

Developing and deploying a Node.js app from Docker to Kubernetes.

```
PutTY (inactive)
login as: ubuntu
* Authenticating with public key "imported-openssh-key"
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-1045-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Sat Sep 25 10:09:28 UTC 2021

System load:  0.0          Processes:    103
Usage of /:   14.5% of 9.63GB   Users logged in: 1
Memory usage: 24%          IPv4 address for eth0: 172.31.12.226
Swap usage:   0%

1 update can be applied immediately.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Sat Sep 25 10:04:27 2021 from 13.233.177.4
ubuntu@ip-172-31-12-226:~$ curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.33.1/install.sh | bash
% Total % Received % Xferd Average Speed Time Time Time Current
         Dload Upload Total Spent Left Speed
100 11388 100 11388 0 0 53971 0 --:--:-- --:--:-- --:--:-- 53971
=> nvm is already installed in /home/ubuntu/.nvm, trying to update using git
=> Compressing and cleaning up git repository
Enumerating objects: 8441, done.
Counting objects: 100% (8441/8441), done.
Compressing objects: 100% (8356/8356), done.
Writing objects: 100% (8441/8441), done.
Total 8441 (delta 5663), reused 2778 (delta 0)

=> nvm source string already in /home/ubuntu/.bashrc
=> bash_completion source string already in /home/ubuntu/.bashrc
=> You currently have modules installed globally with 'npm'. These will no
=> longer be linked to the active version of Node when you install a new node
=> with 'nvm'; and they may (depending on how you construct your '$PATH')
=> override the binaries of modules installed with 'nvm':

/home/ubuntu/.nvm/versions/node/v16.10.0/lib
└─ corepack@0.9.0
=> If you wish to uninstall them at a later point (or re-install them under your
=> 'nvm' Nodes), you can remove them from the system Node as follows:

    $ nvm use system
    $ npm uninstall -g a_module

=> Close and reopen your terminal to start using nvm or run the following to use it now:
```

```
PutTY (inactive)
Processing triggers for libc-bin (2.31-0ubuntu9.2) ...
ubuntu@ip-172-31-12-226:~$ docker --version
Docker version 20.10.7, build 20.10.7-0ubuntu1-20.04.1
ubuntu@ip-172-31-12-226:~$ mkdir nodejs
ubuntu@ip-172-31-12-226:~$ cd nodejs/
ubuntu@ip-172-31-12-226:~/nodejs$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See 'npm help init' for definitive documentation on these fields
and exactly what they do.

Use 'npm install <pkg>' afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (nodejs) Anu
Sorry, name can no longer contain capital letters.
package name: (nodejs) anu
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author:
license: (ISC)
About to write to /home/ubuntu/nodejs/package.json:

{
  "name": "anu",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}

Is this OK? (yes) yes
npm notice
npm notice New patch version of npm available! 7.24.0 -> 7.24.1
npm notice Changelog: https://github.com/npm/cli/releases/tag/v7.24.1
npm notice Run npm install -g npm@7.24.1 to update!
npm notice
ubuntu@ip-172-31-12-226:~/nodejs$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.
```

```
See 'docker run --help'.
ubuntu@ip-172-31-12-226:~/nodejs$ docker kill nodongo
Error response from daemon: Cannot kill container: nodongo: Container 6048a5800397e24f2538f1936af31d538a736ce6ebab3ae68fa540c244b18b74 is not running
ubuntu@ip-172-31-12-226:~/nodejs$ docker run -d --name nodongo -p 3000:3000 node-server
docker: Error response from daemon: Conflict. The container name "/nodongo" is already in use by container "6048a5800397e24f2538f1936af31d538a736ce6ebab3ae68fa540c244b18b74". You have to re
move (or rename) that container to be able to reuse that name.
See 'docker run --help'.
ubuntu@ip-172-31-12-226:~/nodejs$ docker run -d -p 3000:3000 node-server
8d2bad4d4db21a3a1781f9b2800e761f66c31cc2aa7bbe0874068586824ed7df1
ubuntu@ip-172-31-12-226:~/nodejs$ docker tag node-server anu33317/nodejs-starter:1.1
ubuntu@ip-172-31-12-226:~/nodejs$ docker push anu33317/nodejs-starter:1.1
The push refers to repository [docker.io/anu33317/nodejs-starter]
03c358d6ea66: Preparing
e4442293713c: Preparing
004b2812ce84: Preparing
a652bbd81bf4: Preparing
ed09928f5a32: Preparing
ee50c22fd6c: Waiting
d8183b2c9c73: Waiting
5aea01ea0a0f: Waiting
05f4935ad90a: Waiting
c96f2308ab16: Waiting
39c2f9ead82d: Waiting
0dabcc98eeef: Waiting
6885f9305c0a: Waiting
denied: requested access to the resource is denied
ubuntu@ip-172-31-12-226:~/nodejs$ docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com to create one.
Username: anu33317
Password:
WARNING! Your password will be stored unencrypted in /home/ubuntu/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
ubuntu@ip-172-31-12-226:~/nodejs$ docker push anu33317/nodejs-starter:1.1
The push refers to repository [docker.io/anu33317/nodejs-starter]
03c358d6ea66: Pushed
e4442293713c: Pushed
004b2812ce84: Pushed
a652bbd81bf4: Pushed
ed09928f5a32: Mounted from library/node
ee50c22fd6c: Mounted from library/node
d8183b2c9c73: Mounted from library/node
5aea01ea0a0f: Mounted from library/node
05f4935ad90a: Mounted from library/node
c96f2308ab16: Mounted from library/node
39c2f9ead82d: Mounted from library/node
0dabcc98eeef: Mounted from library/node
6885f9305c0a: Mounted from library/node
1.1: digest: sha256:bcc7c8ad630f7ef5d5f2b3d128914ce632e7aa78cfaea4308582dec2b0ab6580 size: 3050
ubuntu@ip-172-31-12-226:~/nodejs$ []
```

