

# Deep Dive into the Prometheus Operator Ecosystem

**Jayapriya Pai, Red Hat**

**3 June 2023 | Bangalore, India**

# About me



- **Jayapriya Pai (Jp)**
- Senior Software Engineer
- OpenShift In-cluster monitoring team Red Hat
- Contributor and member in Prometheus-Operator
- *github: @slashpai*

# Takeaways



Configure monitoring in Kubernetes with Prometheus Operator



How Prometheus Operator works and available features



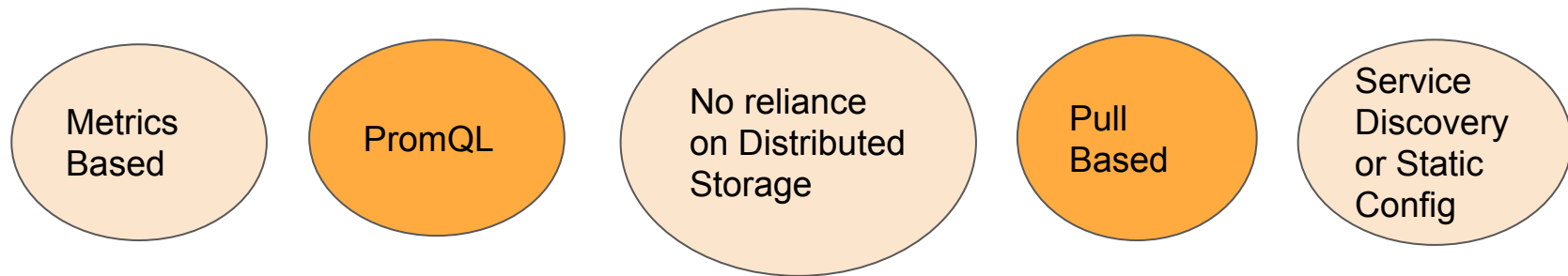
Deploying Monitoring Stack, Examples

# Prometheus

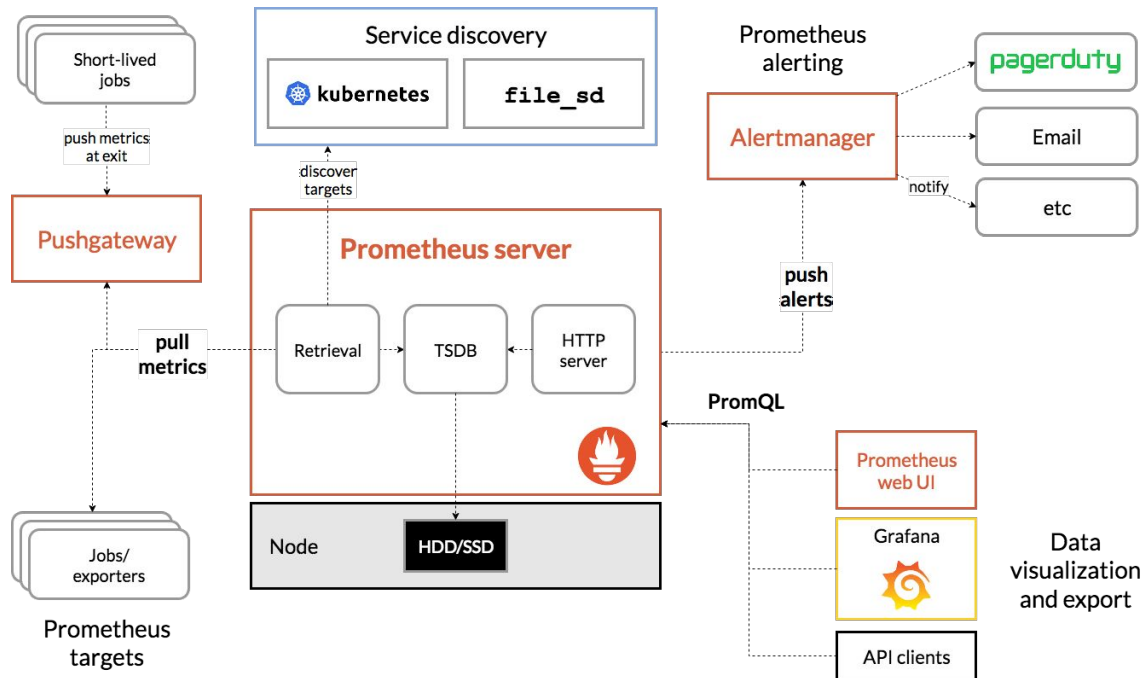
# Prometheus

“Prometheus is an open-source systems monitoring and alerting toolkit originally built at SoundCloud”

CNCF Project since 2016 and second hosted project after Kubernetes




# Prometheus Architecture






source: <https://prometheus.io/docs/introduction/overview>

# Prometheus-Operator



## prometheus-operator

Manage Prometheus deploys on Kubernetes. This is an independent project from the Prometheus project.

