

Васильев Егор

Positive Technologies



Тестировать Helm-чарты проще, чем их писать

О себе



- DevOps
- Работал в крупных международных компаниях (Veeam Software, JetBrains, Huawei)
- Веду технический блог <http://blog.bissquit.com>

Аудитория

- QA-инженеры любого уровня, не работавшие с Kubernetes и которым не приходилось сталкиваться с тестированием инфраструктуры, но которым **очень хочется** влиться в тему
- Инженеры разных специализаций (dev, ops, other), которые хотят повысить качество своих деплоев в Kubernetes

Как возникла идея доклада

QA тестируют приложения, но мало кто тестирует инфраструктуру, хотя

иногда качество инфраструктуры влияет на стабильность работы приложения даже больше, чем качество самого приложения

О чем поговорим?

- Что такое Helm
- Создадим чарт
- Узнаем как тестировать чарты
 - Подготовим окружение
 - Подеплоим чарт в кластер Kubernetes
- Создадим пайплайн

01



Что такое Helm?

Что такое Helm

Как это видит официальная документация:

The package manager for Kubernetes, CNCF graduated



Что такое Helm

Как это видим мы:

Шаблонизатор с GO templates под капотом,

который позволяет сделать множество громоздких
yaml-манифестов Kubernetes более гибкими для
депоя в разные окружения



Что такое Helm

```
apiVersion: v1
kind: Service
metadata:
  name: rate-limits-exporter
  labels:
    app.kubernetes.io/name: rate-limits-exporter
spec:
  type: ClusterIP
  ports:
    - port: 80
      targetPort: http
      protocol: TCP
      name: http
  selector:
    app.kubernetes.io/name: rate-limits-exporter
```

Что такое Helm

```
apiVersion: v1
kind: Service
metadata:
  name: {{ include "rate-limits-exporter.fullname" . }}
  labels:
    {{- include "rate-limits-exporter.labels" . | nindent 4 }}
spec:
  type: {{ .Values.service.type }}
  ports:
    - port: {{ .Values.service.port }}
      targetPort: http
      protocol: TCP
      name: http
  selector:
    {{- include "rate-limits-exporter.selectorLabels" . | nindent 4 }}
```

Что такое Helm

_helpers.tpl

```
1 {{- define "rate-limits-exporter.fullname" -}}
  {{- if .Values.fullnameOverride }}
2 {{- .Values.fullnameOverride | trunc 63 | trimSuffix "-" }}
  {{- else }}
  {{- $name := default .Chart.Name .Values.nameOverride }}
3 {{- if contains $name .Release.Name }}
  {{- .Release.Name | trunc 63 | trimSuffix "-" }}
  {{- else }}
  {{- printf "%s-%s" .Release.Name $name | trunc 63 | trimSuffix "-" }}
  {{- end }}
  {{- end }}
  {{- end }}
```

Что такое Helm

```
apiVersion: v1
kind: Service
metadata:
  name: {{ include "rate-limits-exporter.fullname" . }}
  labels:
    {{- include "rate-limits-exporter.labels" . | nindent 4 }}
spec:
  type: {{ .Values.service.type }}
  ports:
    - port: {{ .Values.service.port }}
      targetPort: http
      protocol: TCP
      name: http
  selector:
    {{- include "rate-limits-exporter.selectorLabels" . | nindent 4 }}
```

Что такое Helm

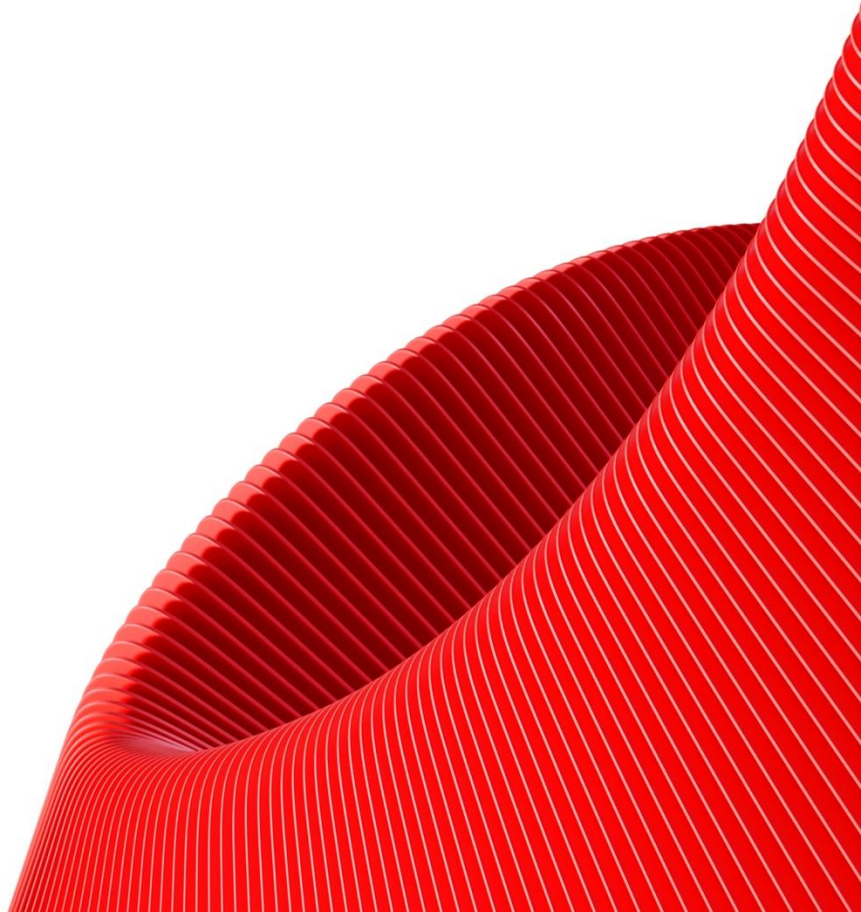
values.yaml

```
service:  
  type: ClusterIP  
  port: 80
```

02



Создаем чарт



Создаем чарт

Пример: кастомный экспортер метрик для Prometheus - rate-limits-exporter (показывает лимиты по количеству скачиваний образов DockerHub)


What's the download rate limit on Docker Hub?

Docker Hub limits the number of Docker image downloads, or pulls, based on the account type of the user pulling the image. Pull rate limits are based on individual IP address.

User type	Rate limit
Anonymous users	100 pulls per 6 hours per IP address
Authenticated users	200 pulls per 6 hour period
Users with a paid Docker subscription ↗	Up to 5000 pulls per day

Создаем чарт

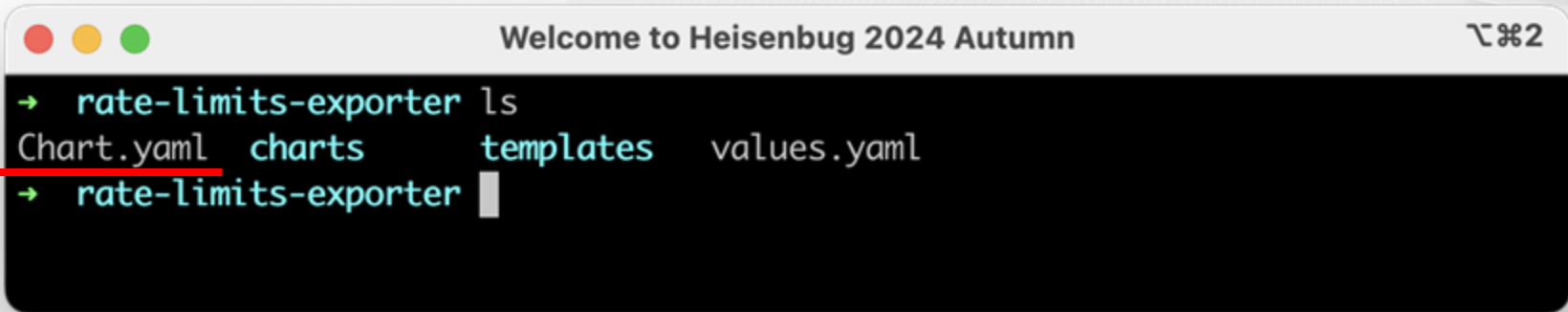
- Установим Helm
- Вызовем команду `helm create <chart-name>` для создания начальной структуры каталогов



```
Welcome to Heisenbug 2024 Autumn ⌘2  
→ conference helm create rate-limits-exporter  
Creating rate-limits-exporter  
→ conference cd rate-limits-exporter  
→ rate-limits-exporter █
```

Создаем чарт

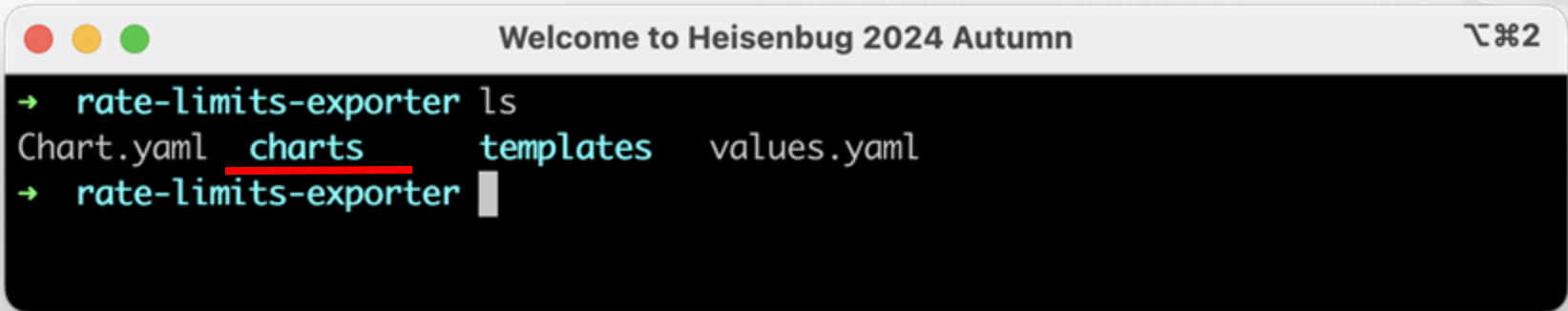
Общая информация о чарте (авторы, версии, описание)



```
Welcome to Heisenbug 2024 Autumn  ⌘2  
→ rate-limits-exporter ls  
Chart.yaml  charts  templates  values.yaml  
→ rate-limits-exporter
```

