

第 4 节

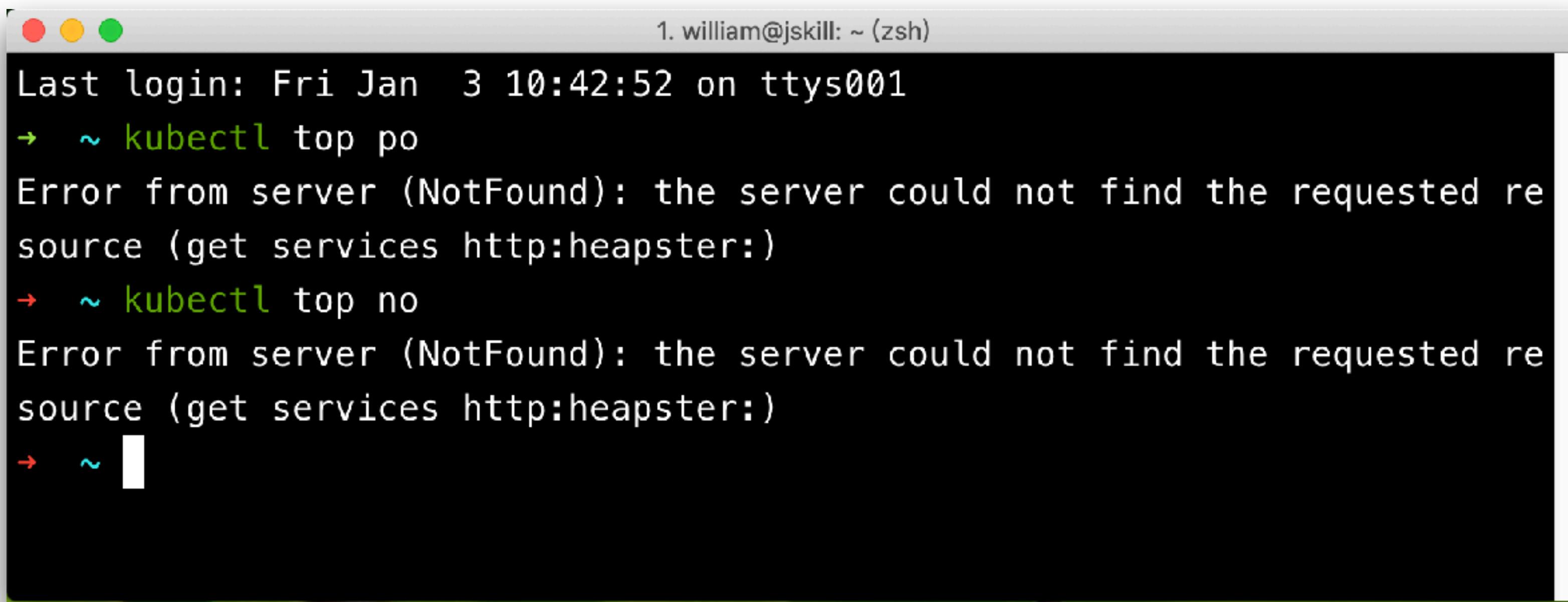
Java/Spring应用在K8s中的内存配置

本课内容

- 启用K8s的Metrics Server组件
- JVM内存结构
- Java/Spring应用在K8s中内存设置实践



默认未启用Metrics Server

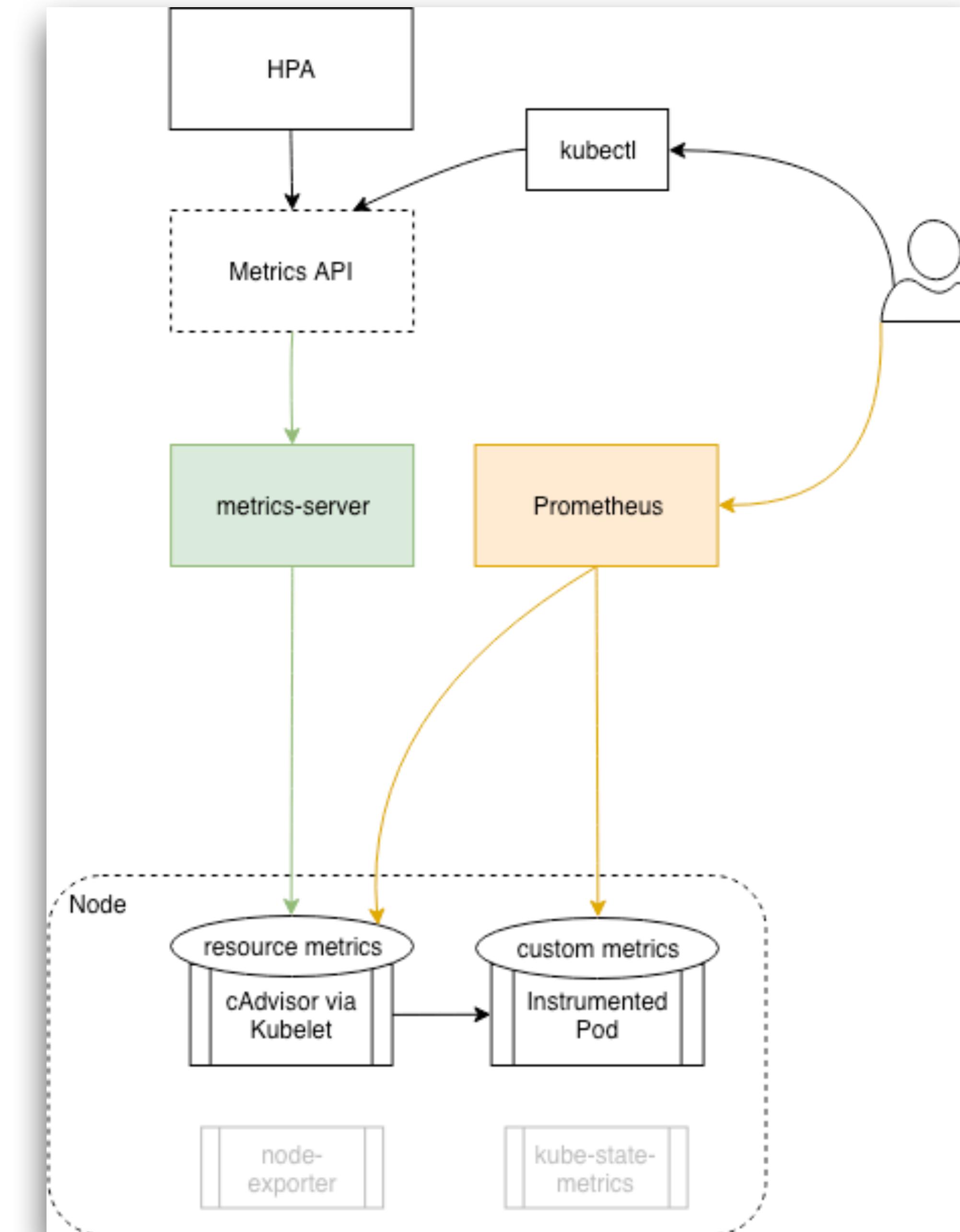


A screenshot of a macOS terminal window titled "1. william@jskill: ~ (zsh)". The window shows the following command history:

```
Last login: Fri Jan  3 10:42:52 on ttys001
→ ~ kubectl top po
Error from server (NotFound): the server could not find the requested resource (get services http:heapster:)
→ ~ kubectl top no
Error from server (NotFound): the server could not find the requested resource (get services http:heapster:)
→ ~ █
```

