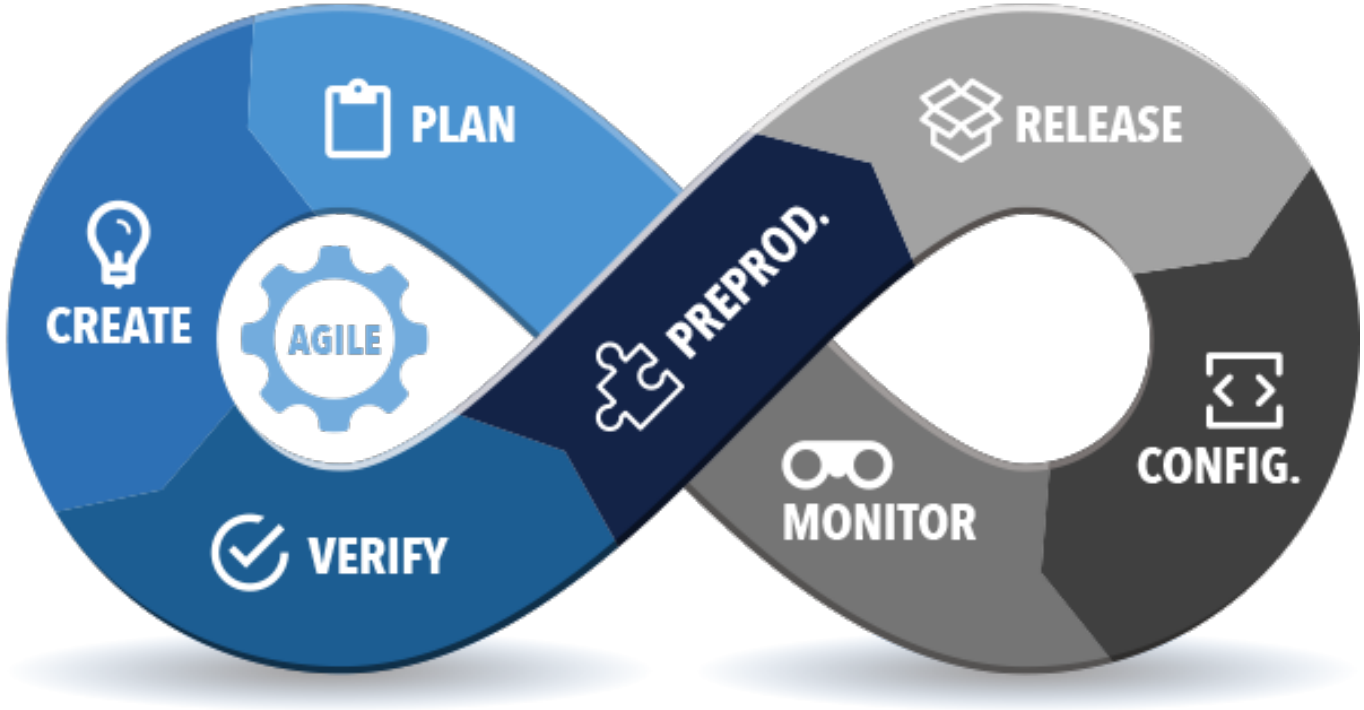
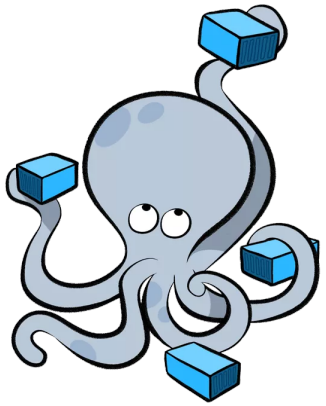



***4. Kubernetes***

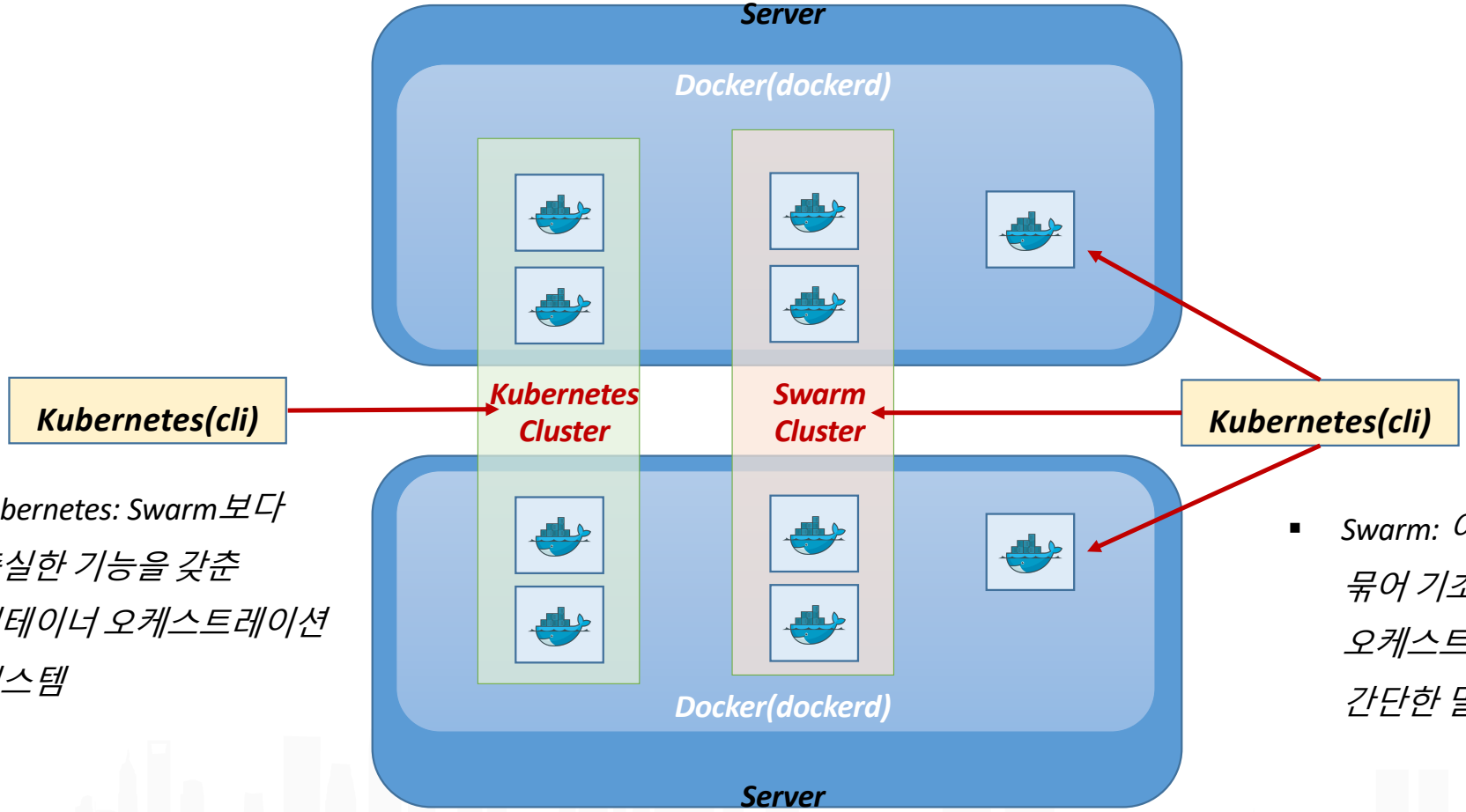
# Kubernetes

- ***Docker Container 운영을 자동화하기 위한 컨테이너 오케스트레이션 툴***
  - 컨테이너 배포 및 배치 전략
  - Scale in/Scale out
  - Service discovery
  - 기타 운용
- ***구글의 Borg 프로젝트에서 시작***
- ***2017년 Docker에 정식으로 통합된 사실상 표준***
  - DockerCon EU 2017



