kubernetes集群应用项目部署

一、部署前准备工作及注意事项

部署项目情况

业务架构

- dubbo
- spring cloud

第三方服务

- 数据库服务 mysql
- 缓存服务 redis
- 协调服务 zookeeper
- 消息中间件服务 rabbitmq kafka
- 注册服务 eruka

服务之间的通信方式

项目所需要的资源

- 硬件资源

部署项目所需要的k8s资源

多套k8s集群资源

- 测试环境

- 开发环境
- 生产环境

使用namespace隔离项目或环境

- 每项目独立使用namespace
- 每环境独立使用namespace

有状态应用部署

- statefulset
- pv
- pvc

无状态应用部署

- deployment 例如wordpress部署为无状态

暴露外部访问

- service 例如mysql部署 headless service
- ingress 例如暴露wordpress对外访问

密钥及配置管理

- secret 例如k8s集群使用harbor服务器
- configmap 例如mysql配置文件

项目基础镜像

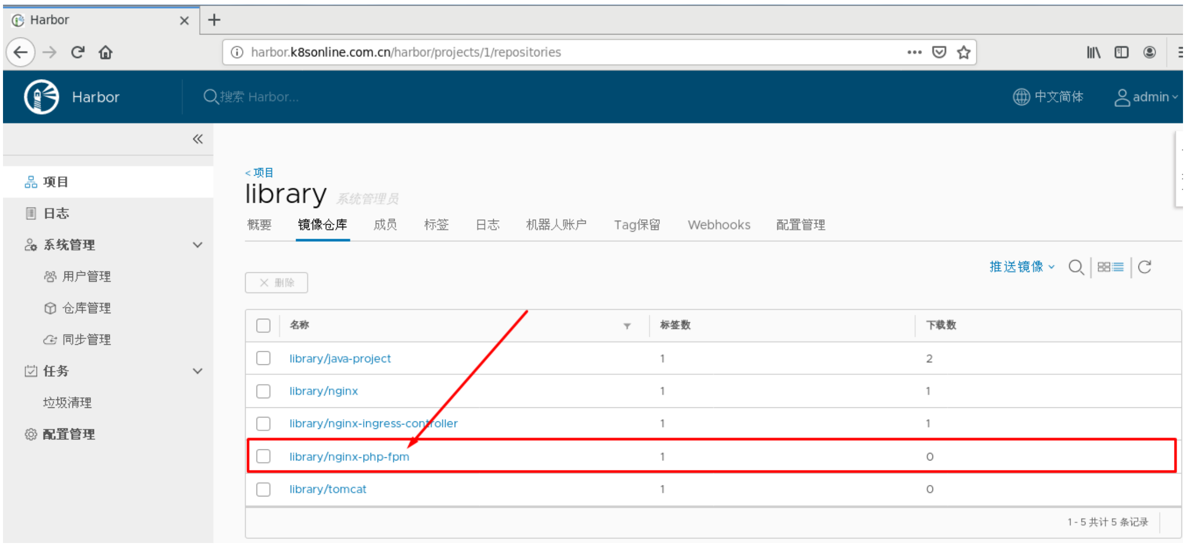
> 使用nginx与php环境发布php项目，需要nginx-php环境镜像。可以直接下载，也可以选择定制。

获取nginx-php镜像

```
可采用直接下载，或进行定制
[root@harborserver ~]# docker pull nginx-php

nginx-php镜像在文档及相关资源目录中提供，不可直接在dockerhub下载。
```

上传nginx-php镜像到harbor服务器

```
[root@harborserver ~]# docker tag nginx-php:latest
harbor.k8sonline.com.cn/library/nginx-php:v1
[root@harborserver ~]# docker push
harbor.k8sonline.com.cn/library/nginx-php:v1
```



名称	标签数	下载数
library/java-project	1	2
library/nginx	1	1
library/nginx-ingress-controller	1	1
library/nginx-php-fpm	1	0
library/tomcat	1	0

1 - 5 共计 5 条记录

项目文件

项目代码

```
[root@harborserver ~]# wget https://cn.wordpress.org/latest-
zh_CN.tar.gz
由于wordpress网站访问量过大，请提前尝试下载。


[root@harborserver ~]# tar xf latest-zh_CN.tar.gz

[root@harborserver ~]# ls
php-project
latest-zh_CN.tar.gz
wordpress #此为解压出来的目录


[root@harborserver ~]# mkdir php-project

[root@harborserver ~]# cp -r wordpress php-project/
[root@harborserver ~]# ls php-project/
wordpress

[root@harborserver ~]# ls php-project/wordpress/
index.php        wp-blog-header.php    wp-includes        wp-
settings.php
license.txt      wp-comments-post.php  wp-links-opml.php  wp-
signup.php
readme.html      wp-config-sample.php  wp-load.php        wp-
trackback.php
wp-activate.php  wp-content            wp-login.php
xmlrpc.php
wp-admin         wp-cron.php           wp-mail.php
```