metrics-server

<https://github.com/kubernetes-sigs/metrics-server>



更新metrics-server-deployment.yml

```
30      containers:
31        - name: metrics-server
32          image: k8s.gcr.io/metrics-server-amd64:v0.3.6
33          args:
34            - --cert-dir=/tmp
35            - --secure-port=4443
36          ports:
37            - name: main-port
38              containerPort: 4443
39              protocol: TCP
40          securityContext:
41            readOnlyRootFilesystem: true
42            runAsNonRoot: true
43            runAsUser: 1000
44          imagePullPolicy: Always
45          command:
46            - /metrics-server
47            - --kubelet-insecure-tls
48          volumeMounts:
49            - name: tmp-dir
50              mountPath: /tmp
```

deploy/1.8+/metrics-server-deployment.yml

发布metrics-server

```
1. william@jskill: ~/temp/metrics-server (zsh)
x ..etrics-server (zsh) #1 x ~ (zsh) ● #2
→ metrics-server git:(master) ✘ pwd
/Users/william/temp/metrics-server
→ metrics-server git:(master) ✘ ls
CONTRIBUTING.md    OWNERS           SECURITY_CONTACTS  deploy      hack
LICENSE            OWNERS_ALIASES   cmd                go.mod      pkg
Makefile           README.md       code-of-conduct.md go.sum      test
→ metrics-server git:(master) ✘ kubectl apply -f deploy/1.8+
clusterrole.rbac.authorization.k8s.io/system:aggregated-metrics-reader created
clusterrolebinding.rbac.authorization.k8s.io/metrics-server:system:auth-delegator created
rolebinding.rbac.authorization.k8s.io/metrics-server-auth-reader created
apiservice.apiregistration.k8s.io/v1beta1.metrics.k8s.io created
serviceaccount/metrics-server created
deployment.apps/metrics-server created
service/metrics-server created
clusterrole.rbac.authorization.k8s.io/system:metrics-server created
clusterrolebinding.rbac.authorization.k8s.io/system:metrics-server created
→ metrics-server git:(master) ✘ kubectl get po -n kube-system
NAME                      READY   STATUS    RESTARTS   AGE
coredns-584795fc57-hhlr6   1/1     Running   425        115d
coredns-584795fc57-nv8wq   1/1     Running   424        115d
etcd-docker-desktop        1/1     Running   258        115d
kube-apiserver-docker-desktop  1/1     Running   258        115d
kube-controller-manager-docker-desktop  1/1     Running   258        115d
kube-proxy-vv5sp           1/1     Running   258        115d
kube-scheduler-docker-desktop  1/1     Running   258        115d
metrics-server-564fbf75b5-qjmrh  1/1     Running   0          65s
```

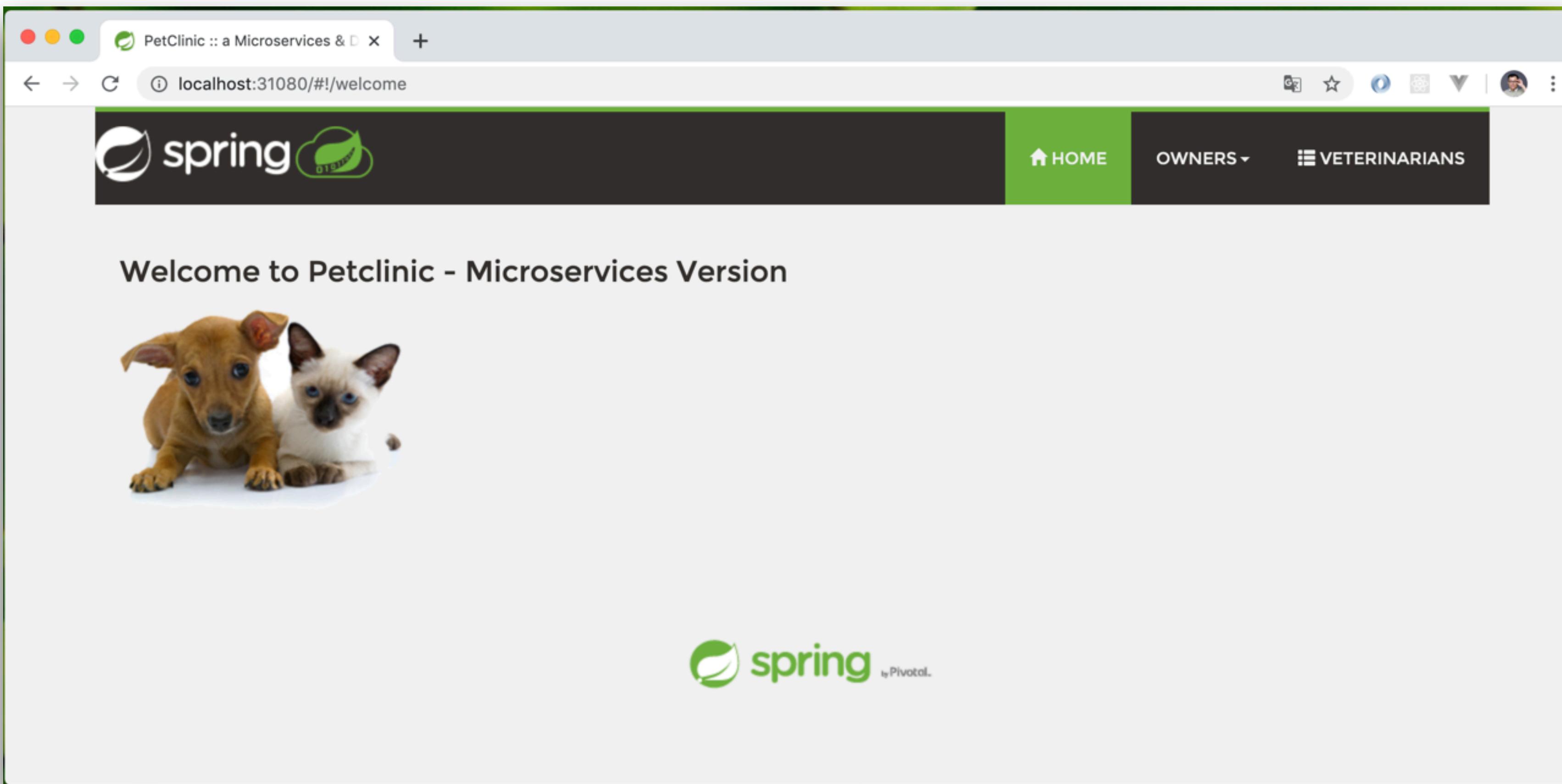
kubectl top

```
1. william@jskill: ~/temp/metrics-server (zsh)
..metrics-server (zsh) 261 ~ (zsh) 262
→ metrics-server git:(master) ✘ kubectl top no
NAME          CPU(cores)   CPU%   MEMORY(bytes)   MEMORY%
docker-desktop  265m        6%    999Mi          25%
→ metrics-server git:(master) ✘ kubectl top po
→ metrics-server git:(master) ✘ kubectl top po -n kube-system
NAME          CPU(cores)   MEMORY(bytes)
coredns-584795fc57-hhlr6      5m     8Mi
coredns-584795fc57-nv8wq      5m     8Mi
etcd-docker-desktop           32m    59Mi
kube-apiserver-docker-desktop 55m    276Mi
kube-controller-manager-docker-desktop 27m    44Mi
kube-proxy-vv5sp               2m     12Mi
kube-scheduler-docker-desktop 3m     11Mi
metrics-server-564fbf75b5-qjmrh 1m     11Mi
→ metrics-server git:(master) ✘
```