kubernetes

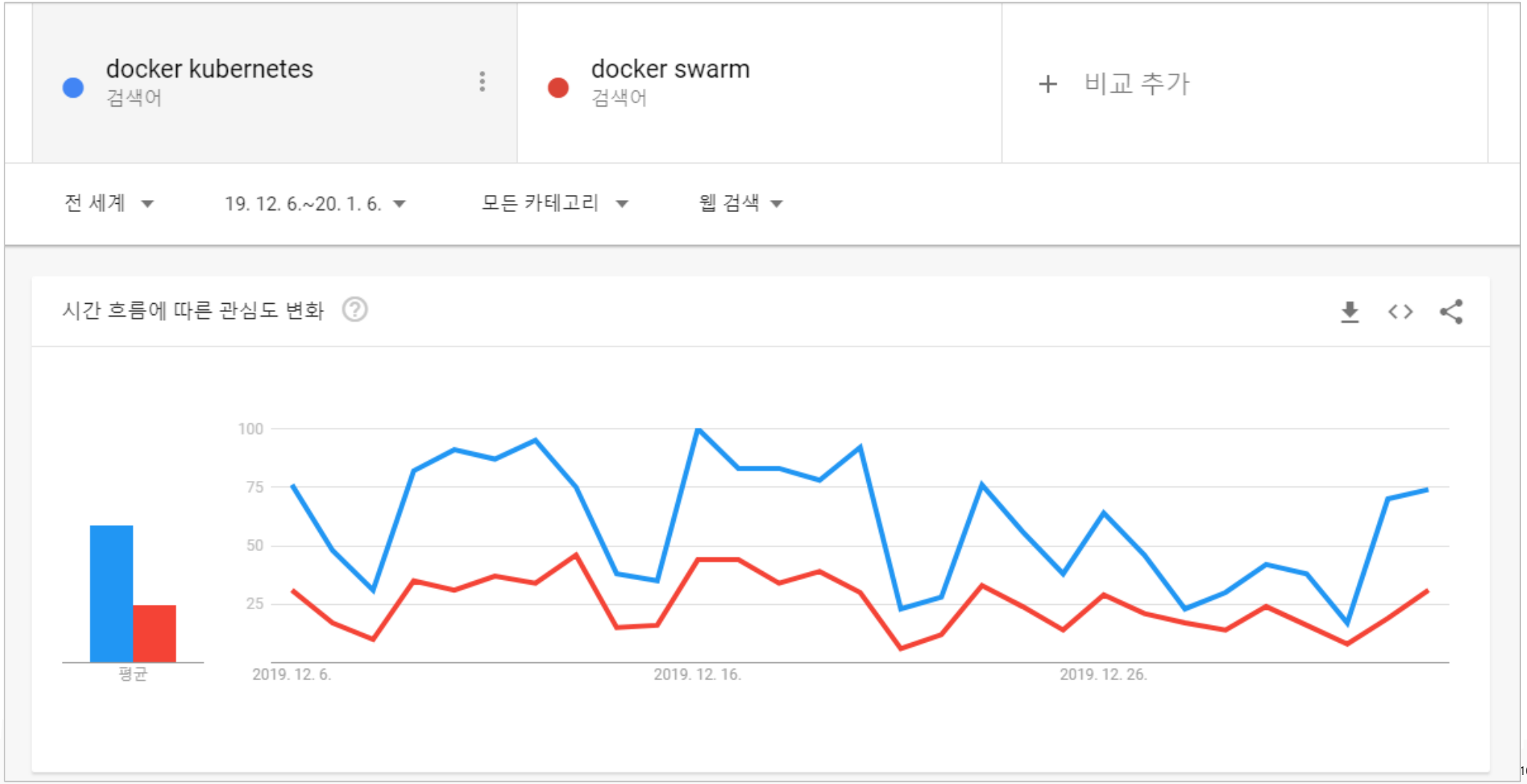
# Kubernetes




- Kubernetes: Swarm보다 충실한 기능을 갖춘 컨테이너 오케스트레이션 시스템

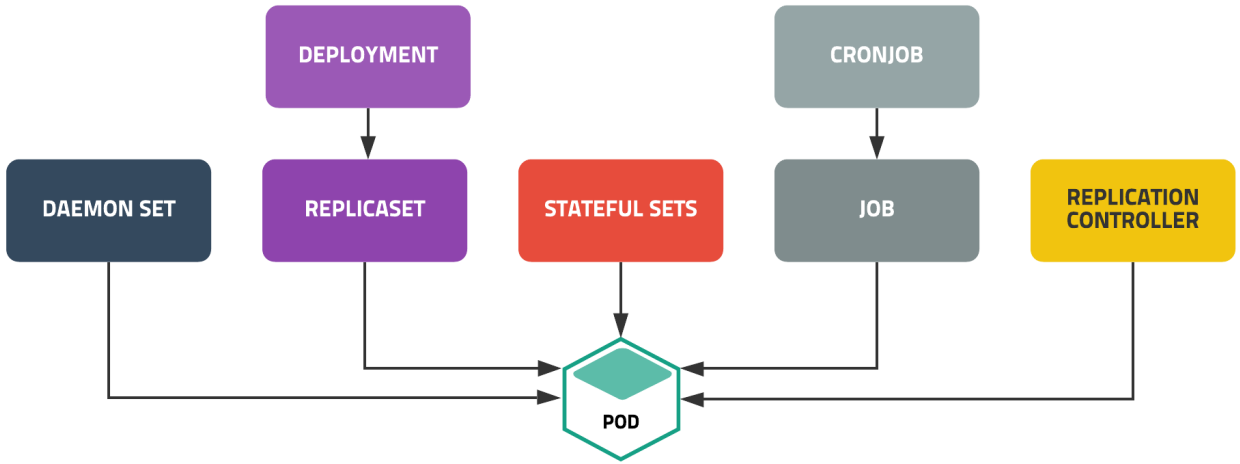
- Swarm: 여러 대의 호스트를 묶어 기초적인 컨테이너 오케스트레이션 기능 제공, 간단한 멀티 컨테이너 구축

# Kubernetes


docker kubernetes
검색어
docker swarm
검색어
+ 비교 추가
전 세계
19. 12. 6.~20. 1. 6.
모든 카테고리
웹 검색
시간 흐름에 따른 관심도 변화
100
75
50
25
평균
2019. 12. 6.
2019. 12. 16.
2019. 12. 26.