项目资源清单文件

```
[root@nginxk8syaml ~]# mkdir /usr/share/nginx/html/php-project

[root@nginxk8syaml ~]# ls /usr/share/nginx/html/php-project
deployment.yaml  mysql.yaml ingress.yaml   namespace.yaml
 service.yaml
```

项目资源清单文件内容

namespace.yaml

```
[root@nginxk8syaml php-project]# cat namespace.yaml
apiVersion: v1
kind: Namespace
metadata:
  name: phpproject
```

mysql.yaml

```
[root@nginxk8syaml php-project]# cat mysql.yaml
apiVersion: v1
kind: Service
metadata:
  name: phpmysql
  namespace: phpproject
spec:
  ports:
  - port: 3306
    name: mysql
  clusterIP: None
  selector:
    app: mysql-php
```

```yaml
---

apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: db
  namespace: phpproject
spec:
  selector:
    matchLabels:
      app: mysql-php
  serviceName: "phpmysql"
  template:
    metadata:
      labels:
        app: mysql-php
    spec:
      containers:
      - name: mysql
        image: mysql:5.7
        env:
        - name: MYSQL_ROOT_PASSWORD
          value: "123456"
        - name: MYSQL_DATABASE
          value: wordpress
        ports:
        - containerPort: 3306
        volumeMounts:
        - mountPath: "/var/lib/mysql"
          name: mysql-data
  volumeClaimTemplates:
  - metadata:
      name: mysql-data
    spec:
      accessModes: ["ReadWriteMany"]
      storageClassName: "managed-nfs-storage"
      resources:
        requests:
          storage: 3Gi
```

deployment.yaml

```
[root@nginxk8syaml php-project]# cat deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: php-project
  namespace: phpproject
spec:
  replicas: 2
  selector:
    matchLabels:
      project: phpweb
      app: php-project
  template:
    metadata:
      labels:
        project: phpweb
        app: php-project
    spec:
      imagePullSecrets:
      - name: harborreg
      containers:
      - name: nginx-php
        image: harbor.k8sonline.com.cn/library/php-project:v1
        imagePullPolicy: Always
        ports:
        - containerPort: 80
          name: web
          protocol: TCP
        resources:
          requests:
            cpu: 0.5
            memory: 256Mi
          limits:
            cpu: 1
            memory: 1Gi
        livenessProbe:
          httpGet:
            path: /index.php
            port: 80
          initialDelaySeconds: 6
          timeoutSeconds: 20
        readinessProbe:
          httpGet:
            path: /index.php
            port: 80
          initialDelaySeconds: 6
          timeoutSeconds: 20
```

service.yaml

```
[root@nginxk8syaml php-project]# cat service.yaml
apiVersion: v1
kind: Service
metadata:
  name: php-project
  namespace: phpproject
spec:
  selector:
    project: phpweb
    app: php-project
  ports:
  - name: web
    port: 80
    targetPort: 80
```

ingress.yaml

```
[root@nginxk8syaml php-project]# cat ingress.yaml
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: php-project
  namespace: phpproject
spec:
  rules:
    - host: php.k8sonline.com.cn
      http:
        paths:
        - path: /
          backend:
            serviceName: php-project
            servicePort: 80
```

编排部署

由于最终部署交付物为容器镜像，需要基于项目资料生成最终镜像。

项目资料

- php(wordpress)项目代码
- 项目资源清单

项目镜像构建

- 手动构建镜像(Dockerfile)
- 自动构建镜像(ci/cd)