 152 followers  <https://prometheus-operator.dev/>  @PromOperator




[Overview](#) [Repositories](#) 4 [Projects](#) [Packages](#) [Teams](#) 4 [People](#) 13

## Popular repositories

**prometheus-operator**

Public




Prometheus Operator creates/configures/manages Prometheus clusters atop Kubernetes

 Go  8k  3.6k

**kube-prometheus**




Public

Use Prometheus to monitor Kubernetes and applications running on Kubernetes

 Jsonnet  5.3k  1.7k

**runbooks**




Public

 HTML  58  95

**website**

Public

The Prometheus Operator & kube-prometheus website

 Shell  10  15

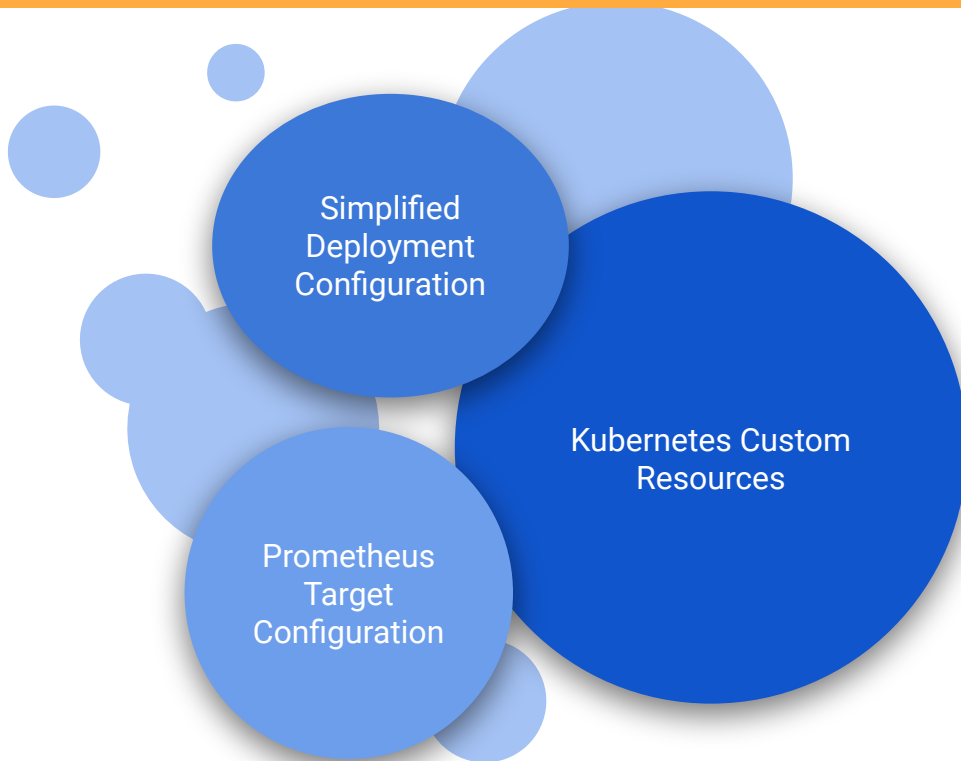
# Why Prometheus-Operator?



# Typical Monitoring Stack



# Prometheus-Operator



# How Prometheus Operator Works

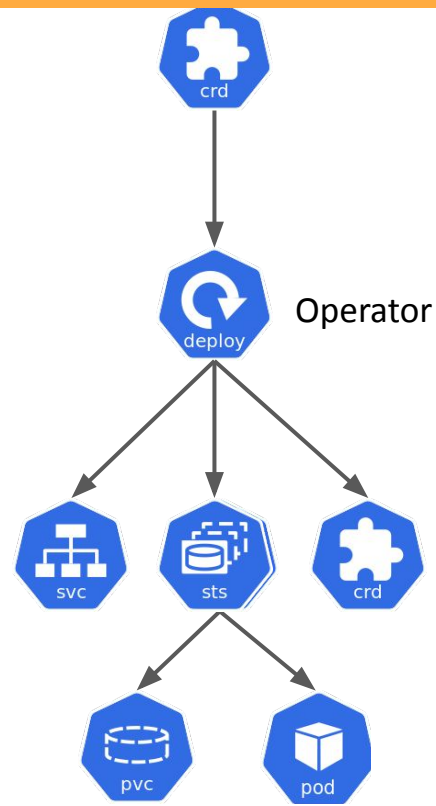
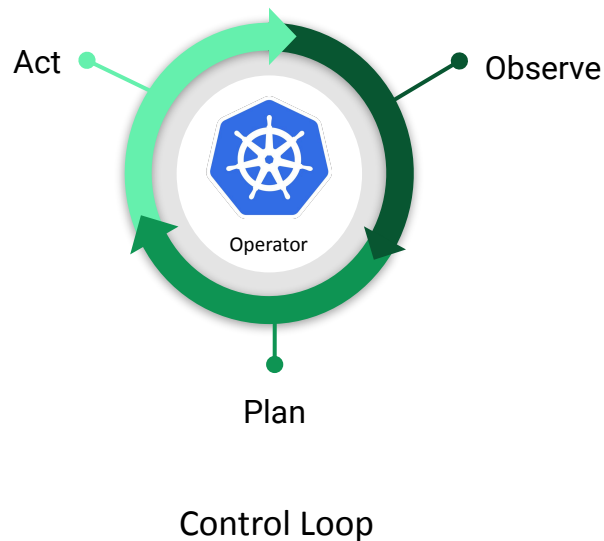
# Operator Pattern

“Operators were introduced by CoreOS as a class of software that operates other software, putting operational knowledge collected by humans into software”

From <https://web.archive.org/web/20170129131616/https://coreos.com/blog/introducing-operators.html>