Создаем чарт

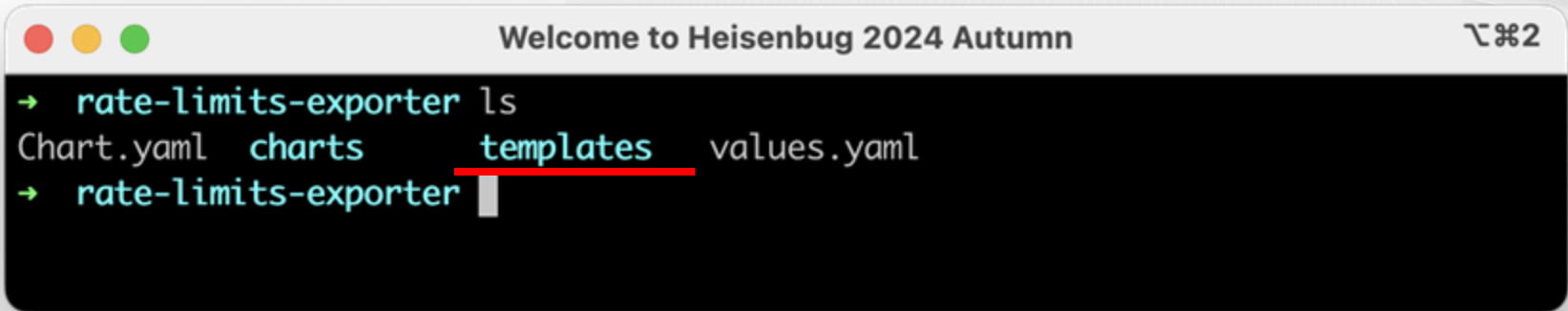
Любые другие чарты, от которых наш чарт зависит



```
Welcome to Heisenbug 2024 Autumn  ⌘2  
→ rate-limits-exporter ls  
Chart.yaml charts templates values.yaml  
→ rate-limits-exporter
```

Создаем чарт

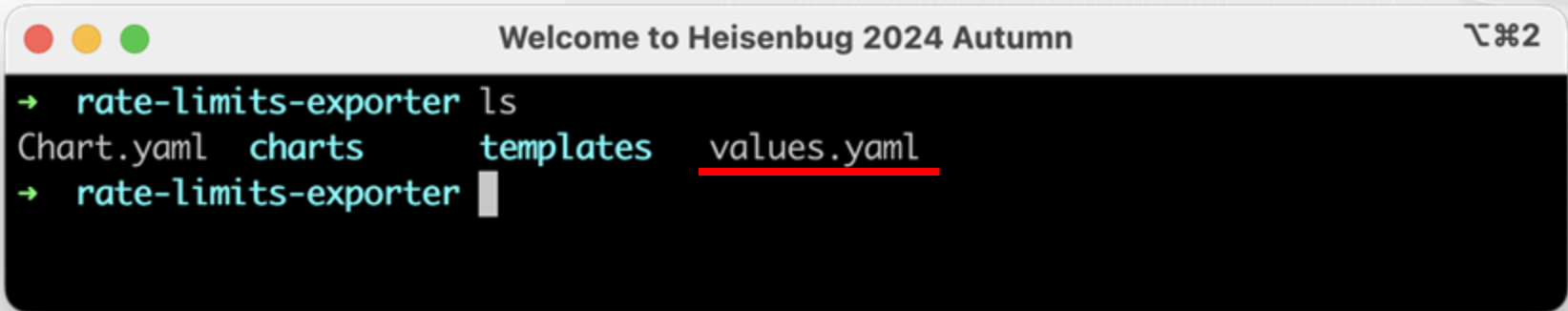
Шаблоны нашего чарта (+ values.yaml = Kubernetes manifests)



```
Welcome to Heisenbug 2024 Autumn  ⌘2  
→ rate-limits-exporter ls  
Chart.yaml  charts  templates  values.yaml  
→ rate-limits-exporter
```

Создаем чарт

Конфигурация чарта по умолчанию



```
Welcome to Heisenbug 2024 Autumn  ⌘2  
→ rate-limits-exporter ls  
Chart.yaml  charts  templates  values.yaml  
→ rate-limits-exporter
```


Создаем чарт

LICENSE - информация о лицензии

README.md - документация к чарту

values.schema.json - схема формата json, описывает структуру values.yaml

crds/ - каталог с Custom Resource Definitions

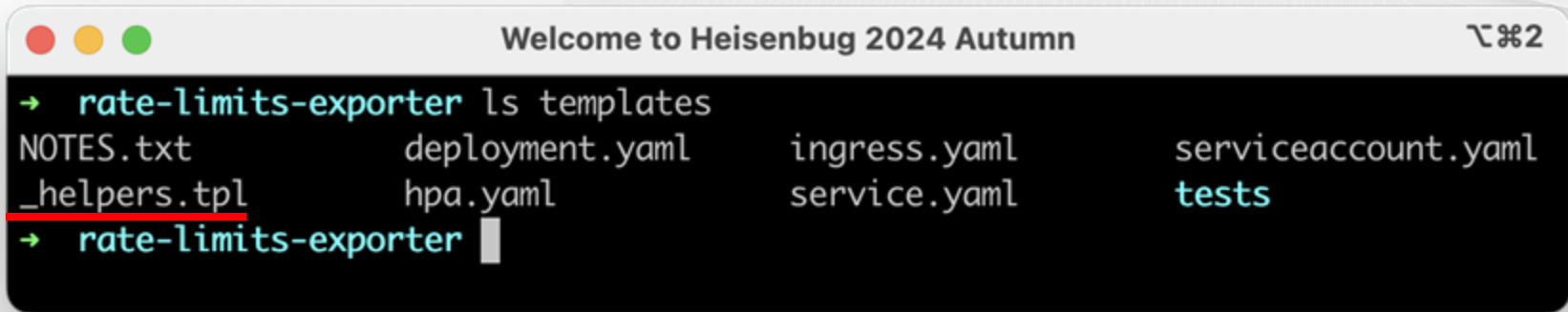
Создаем чарт

Заметки после успешной установки чарта

```
Welcome to Heisenbug 2024 Autumn ~⌘2  
→ rate-limits-exporter ls templates  
NOTES.txt           deployment.yaml      ingress.yaml         serviceaccount.yaml  
_helpers.tpl        hpa.yaml            service.yaml        tests  
→ rate-limits-exporter
```

Создаем чарт

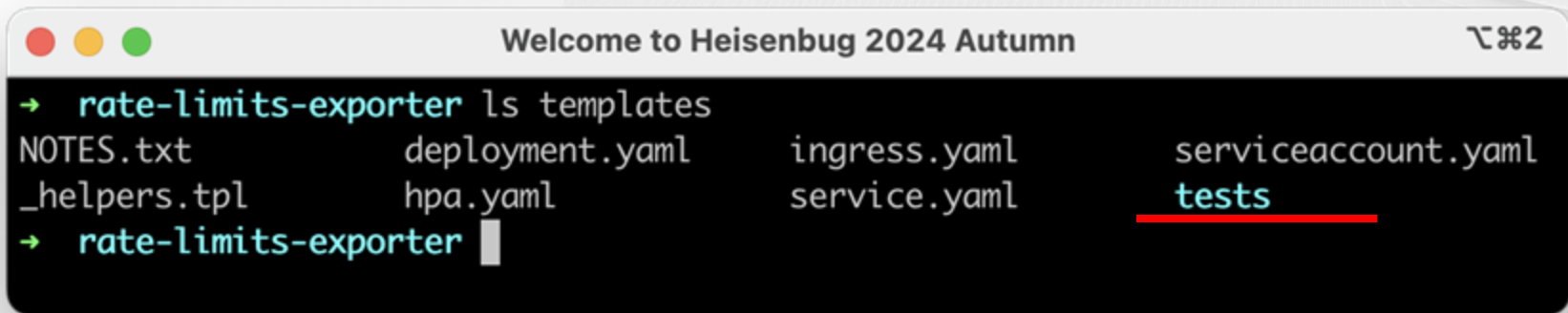
Файлы с `_` в начале содержат переиспользуемый код



```
Welcome to Heisenbug 2024 Autumn ~%2  
→ rate-limits-exporter ls templates  
NOTES.txt          deployment.yaml    ingress.yaml       serviceaccount.yaml  
_helpers.tpl     hpa.yaml          service.yaml       tests  
→ rate-limits-exporter
```

Создаем чарт

Магия Helm для тестирования чартов



```
Welcome to Heisenbug 2024 Autumn ⌘⌘2  
→ rate-limits-exporter ls templates  
NOTES.txt          deployment.yaml    ingress.yaml       serviceaccount.yaml  
_helpers.tpl       hpa.yaml          service.yaml       tests  
→ rate-limits-exporter
```

Создаем чарт

Пример Helm-чарта на GitHub



03



Как тестировать чарты

Тестирование чарта

Наверняка нам понадобится готовый кластер Kubernetes



Стоп! До этого момента мы можем использовать еще множество инструментов, не стоит забегать вперед