# Kubernetes

- ***Kubernetes 설치***
  - Minikube
    - 이전에는 로컬 환경에서 Kubernetes를 구축하기 위해 사용
  - 2018년부터 안정버전에서도 설치 가능
  - 윈도우 설치)
    - 바탕화면 트레이 > 도커 아이콘 (오른쪽 클릭) > Setting 메뉴 > Kebernetes 탭
    - Enable Kubernetes 선택 → Install




*Kubernetes 배포 방식*

# Kubernetes

- ***kubectl 설치***
  - Kubernetes를 다루기 위한 Command Line Interface
  - 윈도우 설치)
    - https://storage.googleapis.com/kubernetes-release/release/v1.17.0/bin/windows/amd64/kubectl.exe
    - 다운로드, PATH 추가
  - $ kubectl version

```
kubectl version
Client Version: version.Info{Major:"1", Minor:"17", GitVersion:"v1.17.0", GitCommit:"70132b0f130acc0bed193d9ba59dd186f0e634cf",
GitTreeState:"clean", BuildDate:"2019-12-07T21:20:10Z", GoVersion:"go1.13.4", Compiler:"gc", Platform:"darwin/amd64"}
Server Version: version.Info{Major:"1", Minor:"14", GitVersion:"v1.14.6", GitCommit:"96fac5cd13a5dc064f7d9f4f23030a6aeface6cc",
GitTreeState:"clean", BuildDate:"2019-08-19T11:05:16Z", GoVersion:"go1.12.9", Compiler:"gc", Platform:"linux/amd64"}
```

# Kubernetes

- ***Dashboard 설치***
  - Kubernetes에 배포된 컨테이너 등에 대한 정보를 보여주는 관리 도구
  - $ kubectl apply -f

    https://raw.githubusercontent.com/kubernetes/dashboard/v1.8.3/src/deploy/recommended/kubernetes-dashboard.yaml
  - $ kubectl get pod --namespace=kube-system -l k8s-app=kubernetes-dashboard

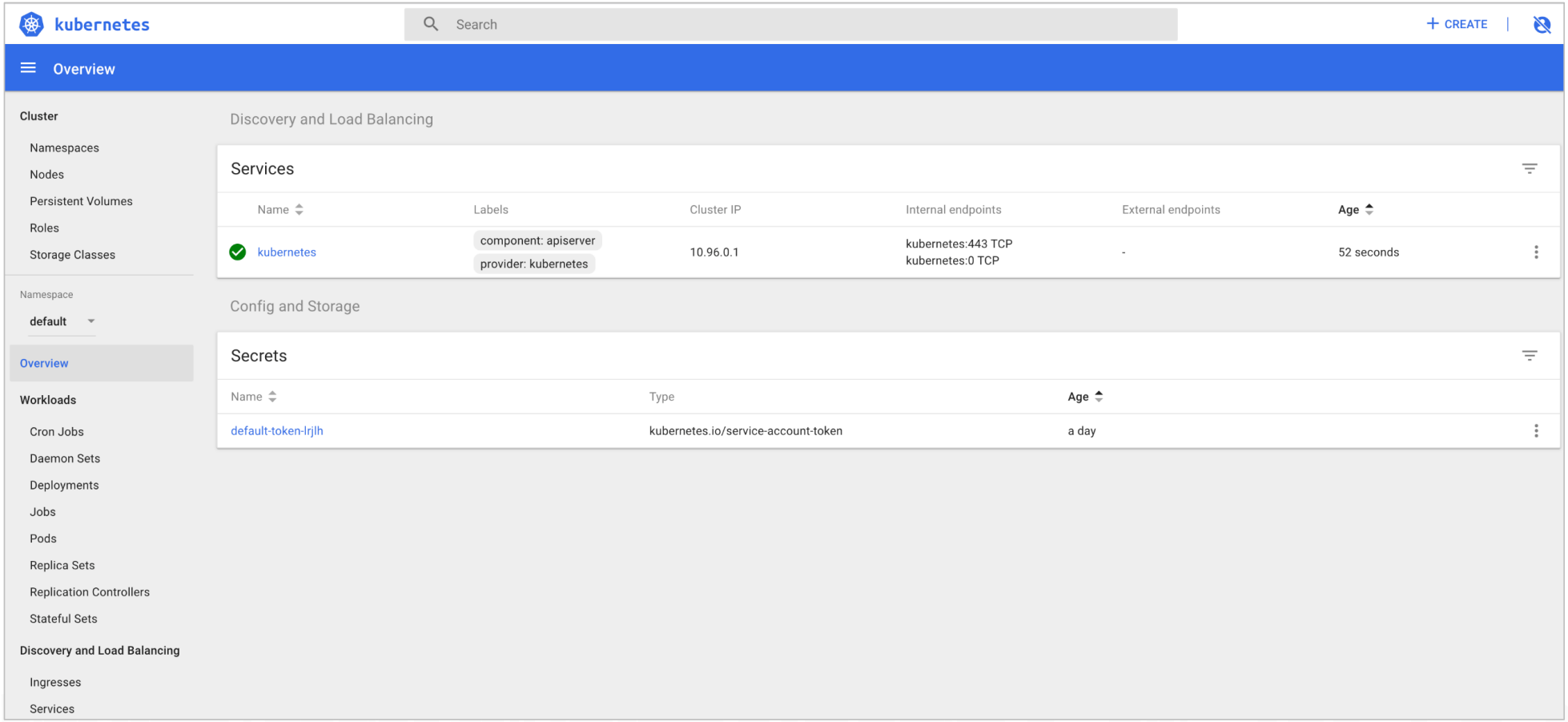
```
NAME                                    READY   STATUS    RESTARTS   AGE
kubernetes-dashboard-6fd7f9c494-2bkjp   1/1     Running   0          30h
```

  - $ kubectl proxy (웹 브라우저로 대시보드를 사용할 수 있도록 프록시 서버 설정)

http://localhost:8001/api/v1/namespaces/kube-system/services/https:kubernetes-dashboard:/proxy/