发布Petclinic微服务版 + 查看资源使用

```
1. william@jskill: ~/mydev/github/spring2go/spring-petclinic-msa/k8s/local (zsh)
→ local git:(master) ls
customers-svc.yml gateway-svc.yml    vets-svc.yml      visits-svc.yml    web-app.yml
→ local git:(master) kubectl apply -f .
deployment.apps/customers created
service/customers created
deployment.apps/gateway created
service/gateway created
deployment.apps/vets created
service/vets created
deployment.apps/visits created
service/visits created
deployment.apps/web created
service/web created
→ local git:(master) kubectl top no
NAME          CPU(cores)   CPU%   MEMORY(bytes)  MEMORY%
docker-desktop  2398m       59%   2015Mi        52%
→ local git:(master) kubectl top po
NAME          CPU(cores)   MEMORY(bytes)
customers-7b78b85c7d-w7zq6  550m     184Mi
gateway-558d7f976c-btpkm   16m      127Mi
vets-8bf75546c-829jx      593m     179Mi
visits-57f4598df9-76rdh   427m     185Mi
web-6d46d5c849-mgxct     122m     146Mi
→ local git:(master) █
```

浏览器校验成功



查看Dashboard令牌

```
1. william@jskill: ~/mydev/github/spring2go/spring-petclinic-msa/kBs/local [zsh]

→ local git:(master) kubectl get ns
NAME           STATUS   AGE
default        Active   115d
docker         Active   115d
kube-node-lease Active   115d
kube-public    Active   115d
kube-system    Active   115d
kubernetes-dashboard Active  10d

→ local git:(master) kubectl get secret -n kubernetes-dashboard
NAME              TYPE          DATA  AGE
default-token-745bg  kubernetes.io/service-account-token  3      10d
kubernetes-dashboard-certs  Opaque          0      10d
kubernetes-dashboard-csrf  Opaque          1      10d
kubernetes-dashboard-key-holder  Opaque          2      10d
kubernetes-dashboard-token-qsbwf  kubernetes.io/service-account-token  3      10d

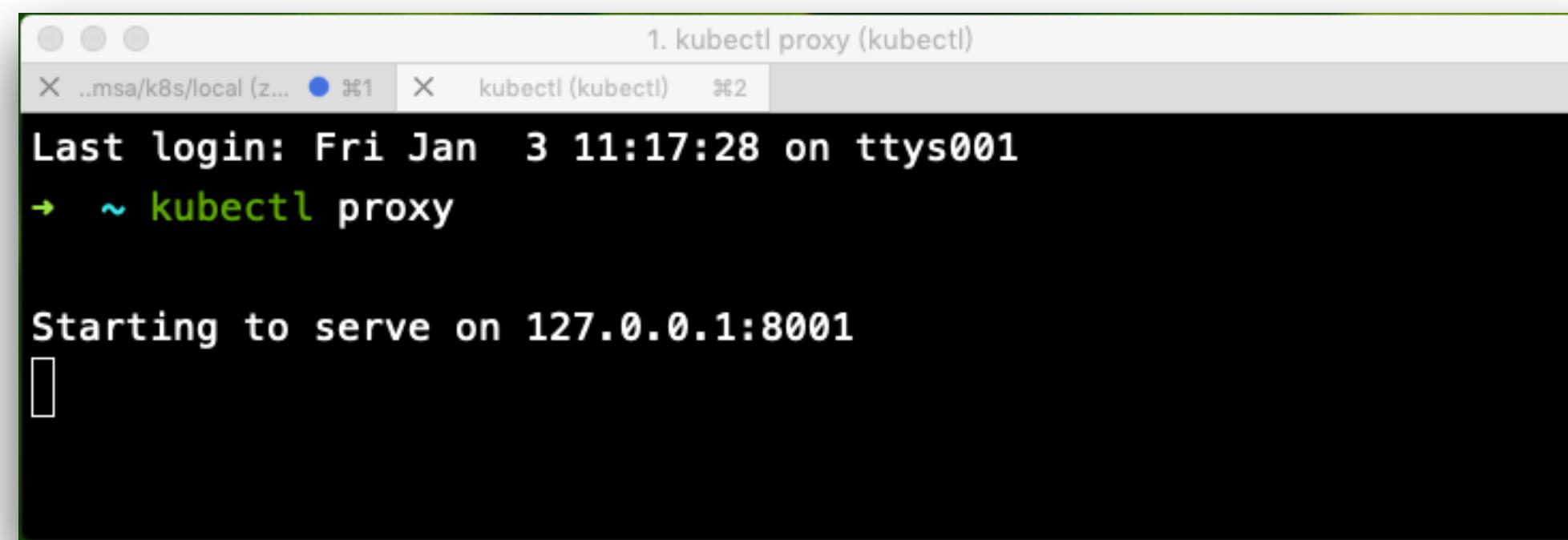
→ local git:(master) kubectl describe secret kubernetes-dashboard-token-qsbwf -n kubernetes-dashboard
Name:         kubernetes-dashboard-token-qsbwf
Namespace:    kubernetes-dashboard
Labels:       <none>
Annotations: kubernetes.io/service-account.name: kubernetes-dashboard
              kubernetes.io/service-account.uid: 4d4080b8-2557-11ea-98a9-025000000001

Type:        kubernetes.io/service-account-token

Data
=====
ca.crt:     1025 bytes
namespace:  20 bytes

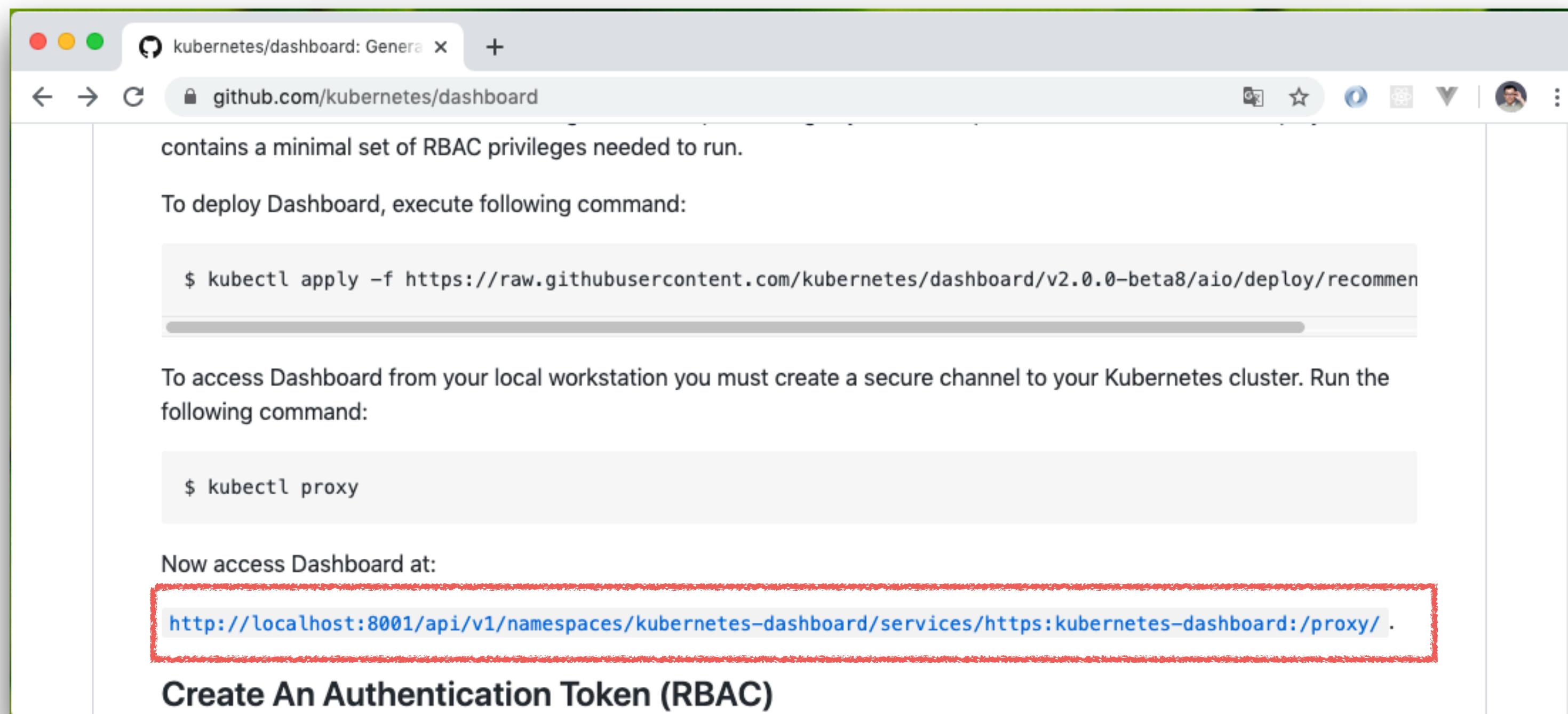
token:      eyJhbGciOiJSUzI1NiIsImtpZCI6IiJ9.eyJpc3MiOiJrdWJlc...  
[REDACTED]
```