03

pt

Как тестировать чарты

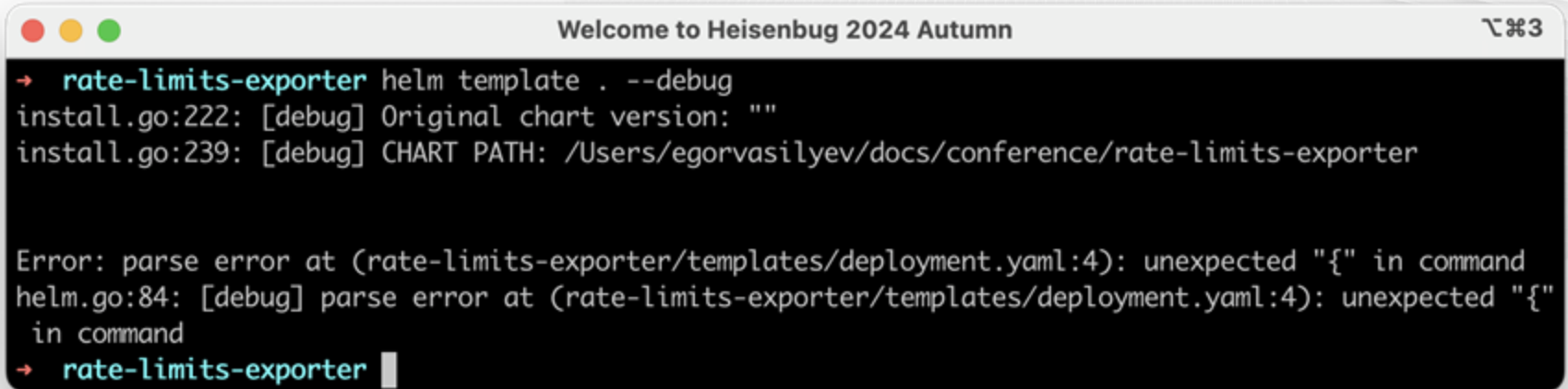
Встроенные инструменты Helm

Тестирование чарта

- Проверим, а способен ли Helm сделать рендер нашего чарта
- Если все хорошо, на выходе получите готовые к деплою манифесты

```
helm template <chart-name>
```

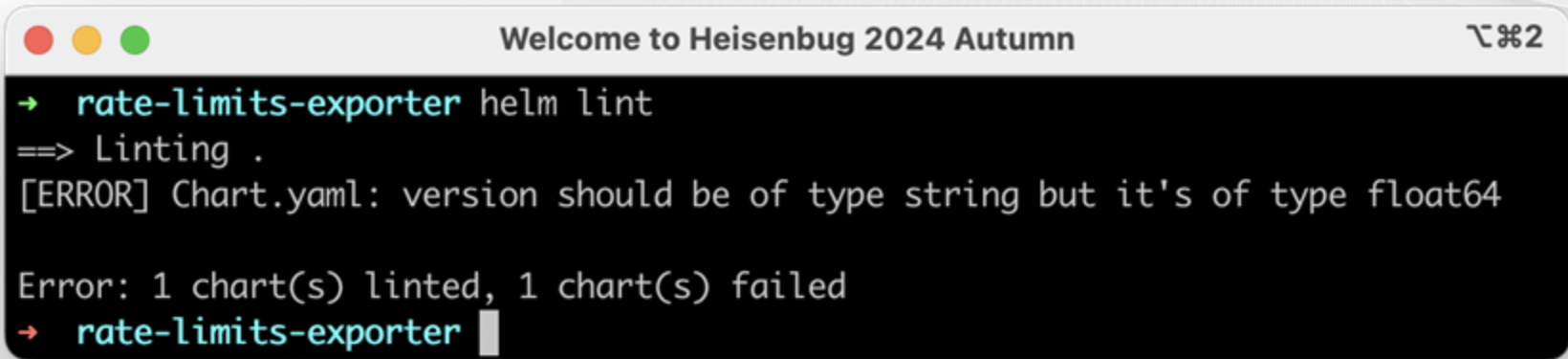
```
helm template .
```



```
Welcome to Heisenbug 2024 Autumn  ㄒ⌘3  
→ rate-limits-exporter helm template . --debug  
install.go:222: [debug] Original chart version: ""  
install.go:239: [debug] CHART PATH: /Users/egorvasilyev/docs/conference/rate-limits-exporter  
  
Error: parse error at (rate-limits-exporter/templates/deployment.yaml:4): unexpected "{" in command  
helm.go:84: [debug] parse error at (rate-limits-exporter/templates/deployment.yaml:4): unexpected "{"  
in command  
→ rate-limits-exporter
```

Тестирование чарта

- Полезный встроенный инструмент - **helm lint**, который проверит чарт и выдаст критические ошибки, если чарт не сможет подеплоиться

A terminal window titled "Welcome to Heisenbug 2024 Autumn" with a window control bar (red, yellow, green buttons) and a zoom icon. The terminal shows the command "rate-limits-exporter helm lint" being executed. The output indicates a linting error: "[ERROR] Chart.yaml: version should be of type string but it's of type float64". The summary shows "Error: 1 chart(s) linted, 1 chart(s) failed". The prompt "rate-limits-exporter" is shown at the bottom with a cursor.

```
→ rate-limits-exporter helm lint
==> Linting .
[ERROR] Chart.yaml: version should be of type string but it's of type float64

Error: 1 chart(s) linted, 1 chart(s) failed
→ rate-limits-exporter
```

А еще helm lint проверит соответствие схеме **values.schema.json**

Тестирование чарта

- Если чарт не соответствует принятым соглашениям и рекомендациям, **helm lint** выдаст предупреждение

```
Welcome to Heisenbug 2024 Autumn  ~#3
→ rate-limits-exporter helm lint
==> Linting .
[INFO] Chart.yaml: icon is recommended
[WARNING] templates/service.yaml: object name does not conform to Kubernetes naming requirements: "A-test-release-rate-limits-exporter": metadata.name: Invalid value: "A-test-release-rate-limits-exporter": a DNS-1035 label must consist of lower case alphanumeric characters or '-', start with an alphabetic character, and end with an alphanumeric character (e.g. 'my-name', or 'abc-123', regex used for validation is '[a-z]([-a-z0-9]*[a-z0-9])?')

1 chart(s) linted, 0 chart(s) failed
→ rate-limits-exporter █
```


03

pt

Как тестировать чарты

Валидация схемы

Тестирование чарта



ci passing homebrew 0.6.7 go report A+ GO reference

Kubeconform is a Kubernetes manifest validation tool. Incorporate it into your CI, or use it locally to validate your Kubernetes configuration!

Тестирование чарта

spec:

thisFieldDoesntExist: "some-data"

{{- with .Values.imagePullSecrets }}

imagePullSecrets:

{{- toYaml . | nindent 8 }}

{{- end }}

{{- if .Values.serviceAccount.create }}

serviceAccountName: {{ include "rate-limits-exporter.serviceAccountName" . }}

{{- end }}

securityContext:

Тестирование чарта

helm template . | kubeconform -summary -strict -verbose

```
Welcome to Heisenbug 2024 Autumn  2
rate-limits-exporter helm template . | kubeconform -summary -strict -verbose
stdin - ConfigMap release-name-rate-limits-exporter-autotests is valid
stdin - ServiceAccount release-name-rate-limits-exporter is valid
stdin - Service release-name-rate-limits-exporter is valid
stdin - Secret release-name-rate-limits-exporter is valid
stdin - Deployment release-name-rate-limits-exporter is invalid: problem validating schema. Check JSON
formatting: jsonschema: '/spec/template/spec' does not validate with https://raw.githubusercontent.com/yannh/kubernet
es-json-schema/master/master-standalone-strict/deployment-apps-v1.json#/properties/spec/properties/template/properties
/spec/properties/template/properties/spec/additionalProperties: additionalProperties 'thisFieldDoesntExist' not allowe
d
stdin - Pod release-name-rate-limits-exporter-autotests is valid
stdin - Pod release-name-rate-limits-exporter-test-connection is valid
Summary: 7 resources found parsing stdin - Valid: 6, Invalid: 1, Errors: 0, Skipped: 0
rate-limits-exporter echo $?
1
rate-limits-exporter
```

03

pt

Как тестировать чарты

Статические анализаторы

Тестирование чарта

kube-score



go report A+ Test Go passing release v1.18.0 Github stars 2.6k downloads 2.8M License MIT

`kube-score` is a tool that performs static code analysis of your Kubernetes object definitions.

The output is a list of recommendations of what you can improve to make your application more secure and resilient.

You can test `kube-score` out in the browser with the [online demo](#) ([source](#)).

Тестирование чарта



Kubernetes object analysis with recommendations for improved reliability and security.

`kube-score` is a tool that does static code analysis of your Kubernetes object definitions. The output is a list of recommendations of what you can improve to make your application more secure and resilient.

`kube-score` is [open-source and available under the MIT-license](#). For more information about how to use `kube-score`, see [zegl/kube-score](#) on GitHub. Use this website to easily test `kube-score`, just paste your object definition YAML or JSON in the box below.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: statefulset-test-1
spec:
  template:
    metadata:
      labels:
        app: foo
    spec:
      containers:
        - name: foobar
          image: foo:bar
---
apiVersion: policy/v1beta1
kind: PodDisruptionBudget
metadata:
```