# Kubernetes

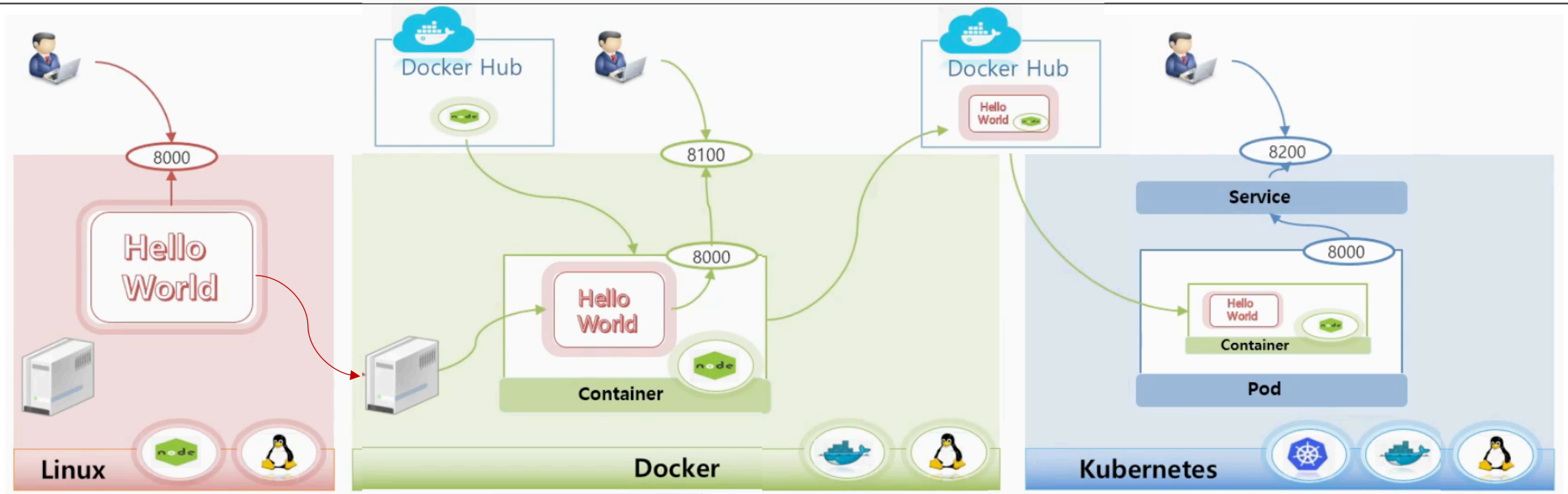
## Dashboard 설치



# Kubernetes 주요 개념

| Resource or Object | 용도 |
|---|---|
| Node | 컨테이너가 배치되는 서버 |
| Namespace | 쿠버네티스 클러스터 안의 가상 클러스터 |
| Pod | 컨테이너의 집합 중 가장 작은 단위, 컨테이너의 실행 방법 정의 |
| Replica Set | 같은 스펙을 갖는 파드를 여러 개 생성하고 관리하는 역할 |
| Deployment | 레플리카 세트의 리비전을 관리 |
| Service | 파드의 집합에 접근하기 위한 경로를 정의 |
| Ingress | 서비스를 쿠버네티스 클러스터 외부로 노출 |
| ConfigMap | 설정 정보를 정의하고 파드에 전달 |
| Persistent Volume | 파드가 사용할 스토리지의 크기 및 종류를 정의 |
| Persistent Volume Claim | 퍼시스턴트 볼륨을 동적으로 확보 |

# Kubernetes Demo




**Dockerfile**

```
FROM node:slim
EXPOSE 8000
COPY hello.js .
CMD node hello.js
```

**Pod**

```
apiVersion: v1
kind: Pod
metadata:
  name: hello-pod
  labels:
    app: hello
spec:
  containers:
  - name: hello-container
    image: tmkube/hello
    ports:
    - containerPort: 8000
```

**Service**

```
apiVersion: v1
kind: Service
metadata:
  name: hello-svc
spec:
  selector:
    app: hello
  ports:
    - port: 8200
      targetPort: 8000
  externalIPs:
  - 192.168.0.30
```

# Kubernetes Demo

- ***Linux version***

$ vi hello.js

```
1 var http = require('http');
2 var content = function(req, resp) {
3  resp.end("Hello Kubernetes!" + "\n");
4  resp.writeHead(200);
5 }
6 var w = http.createServer(content);
7 w.listen(8000);
```

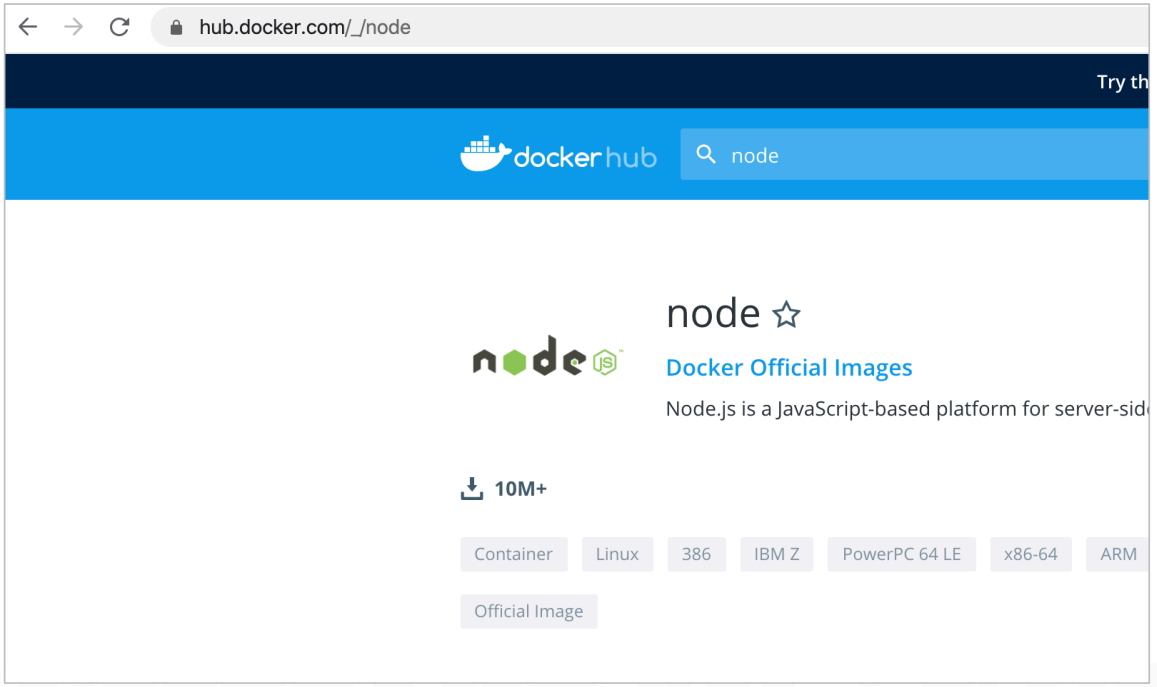
$ node hello.js

# Kubernetes Demo

- ***Docker version***

$ vi Dockerfile.js

```
1 FROM node:slim
2
3 EXPOSE 8000
4
5 COPY hello.js .
6
7 CMD node hello.js
```



# Kubernetes Demo

- ***Docker version***

$ docker build –t edowon0623/hello .

$ docker images

```
docker build -t edowon0623/hello .
Sending build context to Docker daemon  3.072kB
Step 1/4 : FROM node:slim
slim: Pulling from library/node
804555ee0376: Already exists
2706bdf80250: Pull complete
3a1861ab5a61: Pull complete
2a2939395b29: Pull complete
738ba111f3fd: Pull complete
Digest: sha256:65600ac92bd94a647ddfebea5ef141b44ef02ffed9e431389000b245c3b84330
Status: Downloaded newer image for node:slim
 ---> b515d98fd8cd
Step 2/4 : EXPOSE 8000
 ---> Running in 696e95575d7f
Removing intermediate container 696e95575d7f
 ---> c19d3c1cc743
Step 3/4 : COPY hello.js .
 ---> 786edbe52048
Step 4/4 : CMD node hello.js
 ---> Running in efd1148f0070
Removing intermediate container efd1148f0070
 ---> 2eb447ee279a
Successfully built 2eb447ee279a
Successfully tagged edowon0623/hello:latest
```