# Operator = CRD + Custom Controllers

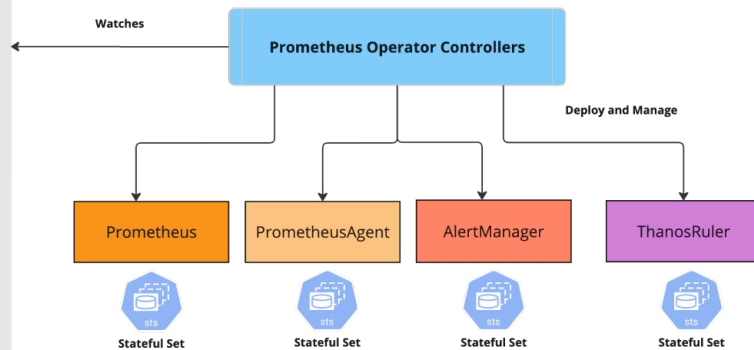
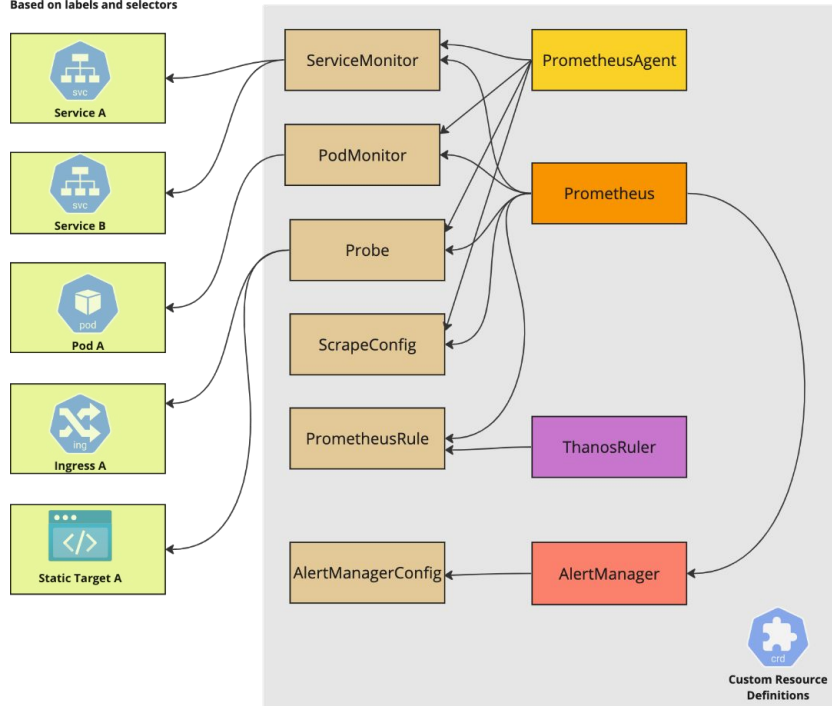
CR is Extension of Kubernetes API that provides a place where you can store and retrieve structured data—the desired state of your application and controller watches CR's and acts upon them



*TLDR: "Operators are Software SRE's"*

# Prometheus Operator Architecture

Based on labels and selectors



# Prometheus-Operator CRD's

01

StatefulSet

- Prometheus
- AlertManager
- ThanosRuler
- PrometheusAgent

02

Config Objects

- ServiceMonitor
- PodMonitor
- Probe
- PrometheusRule
- AlertManagerConfig
- ScrapeConfig



More details <https://github.com/prometheus-operator/prometheus-operator/blob/main/Documentation/api.md>

# Prometheus Custom Resource

## Important Fields:

- Selectors:
  - serviceMonitorNamespaceSelector, serviceMonitorSelector
  - podMonitorNamespaceSelector, podMonitorSelector
  - probeNamespaceSelector, probeSelector
  - scrapeConfigNamespaceSelector, scrapeConfigSelector
- version
- replicas
- alerting
- resources

```
apiVersion: monitoring.coreos.com/v1
kind: Prometheus
metadata:
  name: prometheus
  labels:
    prometheus: prometheus
spec:
  replicas: 2
  resources:
    requests:
      memory: 400Mi
  serviceMonitorSelector:
    matchLabels:
      team: frontend
  alerting:
    alertmanagers:
      - namespace: default
        name: alertmanager
        port: web
```

More Details: <https://github.com/prometheus-operator/prometheus-operator/blob/main/Documentation/design.md#prometheus>



# Collecting Metrics

# ServiceMonitor, PodMonitor and Probe

- Configure targets to monitor in the cluster
  - No need to learn Prometheus specific configuration
- ServiceMonitor
  - Selects pods via service (and their endpoints) based on label selector
    - Eg: <https://github.com/slashpai/prometheus-operator-examples/tree/main/servicemonitor>
- PodMonitor
  - Selects pods directly based on label selector
    - Eg: <https://github.com/slashpai/prometheus-operator-examples/tree/main/podmonitor>
- Probe
  - Monitor static target or ingress objects
  - Probe can be used to describe blackbox checks
  - To work with probe blackbox\_exporter has to be deployed in the cluster
  - Selects ingress objects based on label selector
    - Eg: <https://github.com/slashpai/prometheus-operator-examples/tree/main/probe>