```
apps/v1/Deployment statefulset-test-1
[CRITICAL] Container Resources
  - foobar -> CPU limit is not set
```

Тестирование чарта

Идентификатор проверки

Описание

Цель

ID	Target	Description	Enabled
deployment-strategy	Deployment	Makes sure that all Deployments targeted by service use RollingUpdate strategy	default
deployment-replicas	Deployment	Makes sure that Deployment has multiple replicas	default
ingress-targets-service	Ingress	Makes sure that the Ingress targets a Service	default
cronjob-has-deadline	CronJob	Makes sure that all CronJobs has a configured deadline	default
cronjob-restartpolicy	CronJob	Makes sure CronJobs have a valid RestartPolicy	default
container-resources	Pod	Makes sure that all pods have resource limits and requests set. The --ignore-container-cpu-limit flag can be used to disable the requirement of having a CPU limit	default
container-resource-requests-equal-limits	Pod	Makes sure that all pods have the same requests as limits on resources set.	optional
container-cpu-requests-equal-limits	Pod	Makes sure that all pods have the same CPU requests as limits set.	optional
container-memory-requests-equal-limits	Pod	Makes sure that all pods have the same memory requests as limits set.	optional
container-image-tag	Pod	Makes sure that a specific version latest tag is used	default

Тестирование чарта

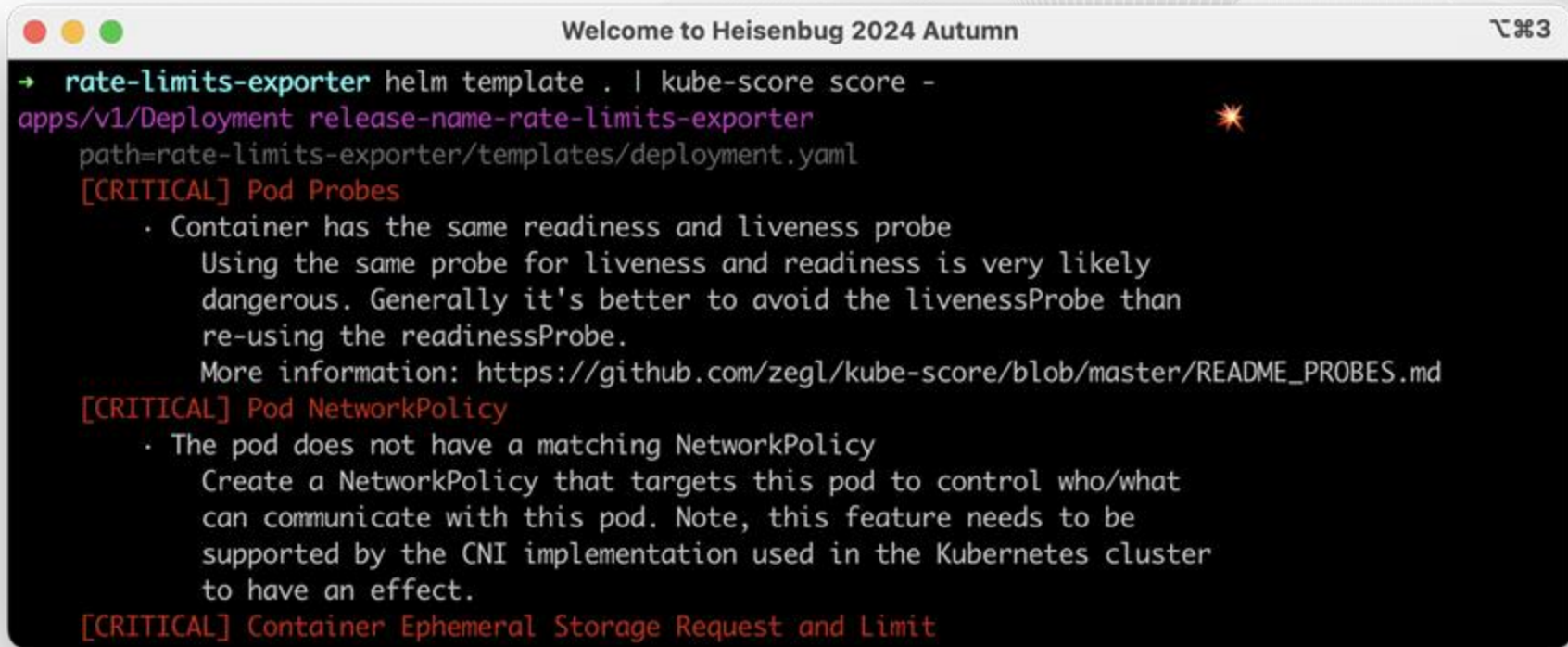
```
kube-score score --ignore-test pod-networkpolicy
```

OR

```
apiVersion: v1
kind: Pod
metadata:
  name: "{{ include "rate-limits-exporter.fullname" . }}-test-connection"
  labels:
    {{- include "rate-limits-exporter.labels" . | nindent 4 }}
  annotations:
    "helm.sh/hook": test
    "kube-score/ignore": container-ephemeral-storage-request-and-limit,pod-probes
```

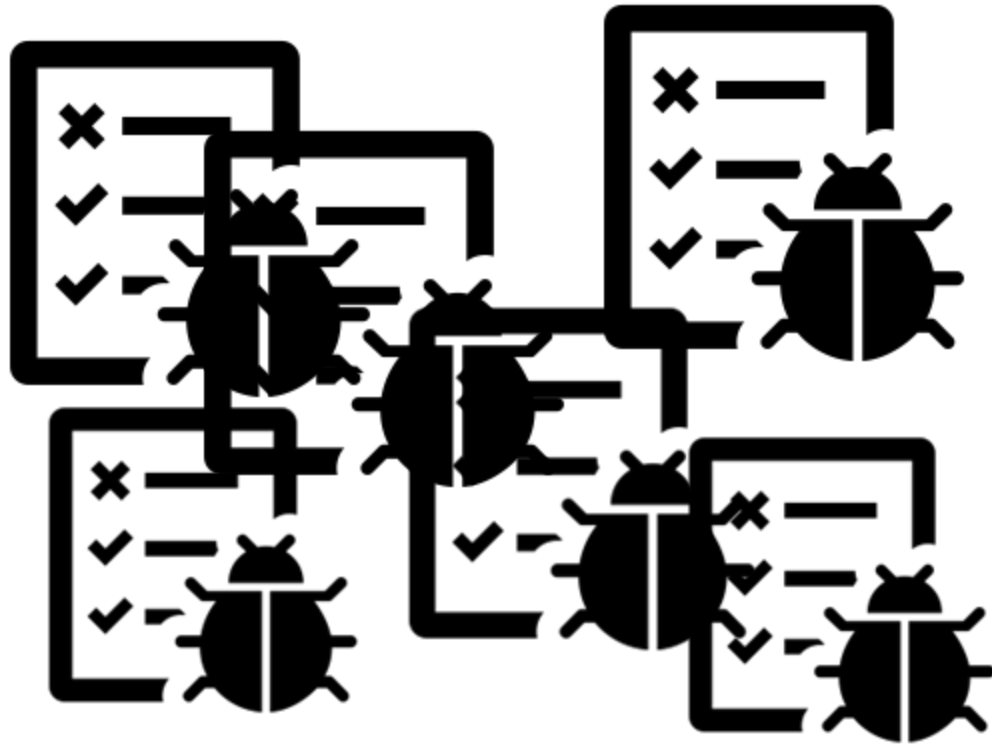

Тестирование чарта

