OTUS

Онлайн-образование

Не забыл включить запись

Меня хорошо видно && слышно?

Ставьте плюсы, если все хорошо

Напишите в чат, если есть проблемы

Мониторинг k8s

План

- Метрики предоставляемые Kubernetes
- Компоненты мониторинга
- Обзор основных метрик
- Мониторинг дополнительных компонентов
- Prometheus
- Prometheus operator
- Альтернативный оператор

Метрики и мониторинг

ОПИСАНИЕ ВСЕХ КОМПОНЕНТОВ МОНИТОРИНГА, И КАК ЕГО ИСПОЛЬЗОВАТЬ ОПИСАНЫ В ДОКУМЕНТЕ

KUBERNETES MONITORING ARCHITECTURE

Метрики и мониторинг

PIPELINES

- Core metrics pipeline
  - Kubelet
  - Resource estimator
  - Metrics-server
  - API server
- Monitoring pipeline
  - Expose metrics to end-users

Метрики и мониторинг

- Metric-server предоставляет данные для HPA через адаптер
- Использование "Infrastore" в качестве хранилища данных
- Пользовательские системы мониторинга не взаимодействуют напрямую с metric-server
- cAdvisor должен быть источником метрик контейнеров
- Все компоненты k8s должны предоставлять (если должны) метрики в формате пригодном для Prometheus

Метрики и мониторинг

- Core metrics
  - CPU
  - RAM
  - Disk
  - Network

Подходы к мониторингу

- The Four Golden Signals
- The USE Method
- The RED Method

The Four Golden Signals

Подход был озвучен в "SRE handbook"

- Latency — Время необходимое на обслуживание запроса
- Traffic — Количество запросов отправляемых на вашу систему
- Errors — Количество ошибок
- Saturation — Насколько полон ваш сервис ^_^

The USE Method

Подход озвучил Брендан Грэг

- Resource - все компоненты физческого\виртуального сервера (CPU, Disk, RAM, etc.)
- Utilization - время которое затрачивает ресурс на выполнение задач
- Saturation - показатель указывающий на количество тасков которые не могут быть выполнены, при этом попадая в очередь
- Errors - количество ошибок

The RED Method

Подход озвучил Том Вилки

`The USE method is for resources and the RED method is for my services`

- Rate - количество запросов в секунду
- Errors - количество запросов завершившихся с ошибкой
- Duration - количество времени которое занимает каждый запрос

Мониторинг (Метрики контейнеров)

- Kubelet предоставляет метрики доступные по /metrics
- Cadvisor => cgroups
- CPU, RAM, DISK, Network

Мониторинг Контейнеры CPU

- container_cpu_user_seconds_total - общее пользовательское время (user ns)
- container_cpu_system_seconds_total - общее системное время (kernel ns)
- container_cpu_usage_seconds_total - сумма

Мониторинг Контейнеры CPU

```
sum(rate(container_cpu_cfs_throttled_seconds_total[5m]))
by (container_name)
```

МОНИТОРИНГ ВРЕМЕНИ КОГДА НАШИ КОНТЕЙНЕРЫ НАЧИНАЮТ "ТРОТЛИТЬ"

Мониторинг Контейнеры RAM

- container_memory_working_set_bytes

```
sum(container_memory_working_set_bytes{name!~"POD"})
by (name)
```

OOMKILLER ВСЕ ВИДИТ ^_^

Мониторинг Контейнеры DISK

- container_fs_io_time_seconds_total
- container_fs_io_time_weighted_seconds_total

```
sum(rate(container_fs_writes_bytes_total[5m])) by (container_name
sum(rate(container_fs_reads_bytes_total[5m])) by (container_name,
```

Мониторинг Контейнеры Network

```
sum(rate(container_network_receive_bytes_total[5m])) by (name)
sum(rate(container_network_transmit_bytes_total[5m])) by (name)
```

- container_network_receive_packets_dropped_total
- container_network_transmit_packets_dropped_total
- container_network_receive_errors_total
- container_network_transmit_errors_total