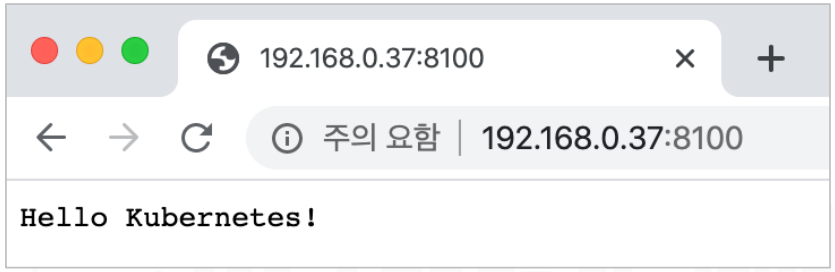
# Kubernetes Demo

- ***Docker version***

$ docker run -d -p 8100:8000 edowon0623/hello

```
 docker run -d -p 8100:8000 edowon0623/hello:latest
46cf06dc11297c35eae06c88202e8255b7ba93798c7e53be00594fcff20e344c
 dowon@DOWON-MacBook  ~/Desktop/Work/docker/kubernetes/getstarted
```

$ docker exec -it edowon0623/hello bash




```
 docker exec -it 46cf06dc1129 bash
root@46cf06dc1129:/# ls -al
total 76
drwxr-xr-x   1 root root 4096 Jan  5 23:04 .
drwxr-xr-x   1 root root 4096 Jan  5 23:04 ..
-rwxr-xr-x   1 root root    0 Jan  5 23:04 .dockerenv
drwxr-xr-x   2 root root 4096 Dec 24 00:00 bin
drwxr-xr-x   2 root root 4096 Sep  8 10:51 boot
drwxr-xr-x   5 root root  340 Jan  5 23:04 dev
drwxr-xr-x   1 root root 4096 Jan  5 23:04 etc
-rw-r--r--   1 root root  179 Jan  5 22:54 hello.js
drwxr-xr-x   1 root root 4096 Dec 28 23:33 home
drwxr-xr-x   1 root root 4096 Dec 24 00:00 lib
drwxr-xr-x   2 root root 4096 Dec 24 00:00 lib64
drwxr-xr-x   2 root root 4096 Dec 24 00:00 media
drwxr-xr-x   2 root root 4096 Dec 24 00:00 mnt
drwxr-xr-x   1 root root 4096 Dec 28 23:35 opt
dr-xr-xr-x 253 root root    0 Jan  5 23:04 proc
drwx------   1 root root 4096 Dec 28 23:34 root
drwxr-xr-x   3 root root 4096 Dec 24 00:00 run
drwxr-xr-x   2 root root 4096 Dec 24 00:00 sbin
drwxr-xr-x   2 root root 4096 Dec 24 00:00 srv
dr-xr-xr-x  13 root root    0 Dec 30 16:41 sys
drwxrwxrwt   1 root root 4096 Dec 28 23:34 tmp
drwxr-xr-x   1 root root 4096 Dec 24 00:00 usr
drwxr-xr-x   1 root root 4096 Dec 24 00:00 var
```

# Kubernetes Demo

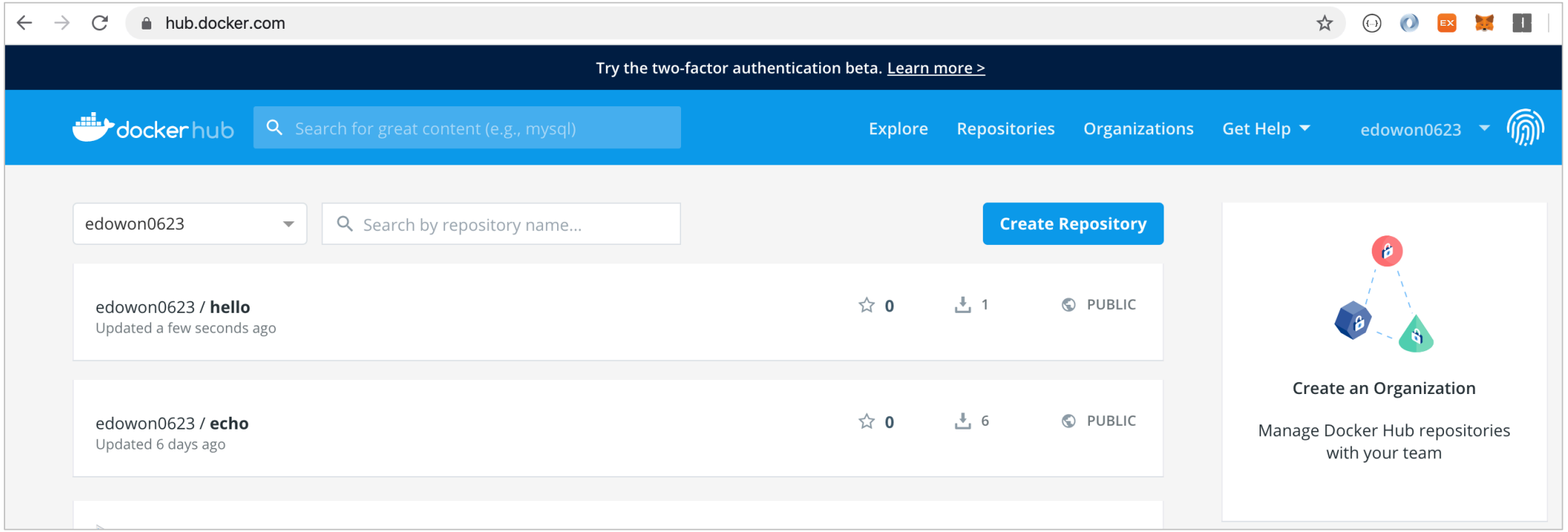
- ***Kubernetes version***

```
docker login
Authenticating with existing credentials...
Login Succeeded
 dowon@DOWON-MacBook ~/Desktop/Work/docker/kubernetes/getstarted
```

```
docker push edowon0623/hello:latest
The push refers to repository [docker.io/edowon0623/hello]
ce63c2b86199: Pushed
0bd549a35bb4: Mounted from library/node
a954d4bc6a8d: Mounted from library/node
a2357d53df3b: Mounted from library/node
62dac45972d5: Mounted from library/node
814c70fdae62: Mounted from library/node
latest: digest: sha256:c46636d9b118c67c2248b59bb6df93f16d12004be86f41db44cdbca3bc521396 size: 1574
```