# ServiceMonitor Example

```
! service.yaml × ... ! service-monitor.yaml ●
servicemonitor > ! service.yaml > Google Cloud Code > [k] kind
1 kind: Service
2 apiVersion: v1
3 metadata:
4   name: prometheus-example-app
5   namespace: default
6   labels:
7     app.kubernetes.io/name: prometheus-example-app
8 spec:
9   selector:
10    app.kubernetes.io/name: prometheus-example-app
11   ports:
12     - name: web
13     port: 8080

servicemonitor > ! service-monitor.yaml > Google Cloud Code > {} spec > [ ] endpoints
1 apiVersion: monitoring.coreos.com/v1
2 kind: ServiceMonitor
3 metadata:
4   name: prometheus-example-app
5   namespace: default
6   labels:
7     app.kubernetes.io/name: prometheus-example-app
8 spec:
9   selector:
10    matchLabels:
11      app.kubernetes.io/name: prometheus-example-app
12   endpoints:
13     - port: web
```

# ServiceMonitor Example

serviceMonitor/default/prometheus-example-app/0 (3/3 up) [show less](#)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
<a href="http://172.17.0.17:8080/metrics">http://172.17.0.17:8080/metrics</a>	UP	<code>container="prometheus-example-app"</code> <code>endpoint="web"</code> <code>instance="172.17.0.17:8080"</code> <code>job="prometheus-example-app"</code> <code>namespace="default"</code> <code>pod="prometheus-example-app-86cb447667-g9fik"</code> <code>service="prometheus-example-app"</code>	25.34s ago	1.814ms	
<a href="http://172.17.0.18:8080/metrics">http://172.17.0.18:8080/metrics</a>	UP	<code>container="prometheus-example-app"</code> <code>endpoint="web"</code> <code>instance="172.17.0.18:8080"</code> <code>job="prometheus-example-app"</code> <code>namespace="default"</code> <code>pod="prometheus-example-app-86cb447667-8jxrx"</code> <code>service="prometheus-example-app"</code>	24.4s ago	2.463ms	
<a href="http://172.17.0.19:8080/metrics">http://172.17.0.19:8080/metrics</a>	UP	<code>container="prometheus-example-app"</code> <code>endpoint="web"</code> <code>instance="172.17.0.19:8080"</code> <code>job="prometheus-example-app"</code> <code>namespace="default"</code> <code>pod="prometheus-example-app-86cb447667-fjcz2"</code> <code>service="prometheus-example-app"</code>	18.285s ago	1.351ms	

# Probe Example: Static Target

- Monitor Static Targets

```
apiVersion: monitoring.coreos.com/v1
kind: Probe
metadata:
  name: probe-static-target
  namespace: default
  labels:
    app.kubernetes.io/name: probe-static-target
spec:
  prober:
    url: blackbox-exporter.monitoring.svc:19115
  targets:
    staticConfig:
      labels:
        environment: prometheus.io
      static:
        - 'https://demo.do.prometheus.io'
        - 'https://node.demo.do.prometheus.io'
```

# Probe Example: Static Target

probe/default/probe-static-target (2/2 up) [show less](#)


Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
<a href="http://blackbox-exporter.monitoring.svc:19115/probe">http://blackbox-exporter.monitoring.svc:19115/probe</a> <code>target="https://demo.do.prometheus.io"</code>	UP	<code>environment="prometheus.io"</code> <code>instance="https://demo.do.prometheus.io"</code> <code>job="probe/default/probe-static-target"</code> <code>namespace="default"</code>	38.632s ago	561.515ms	
<a href="http://blackbox-exporter.monitoring.svc:19115/probe">http://blackbox-exporter.monitoring.svc:19115/probe</a> <code>target="https://node.demo.do.prometheus.io"</code>	UP	<code>environment="prometheus.io"</code> <code>instance="https://node.demo.do.prometheus.io"</code> <code>job="probe/default/probe-static-target"</code> <code>namespace="default"</code>	33.473s ago	549.245ms	

# Probe Example: Ingress

- Monitor Ingress Resource

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: example-app-ingress
  namespace: default
  labels:
    app.kubernetes.io/name: example-app
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /$1
spec:
  rules:
  - host: example-app
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: example-app
            port:
              number: 8080
```

```
apiVersion: monitoring.coreos.com/v1
kind: Probe
metadata:
  name: probe-ingress
  namespace: default
  labels:
    app.kubernetes.io/name: probe-ingress
spec:
  probe:
    url: blackbox-exporter.monitoring.svc:19115
  targets:
    ingress:
      selector:
        matchLabels:
          app.kubernetes.io/name: example-app
```



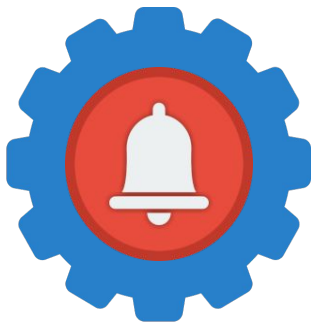
# Probe Example: Ingress

probe/default/probe-ingress (1/1 up) [show less](#)

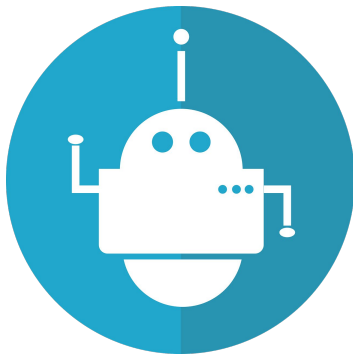
Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
<a href="http://blackbox-exporter.monitoring.svc:19115/probe">http://blackbox-exporter.monitoring.svc:19115/probe</a> <a href="#">target="http://example-app/"</a>	UP	<a href="#">ingress="example-app-ingress"</a> <a href="#">instance="http://example-app/"</a> <a href="#">job="probe/default/probe-ingress"</a> <a href="#">namespace="default"</a>	5.275s ago	9.925ms	



# What's latest in Prometheus-Operator?



AlertManagerConfig  
v1beta1  
(from v0.57)



Prometheus Agent  
v1alpha1  
(from v0.64)



ScrapeConfig  
v1alpha1  
(from v0.65)

# Additional Scrape Config

First, you will need to create the additional configuration. Below we are making a simple "prometheus" config. Name this `prometheus-additional.yaml` or something similar.