启用Dashboard



```
Last login: Fri Jan  3 11:17:28 on ttys001
→ ~ kubectl proxy

Starting to serve on 127.0.0.1:8001
[
```



kubernetes/dashboard: Genera x +

← → C github.com/kubernetes/dashboard

contains a minimal set of RBAC privileges needed to run.

To deploy Dashboard, execute following command:

```
$ kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.0.0-beta8/aio/deploy/recommen
```

To access Dashboard from your local workstation you must create a secure channel to your Kubernetes cluster. Run the following command:

```
$ kubectl proxy
```

Now access Dashboard at:

<http://localhost:8001/api/v1/namespaces/kubernetes-dashboard/services/https:kubernetes-dashboard:/proxy/>

Create An Authentication Token (RBAC)

查看节点资源使用

Kubernetes Dashboard

localhost:8001/api/v1/namespaces/kubernetes-dashboard/services/https:kubernetes-dashboard:/proxy/#/node/docker-desktop?namespace=default

kubernetes

Search

Cluster > Nodes > docker-desktop

Cluster

- Cluster Roles
- Namespaces
- Nodes**
- Persistent Volumes
- Storage Classes

Namespace

default

CPU Usage

Memory Usage

Metadata

Name: docker-desktop Creation time: Sep 9, 2019 Age: 3.months UID: 96f97889-d2f3-11e9-bfdc-025000000001

Labels: beta.kubernetes.io/arch: amd64, beta.kubernetes.io/os: linux, kubernetes.io/arch: amd64, kubernetes.io/hostname: docker-desktop, Show all

Annotations: kubeadm.alpha.kubernetes.io/cr-socket: /var/run/dockershim.sock, node.alpha.kubernetes.io/ttl: 0, volumes.kubernetes.io/controller-managed-attach-detach: true

Resource information

Addresses: InternalIP: 192.168.65.3, Hostname: docker-desktop

System information

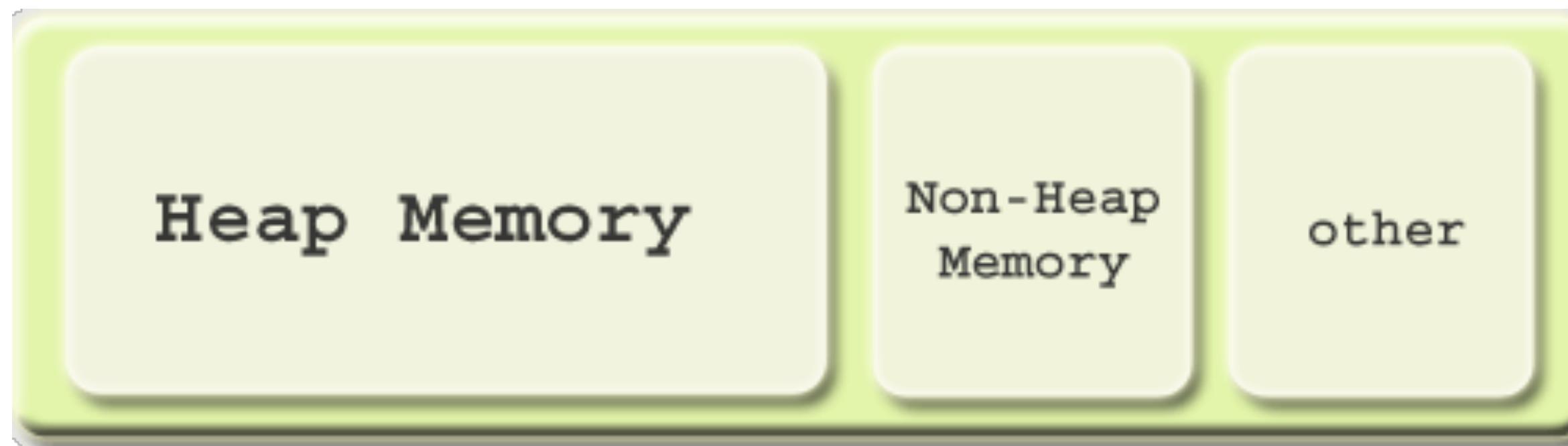
| Machine ID | System UUID | Boot ID | Kernel version |
|--------------------------------------|--------------------------------------|--------------------------------------|------------------|
| 8cc60ad8-9f9d-4ef3-9895-090c13014c81 | 68784BBF-0000-0000-AB59-2BCB8E63FADF | e5fd1cc2-d211-47a8-85ec-0cf7a30607d2 | 4.9.184-linuxkit |