```
helm template . | kube-score score -
```



```
Welcome to Heisenbug 2024 Autumn ~#3  
→ rate-limits-exporter helm template . | kube-score score -  
apps/v1/Deployment release-name-rate-limits-exporter  
path=rate-limits-exporter/templates/deployment.yaml  
[CRITICAL] Pod Probes  
  · Container has the same readiness and liveness probe  
    Using the same probe for liveness and readiness is very likely  
    dangerous. Generally it's better to avoid the livenessProbe than  
    re-using the readinessProbe.  
    More information: https://github.com/zegl/kube-score/blob/master/README\_PROBES.md  
[CRITICAL] Pod NetworkPolicy  
  · The pod does not have a matching NetworkPolicy  
    Create a NetworkPolicy that targets this pod to control who/what  
    can communicate with this pod. Note, this feature needs to be  
    supported by the CNI implementation used in the Kubernetes cluster  
    to have an effect.  
[CRITICAL] Container Ephemeral Storage Request and Limit
```

Тестирование чарта



BRACE YOURSELVES

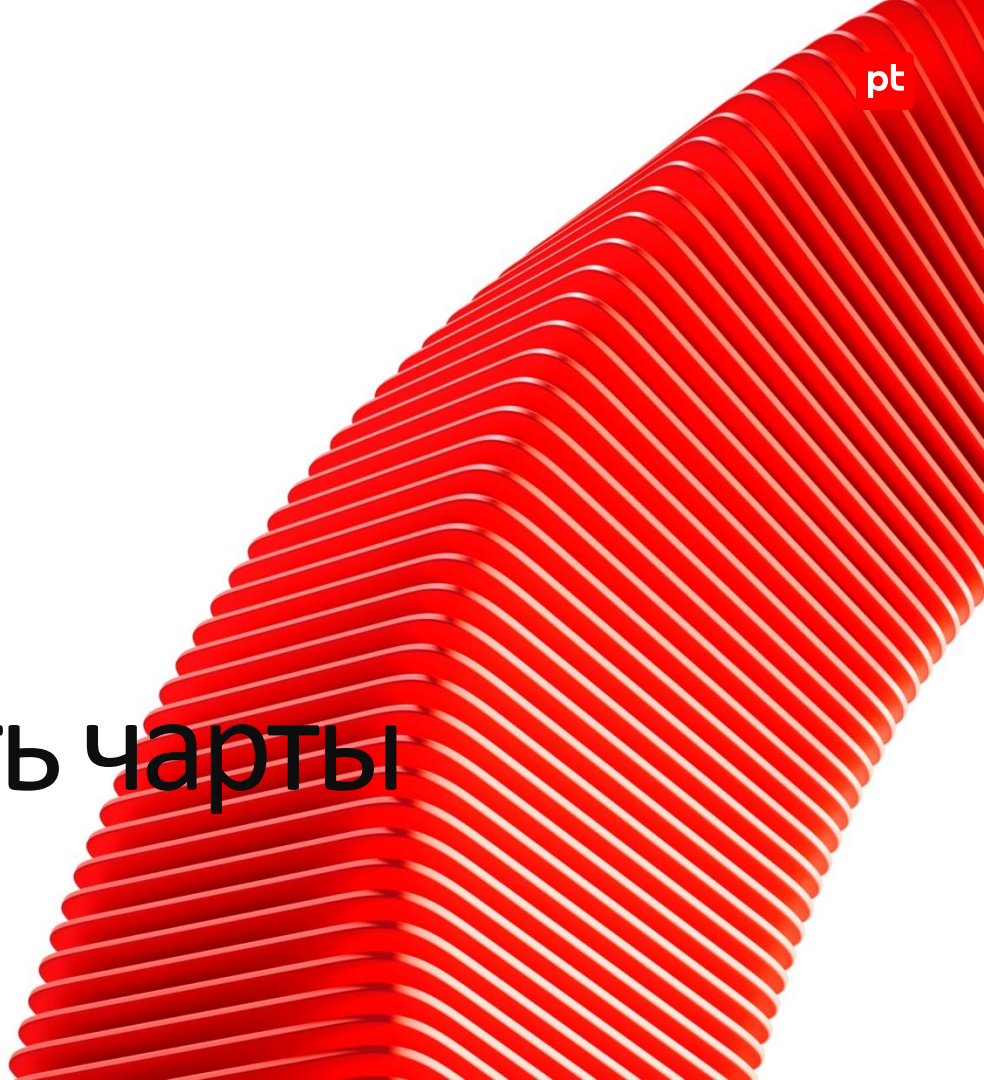


03

pt

Как тестировать чарты

Промежуточные итоги



Тестирование чарта

helm template
helm lint
kubecconform
kube-score

Инструменты для
тестирования чарта
локально без
необходимости
деплоить в кластер
Kubernetes

03

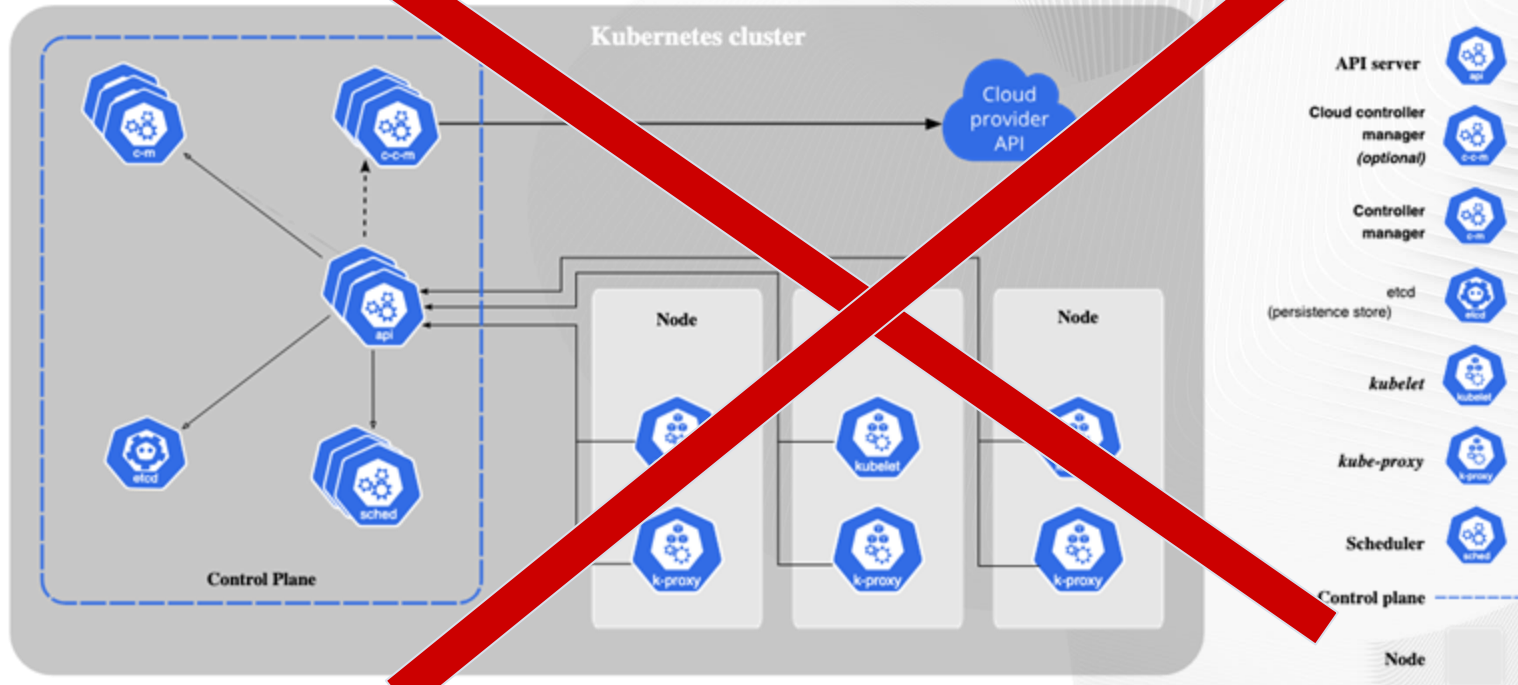
pt

Как тестировать чарты

Готовим окружение

Готовим окружение

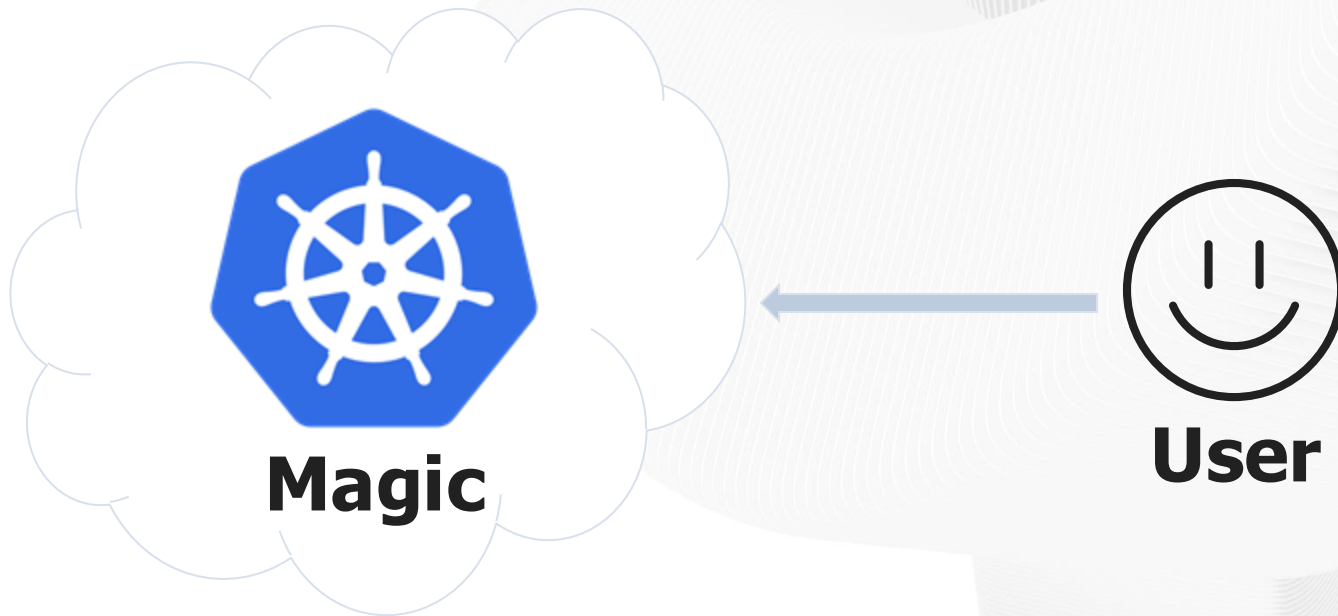
Архитектура Kubernetes



Готовим окружение

Kubernetes Cluster

Архитектура Kubernetes



Готовим окружение

Нам понадобится:

- kubectl
- kind

ГОТОВИМ ОКРУЖЕНИЕ

kubectl - Kubernetes command-line tool

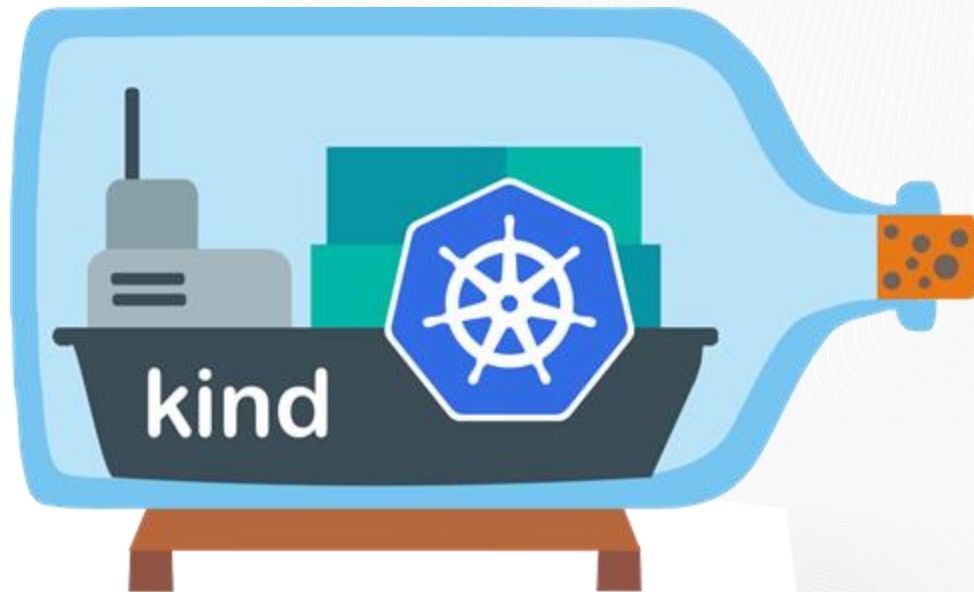
supported within one minor version (older or newer) of kube-apiserver (kube-apiserver is at 1.30 - kubectl is supported at 1.31, 1.30, and 1.29)



<https://kubernetes.io/releases/version-skew-policy/>

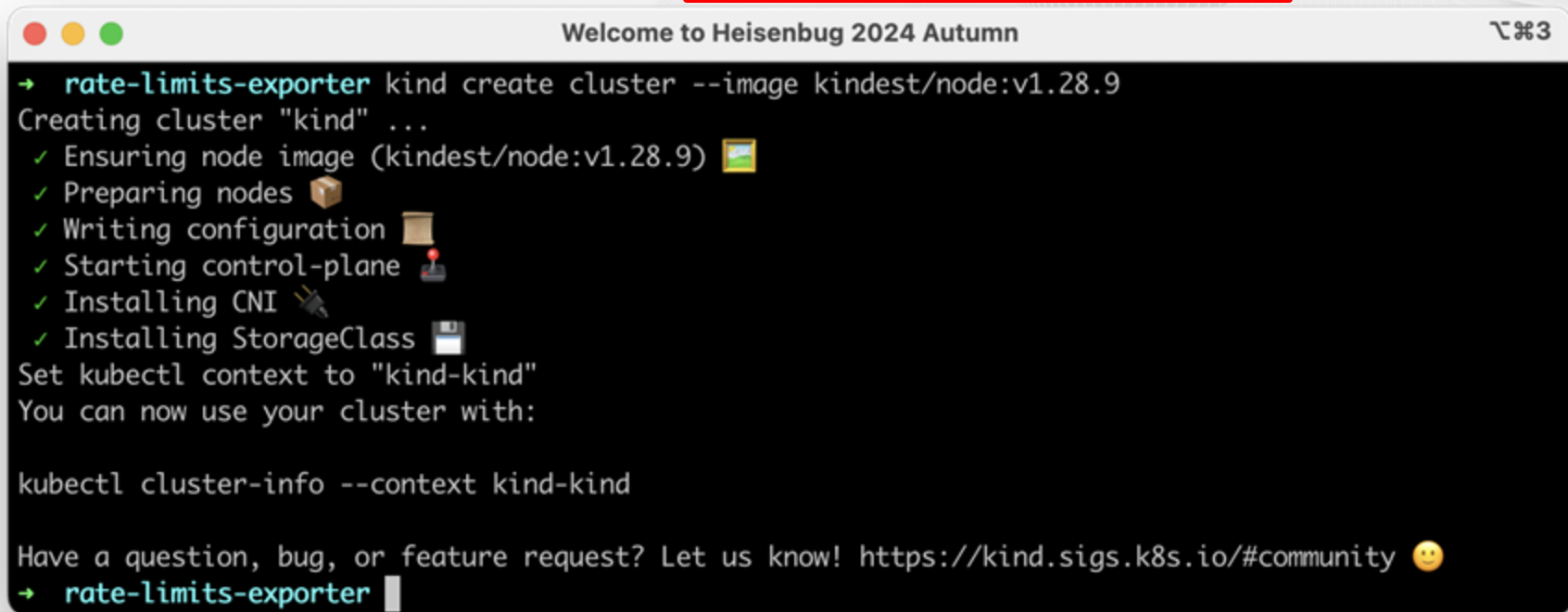
Готовим окружение

kind - Kubernetes in Docker - инструмент для запуска кластеров Kubernetes локально с использованием контейнеров Docker



ГОТОВИМ окружение

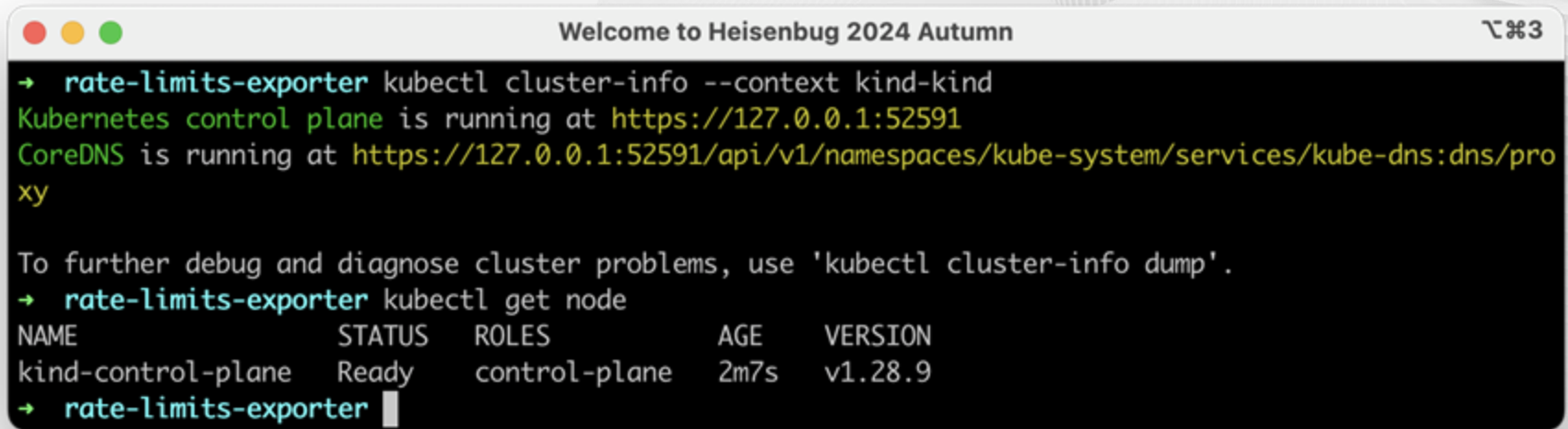
```
kind create cluster --image kindest/node:v1.28.9
```