项目部署工作流程

- 安装kubectl命令行工具

默认部署集群时安装

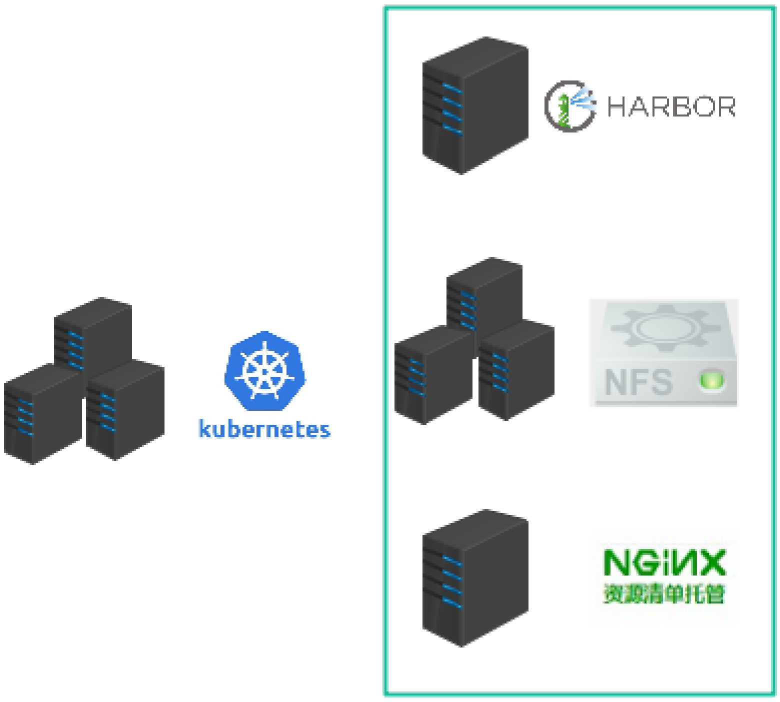
- 创建镜像

- 编写资源清单文件

- 使用资源清单文件创建项目service

- 使用ingress实现项目对外访问

项目拓扑



二、存储准备

> 本次使用NFS服务做为K8S集群后端存储

pv动态供给存储配置

> 参考连接：https://github.com/kubernetes-incubator/external-storage/tree/master/nfs-client/deploy

创建NFS服务

在k8s集群内或集群外配置NFS服务，用于为k8s集群内有状态应用部署存储数据，即可提供持久化存储。

服务端

前置条件

```
[root@nfsserver ~]# cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
TYPE=Ethernet
ONBOOT=yes
BOOTPROTO=static
IPADDR=192.168.122.250
NETMASK=255.255.255.0
GATEWAY=192.168.122.1
DNS1=192.168.122.254
DNS2=119.29.29.29
```

```
[root@nfsserver ~]# yum -y install nfs-utils rpcbind

[root@nfsserver ~]# mkdir -p /vdb/k8spublic

[root@nfsserver ~]# cat /etc/exports
/vdb/k8spublic  *(rw,sync,no_root_squash)

[root@nfsserver ~]# systemctl enable nfs-server;systemctl start nfs-server
```

客户端

> 所有集群内节点全部安装NFS客户端

```
[root@XXX ~]# yum -y install nfs-utils rpcbind

XXX替换为master1 work1 work2
```

获取持久化存储配置文件

> 把https://github.com/kubernetes-incubator/external-storage/克隆到k8s集群master节点,提醒:由于网速原因，可提前下载。文件内容或有变化，请参考下载文件内容。

```
[root@master1 ~]# git clone https://github.com/kubernetes-
incubator/external-storage/
```

```
[root@master1 ~]# ls
external-storage
[root@master1 ~]# cd external-storage/
[root@master1 external-storage]# ls
aws                  flex         LICENSE       OWNERS
    test.sh
ceph                 gluster      local-volume  README.md
   unittests.sh
code-of-conduct.md   Gopkg.lock   Makefile      RELEASE.md
    vendor
CONTRIBUTING.md      Gopkg.toml   nfs           repo-infra
```

```
deploy.sh           hack        nfs-client
 SECURITY_CONTACTS
digitalocean        iscsi       openebs       snapshot
[root@master1 external-storage]# cd nfs-client/
[root@master1 nfs-client]# ls
CHANGELOG.md  cmd  deploy  docker  Makefile  OWNERS
 README.md
[root@master1 nfs-client]# cd deploy/
[root@master1 deploy]# ls
class.yaml           deployment.yaml  rbac.yaml        test-
pod.yaml
deployment-arm.yaml  objects          test-claim.yaml
```

查看或修改资源清单文件

查看class.yaml文件

用于动态PV创建的实现

```
[root@nginxk8syaml pv]# cat class.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: managed-nfs-storage #部署项目时需要指定此处名称，告诉应用使
用哪一个StorageClass自动创建pv
provisioner: fuseim.pri/ifs # or choose another name, must
match deployment's env PROVISIONER_NAME'
parameters:
  archiveOnDelete: "false"
```

