodeAffinity:
   required:
    nodeSelectorTerms:
    - matchExpressions:
     - key: pv
      operator: In
      values:
      - worker1
  persistentVolumeReclaimPolicy: Retain
  volumeMode: Filesystem
```

c. 编辑 values.yaml, 设置 playbooksPersistentVolume: true

```
helm get values robusta -o yaml > values.yaml
echo "playbooksPersistentVolume: true" >> values.yaml
helm upgrade robusta robusta/robusta --values=values.yaml
helm upgrade robusta ./robusta/helm/robusta/ --namespace=robusta --
values ./values.yaml
```

d. Push 自定义 playbooks 到 Robusta

```
robusta playbooks push ./my_action_root
```

e. 编辑 values.yaml 文件,使自定义的 action 得以生效

```
customPlaybooks:
- actions:
```

```
- my_node_health_watcher: {}triggers:- on_node_update: {}
```

- 3. 集成 Prometheus 和 Robusta
  - a. 编辑 alertManager 配置文件, account\_id 和 signing\_key 可以从 values.yaml 文件的 global config 中获得

```
receivers:
- name: 'robusta'
  webhook_configs:
   - url: 'http://10.0.4.17:30080/api/alerts'
    http_config:
     authorization:
     # Replace <TOKEN> with a string in the format `<ACCOUNT_ID>
<SIGNING KEY>`
      credentials: '8b01ca9a-437b-41b6-adf9-11d6cc0b7c97 8b01ca9a-437b-
41b6-adf9-11d6cc0b7c97'
    send_resolved: true
route:
receiver: 'robusta'
routes:
- receiver: 'robusta'
 group by: [ '...' ]
  group_wait: 1s
  group_interval: 1s
  matchers:
  - severity =~ ".*"
  repeat_interval: 4h
  continue: true
```

b. 编辑 Prometheus 的 alert 相关配置,添加 label: cluster\_name: lizhutest

```
groups:
- name: example
rules:
# Alert for any instance that is unreachable for >2 minutes.
- alert: service_down
    expr: up == 0
    for: 2m
    labels:
        severity: page
        cluster_name: lizhutest
        annotations:
        summary: "Instance {{ $labels.instance }} down"
```

```
description: "{{ $labels.instance }} of job {{ $labels.job }} has been down
for more than 2 minutes."
   - alert: high_load
   expr: node_load1 > 0.5
   for: 2m
   labels:
      severity: page
      cluster_name: lizhutest
      annotations:
      summary: "Instance {{ $labels.instance }} under high load"
      description: "{{ $labels.instance }} of job {{ $labels.job }} is under high load."
```

c. 编辑 values.yaml 文件,使 Robusta 可以从 Prometheus 中获取所需要的信息

```
disableCloudRouting: false
globalConfig:
    prometheus_auth: Bearer <YOUR TOKEN> # or any other auth header
    alertmanager_auth: Basic <USER:PASSWORD base64-encoded>
    alertmanager_url: ""
    grafana_url: ""
    prometheus_url: <a href="http://ip:9090">http://ip:9090</a>
# Additional query string parameters to be appended to the Prometheus connection
URL (optional)
    prometheus_url_query_string: "demo-query=example-data&another-query=value"
```

4. 测试 alert manager 是否正常运行

```
customPlaybooks:
- actions:
- my_node_health_watcher: {}
triggers:
- on_node_update: {}
- actions:
- stack_overflow_enricher: {}
triggers:
- on_prometheus_alert: {}
```

helm upgrade robusta robusta/robusta –values=./values.yaml
helm upgrade robusta ./robusta/helm/robusta/ --namespace=robusta –values ./values.yaml
robusta demo-alert --alertmanager-url http://10.0.4.20:9093 --labels cluster\_name=lizhutest

- 5. 配置 Prometheus alert playbooks
- 6. Tips

- a. Robusta 的 runner 和 forwarder 不能运行在同一个 node 上,否则 k8s alert 不能发送到配置的 sink 内。但是 prometheus 的 alert 发送没有问题。
- b. 给 robusta-runner 服务配置 ingress,使 alert 能访问到 robusta 的 runner
- c. Robusta 默认会监测 pod 是否 crashloopbackoff, imagepullbackoff 和 oom kill, 所以如果不需要一些默认提醒可以在 chart 的 values.yaml 内将该 trigger 注释掉。