Мониторинг K8s API Server

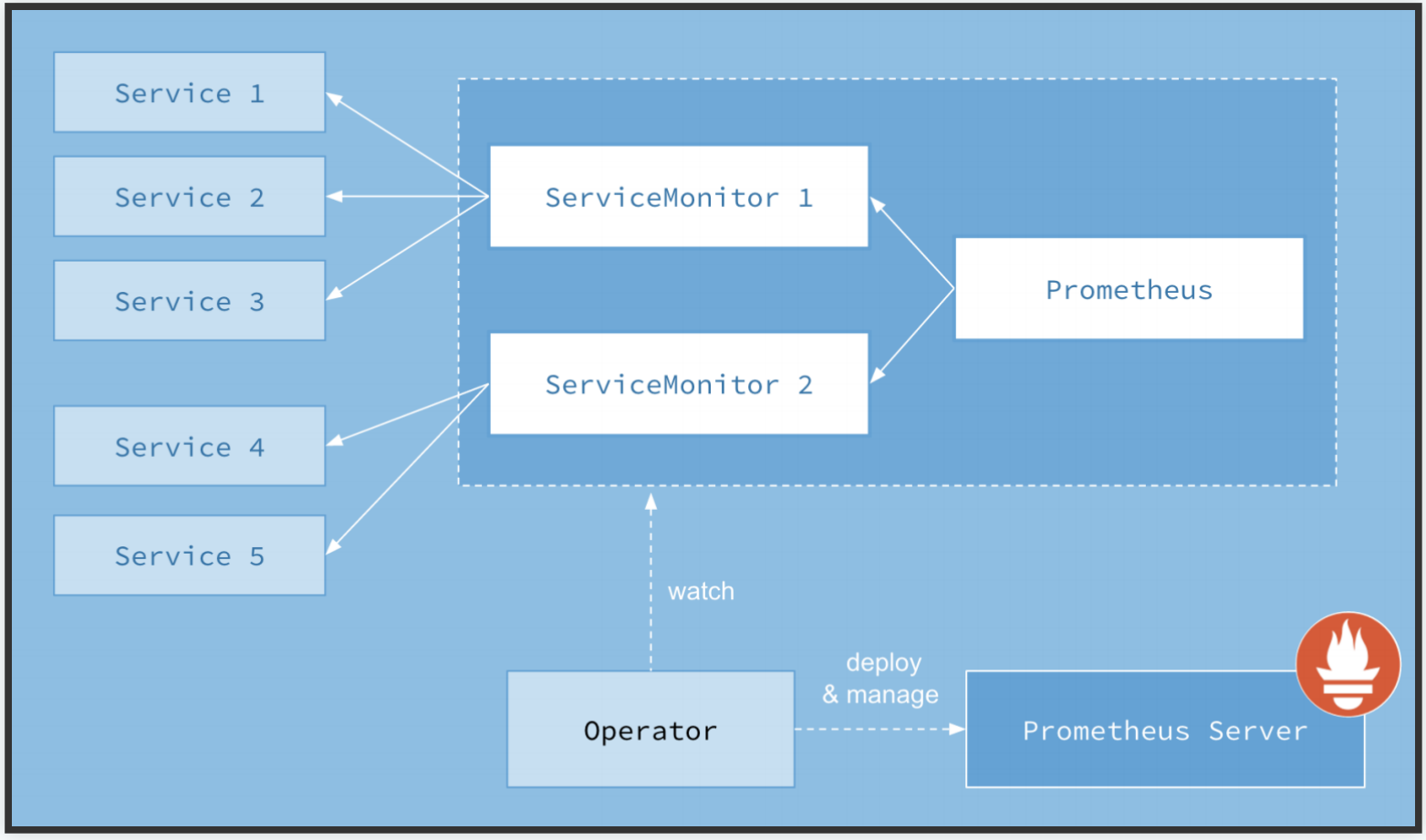
- Request Rates and Latencies
- Performance of controller work queues
- Etcd helper cache work queues and cache performance
- General process status (File Descriptors/Memory/CPU Seconds)
- Golang status (GC/Memory/Threads)

