Подключаемся к стенду

!!! ВНИМАНИЕ: Перед тем, как работать со стендом, его необходимо запустить.

Доступы рядом с кнопкой запуска стенда.

1. Подключаемся по SSH к sbox.slurm.io:

```
ssh s<номер вашего логина>@sbox.slurm.io
```

2. Подключаемся к своему кластеру Kubernetes:

```
ssh master-1.s<ваш номер логина>
```

3. Становимся root:

```
sudo -s
```

Добавляем дашборд в Grafana

!!! ВНИМАНИЕ: Работы продолжаем на стенде из предыдущего шага

1. Откройте веб интерфейс Grafana в браузере (в режиме "инкогнито")

```
grafana.s<ваш номер логина>.edu.slurm.io
```

2. Создайте дашборд с графиками Latency и Errors для API server как показано в видео.

P.S. Вы также можете импортнуть готовый дашборд из видео https://grafana.com/grafana/dashboards/10907 но рекомендуем сначала попробовать составить нужные графики самостоятельно.

Устанавливаем Kibana

!!! ВНИМАНИЕ: Работы продолжаем на стенде из предыдущего шага

1. Скачайте values чарта `elastic/kibana`

```
helm show values elastic/kibana > kibana-values.yaml
```

2. Отредактируйте values для Kibana. Найдите секцию `ingress`. Поменяйте в ней `enabled: false` на `enabled: true`, в hosts пропишите адрес - `kibana.s<ваш номер логина>.edu.slurm.io`

3. Установите чарт в кластер как в предыдущих шагах с использованием сохраненных values в неймспейс `logging`

```
helm upgrade -i kibana elastic/kibana -f kibana-values.yaml -n logging
```

```
kubectl get pods --namespace=logging -w
```

4. Проверьте в браузере, что у вас открывается интерфейс Kibana по адресу (в режиме "инкогнито") `kibana.s<ваш номер логина>.edu.slurm.io`

Полезные метрики для мониторинга Control Plane

ETCD

etcd_server_leader_changes_seen_total - как часто меняется лидер в кластере Etcd

grpc_server_handled_total - все запросы в Etcd (например можно мониторить ошибки - grpc_server_handled_total{grpc_code!="OK"})

grpc_server_handling_seconds_bucket - на основании этой метрики можно построить Latency запросов в Etcd (см пример с API сервером из видео)

etcd_network_peer_round_trip_time_seconds_bucket - на основании этой метрики можно построить Latency общения инстансов Etcd между собой

etcd_disk_wal_fsync_duration_seconds_bucket - позволяет построить график скорости работы с диском (один из самых критичных для Etcd показателей)

Kube API Server

apiserver_request_count - позволяет посчитать rps и количество ошибок в запросах к API серверу Кубернетиса

apiserver_client_certificate_expiration_seconds_count - позволяет следить за временем жизни сертификатов в кластере

apiserver_request_latencies_bucket - на основании этой метрики можно построить Latency запросов в kibe-api

Kubelet (Node)

kube_node_status_condition - позволяет проверить статус ноды по множеству различных показателей (например {condition=«Ready»,status="true"} позволяет понять что нода находится в статусе Ready или {condition="MemoryPressure",status="true"} позволяет узнать что на ноде не хватает оперативной памяти)

Полезные алерты для кластера Kubernetes

Прежде всего рекомендуется ознакомиться со списком алертов тут (https://awesome-prometheus-alerts.grep.to/rules#kubernetes) и

Тут:

https://github.com/prometheus-operator/kube-prometheus/blob/main/manifests/kube-prometheus-prometheusRule.yaml

https://github.com/prometheus-operator/kube-prometheus/blob/main/manifests/kube-state-metrics-prometheusRule.yaml

https://github.com/prometheus-operator/kube-prometheus/blob/main/manifests/node-exporter-prometheusRule.yaml

1. Добавьте в Prometheus на созданном ранее стенде несколько алертов из списка выше (например для мониторинга состояния подов)
2. Попробуйте вызвать срабатывание алерта (например создайте в кластере под с заведомо фейлящейся командой)
3. Посмотрите в веб интерфейсе прометеуса описание сработавшего алерта (вы можете установить чекбокс «show annotaiotns» и увидеть как рендерятся шаблоны desctiption для сработавшего алерта)