查看Pod资源使用

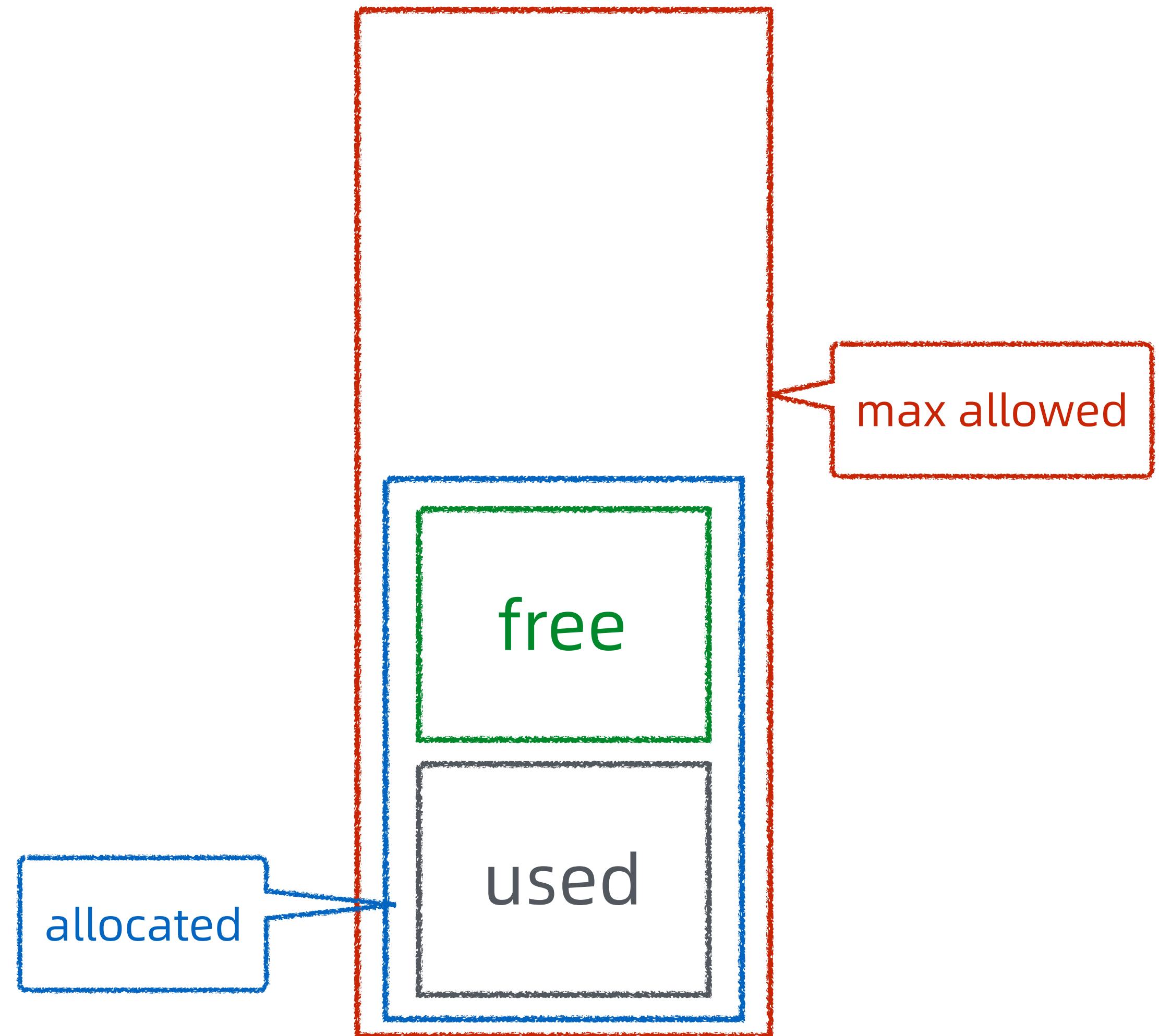
The screenshot shows the Kubernetes Dashboard interface. The left sidebar is titled "Cluster" and contains several sections: Cluster Roles, Namespaces, Nodes (which is selected), Persistent Volumes, Storage Classes, Namespace (set to default), Overview, Workloads, Cron Jobs, Daemon Sets, Deployments, Jobs, Pods (selected), Replica Sets, Replication Controllers, Stateful Sets, Discovery and Load Balancing, Ingresses, Services, Config and Storage, Config Maps, and Persistent Volume Claims. The main content area is titled "Pods" and displays a table with the following data:

| Name | Namespace | Labels | Node | Status | Restarts | CPU Usage (cores) | Memory Usage (bytes) | Age |
|--|----------------------|---|----------------|---------|----------|-------------------|----------------------|------------|
| customers-7b78b85c7d-w7zq6 | default | app: customers pod-template-hash: 7b78b85c7d | docker-desktop | Running | 0 | 3.00m | 182.87Mi | 21.minutes |
| gateway-558d7f976c-btpkm | default | app: gateway pod-template-hash: 558d7f976c | docker-desktop | Running | 0 | 3.00m | 128.95Mi | 21.minutes |
| vets-8bf75546c-829jx | default | app: vets pod-template-hash: 8bf75546c | docker-desktop | Running | 0 | 3.00m | 179.78Mi | 21.minutes |
| visits-57f4598df9-76ch | default | app: visits pod-template-hash: 57f4598df9 | docker desktop | Running | 0 | 3.00m | 184.79Mi | 21.minutes |
| web-6d46d5c849-mgxct | default | app: web pod-template-hash: 6d46d5c849 | docker desktop | Running | 0 | 3.00m | 146.18Mi | 21.minutes |
| metrics-server-564fbf75b5-qjmrh | kube-system | k8s-app: metrics-server pod-template-hash: 564fbf75b5 | docker-desktop | Running | 0 | 1.00m | 10.65Mi | 28.minutes |
| dashboard-metrics-scraper-69fcc6d9df-vtsm6 | kubernetes-dashboard | k8s-app: dashboard-metrics-scraper pod-template-hash: 69fcc6d9df | docker-desktop | Running | 39 | 11.00m | 8.85Mi | 10.days |
| kubernetes-dashboard-6d75768647-vbdfr | kubernetes-dashboard | k8s-app: kubernetes-dashboard pod-template-hash: 6d75768647 | docker-desktop | Running | 32 | 17.00m | 28.44Mi | 10.days |
| compose-api-57ff65b8c7-777m4 | docker | com.docker.deploy-nameSpace: docker com.docker.fry: compose.api Show all com.docker.default-service-type | docker-desktop | Running | 340 | 8.00m | 18.90Mi | 3.months |
| compose-5c67d745f6-6brc7 | docker | com.docker.deploy-nameSpace: docker Show all | docker-desktop | Running | 258 | 1.00m | 7.52Mi | 3.months |