Meet - cww-pggz-jj

DevOps Student details - Google She...

Developing and deploying a Node.js...

Connect to instance | EC2 Managem...

i-04c73d4bf8d27528d (Ubuntu) | X

https://ap-south-1.console.aws.amazon.com/ec2/v2/connect/ubuntu/i-04c73d4bf8d27528d

Must Do Coding Q... Roadmap To Learn... People | CALYXPOD Cisco Jobs Linked List Data Str... Internship Jobs 202... W3Schools Online... Current Local Time... BuilderIO/figma-ht... niinpatel/nodejs-im...

Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-1045-aws x86_64)

* Documentation: <https://help.ubuntu.com>

* Management: <https://landscape.canonical.com>

* Support: <https://ubuntu.com/advantage>

System information as of Sat Sep 25 10:47:04 UTC 2021

| | | | |
|---------------|-----------------|---------------------------|---------------|
| System load: | 0.07 | Processes: | 108 |
| Usage of /: | 30.6% of 9.63GB | Users logged in: | 1 |
| Memory usage: | 38% | IPv4 address for docker0: | 172.17.0.1 |
| Swap usage: | 0% | IPv4 address for eth0: | 172.31.12.226 |

125 updates can be applied immediately.
57 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Last login: Sat Sep 25 10:28:53 2021 from 13.233.177.1
ubuntu@ip-172-31-12-226:~\$ curl http://localhost:3000
LaxmiNarayana JaganMohan Adityaubuntu@ip-172-31-12-226:~\$

i-04c73d4bf8d27528d (Ubuntu)

Public IPs: 13.232.119.51 Private IPs: 172.31.12.226

Meet - cwr-ppgz-jj

DevOps Student details - Google

Connect to instance | EC2 Manu...