查看rbac.yaml文件

用于赋予deployment部署的pod拥有权限访问api-server，可以访问k8s集群中pvc资源，简言之，用于授权

```
[root@nginxk8syaml pv]# cat rbac.yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: nfs-client-provisioner  # replace with namespace where provisioner is deployed

  namespace: default
---
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: nfs-client-provisioner-runner
rules:
  - apiGroups: [""]
    resources: ["persistentvolumes"]
    verbs: ["get", "list", "watch", "create", "delete"]
  - apiGroups: [""]
    resources: ["persistentvolumeclaims"]
    verbs: ["get", "list", "watch", "update"]
  - apiGroups: ["storage.k8s.io"]
    resources: ["storageclasses"]
    verbs: ["get", "list", "watch"]
  - apiGroups: [""]
    resources: ["events"]
    verbs: ["create", "update", "patch"]
---
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: run-nfs-client-provisioner
subjects:
  - kind: ServiceAccount
    name: nfs-client-provisioner  # replace with namespace where provisioner is deployed

    namespace: default
roleRef:
  kind: ClusterRole
  name: nfs-client-provisioner-runner
  apiGroup: rbac.authorization.k8s.io
---
kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: leader-locking-nfs-client-provisioner  # replace with namespace where provisioner is deployed

  namespace: default
rules:
  - apiGroups: [""]
    resources: ["endpoints"]
    verbs: ["get", "list", "watch", "create", "update", "patch"]
---
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: leader-locking-nfs-client-provisioner
subjects:
  - kind: ServiceAccount
    name: nfs-client-provisioner # replace with namespace where provisioner is deployed

    namespace: default
roleRef:
  kind: Role
  name: leader-locking-nfs-client-provisioner
  apiGroup: rbac.authorization.k8s.io
```

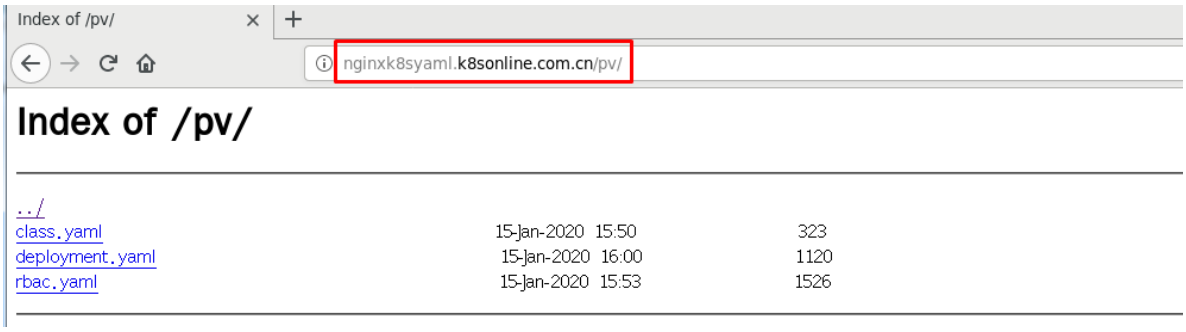
修改deployment.yaml文件

用于创建一个名称叫nfs-client-provisioner的pod，是nfs在k8s集群中的提供者，用于操作nfs创建共享目录

```
[root@nginxk8syaml pv]# cat deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nfs-client-provisioner
  labels:
    app: nfs-client-provisioner # replace with namespace
where provisioner is deployed

  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nfs-client-provisioner
  strategy:
    type: Recreate
  selector:
    matchLabels:
      app: nfs-client-provisioner
  template:
    metadata:
      labels:
        app: nfs-client-provisioner
    spec:
      serviceAccountName: nfs-client-provisioner
      containers:
        - name: nfs-client-provisioner
          image: quay.io/external_storage/nfs-client-
provisioner:latest #提前下载镜像
          volumeMounts:
            - name: nfs-client-root
              mountPath: /persistentvolumes
          env:
            - name: PROVISIONER_NAME
              value: fuseim.pri/ifs
            - name: NFS_SERVER
              value: nfs.k8sonline.com.cn #NFS服务器IP或域名
            - name: NFS_PATH
              value: /vdb/k8spublic #为NFS服务器共享目录
      volumes:
        - name: nfs-client-root
          nfs:
            server: nfs.k8sonline.com.cn #NFS服务器IP或域名
            path: /vdb/k8spublic #为NFS服务器共享目录
```

应用资源清单文件

查看资源清单托管服务器



Index of /pv/

```
../
class.yaml                                         15-Jan-2020 15:50                 323
deployment.yaml                                    15-Jan-2020 16:00                1120
rbac.yaml                                          15-Jan-2020 15:53                1526
```