JVM内存结构



Heap Usage



JVM内存监控代码

```
Runtime runtime = Runtime.getRuntime();

NumberFormat format = NumberFormat.getInstance();

long maxMemory = runtime.maxMemory();
long allocatedMemory = runtime.totalMemory();
long freeMemory = runtime.freeMemory();
long mb = 1024 * 1024;
String mega = " MB";

long physicalMemory;
try {
    physicalMemory = ((com.sun.management.OperatingSystemMXBean) ManagementFactory
        .getOperatingSystemMXBean()).getTotalPhysicalMemorySize();
} catch (Exception e) {
    physicalMemory = -1L;
}

int availableCores = Runtime.getRuntime().availableProcessors();

logger.info("===== System Info =====");
logger.info("Java version: " + System.getProperty("java.vendor") + " " + System.getProperty("java.version"));
logger.info("Operating system: " + System.getProperty("os.name") + " " + System.getProperty("os.version"));
logger.info("CPU Cores: " + availableCores);
if (physicalMemory != -1L) {
    logger.info("Physical Memory: " + format.format( number: physicalMemory / mb) + mega);
}
logger.info("===== JVM Memory Info =====");
logger.info("Max allowed memory: " + format.format( number: maxMemory / mb) + mega);
logger.info("Allocated memory:" + format.format( number: allocatedMemory / mb) + mega);
logger.info("Used memory in allocated: " + format.format( number: (allocatedMemory - freeMemory) / mb) + mega);
logger.info("Free memory in allocated: " + format.format( number: freeMemory / mb) + mega);
logger.info("Total free memory: " + format.format( number: (freeMemory + (maxMemory - allocatedMemory)) / mb) + mega);
logger.info("Heap Memory Usage: " + ManagementFactory.getMemoryMXBean().getHeapMemoryUsage());
logger.info("Non-Heap Memory Usage: " + ManagementFactory.getMemoryMXBean().getNonHeapMemoryUsage());
logger.info("=====\\n");
```

查看CustomerService Pod监控日志

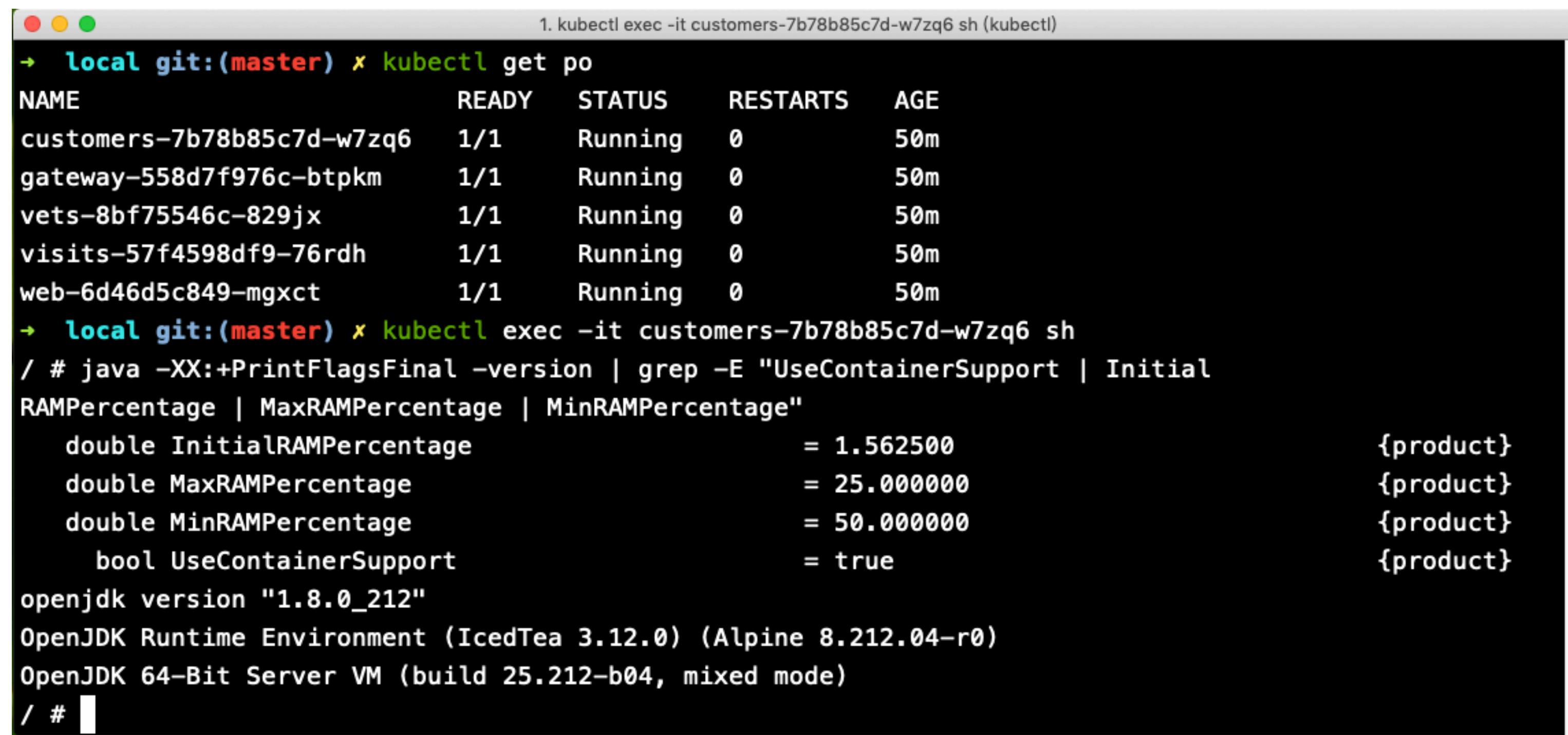
```
1. william@jskill: ~/mydev/github/spring2go/spring-petclinic-msa/k8s/local (zsh)
2020-01-03 03:15:29.916 INFO 1 --- [main] org.hibernate.dialect.Dialect      : HHH000400: Using dialect: org.hibernate.dialect.HSQLDialect
2020-01-03 03:15:31.609 INFO 1 --- [main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2020-01-03 03:15:32.319 INFO 1 --- [main] o.h.h.i.QueryTranslatorFactoryInitiator : HHH000397: Using ASTQueryTranslatorFactory
2020-01-03 03:15:33.117 INFO 1 --- [main] o.s.s.concurrent.ThreadPoolTaskExecutor : Initializing ExecutorService 'applicationTaskExecutor'
2020-01-03 03:15:33.263 WARN 1 --- [main] aWebConfiguration$JpaWebMvcConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be performed during view rendering. Explicitly configure spring.jpa.open-in-view to disable this warning
2020-01-03 03:15:34.227 INFO 1 --- [main] o.s.b.a.e.EndpointLinksResolver      : Exposing 17 endpoint(s) beneath base path '/actuator'
2020-01-03 03:15:34.438 INFO 1 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context path ''
2020-01-03 03:15:34.441 INFO 1 --- [main] c.s.s.p.c.CustomersServiceApplication : Started CustomersServiceApplication in 19.832 seconds (JVM running for 21.178)

2020-01-03 03:15:34.465 INFO 1 --- [main] c.s.s.p.c.CustomersServiceApplication
2020-01-03 03:15:34.466 INFO 1 --- [main] c.s.s.p.c.CustomersServiceApplication
2020-01-03 03:15:34.466 INFO 1 --- [main] c.s.s.p.c.CustomersServiceApplication
2020-01-03 03:15:34.466 INFO 1 --- [main] c.s.s.p.c.CustomersServiceApplication
2020-01-03 03:15:34.467 INFO 1 --- [main] c.s.s.p.c.CustomersServiceApplication
2020-01-03 03:15:34.467 INFO 1 --- [main] c.s.s.p.c.CustomersServiceApplication
=
2020-01-03 03:15:34.468 INFO 1 --- [main] c.s.s.p.c.CustomersServiceApplication
2020-01-03 03:15:34.468 INFO 1 --- [main] c.s.s.p.c.CustomersServiceApplication
2020-01-03 03:15:34.468 INFO 1 --- [main] c.s.s.p.c.CustomersServiceApplication
2020-01-03 03:15:34.469 INFO 1 --- [main] c.s.s.p.c.CustomersServiceApplication
2020-01-03 03:15:34.469 INFO 1 --- [main] c.s.s.p.c.CustomersServiceApplication
2020-01-03 03:15:34.470 INFO 1 --- [main] c.s.s.p.c.CustomersServiceApplication
committed = 70197248(68552K) max = 1001521152(978048K)
2020-01-03 03:15:34.470 INFO 1 --- [main] c.s.s.p.c.CustomersServiceApplication
committed = 91226112(89088K) max = -1(-1K)
2020-01-03 03:15:34.471 INFO 1 --- [main] c.s.s.p.c.CustomersServiceApplication

: ===== System Info =====
: Java version: IcedTea 1.8.0_212
: Operating system: Linux 4.9.184-linuxkit
: CPU Cores: 1
: Physical Memory: 3,947 MB
: ===== JVM Memory Info =====

: Max allowed memory: 955 MB
: Allocated memory: 66 MB
: Used memory in allocated: 48 MB
: Free memory in allocated: 18 MB
: Total free memory: 907 MB
: Heap Memory Usage: init = 65011712(63488K) used = 50446648(49264K) c
: Non-Heap Memory Usage: init = 2555904(2496K) used = 87240064(85195K)
: =====
```