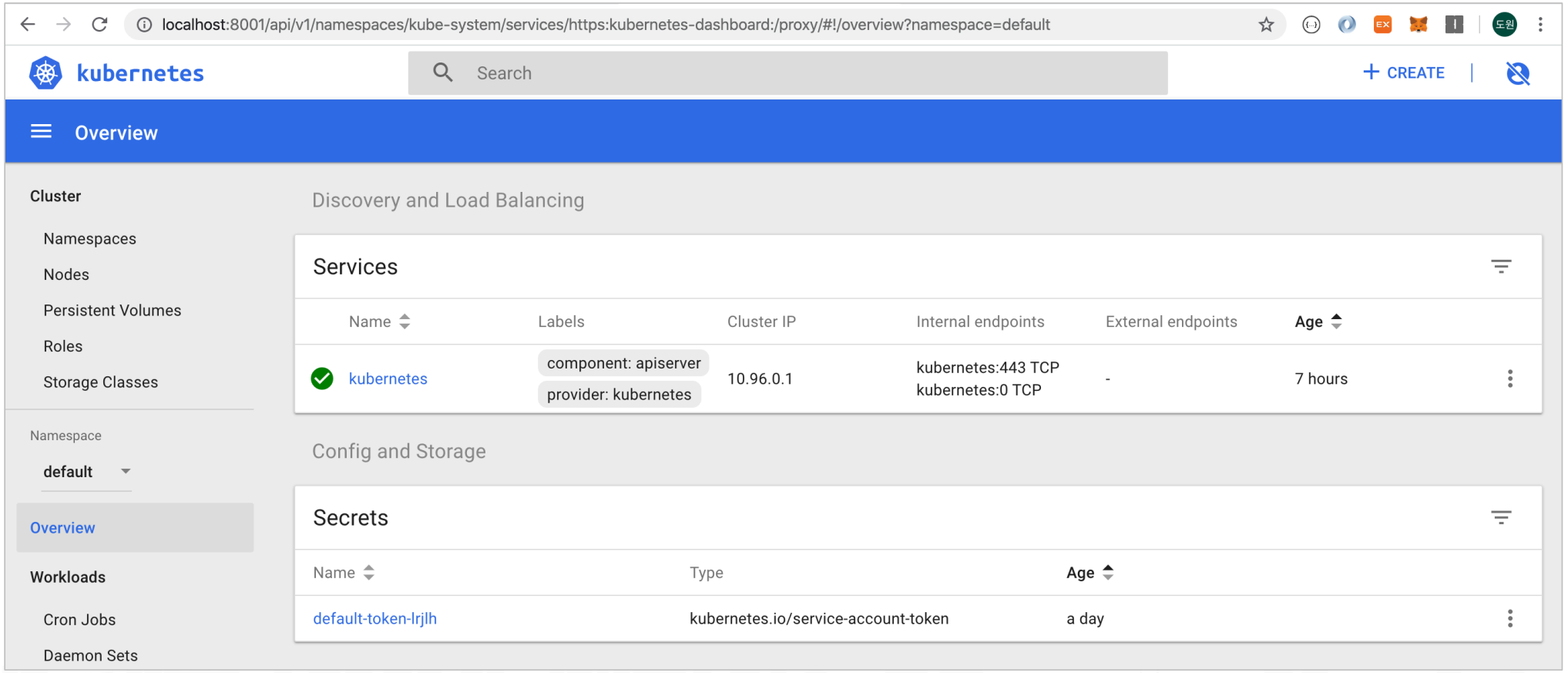
# Kubernetes Demo

- ***Kubernetes version***



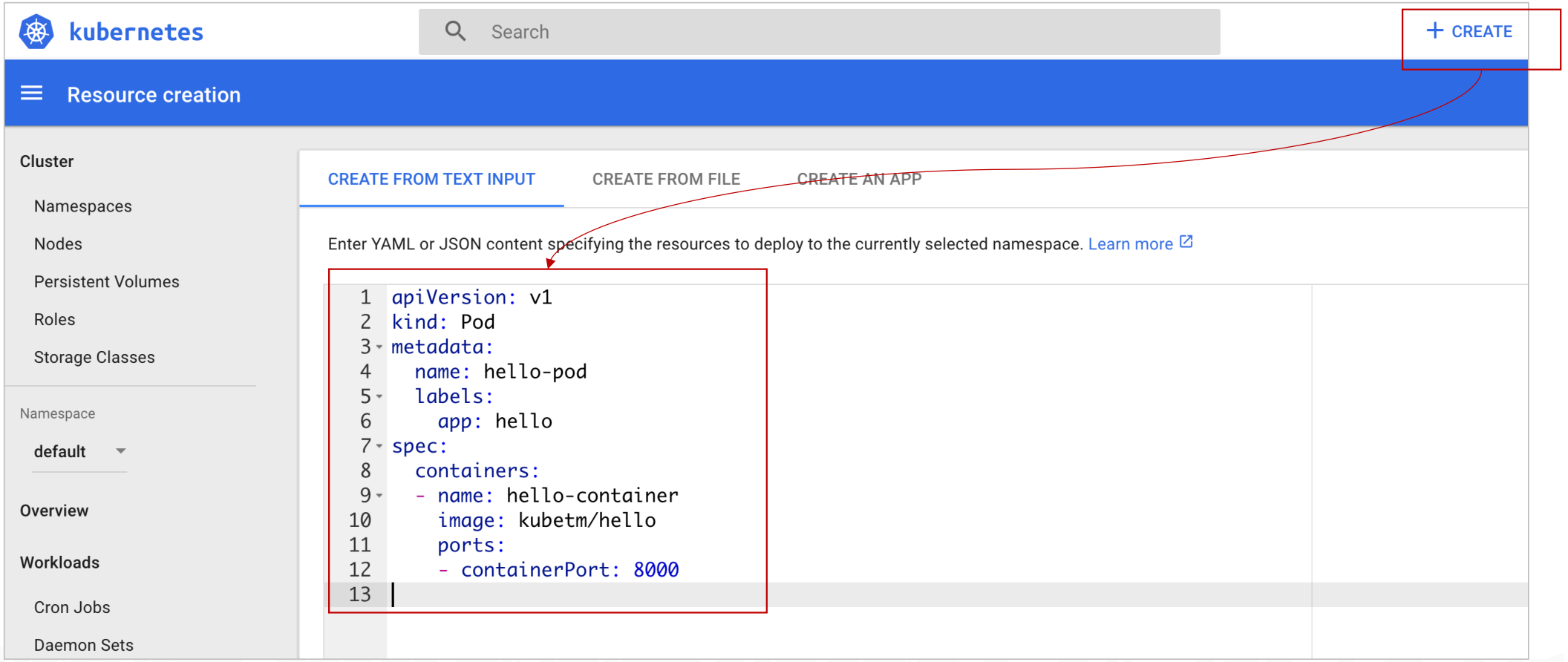
# Kubernetes Demo

- ***Kubernetes version***



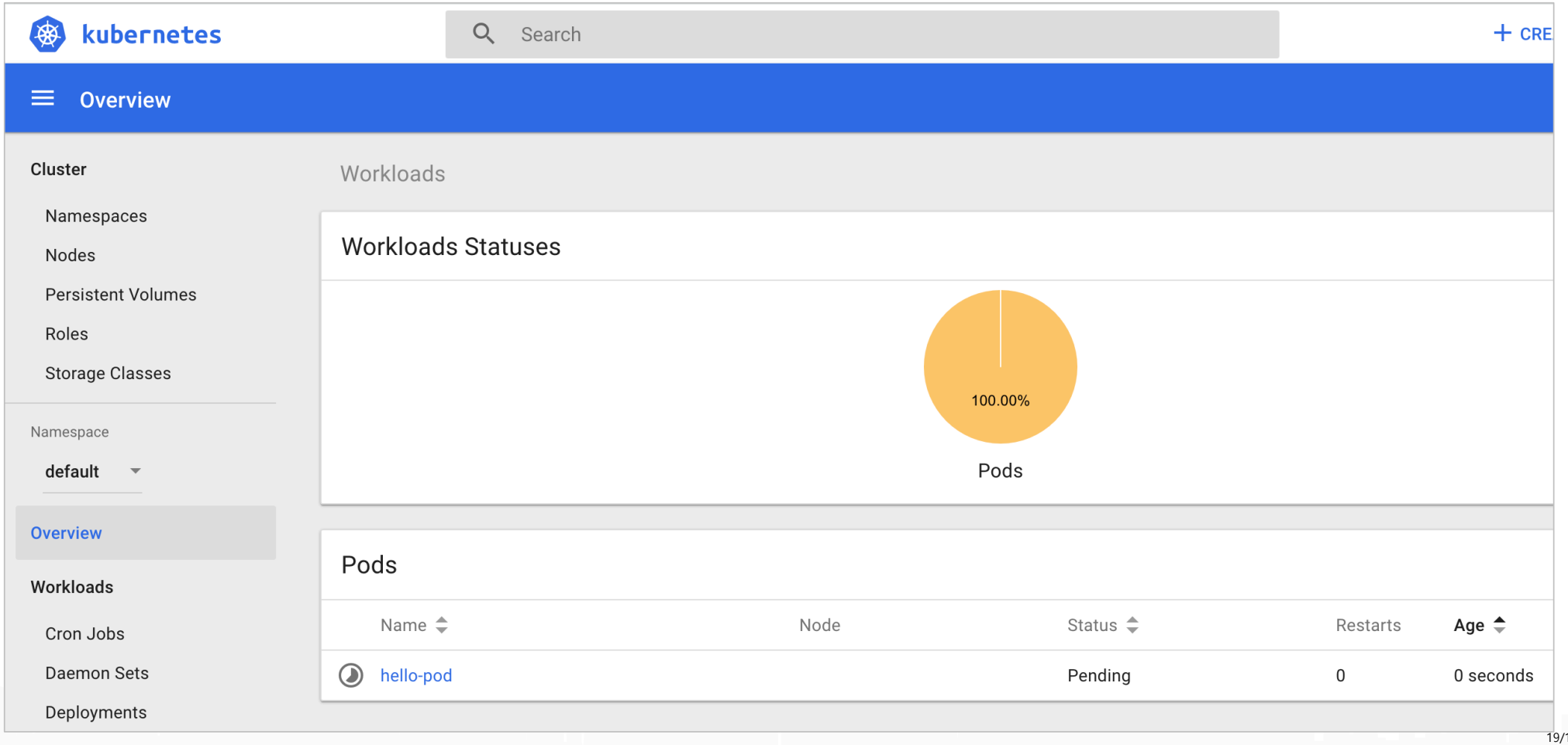