```
Welcome to Heisenbug 2024 Autumn ⌘#3  
→ rate-limits-exporter kind create cluster --image kindest/node:v1.28.9  
Creating cluster "kind" ...  
✓ Ensuring node image (kindest/node:v1.28.9) 📄  
✓ Preparing nodes 📦  
✓ Writing configuration 📄  
✓ Starting control-plane 🚦  
✓ Installing CNI 🛠️  
✓ Installing StorageClass 📄  
Set kubectl context to "kind-kind"  
You can now use your cluster with:  
  
kubectl cluster-info --context kind-kind  
  
Have a question, bug, or feature request? Let us know! https://kind.sigs.k8s.io/#community 😊  
→ rate-limits-exporter
```

Готовим окружение

```
kubectl cluster-info --context kind-kind  
kubectl get node
```



Welcome to Heisenbug 2024 Autumn

```
→ rate-limits-exporter kubectl cluster-info --context kind-kind  
Kubernetes control plane is running at https://127.0.0.1:52591  
CoreDNS is running at https://127.0.0.1:52591/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy  
  
To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.  
→ rate-limits-exporter kubectl get node  
NAME                STATUS    ROLES    AGE   VERSION  
kind-control-plane  Ready    control-plane  2m7s  v1.28.9  
→ rate-limits-exporter
```

Окружение готово, можно продолжать!

03

pt


Как тестировать чарты

Деплоим чарт

Деплоим чарт

- Используем **helm upgrade --install** для установки чарта
- Команда **helm install** тоже подходит, но при повторном выполнении появится ошибка, поскольку такой Helm-чарт уже будет присутствовать

```
helm upgrade \  
  --install \  
1 --namespace awesome-app \  
2 --create-namespace \  
3 --values values.yaml \  
4 --wait \  
  rate-limits-exporter .
```



Деплоим чарт

```
Welcome to Heisenbug 2024 Autumn ⌘#3  
→ rate-limits-exporter helm upgrade --install --namespace awesome-app --create-namespace --values va  
lues.yaml --wait rate-limits-exporter .  
Release "rate-limits-exporter" does not exist. Installing it now.  
NAME: rate-limits-exporter  
LAST DEPLOYED: Fri Aug 23 22:15:48 2024  
NAMESPACE: awesome-app  
STATUS: deployed  
REVISION: 1  
NOTES:  
1. Get the application URL by running these commands:  
   export POD_NAME=$(kubectl get pods --namespace awesome-app -l "app.kubernetes.io/name=rate-limits-e  
xporter,app.kubernetes.io/instance=rate-limits-exporter" -o jsonpath="{.items[0].metadata.name}")  
   export CONTAINER_PORT=$(kubectl get pod --namespace awesome-app $POD_NAME -o jsonpath="{.spec.conta  
iners[0].ports[0].containerPort}")  
   echo "Visit http://127.0.0.1:8080 to use your application"  
   kubectl --namespace awesome-app port-forward $POD_NAME 8080:$CONTAINER_PORT  
→ rate-limits-exporter
```

Деплоим чарт

- Проверим какие ресурсы были созданы в кластере
- Выполним **kubectl get**, укажем пространство имен и типа ресурса

```
Welcome to Heisenbug 2024 Autumn ~#3  
→ rate-limits-exporter kubectl --namespace awesome-app get all  
NAME                READY  STATUS   RESTARTS  AGE  
pod/rate-limits-exporter-64b8f47764-wqb69  1/1    Running  0          18m  
  
NAME                TYPE          CLUSTER-IP    EXTERNAL-IP  PORT(S)    AGE  
service/rate-limits-exporter  ClusterIP    10.96.139.125 <none>       80/TCP     18m  
  
NAME                READY  UP-TO-DATE  AVAILABLE  AGE  
deployment.apps/rate-limits-exporter  1/1    1           1          18m  
  
NAME                DESIRED  CURRENT  READY  AGE  
replicaset.apps/rate-limits-exporter-64b8f47764  1       1       1     18m  
→ rate-limits-exporter
```

Деплоим чарт

- Важно помнить, что Helm хранит состояние
- Helm 3 делает это по умолчанию в секретах того неймспейса, куда деплоится чарт
- Helm 2 делал это по-другому

```
Welcome to Heisenbug 2024 Autumn ⌘⌘2
```

```
→ rate-limits-exporter kubectl --namespace awesome-app get secrets
```

NAME	TYPE	DATA	AGE
rate-limits-exporter	Opaque	0	23m
sh.helm.release.v1.rate-limits-exporter.v1	helm.sh/release.v1	1	23m
sh.helm.release.v1.rate-limits-exporter.v2	helm.sh/release.v1	1	29s