应用资源清单文件

```
[root@master1 ~]# kubectl apply -f
http://nginxk8syaml.k8sonline.com.cn/pv/class.yaml
storageclass.storage.k8s.io/managed-nfs-storage created


[root@master1 ~]# kubectl apply -f
http://nginxk8syaml.k8sonline.com.cn/pv/rbac.yaml
serviceaccount/nfs-client-provisioner created
clusterrole.rbac.authorization.k8s.io/nfs-client-provisioner-
runner created
clusterrolebinding.rbac.authorization.k8s.io/run-nfs-client-
provisioner created
role.rbac.authorization.k8s.io/leader-locking-nfs-client-
provisioner created
rolebinding.rbac.authorization.k8s.io/leader-locking-nfs-
client-provisioner created
```

```
[root@master1 ~]# kubectl apply -f
http://nginxk8syaml.k8sonline.com.cn/pv/deployment.yaml
deployment.apps/nfs-client-provisioner created

```

验证nfs-client pod是否运行

```
[root@master1 ~]# kubectl get pods
NAME                                      READY   STATUS
 RESTARTS   AGE
nfs-client-provisioner-5786f95795-8bpmc   1/1     Running   0
         19s
```

三、项目准备

项目部署思路

需要部署的应用

- mysql
- nginx-php-fpm

部署应用涉及状态

- nginx-php-fpm 部署为无状态应用
- mysql 部署为有有状态应用

部署应用涉及控制器及资源

- nginx-php-fpm
  - deployment控制器
  - service
  - ingress 暴露给集群外用户访问
- mysql
  - statefulset控制器
  - headless service
  - pv,pvc 动态供给存储资源

在k8s集群中部署ingress控制器

> ingress controller以daemonSet方式部署，访问时可以绑定K8S集群任一IP地址。

在harbor服务器获取镜像

```
[root@harborserver ~]# docker pull quay.io/kubernetes-ingress-controller/nginx-ingress-controller:master


[root@harborserver ~]# docker tag quay.io/kubernetes-ingress-controller/nginx-ingress-controller:master harbor.k8sonline.com.cn/library/nginx-ingress-controller:master

[root@harborserver ~]# docker push harbor.k8sonline.com.cn/library/nginx-ingress-controller:master
```

在nginxk8syaml服务器修改ingress controller资源清单文件

下载ingress controller资源清单文件

```
[root@nginxk8syaml html]# wget https://raw.githubusercontent.com/kubernetes/ingress-nginx/master/deploy/static/mandatory.yaml
```

> 会使用K8S集群物理机TCP 80 端口，在启动前，请检查此端口是否被占用。

修改

```
[root@nginxk8syaml html]# vim mandatory.yaml

修改
   prometheus.io/port: "10254"
   prometheus.io/scrape: "true"
 spec:
  hostNetwork: true  #需要添加，不能添加引号。
  serviceAccountName: nginx-ingress-serviceaccount
  containers:
   - name: nginx-ingress-controller

......
  containers:
   - name: nginx-ingress-controller
    image: harbor.k8sonline.com.cn/library/nginx-ingress-controller:master  #修改镜像
```

在k8s集群master节点应用ingress controller资源清单文件

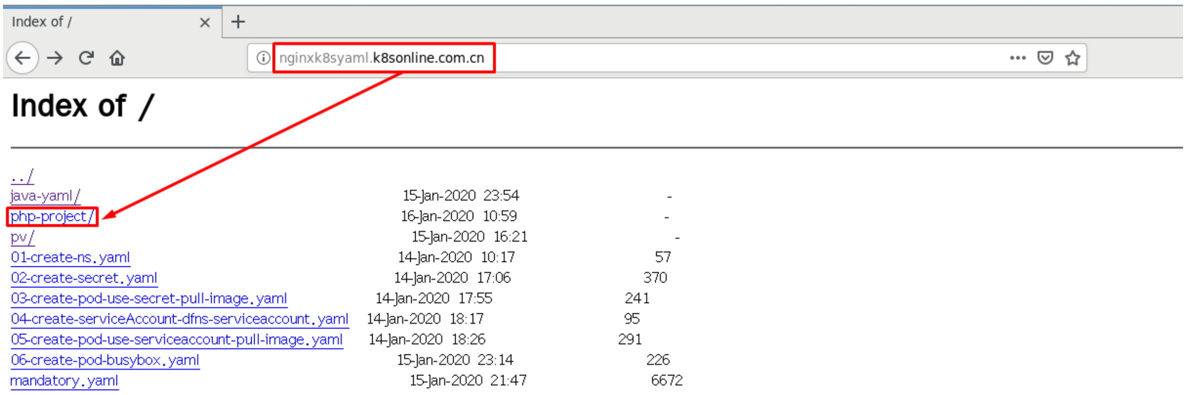
```
[root@master1 ~]# kubectl apply -f
http://nginxk8syaml.k8sonline.com.cn/mandatory.yaml
namespace/ingress-nginx created
configmap/nginx-configuration created
configmap/tcp-services created
configmap/udp-services created
serviceaccount/nginx-ingress-serviceaccount created
clusterrole.rbac.authorization.k8s.io/nginx-ingress-
clusterrole created
role.rbac.authorization.k8s.io/nginx-ingress-role created
rolebinding.rbac.authorization.k8s.io/nginx-ingress-role-
nisa-binding created
clusterrolebinding.rbac.authorization.k8s.io/nginx-ingress-
clusterrole-nisa-binding created
deployment.apps/nginx-ingress-controller created
limitrange/ingress-nginx created
```