Docker Hub

Developing and deploying a No...

Deploy Containers Using YA...

https://www.katacoda.com/courses/kubernetes/creating-kubernetes-yaml-definitions

Must Do Coding Q... Roadmap To Learn... People | CALYXPOD Cisco Jobs Linked List Data Str... Internship Jobs 202... W3Schools Online... Current Local Time... BuilderIO/figma-ht... niinpatel/nodejs-im...

O'REILLYKatacodaKATACODA OVERVIEW & SOLUTIONS

TRY O'REILLYLOG IN

Deploy Containers Using YAML

Step 2 of 3

Labels:

app: webapp1

spec:

type: NodePort

ports:

- port: 80

nodePort: 30080

selector:

app: webapp1

All Kubernetes objects are deployed in a consistent way using *kubectl*.

Deploy the Service with

kubectl create -f service.yaml ↵

As before, details of all the Service objects deployed with *kubectl get svc* ↵. By describing the object it's possible to discover more details about the configuration

kubectl describe svc webapp1-svc ↵

curl host01:30080 ✓

CONTINUE

deployment.yaml

service.yaml

Terminal

1 # This is your Editor pane.
2 apiVersion: apps/v1 #1
3 kind: Deployment #2
4 metadata: #3
5 name: nodejs-deployment #4
6 spec: #5
7 replicas: 2 #6
8 selector: #7
9 matchLabels: #7
10 app: nodejs #7
11 template: #8
12 metadata: #9
13 labels: #10
14 app: nodejs #11
15 spec: #12
16 containers: #13
17 - name: nodongo #14
18 image: anu33317/nodejs-starter:1.1 #15
19 ports: #16
20 - containerPort: 3000 #17

\$ curl https://172.17.15.61:32045
curl: (7) Failed to connect to 172.17.15.61 port 32045: No route to host
\$ curl host01:30080
curl: (7) Failed to connect to host01 port 30080: Connection refused
\$ curl host01:32045
LaxmiNarayana JaganMohan Aditya\$

Meet - cwr-ppgz-jj

DevOps Student details - Google

Connect to instance | EC2 Manu...

Docker Hub

Developing and deploying a No...

Deploy Containers Using YA...

https://www.katacoda.com/courses/kubernetes/creating-kubernetes-yaml-definitions

Must Do Coding Q... Roadmap To Learn... People | CALYXPOD Cisco Jobs Linked List Data Str... Internship Jobs 202... W3Schools Online... Current Local Time... BuilderIO/figma-ht... niinpatel/nodejs-im...

O'REILLYKatacodaKATACODA OVERVIEW & SOLUTIONS

TRY O'REILLYLOG IN

Deploy Containers Using YAML

Step 2 of 3

Labels:

app: webapp1

spec:

type: NodePort

ports:

- port: 80

nodePort: 30080

selector:

app: webapp1

All Kubernetes objects are deployed in a consistent way using *kubectl*.

Deploy the Service with

kubectl create -f service.yaml ↵

As before, details of all the Service objects deployed with *kubectl get svc* ↵. By describing the object it's possible to discover more details about the configuration

kubectl describe svc webapp1-svc ↵

curl host01:30080 ✓

CONTINUE

deployment.yaml

service.yaml

Terminal

1 # This is your Editor pane.
2 apiVersion: v1
3 kind: ConfigMap
4 metadata:
5 namespace: metallb-system
6 name: config
7 data:
8 config: |
9 address-pools:
10 - name: default
11 protocol: layer2
12 addresses:
13 - 172.17.15.61-172.17.15.71
14

\$ curl https://172.17.15.61:32045
curl: (7) Failed to connect to 172.17.15.61 port 32045: No route to host
\$ curl host01:30080
curl: (7) Failed to connect to host01 port 30080: Connection refused
\$ curl host01:32045
\$

Deploy Containers Using YAML

Step 2 of 3

```
name: webapp1-svc
labels:
  app: webapp1
spec:
  type: NodePort
  ports:
    - port: 80
      nodePort: 30080
  selector:
    app: webapp1
```

All Kubernetes objects are deployed in a consistent way using *kubectl*.