```
→ rate-limits-exporter █
```

03

pt

Как тестировать чарты

Тестируем чарт

- Helm имеет **встроенный инструмент тестирования чартов**
- **Основан на механизме хуков**, которые запускаются при наступлении определенных событий
- Нас интересует конечно же событие **test**

Annotation Value	Description
pre-install	Executes after templates are rendered, but before any resources are created in Kubernetes
post-install	Executes after all resources are loaded into Kubernetes
pre-delete	Executes on a deletion request before any resources are deleted from Kubernetes
post-delete	Executes on a deletion request after all of the release's resources have been deleted
pre-upgrade	Executes on an upgrade request after templates are rendered, but before any resources are updated
post-upgrade	Executes on an upgrade request after all resources have been upgraded
pre-rollback	Executes on a rollback request after templates are rendered, but before any resources are rolled back
post-rollback	Executes on a rollback request after all resources have been modified
test	Executes when the Helm test subcommand is invoked (view test docs)

Тестируем чарт

- Обычный манифест Kubernetes
- Должен иметь аннотацию **helm.sh/hook: test**
- **Внутри манифеста - под**, который запускает задачи
- Тест считается успешным, если **контейнер пода завершился** со статусом 0 (exit 0)

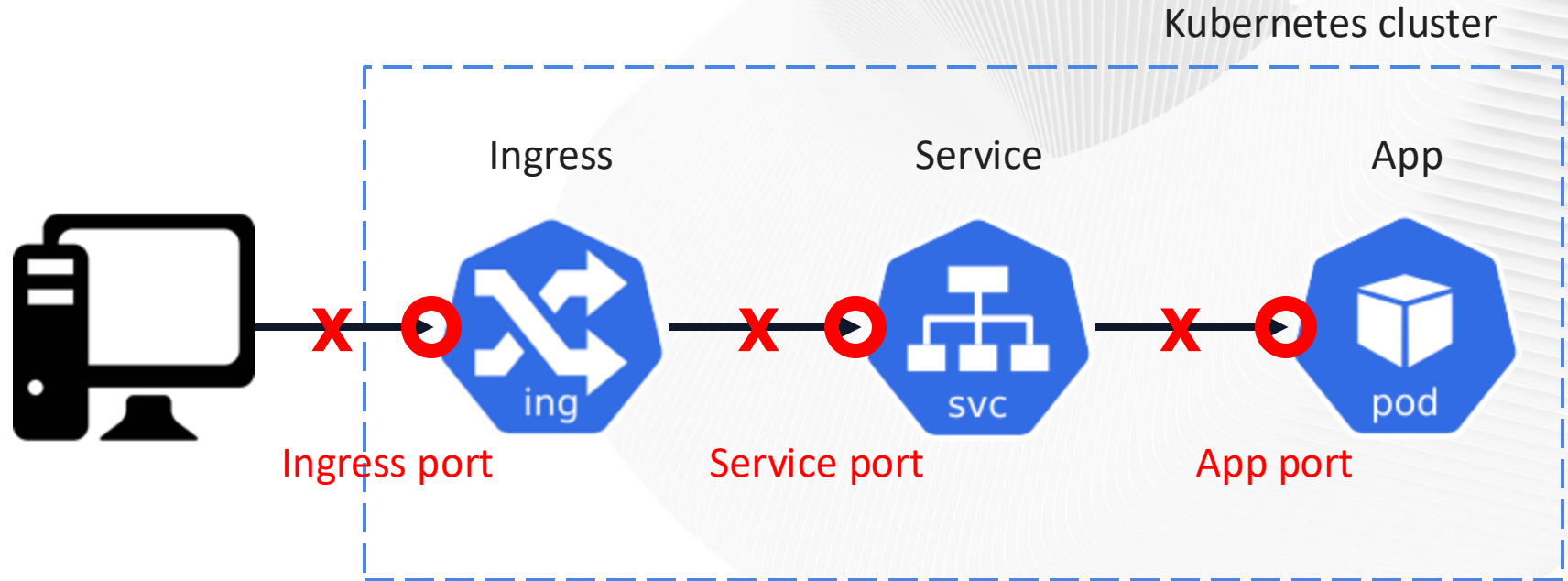
Тестируем чарт

- Можно определить **сколько угодно** тестовых манифестов
- Все тестовые манифесты должны находиться в каталогах **templates/** или **templates/tests/**
- Запускаем тесты командой **helm test**
- Helm дожидается их выполнения и выводит соответствующий статус

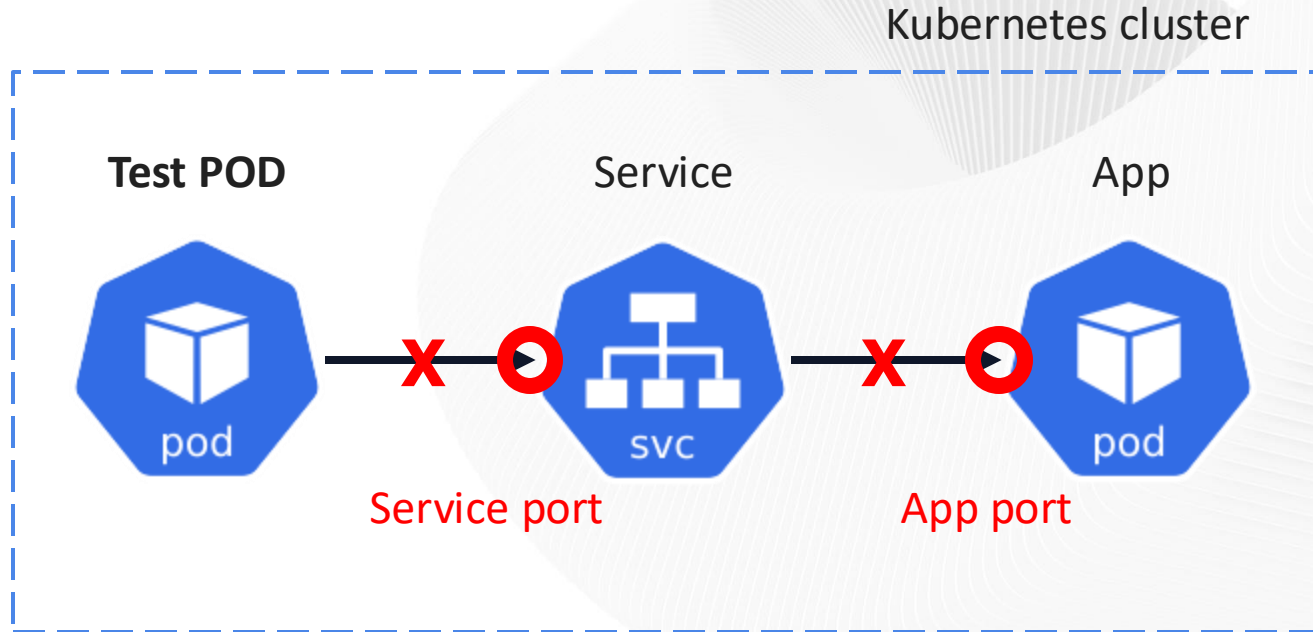
```
Welcome to Heisenbug 2024 Autumn  ⌘%2
→ rate-limits-exporter
→ rate-limits-exporter helm --namespace awesome-app
test rate-limits-exporter
NAME: rate-limits-exporter
LAST DEPLOYED: Fri Aug 23 22:39:12 2024
NAMESPACE: awesome-app
STATUS: deployed
REVISION: 2
TEST SUITE:      rate-limits-exporter-autotests
Last Started:   Sat Aug 24 22:00:34 2024
Last Completed: Sat Aug 24 22:02:22 2024
Phase:          Succeeded
TEST SUITE:      rate-limits-exporter-test-connection
Last Started:   Sat Aug 24 22:02:22 2024
Last Completed: Sat Aug 24 22:02:25 2024
Phase:          Succeeded
```

Что именно мы тестируем?

Тестируем чарт



Тестируем чарт



Как будет выглядеть тест?

Тестируем чарт

```
apiVersion: v1
```

```
kind: Pod
```

```
metadata:
```

```
  name: "{{ include \"rate-limits-exporter.fullname\" . }}-test-connection"
```

```
  labels:
```

```
    {{- include \"rate-limits-exporter.labels\" . | nindent 4 }}
```

```
  annotations:
```

```
    \"helm.sh/hook\": test
```

```
    \"kube-score/ignore\": pod-probes,container-image-pull-policy,pod-networkpolicy
```

```
spec:
```

```
  containers:
```

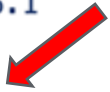
```
    - name: wget
```

```
      image: busybox:1.36.1
```

```
      workingDir: /tmp
```

```
      command: ['wget']
```

```
      args: ['{{ include \"rate-limits-exporter.fullname\" . }}:{{ .Values.service.port }}']
```



А что на счет автотестов?

Тестируем чарт

```
import os
import re
import requests
from pytest_check import check

url = os.getenv('APP_URL', 'http://0.0.0.0:8080')

def test_status():
    r = requests.get(url)
    # using soft assertion here
    check.equal(r.status_code, 200)
    check.equal(r.encoding, 'utf-8')

def test_metrics():
    r = requests.get(f'{url}/metrics')
    assert re.search(r'dockerhub_ratelimit_scrape_error\{.*\} 0|1', r.text)

def test_healthz():
    r = requests.get(f'{url}/healthz')
    assert r.text == 'Ok'
```

Где хранить?
Как запаковать?

Тестируем чарт

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: "{{ include "rate-limits-exporter.fullname" . }}-autotests"
  labels:
    {{- include "rate-limits-exporter.labels" . | nindent 4 }}
# accessing files using helm https://helm.sh/docs/chart\_template\_guide/accessing\_files/
data:
  {{- range $path, $_ := .Files.Glob "auto-tests/*" }}
  {{ base $path | indent 2 }}: |
  {{ $.Files.Get $path | indent 4 }}
  {{- end }}
```

Тестируем чарт

```
apiVersion: v1
kind: Pod
metadata:
  name: "{{ include "rate-limits-exporter.fullname" . }}-autotests"
  labels:
    {{- include "rate-limits-exporter.labels" . | nindent 4 }}
  annotations:
    "helm.sh/hook": test
    "kube-score/ignore": pod-probes,container-image-pull-policy,pod-networkpolicy,container-security-spec:
containers:
  - name: autotests
    image: python:3.9.19-slim-bookworm
    workingDir: /opt
    command: ["/bin/bash", "-c"]
    args: ['python3 -m pip install -r requirements.txt ; python3 -m pytest -v']
    env:
      - name: APP_URL
        value: "http://{{ include "rate-limits-exporter.fullname" . }}:{{ .Values.service.port }}"
```