查看ingress controller部署主机

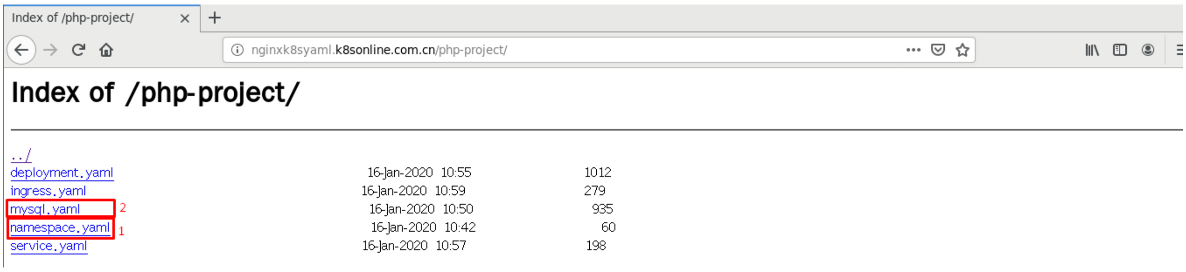
> 稍后部署ingress对象，查看ingress controller运行主机，获取IP地址后便于解析访问，也可以使用DaemonSet控制器部署，这样可以把域名解析到更多主机。

```
[root@master1 ~]# kubectl get pods -n ingress-nginx
NAME                                        READY   STATUS
 RESTARTS   AGE
nginx-ingress-controller-5c6985f9cc-snqtt   1/1     Running
7          12m

[root@master1 ~]# kubectl get pods -n ingress-nginx -o wide
NAME                                        READY   STATUS
 RESTARTS   AGE   IP                NODE    NOMINATED NODE
READINESS GATES
nginx-ingress-controller-5c6985f9cc-snqtt   1/1     Running
7          12m   192.168.122.102   work2   <none>
<none>
```

在k8s集群中部署mysql数据库应用



```
Index of /

../
java-yaml/                                        15-Jan-2020 23:54       -
php-project/                                      16-Jan-2020 10:59       -
pv/                                               15-Jan-2020 16:21       -
01-create-ns.yaml                                 14-Jan-2020 10:17      57
02-create-secret.yaml                             14-Jan-2020 17:06     370
03-create-pod-use-secret-pull-image.yaml          14-Jan-2020 17:55     241
04-create-serviceAccount-dfns-serviceaccount.yaml 14-Jan-2020 18:17      95
05-create-pod-use-serviceaccount-pull-image.yaml  14-Jan-2020 18:26     291
06-create-pod-busybox.yaml                        15-Jan-2020 23:14     226
mandatory.yaml                                    15-Jan-2020 21:47    6672
```



```
Index of /php-project/

../
deployment.yaml                                   16-Jan-2020 10:55    1012
ingress.yaml                                      16-Jan-2020 10:59     279
mysql.yaml                                        16-Jan-2020 10:50     935
namespace.yaml                                    16-Jan-2020 10:42      60
service.yaml                                      16-Jan-2020 10:57     198
```

创建命名空间

```
[root@master1 ~]# kubectl apply -f
http://nginxk8syaml.k8sonline.com.cn/php-
project/namespace.yaml
namespace/phpproject created
```

部署mysql数据库

```
[root@master1 ~]# kubectl apply -f
http://nginxk8syaml.k8sonline.com.cn/php-project/mysql.yaml
service/phpmysql created
statefulset.apps/db created
```