Deploy the Service with
`kubectl create -f service.yaml ↵`

As before, details of all the Service objects deployed with
`kubectl get svc ↵`. By describing the object it's possible to discover more details about the configuration
`kubectl describe svc webapp1-svc ↵`

`curl host01:30080 ✓`

deployment.yaml

service.yaml

```
1 # This is your Editor pane.
2 apiVersion: v1
3 kind: ConfigMap
4 metadata:
5   namespace: metallb-system
```

Terminal

```
$ minikube start --wait=false
* minikube v1.8.1 on Ubuntu 18.04
* Using the none driver based on user configuration
* Running on localhost (CPUs=2, Memory=2460MB, Disk=145651MB) ...
* OS release is Ubuntu 18.04.4 LTS
* Preparing Kubernetes v1.17.3 on Docker 19.03.6 ...
- kublet.resolv-conf=/run/systemd/resolve/resolv.conf
* Launching Kubernetes ...
* Enabling addons: default-storageclass, storage-provisioner
* Configuring local host environment ...
* Done! kubectl is now configured to use "minikube"
$
$
$ kubectl create -f deployment.yaml
deployment.apps/nodejs-deployment created
$ kubectl get deploy,po
NAME                                     READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/nodejs-deployment      0/2     2             0            12s

NAME                                     READY   STATUS              RESTARTS   AGE
pod/nodejs-deployment-865f95d7c5-dm8mx  0/1     ContainerCreating   0           12s
pod/nodejs-deployment-865f95d7c5-t6b8p  0/1     ContainerCreating   0           12s
$ kubectl expose deployment nodejs-deployment --type="LoadBalancer"
service/nodejs-deployment exposed
$ kubectl get svc
NAME      TYPE          CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
kubernetes ClusterIP   10.96.0.1        <none>            443/TCP          55s
nodejs-deployment LoadBalancer   10.102.150.67    <pending>        3000:30060/TCP   10s
```

Deploy Containers Using YAML

Step 2 of 3

```
name: webapp1-svc
labels:
  app: webapp1
spec:
  type: NodePort
  ports:
    - port: 80
      nodePort: 30080
  selector:
    app: webapp1
```

All Kubernetes objects are deployed in a consistent way using *kubectl*.

Deploy the Service with
`kubectl create -f service.yaml ↵`

As before, details of all the Service objects deployed with
`kubectl get svc ↵`. By describing the object it's possible to discover more details about the configuration
`kubectl describe svc webapp1-svc ↵`

`curl host01:30080 ✓`

deployment.yaml

service.yaml

```
1 # This is your Editor pane.
2 apiVersion: v1
3 kind: ConfigMap
4 metadata:
5   namespace: metallb-system
```

Terminal

```
pod/nodejs-deployment-865f95d7c5-dm8mx  0/1     ContainerCreating   0           12s
pod/nodejs-deployment-865f95d7c5-t6b8p  0/1     ContainerCreating   0           12s
$ kubectl expose deployment nodejs-deployment --type="LoadBalancer"
service/nodejs-deployment exposed
$ kubectl get svc
NAME      TYPE          CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
kubernetes ClusterIP   10.96.0.1        <none>            443/TCP          55s
nodejs-deployment LoadBalancer   10.102.150.67    <pending>        3000:30060/TCP   10s
$ kubectl apply -f https://raw.githubusercontent.com/google/metallb/v0.9.3/manifests/namespace.yaml
namespace/metallb-system created
$ kubectl apply -f https://raw.githubusercontent.com/google/metallb/v0.9.3/manifests/metallb.yaml
podsecuritypolicy.policy/controller created
podsecuritypolicy.policy/speaker created
serviceaccount/controller created
serviceaccount/speaker created
clusterrole.rbac.authorization.k8s.io/metallb-system:controller created
clusterrole.rbac.authorization.k8s.io/metallb-system:speaker created
role.rbac.authorization.k8s.io/config-watcher created
role.rbac.authorization.k8s.io/pod-lister created
clusterrolebinding.rbac.authorization.k8s.io/metallb-system:controller created
clusterrolebinding.rbac.authorization.k8s.io/metallb-system:speaker created
rolebinding.rbac.authorization.k8s.io/config-watcher created
rolebinding.rbac.authorization.k8s.io/pod-lister created
daemonset.apps/speaker created
deployment.apps/controller created
$ kubectl create secret generic -n metallb-system memberlist --from-literal=secretkey="$(openssl rand -base64 128)"
secret/memberlist created
$ minikube ip
172.17.0.28
$ kubectl create -f configmap.yaml
```