```
- job_name: "prometheus"
  static_configs:
  - targets: ["localhost:9090"]
```



Then you will need to make a secret out of this configuration.

```
kubectl create secret generic additional-scrape-configs --from-file=prometheus-additional.yaml --dry-run=client -o yaml
```



Next, apply the generated kubernetes manifest

```
kubectl apply -f additional-scrape-configs.yaml -n monitoring
```



Finally, reference this additional configuration in your `prometheus.yaml` CRD.

```
apiVersion: monitoring.coreos.com/v1
kind: Prometheus
metadata:
  name: prometheus
  labels:
    prometheus: prometheus
spec:
  replicas: 2
  serviceAccountName: prometheus
  serviceMonitorSelector:
    matchLabels:
      team: frontend
  additionalScrapeConfigs:
    name: additional-scrape-configs
    key: prometheus-additional.yaml
```



# ScrapeConfig CR

```
apiVersion: monitoring.coreos.com/v1alpha1
kind: ScrapeConfig
metadata:
  name: scrape-config-static-config-example
  namespace: default
  labels:
    app.kubernetes.io/name: scrape-config-static-config-example
spec:
  staticConfigs:
    - labels:
        job: alertmanager
      targets:
        - alertmanager-main.monitoring.svc.cluster.local:8080
    - labels:
        job: prometheus
      targets:
        - prometheus-k8s.monitoring.svc.cluster.local:9090
```

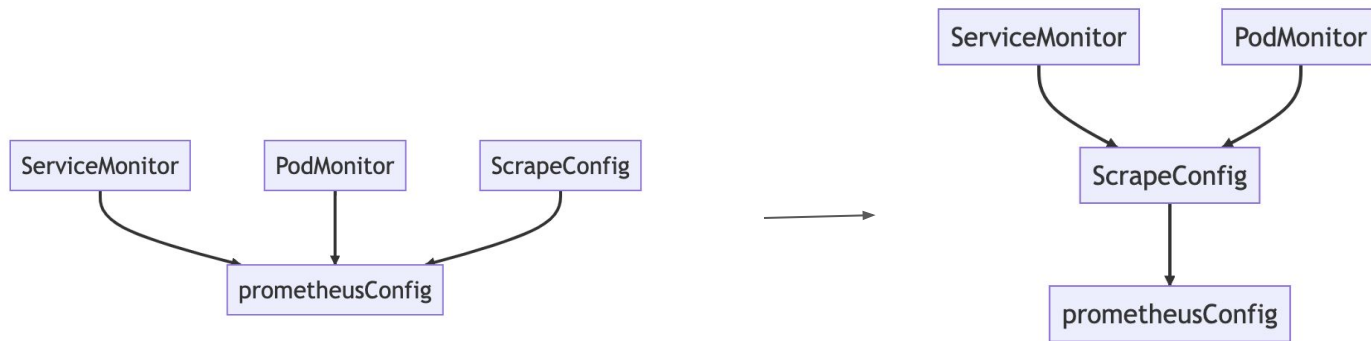
## v1Alpha1

Defines additional scrape configurations the Kubernetes way

Monitor resources outside cluster as well (Say node exporter running outside k8s)

covers file\_sd\_configs, static\_configs and http\_sd\_configs now, more to come

# ScrapeConfig



Act the same as the other CRDs and append scrape configurations to the configuration

# Prometheus Agent

- Optimizes Prometheus for the remote write use case
- Disables querying, alerting, and local storage, and replaces it with a customized TSDB WAL
- Everything else stays the same: scraping logic, service discovery and related configuration

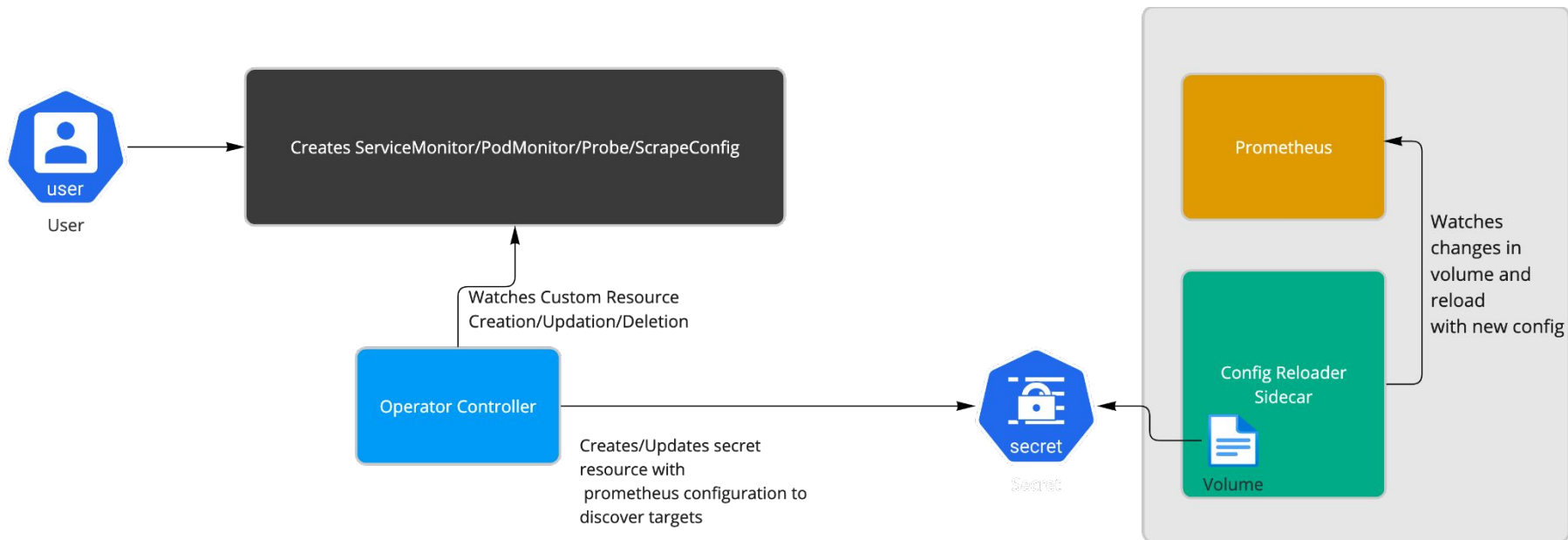
Read more on <https://prometheus.io/blog/2021/11/16/agent/>

# PrometheusAgent CR