# Kubernetes Demo

- ***Kubernetes version***



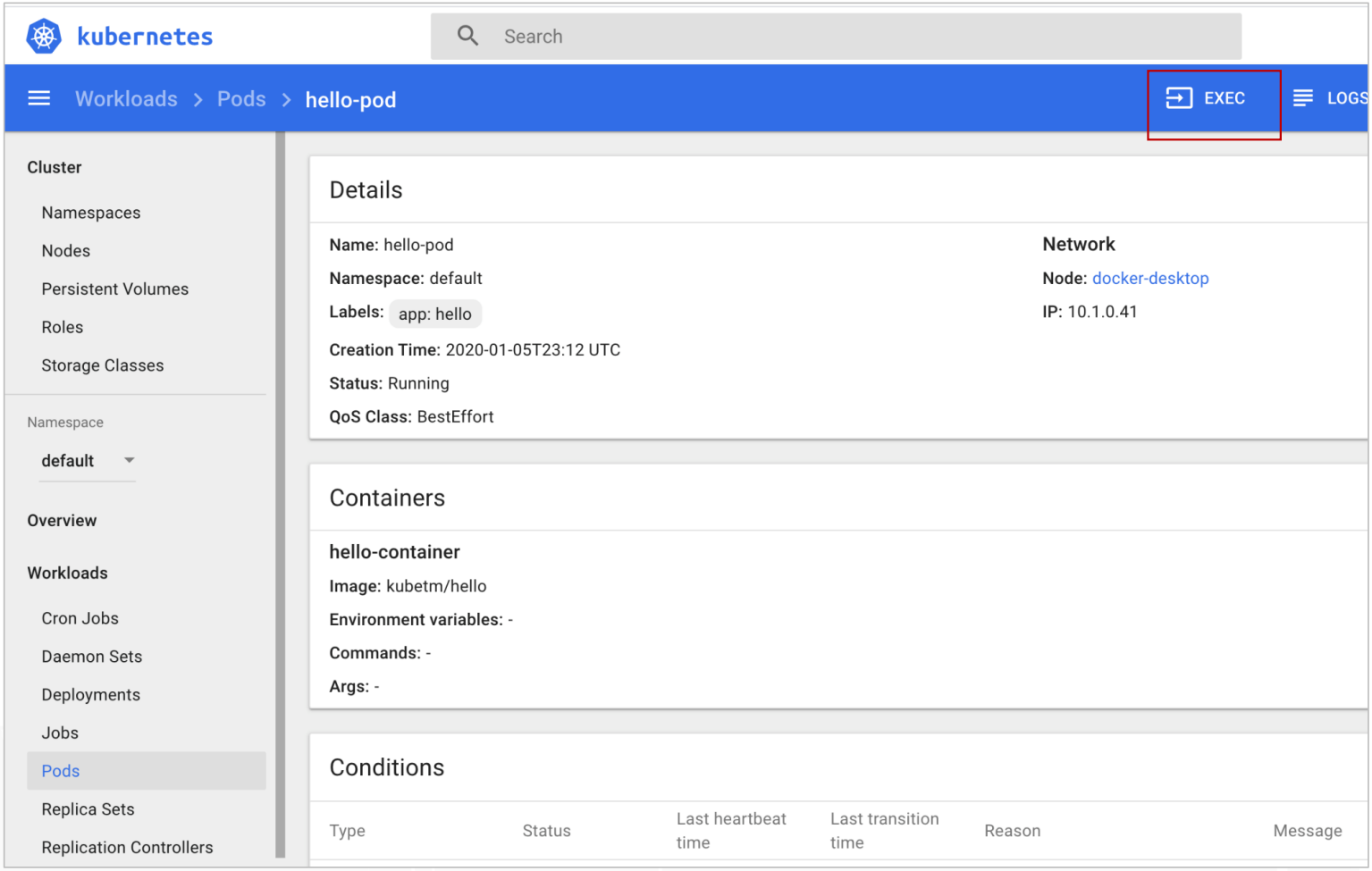
# Kubernetes Demo

- ***Kubernetes version***



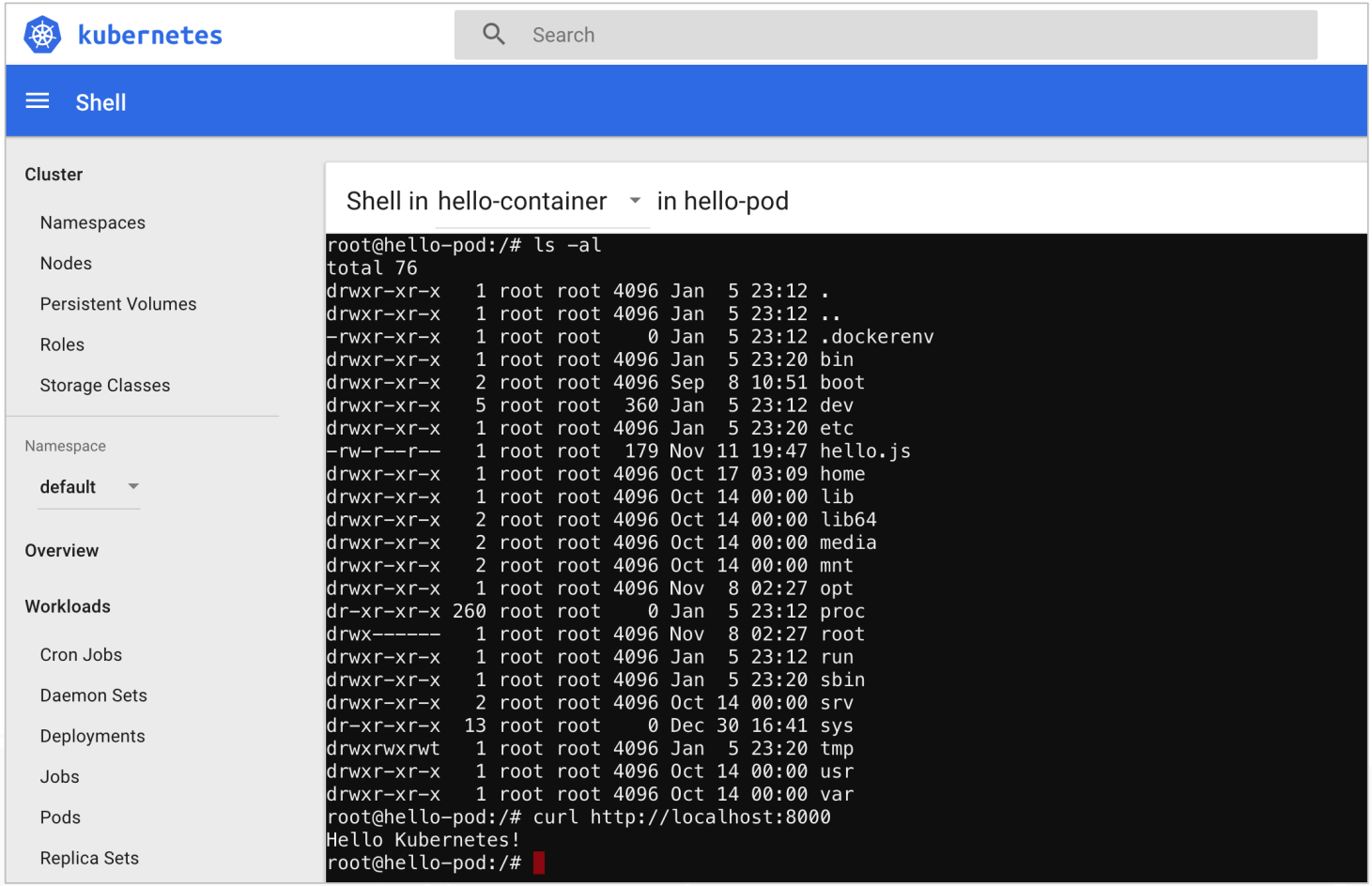
# Kubernetes Demo

- ***Kubernetes version***