查看容器JVM参数



The screenshot shows a terminal window with the following content:

```
1. kubectl exec -it customers-7b78b85c7d-w7zq6 sh (kubectl)
→ local git:(master) ✘ kubectl get po
NAME             READY   STATUS    RESTARTS   AGE
customers-7b78b85c7d-w7zq6   1/1     Running   0          50m
gateway-558d7f976c-btpkm    1/1     Running   0          50m
vets-8bf75546c-829jx       1/1     Running   0          50m
visits-57f4598df9-76rdh     1/1     Running   0          50m
web-6d46d5c849-mgxct      1/1     Running   0          50m
→ local git:(master) ✘ kubectl exec -it customers-7b78b85c7d-w7zq6 sh
/ # java -XX:+PrintFlagsFinal -version | grep -E "UseContainerSupport | Initial
RAMPercentage | MaxRAMPercentage | MinRAMPercentage"
    double InitialRAMPercentage                  = 1.562500                                {product}
    double MaxRAMPercentage                     = 25.000000                               {product}
    double MinRAMPercentage                    = 50.000000                               {product}
    bool UseContainerSupport                  = true                                    {product}
openjdk version "1.8.0_212"
OpenJDK Runtime Environment (IcedTea 3.12.0) (Alpine 8.212.04-r0)
OpenJDK 64-Bit Server VM (build 25.212-b04, mixed mode)
/ #
```

```
java -XX:+PrintFlagsFinal -version | grep -E "UseContainerSupport | Initial
RAMPercentage | MaxRAMPercentage | MinRAMPercentage"
```

<https://stackoverflow.com/questions/54292282/clarification-of-meaning-new-jvm-memory-parameters-initialrampercentage-and-minr>

Petclinic微服务Dockerfile更新

```
FROM openjdk:8-jre-alpine
ARG ARTIFACT_NAME
ARG EXPOSED_PORT

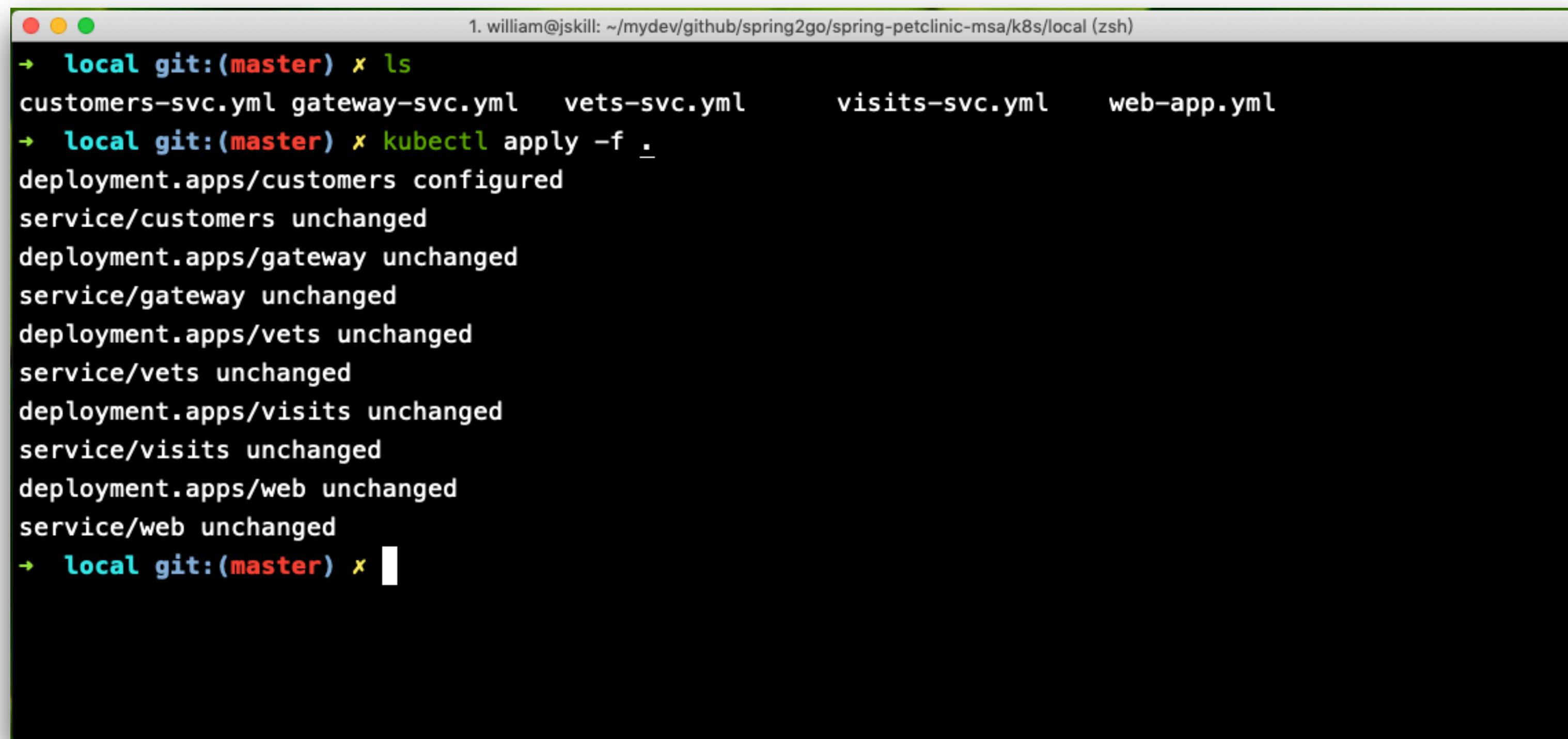
ADD ${ARTIFACT_NAME}.jar /usr/share/app.jar

EXPOSE ${EXPOSED_PORT}
# ENTRYPOINT ["java", "-XX:+UnlockExperimentalVMOptions", "-XX:+UseCGroupMemoryLimitForHeap", "-Djava.security.egd=file:/dev/.urandom","-jar","/usr/share/app.jar"]
ENTRYPOINT exec java ${JAVA_OPTS} -jar /usr/share/app.jar
```