```
apiVersion: monitoring.coreos.com/v1alpha1
kind: PrometheusAgent
metadata:
  name: prometheus-agent
spec:
  replicas: 2
  serviceAccountName: prometheus-agent
  serviceMonitorSelector:
    matchLabels:
      team: frontend
```

v1Alpha1

# How monitoring works?



# Alerting



# Alerting - PrometheusRule



Recording



Alerting

```
apiVersion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
  labels:
    app.kubernetes.io/component: grafana
    app.kubernetes.io/name: grafana
  name: grafana-rules
  namespace: openshift-monitoring
spec:
  groups:
    - name: GrafanaAlerts
      rules:
        - alert: GrafanaRequestsFailing
          annotations:
            message: '{{ $labels.namespace }}/{{ $labels.job }}/{{ $labels.handler }}
              is experiencing {{ $value | humanize }}% errors'
          expr: |
            100 * namespace_job_handler_statuscode:grafana_http_request_duration_seconds_count:rate5m{handler!~/a
              / ignoring (status_code)
              sum without (status_code) (namespace_job_handler_statuscode:grafana_http_request_duration_seconds_coun
              > 50
          for: 5m
          labels:
            severity: warning
        - name: grafana_rules
          rules:
            - expr: |
                sum by (namespace, job, handler, status_code) (rate(grafana_http_request_duration_seconds_count[5m]))
              record: namespace_job_handler_statuscode:grafana_http_request_duration_seconds_count:rate5m
```

We have metrics now, how to inform when something goes wrong?

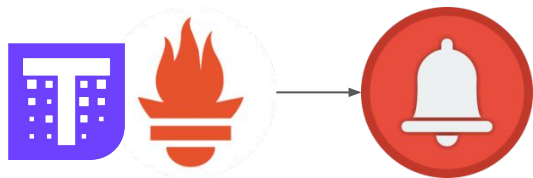
# Admission Webhook

- Prevent invalid alerting and recording rule from causing failures in a deployed Prometheus instance
- Validate PrometheusRule resources upon initial creation or update

Checkout:

<https://github.com/prometheus-operator/prometheus-operator/blob/main/Documentation/user-guides/webhook.md>

# Alerting - AlertManager



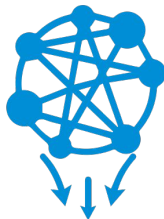
Receive Alerts from Prometheus/Thanos



Alert Definition by PrometheusRule



Deduplication



Aggregation



Notification

# Alerting - AlertManager CRD

- AlertManager StatefulSet
- Automatic Cluster Setup
- Global Configuration
  - Secret “alertmanager-name”
  - AlertManagerConfig CRD
- Wired to Prometheus by alerting.alertmanagers

```
apiVersion: monitoring.coreos.com/v1
kind: Prometheus
metadata:
  name: k8s
  namespace: monitoring
spec:
  alerting:
    alertmanagers:
      - apiVersion: v2
        name: alertmanager-main
        namespace: monitoring
        port: web
```