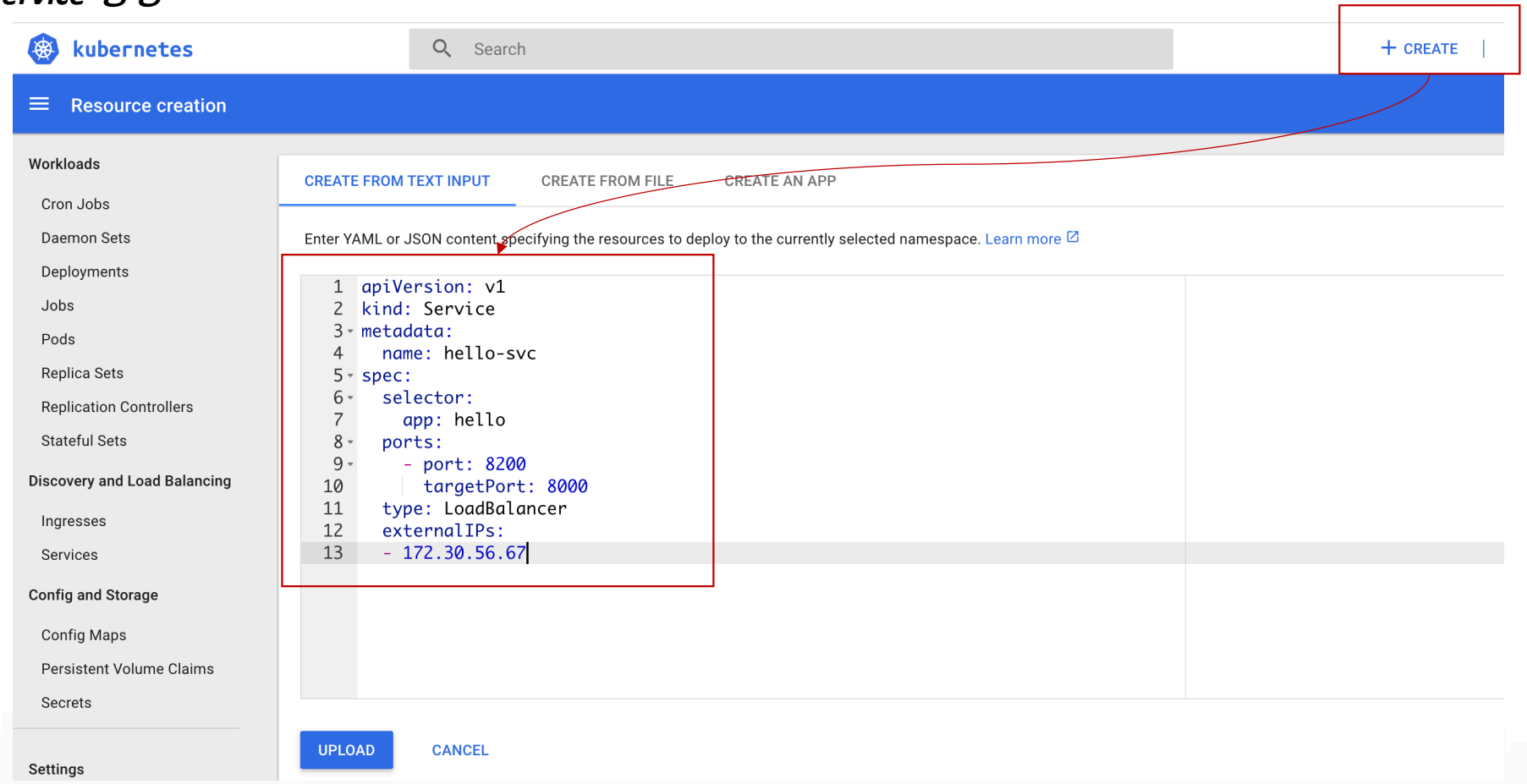
# Kubernetes Demo

- ***Kubernetes version***



# Kubernetes Demo

- **Kubernetes version**
  - **Service 생성**



# Kubernetes Demo

- ***Kubernetes version***

  ***- Service 생성***