查看部署结果

```
[root@master1 ~]# kubectl get ns
NAME              STATUS   AGE
phpproject        Active   79s

[root@master1 ~]# kubectl get statefulset.apps -n phpproject
NAME   READY   AGE
db     1/1     76s

[root@master1 ~]# kubectl get svc -n phpproject
NAME       TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)
 AGE
phpmysql   ClusterIP   None         <none>        3306/TCP
15s

[root@master1 ~]# kubectl get pods -n phpproject
NAME   READY   STATUS    RESTARTS   AGE
db-0   1/1     Running   0          83s
```

验证mysql是否可访问

```
直接进入数据库所在的pod
[root@master1 ~]# kubectl exec -it db-0 sh -n phpproject
# mysql -uroot -p123456
mysql: [Warning] Using a password on the command line
interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.7.29 MySQL Community Server (GPL)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All
rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or
its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current
input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| wordpress          |
+--------------------+
5 rows in set (0.01 sec)


使用busybox 1.28.4镜像启动的pod进行解析或连通性测试
```

```
[root@nginxk8syaml html]# cat 06-create-pod-busybox.yaml
apiVersion: v1
kind: Pod
metadata:
  name: busybox-pod
spec:
  containers:
  - name: busybox-container
    image: busybox:1.28.4  #此镜像nslookup及ping命令都没有问题，不要下载最新版本和1.31。
    imagePullPolicy: IfNotPresent
    command:
    - sleep
    - "3600"
  restartPolicy: Always

应用资源清单文件
[root@master1 ~]# kubectl apply -f http://nginxk8syaml.k8sonline.com.cn/06-create-pod-busybox.yaml

[root@master1 ~]# kubectl get pods
NAME                                          READY   STATUS     RESTARTS   AGE
busybox-pod                                   1/1     Running    0          6s

进入busybox-pod中执行ping命令，测试db-0访问地址

[root@master1 ~]# kubectl exec -it busybox-pod sh
/ # nslookup db-0.phpmysql.phpproject
Server:    10.96.0.10
Address 1: 10.96.0.10 kube-dns.kube-system.svc.cluster.local

Name:      db-0.phpmysql.phpproject
Address 1: 172.16.215.21 db-0.phpmysql.phpproject.svc.cluster.local
/ # ping db-0.phpmysql.phpproject
PING db-0.phpmysql.phpproject (172.16.215.21): 56 data bytes
64 bytes from 172.16.215.21: seq=0 ttl=62 time=0.441 ms
64 bytes from 172.16.215.21: seq=1 ttl=62 time=0.571 ms
64 bytes from 172.16.215.21: seq=2 ttl=62 time=0.334 ms
```

```
64 bytes from 172.16.215.21: seq=3 ttl=62 time=0.544 ms
```

在NFS服务器查看存储目录

```
[root@nfsserver ~]# ls /vdb/k8spublic/
phpproject-mysql-data-db-0-pvc-b8b3d6cf-7c06-4834-bf94-
39d403bcf726

[root@nfsserver ~]# ls /vdb/k8spublic/phpproject-mysql-data-
db-0-pvc-b8b3d6cf-7c06-4834-bf94-39d403bcf726/
auto.cnf         client-key.pem  ib_logfile1
private_key.pem  sys
ca-key.pem       ib_buffer_pool  ibtmp1
 public_key.pem   wordpress
ca.pem           ibdata1         mysql               server-
cert.pem
client-cert.pem  ib_logfile0     performance_schema  server-
key.pem

[root@nfsserver ~]# ls /vdb/k8spublic/phpproject-mysql-data-
db-0-pvc-b8b3d6cf-7c06-4834-bf94-39d403bcf726/wordpress/
db.opt
```

获取php代码

> 在harbor服务器上获取php项目代码，并打包成镜像，上传到harbor仓库。

```
[root@harborserver ~]# wget https://cn.wordpress.org/latest-
zh_CN.tar.gz
```

```
由于wordpress网站访问量过大，请提前尝试下载。


[root@harborserver ~]# tar xf latest-zh_CN.tar.gz

[root@harborserver ~]# ls
php-project
latest-zh_CN.tar.gz
wordpress #此为解压出来的目录



[root@harborserver ~]# mkdir php-project

[root@harborserver ~]# cp -r wordpress php-project/
[root@harborserver ~]# ls php-project/
wordpress

[root@harborserver ~]# ls php-project/wordpress/
index.php        wp-blog-header.php    wp-includes        wp-
settings.php
license.txt      wp-comments-post.php  wp-links-opml.php  wp-
signup.php
readme.html      wp-config-sample.php  wp-load.php        wp-
trackback.php
wp-activate.php  wp-content            wp-login.php
xmlrpc.php
wp-admin         wp-cron.php           wp-mail.php
```