更新customers-svc.yml

```
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   name: customers
5 spec:
6   selector:
7     matchLabels:
8       app: customers
9   replicas: 1
10  template:
11    metadata:
12      labels:
13        app: customers
14    spec:
15      containers:
16        - name: customers
17          image: spring2go/spring-petclinic-customers-service:1.0.0.RELEASE
18          resources:
19            requests:
20              memory: "128Mi"
21            limits:
22              memory: "512Mi"
23          env:
24            - name: JAVA_OPTS
25              value: "-XX:MaxRAMPercentage=80.0"
26            - name: SERVER_PORT
27              value: "8080"
```

更新发布Petclinic微服务



A terminal window titled "1. william@jskill: ~/mydev/github/spring2go/spring-petclinic-msa/k8s/local (zsh)" showing the deployment of Petclinic microservices using Kubernetes. The command run is "local git:(master) x kubectl apply -f .". The output shows the configuration of deployment.apps and service for five services: customers, gateway, vets, visits, and web.

```
1. william@jskill: ~/mydev/github/spring2go/spring-petclinic-msa/k8s/local (zsh)
→ local git:(master) x ls
customers-svc.yml gateway-svc.yml    vets-svc.yml      visits-svc.yml   web-app.yml
→ local git:(master) x kubectl apply -f .
deployment.apps/customers configured
service/customers unchanged
deployment.apps/gateway unchanged
service/gateway unchanged
deployment.apps/vets unchanged
service/vets unchanged
deployment.apps/visits unchanged
service/visits unchanged
deployment.apps/web unchanged
service/web unchanged
→ local git:(master) x
```

查看CustomerService Pod监控日志

```
1. william@jskill: ~/mydev/github/spring2go/spring-petclinic-msa/k8s/local (zsh)
2020-01-03 04:19:24.725 INFO 1 --- [           main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2020-01-03 04:19:25.127 INFO 1 --- [           main] o.h.h.i.QueryTranslatorFactoryInitiator : HHH000397: Using ASTQueryTranslatorFactory
2020-01-03 04:19:25.543 INFO 1 --- [           main] o.s.s.concurrent.ThreadPoolTaskExecutor : Initializing ExecutorService 'applicationTaskExecutor'
2020-01-03 04:19:25.737 WARN 1 --- [           main] aWebConfiguration$JpaWebMvcConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be performed during view rendering. Explicitly configure spring.jpa.open-in-view to disable this warning
2020-01-03 04:19:26.156 INFO 1 --- [           main] o.s.b.a.e.web.EndpointLinksResolver : Exposing 17 endpoint(s) beneath base path '/actuator'
2020-01-03 04:19:26.249 INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context path ''
2020-01-03 04:19:26.251 INFO 1 --- [           main] c.s.s.p.c.CustomersServiceApplication : Started CustomersServiceApplication in 7.785 seconds (JVM running for 8.398)

2020-01-03 04:19:26.265 INFO 1 --- [           main] c.s.s.p.c.CustomersServiceApplication
2020-01-03 04:19:26.266 INFO 1 --- [           main] c.s.s.p.c.CustomersServiceApplication
2020-01-03 04:19:26.267 INFO 1 --- [           main] c.s.s.p.c.CustomersServiceApplication
2020-01-03 04:19:26.267 INFO 1 --- [           main] c.s.s.p.c.CustomersServiceApplication
committed = 72511488(70812K) max = 415629312(405888K)
2020-01-03 04:19:26.267 INFO 1 --- [           main] c.s.s.p.c.CustomersServiceApplication
committed = 91422720(89280K) max = -1(-1K)
2020-01-03 04:19:26.268 INFO 1 --- [           main] c.s.s.p.c.CustomersServiceApplication

: ===== System Info =====
: Java version: IcedTea 1.8.0_212
: Operating system: Linux 4.9.184-linuxkit
: CPU Cores: 1
: Physical Memory: 3,947 MB
: ===== JVM Memory Info =====
: Max allowed memory: 396 MB
: Allocated memory: 69 MB
: Used memory in allocated: 38 MB
: Free memory in allocated: 30 MB
: Total free memory: 357 MB
: Heap Memory Usage: init = 8388608(8192K) used = 40794776(39838K) com
: Non-Heap Memory Usage: init = 2555904(2496K) used = 87523400(85472K)
: =====
```

环境清理

```
1. william@jskill: ~/mydev/github/spring2go/spring-petclinic-msa/k8s/local (zsh)
→ local git:(master) ✘ pwd
/Users/william/mydev/github/spring2go/spring-petclinic-msa/k8s/local
→ local git:(master) ✘ ls
customers-svc.yml gateway-svc.yml    vets-svc.yml      visits-svc.yml    web-app.yml
→ local git:(master) ✘ kubectl delete -f .
deployment.apps "customers" deleted
service "customers" deleted
deployment.apps "gateway" deleted
service "gateway" deleted
deployment.apps "vets" deleted
service "vets" deleted
deployment.apps "visits" deleted
service "visits" deleted
deployment.apps "web" deleted
service "web" deleted
→ local git:(master) ✘ kubectl get all
NAME                  TYPE        CLUSTER-IP   EXTERNAL-IP  PORT(S)   AGE
service/kubernetes   ClusterIP   10.96.0.1   <none>      443/TCP   2d23h
→ local git:(master) ✘
```

本课小结



- 查看node/pod资源使用
 - kubectl top(需启用Metrics Server)
 - K8s Dashboard
- JVM内存结构
 - Heap + Non Heap + 其它
 - 从初使分配开始，随实际使用增加向OS申请，直到允许的最大请求
- JVM设置建议
 - 升级到最新JDK, Java 8u191和Java 10+支持UseContainerSupport
 - 设置K8s内存Limit + MaxRAMPercentage JVM启动参数
 - 实际设置要监控调优