Тестируем чарт

kubectl -namespace awesome-app logs pod/rate-limits-exporter-autotests

```
Welcome to Heisenbug 2024 Autumn ⌘#3
===== test session starts =====
platform linux -- Python 3.9.19, pytest-8.3.1, pluggy-1.5.0 -- /usr/local/bin/python3
cachedir: .pytest_cache
rootdir: /opt
plugins: check-2.3.1
collecting ... collected 3 items

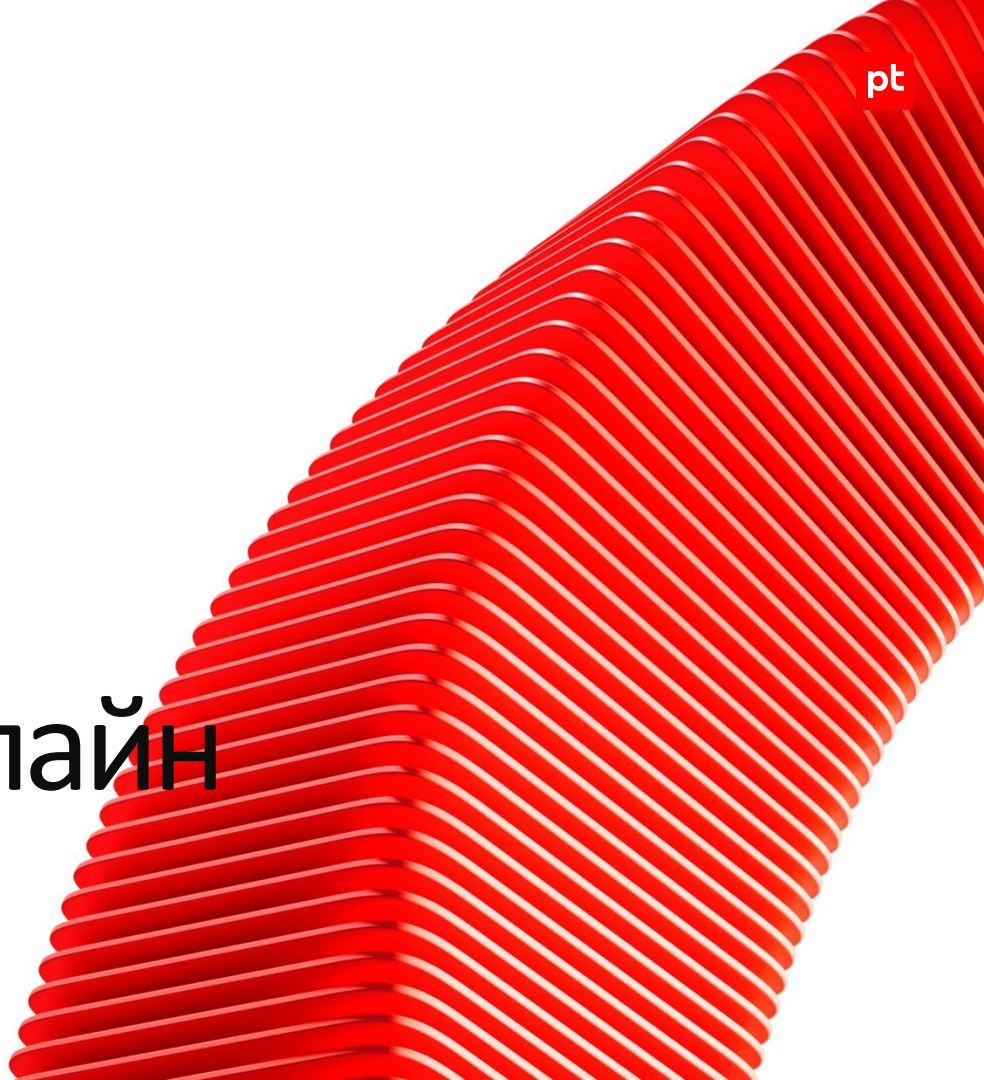
test_rate-limits-exporter.py::test_status PASSED [ 33%]
test_rate-limits-exporter.py::test_metrics PASSED [ 66%]
test_rate-limits-exporter.py::test_healthz PASSED [100%]

===== 3 passed in 4.81s =====
→ rate-limits-exporter
```

04

pt

Создаем пайплайн



Создаем пайплайн

Почему GitHub Actions?

- Доступен всем совершенно бесплатно с безлимитным временем CI/CD (для публичных проектов)
- Минимальный порог входа и большое сообщество



Создаем пайплайн

- 1 - name: 'Install Helm'
uses: azure/setup-helm@v4.2.0 ←
with:
version: \${ env.HELM_VERSION }
- 2 - name: 'Install kubectl'
uses: azure/setup-kubectl@v3 ←
with:
version: \${ env.KUBERNETES_CLUSTER_VERSION }
- 3 - name: 'Install kind and create Kubernetes cluster'
uses: helm/kind-action@v1 ←
with:
version: \${ env.KIND_VERSION }
node_image: kindest/node:\${ env. KUBERNETES_CLUSTER_VERSION }
- 4 - name: 'Install kube-score'
run: |
curl -Lo ./kube-score.tar.gz https://github.com/zegl/kube-score/
tar -zxvf kube-score.tar.gz
chmod +x kube-score
sudo mv kube-score /usr/local/bin/

Pure shell



Создаем пайплайн

```
- name: 'Run Helm lint'  
  run: |  
    helm lint chart  
  
- name: 'Run Kubeconform'  
  working-directory: chart  
  run: |  
    helm template . | kubeconform --kubernetes-version ${KUBERNETES_CLUSTER_VERSION//v}  
  
- name: 'Run kube-score checks'  
  working-directory: chart  
  run: |  
    helm template . | kube-score score -
```

Install

```
- name: 'Run Helm tests'  
  working-directory: chart  
  run: |  
    helm upgrade \  
      --install \  
      --namespace ${NAMESPACE} \  
      --create-namespace \  
      --wait \  
      ${CHART_NAME} .
```

Test

```
    helm --namespace ${NAMESPACE} \  
      test ${CHART_NAME}
```

Создаем пайплайн

test-chart
succeeded 6 minutes ago in 2m 50s

Search logs

> ✓ Set up job	2s
> ✓ Debug	0s
> ✓ Checkout server repository	0s
> ✓ Install Kubeconform	1s
> ✓ Install Helm	0s
> ✓ Install kubectl	0s
> ✓ Install kind and create Kubernetes cluster	46s
> ✓ Install kube-score	1s
> ✓ Run Helm lint	0s
> ✓ Run Kubeconform	0s
> ✓ Run kube-score checks	0s
> ✓ Run Helm tests	1m 52s

Создаем пайплайн

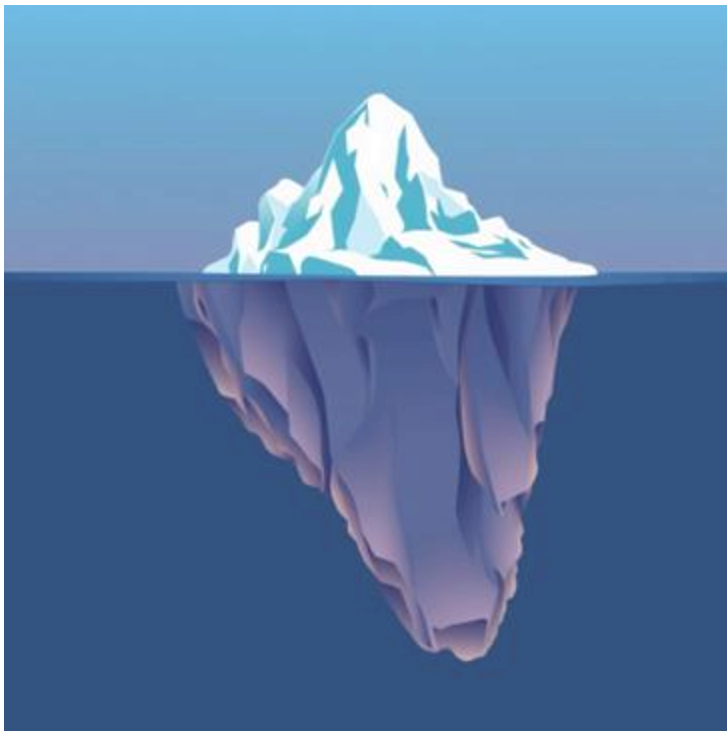
```
test-chart
failed 2 hours ago in 1m 0s

> Run Helm lint 0s
v Run kube-score checks 0s

1 ▶ Run helm template rate-limits-exporter \
13 apps/v1/Deployment release-name-rate-limits-exporter
14 path=rate-limits-exporter/templates/deployment.yaml
15 [CRITICAL] Container Resources
16   · rate-limits-exporter -> CPU limit is not set
17     Resource limits are recommended to avoid resource DDoS. Set
18     resources.limits.cpu
19   · rate-limits-exporter -> Memory limit is not set
20     Resource limits are recommended to avoid resource DDoS. Set
21     resources.limits.memory
22   · rate-limits-exporter -> CPU request is not set
23     Resource requests are recommended to make sure that the application
24     can start and run without crashing. Set resources.requests.cpu
25   · rate-limits-exporter -> Memory request is not set
26     Resource requests are recommended to make sure that the application
27     can start and run without crashing. Set resources.requests.memory
28 [CRITICAL] Pod Probes
29   · Container has the same readiness and liveness probe
30     Using the same probe for liveness and readiness is very likely
31     dangerous. Generally it's better to avoid the livenessProbe than
32     re-using the readinessProbe.
33     More information: https://github.com/zegl/kube-score/blob/master/README\_PROBES.md
```

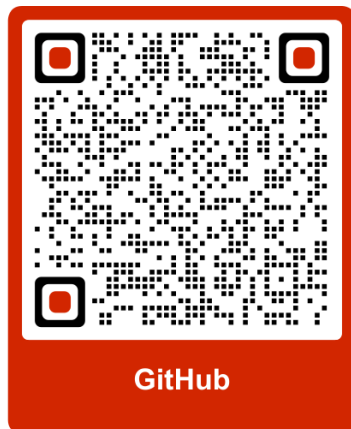
Заключение

Заключение



Application

Infrastructure



Вопросы

Спасибо за внимание!

Товарные знаки, знаки обслуживания, логотипы и иные изображения, использованные в презентации, являются собственностью их правообладателей и приведены исключительно в информационных целях.