```
apiVersion: monitoring.coreos.com/v1
kind: Alertmanager
metadata:
  labels:
    app.kubernetes.io/instance: main
    app.kubernetes.io/name: alertmanager
    app.kubernetes.io/part-of: kube-prometheus
  name: main
  namespace: monitoring
spec:
  image: quay.io/prometheus/alertmanager:v0.24.0
  nodeSelector:
    kubernetes.io/os: linux
  podMetadata:
    labels:
      app.kubernetes.io/instance: main
      app.kubernetes.io/name: alertmanager
      app.kubernetes.io/part-of: kube-prometheus
  replicas: 3
  resources:
    limits:
      cpu: 100m
      memory: 100Mi
    requests:
      cpu: 4m
      memory: 100Mi
  securityContext:
    fsGroup: 2000
    runAsNonRoot: true
    runAsUser: 1000
  serviceAccountName: alertmanager-main
  version: 0.24.0
```

# Managing AlertManager Configuration

Several options to provide the Alertmanager configuration:

- You can use a native Alertmanager configuration file stored in a Kubernetes secret
  - ◆ Secret name: `alertmanager-{ALERTMANAGER_NAME}`
- You can use `spec.alertmanagerConfiguration` to reference an `AlertmanagerConfig` object in the same namespace which defines the main Alertmanager configuration
- You can define `spec.alertmanagerConfigSelector` and `spec.alertmanagerConfigNamespaceSelector` to tell the operator which `AlertmanagerConfigs` objects should be selected and merged with the main Alertmanager configuration.

# Alerting - AlertManagerConfig

- Subsections of AlertManager Configuration
  - Route (Grouping)
    - Combine alerts of similar nature into a single notification.
  - Receiver
    - List of notification receivers.
  - Inhibit Rules
    - Suppress notifications for certain alerts when related alerts are already firing.
  - Time Interval
    - When certain routes are activated or deactivated
- May Merge with other AlertManagerConfig
- Decouple application monitoring from infrastructure management.

```
inhibit_rules:  
  - equal:  
    - namespace  
    - alertname  
    source_matchers:  
      - "severity = critical"  
    target_matchers:  
      - "severity =~ warning|info"  
receivers:  
  - name: Default  
  - name: Critical  
route:  
  group_by:  
    - namespace  
  group_interval: 5m  
  group_wait: 30s  
  receiver: Default  
  repeat_interval: 12h  
  routes:  
    - matchers:  
      - "severity = critical"  
      receiver: Critical
```

# Security - TLS



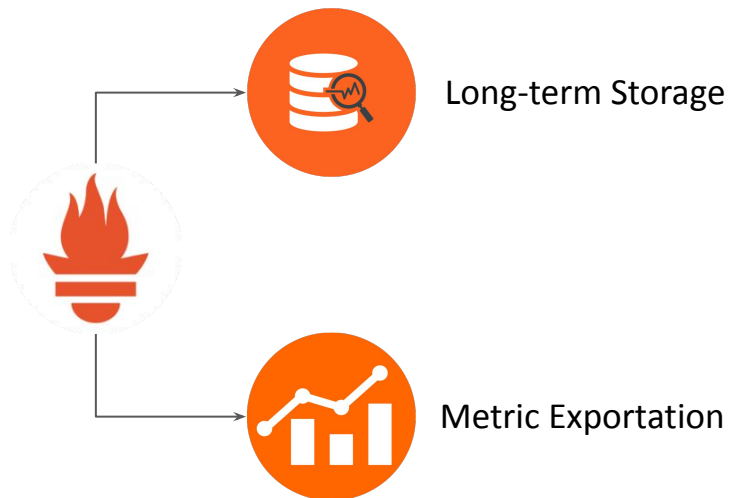
- CA
- Certificate
- Private Key



- Server Name
- Insecure Skip Validation

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: sample-app
  namespace: example-monitoring
spec:
  selector:
    matchLabels:
      app: sample-app
  endpoints:
    - port: mtls
      interval: 30s
      scheme: https
      tlsConfig:
        keySecret:
          name: tls-certs
          key: key.pem
        cert:
          secret:
            name: tls-certs
            key: cert.pem
```