Meet - cwr-ppgi-jj

DevOps Student details - Google

Connect to instance [EC2 Manu...

Docker Hub

Developing and deploying a No...

Deploy Containers Using YA...

https://www.katacoda.com/courses/kubernetes/creating-kubernetes-yaml-definitions

Must Do Coding Q... Roadmap To Learn... People | CALYXPOD Cisco Jobs Linked List Data Str... Internship Jobs 202... W3Schools Online... Current Local Time... BuilderIO/figma-ht... niinpatel/nodejs-im...

O'REILLYKatacoda

KATACODA OVERVIEW & SOLUTIONS

TRY O'REILLY

LOG IN

Deploy Containers UsingYAML

Step 2 of 3

```
name: webapp1-svc
labels:
  app: webapp1
spec:
  type: NodePort
  ports:
    - port: 80
      nodePort: 30080
  selector:
    app: webapp1
```

All Kubernetes objects are deployed in a consistent way using `kubectl`.

Deploy the Service with
`kubectl create -f service.yaml`

As before, details of all the Service objects deployed with
`kubectl get svc`. By describing the object it's possible to discover more details about the configuration
`kubectl describe svc webapp1-svc`

`curl host01:30080` ✓

deployment.yaml

service.yaml

```
1 # This is your Editor pane.
2 apiVersion: v1
3 kind: ConfigMap
4 * metadata:
5   namespace: metallb-system
```

Terminal

```
clusterrolebinding.rbac.authorization.k8s.io/metallb-system:speaker created
rolebinding.rbac.authorization.k8s.io/config-watcher created
rolebinding.rbac.authorization.k8s.io/pod-lister created
daemonset.apps/speaker created
deployment.apps/controller created
$ kubectl create secret generic -n metallb-system memberlist --from-literal=secretkey="$(openssl rand -base64 128)"
secret/memberlist created
$ minikube ip
172.17.0.28
$ kubectl create -f configmap.yaml
error: the path "configmap.yaml" does not exist
$ kubectl create -f service.yaml
configmap/config created
$ kubectl delete svc nodejs-deployment
service "nodejs-deployment" deleted
$ kubectl expose deployment nodejs-deployment --type="LoadBalancer"
service/nodejs-deployment exposed
$ kubectl get svc
NAME                TYPE          CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
kubernetes           ClusterIP     10.96.0.1        <none>            443/TCP          5m14s
nodejs-deployment    LoadBalancer 10.109.32.187    172.17.15.61     3000:32045/TCP   12s
$ curl http://172.17.15.61:32045
curl: (7) Failed to connect to 172.17.15.61 port 32045: No route to host
$ curl https://172.17.15.61:32045
curl: (7) Failed to connect to 172.17.15.61 port 32045: No route to host
$ curl host01:30080
curl: (7) Failed to connect to host01 port 30080: Connection refused
$ curl host01:32045
$ curl host01:32045
LaxmiNarayana JagannMohan Aditya$
```