Мониторинг ETCD

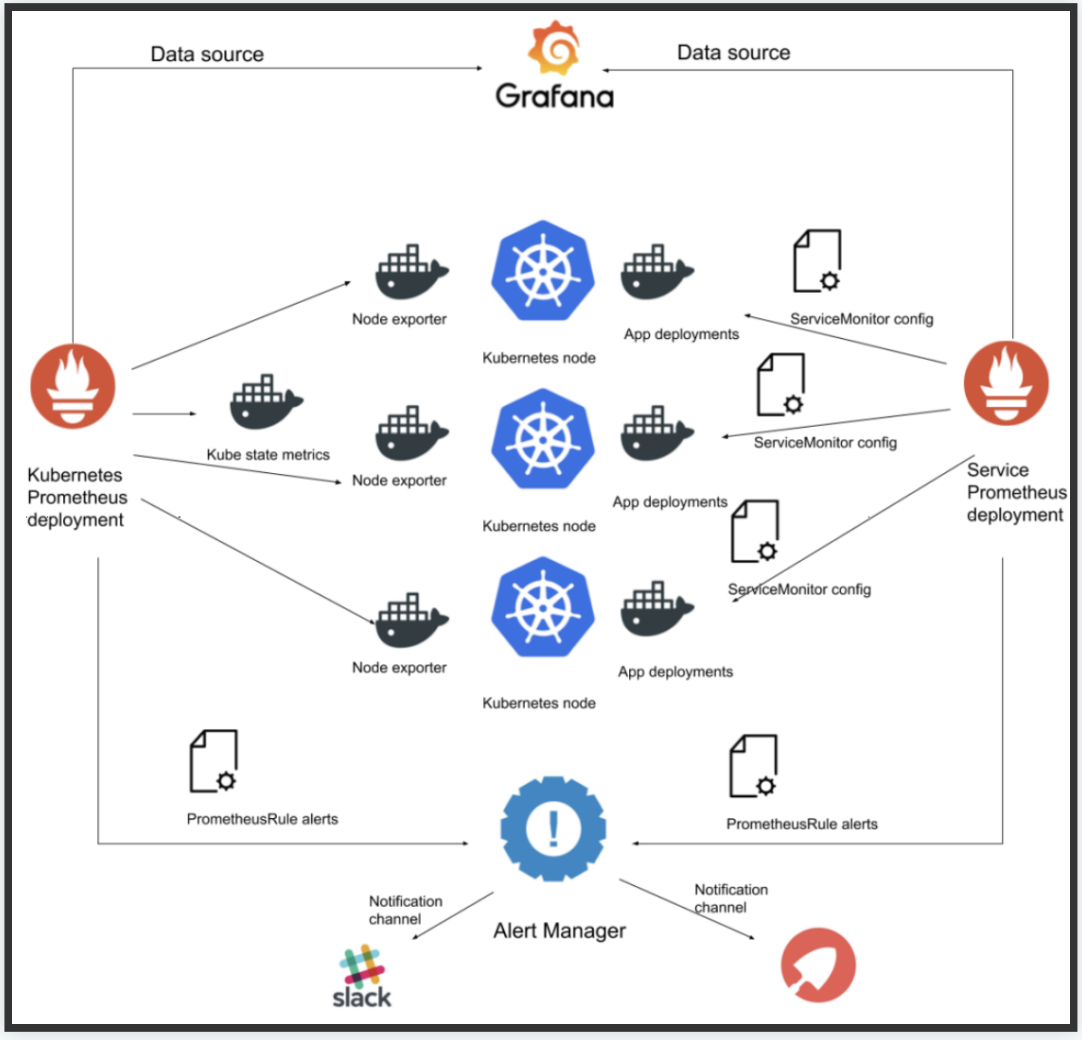
- Leader existence and leader change rate
- Proposals committed/applied/pending/failed
- Disk write performance
- Inbound gRPC stats
- Intra-cluster gRPC stats

Prometheus

Prometheus


Service 1
Service 2
Service 3
Service 4
Service 5
ServiceMonitor 1
ServiceMonitor 2
Prometheus
watch
Operator
deploy
& manage
Prometheus Server

Prometheus


Data source
Grafana
Data source
Node exporter
Kubernetes node
App deployments
ServiceMonitor config
Kube state metrics
Node exporter
Kubernetes node
App deployments
ServiceMonitor config
Kubernetes
Prometheus
deployment
Service
Prometheus
deployment
Node exporter
Kubernetes node
App deployments
ServiceMonitor config
PrometheusRule alerts
PrometheusRule alerts
Notification
channel
Alert Manager
Notification
channel
slack

Prometheus operator

- Helm chart
- Official repo
- Вкатить его руками ^_^

Prometheus operator values.yaml

```yaml
prometheus:
  prometheusSpec:
    retention: 3d
    ### Check this labels: kubectl get prometheus -o yaml -n moni
    serviceMonitorNamespaceSelector: {} ### Namespace for Service
    serviceMonitorSelectorNilUsesHelmValues: false
     serviceMonitorSelector: {} ### matchLabels for ServiceMonito
    # serviceMonitorSelector:
    # matchLabels:
    # prometheus: kube-prometheus
    # release: prometheus-cluster-monitoring
grafana:
  adminPassword: XXXXXXXXXXXXXXXXXXXX
```

Prometheus operator

SERVICE MONITOR

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: example-app
  labels:
    team: frontend
spec:
  selector:
    matchLabels:
      app: example-app
  endpoints:
  - port: web
```