# RemoteWrite



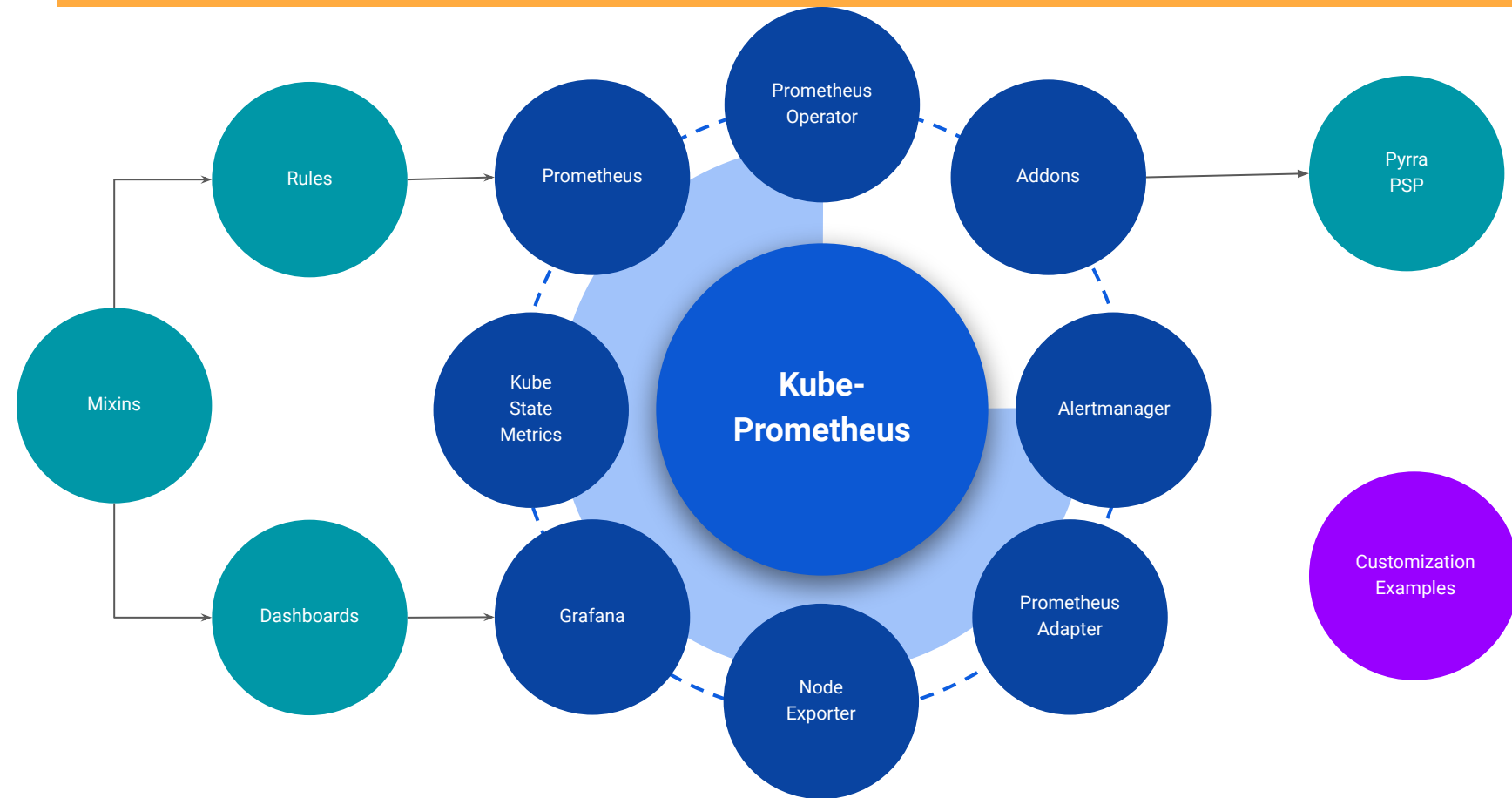
**RemoteWrite:** Forward (stream) all or a subset of metrics collected by Prometheus to the remote location

```
remoteWrite:
- url: https://remote.write.net
  remoteTimeout: 30s
  queueConfig:
    capacity: 1000
    minShards: 1
    maxShards: 10
    maxSamplesPerSend: 120
  oauth2:
    clientId:
    secret:
      name: remote-write-credentials
      key: client-id
    clientSecret:
      name: remote-write-credentials
      key: client-secret
    tokenUrl: https://token.url
    scopes:
      - scope1
```



# Configure Monitoring Stack

# Kube-Prometheus



# Limitations

01

## Config Simplification

- Hide certain complexity from user
- Opinionated Operator

02

## Scaling Prometheus

- Experimental Sharding Support

03

## Release

- Release following Prometheus versions
- New features available after delay

## Summary

# Operator Config Sidenotes

Operator Pattern  
Prometheus Operator

Monitoring Stack  
TLS  
Remote Write

Limitation & Caveats  
New Features

# Interested in Contributing?



Documentation  
Tutorials  
Features  
Bugfixes

GitHub repository view for `prometheus-operator/prometheus-operator` (Public).

Navigation: < Code Issues 212 Pull requests 79 Discussions Actions Projects Security Insights

Alertmanager mTLS config #4241 opened on Sep 2, 2021 by TamasHeuener  
Open 2

Emit Kubernetes events when the operator recognizes invalid r... #3611 opened on Oct 21, 2020 by simonpasquier  
Open 8

Filters: is:issue is:open Labels 19 Milestones 0 New issue

212 Open ✓ 2,267 Closed

Issue	Author	Label	Projects	Milestones	Assignee	Sort
Support new global scrape configuration parameters <code>kind/feature</code>	#5638 opened 1 hour ago by simonpasquier					
ScrapeConfig objects don't work properly when Prometheus sharding is used <code>help wanted</code> <code>kind/bug</code>	#5637 opened 2 hours ago by simonpasquier					
Move PrometheusTracingConfig to CommonPrometheusFields <code>good first issue</code> <code>help wanted</code>	#5635 opened 3 hours ago by ArthurSens					
Add support for consul_sd_config to scrapeConfigs CRD <code>kind/feature</code>	#5629 opened yesterday by hellabp					1
Add support for authentication to the ScrapeConfig CRD <code>help wanted</code> <code>kind/feature</code>	#5620 opened 2 days ago by aedwardstx 2 tasks					1
Add examples for prometheus-agent and scrapeconfig usage <code>kind/documentation</code>	#5607 opened last week by slashpai					6
Support VPA for Prometheus Resource <code>kind/feature</code>	#5594 opened 2 weeks ago by atsa1220					1
Prometheus Operator ScrapConfig CRD cannot support federation <code>help wanted</code> <code>kind/feature</code>	#5592 opened 2 weeks ago by fluzzkitten					

See <https://github.com/prometheus-operator/prometheus-operator/blob/main/CONTRIBUTING.md>



- <https://github.com/prometheus-operator/prometheus-operator/tree/main/Documentation>
- <https://github.com/prometheus-operator/kube-prometheus/tree/main/docs/customizations>
- <https://github.com/slashpai/prometheus-operator-examples>
- <https://www.youtube.com/watch?v=MuhPMXCGiLc>
- <https://prometheus.io/blog/2021/11/16/agent>

# Thank you!



<https://github.com/prometheus-operator>



<https://prometheus-operator.dev>