将php项目代码打包成镜像

编辑Dockerfile文件

```
[root@harborserver ~]# cd php-project/
```

```
[root@harborserver php-project]# ls
wordpress

[root@harborserver php-project]# cd wordpress

[root@harborserver wordpress]# cat Dockerfile
FROM harbor.k8sonline.com.cn/library/nginx-php:v1
LABEL maintainer "k8sonline <admin@k8sonline.com.cn>"
RUN rm -rf /usr/share/nginx/html/*
ADD . /usr/share/nginx/html/
RUN chmod -R +x /usr/local/nginx/html
```

修改wordpress数据库连接文件

```
[root@harborserver php-project]# pwd
/root/php-project
[root@harborserver php-project]# ls
wordpress
[root@harborserver php-project]# cd wordpress/

[root@harborserver php-project]# touch index.html

[root@harborserver wordpress]# ls
Dockerfile   index.html
index.php        wp-blog-header.php    wp-includes        wp-
settings.php
license.txt      wp-comments-post.php  wp-links-opml.php  wp-
signup.php
readme.html      wp-config-sample.php  wp-load.php        wp-
trackback.php
wp-activate.php  wp-content            wp-login.php
xmlrpc.php
wp-admin         wp-cron.php           wp-mail.php
[root@harborserver wordpress]# cp wp-config-sample.php wp-
config.php
```

```
[root@harborserver wordpress]# ls
index.php        wp-blog-header.php    wp-cron.php        wp-
mail.php
license.txt      wp-comments-post.php  wp-includes        wp-
settings.php
readme.html      wp-config.php         wp-links-opml.php  wp-
signup.php
wp-activate.php  wp-config-sample.php  wp-load.php        wp-
trackback.php
wp-admin         wp-content            wp-login.php
xmlrpc.php



修改wordpress连接数据库配置文件
[root@harborserver wordpress]# cat wp-config.php
<?php
/**
 * WordPress基础配置文件。
 *
 * 这个文件被安装程序用于自动生成wp-config.php配置文件，
 * 您可以不使用网站，您需要手动复制这个文件，
 * 并重命名为“wp-config.php”，然后填入相关信息。
 *
 * 本文件包含以下配置选项：
 *
 * * MySQL设置
 * * 密钥
 * * 数据库表名前缀
 * * ABSPATH
 *
 * @link https://codex.wordpress.org/zh-
cn:%E7%BC%96%E8%BE%91_wp-config.php
 *
 * @package WordPress
 */

// ** MySQL 设置 - 具体信息来自您正在使用的主机 ** //
/** WordPress数据库的名称 */
define( 'DB_NAME', 'wordpress' );
/** WordPress数据库的名称为wordpress */
/** MySQL数据库用户名 */
define( 'DB_USER', 'root' );
/** WordPress数据库主机访问用户名为root */

/** MySQL数据库密码 */
define( 'DB_PASSWORD', '123456' );
/** WordPress数据库主机访问密码为123456，与mysql资源清单中定义的保持一致 */

/** MySQL主机 */
define( 'DB_HOST', 'db-0.phpmysql.phpproject' );
/** WordPress数据库主机连接名称，一定要验证后再填写 */

/** 创建数据表时默认的文字编码 */
define( 'DB_CHARSET', 'utf8' );

/** 数据库整理类型。如不确定请勿更改 */
define( 'DB_COLLATE', '' );

/**#@+
 * 身份认证密钥与盐。
 *
 * 修改为任意独一无二的字串！
 * 或者直接访问{@link https://api.wordpress.org/secret-key/1.1/salt/ WordPress.org密钥生成服务}
 * 任何修改都会导致所有cookies失效，所有用户将必须重新登录。
 *
 * @since 2.6.0
 */
define( 'AUTH_KEY',         'put your unique phrase here' );
define( 'SECURE_AUTH_KEY',  'put your unique phrase here' );
define( 'LOGGED_IN_KEY',    'put your unique phrase here' );
define( 'NONCE_KEY',        'put your unique phrase here' );
define( 'AUTH_SALT',        'put your unique phrase here' );
define( 'SECURE_AUTH_SALT', 'put your unique phrase here' );
define( 'LOGGED_IN_SALT',   'put your unique phrase here' );
define( 'NONCE_SALT',       'put your unique phrase here' );

/**#@-*/

/**
 * WordPress数据表前缀。
 *
 * 如果您有在同一数据库内安装多个WordPress的需求，请为每个WordPress设置
 * 不同的数据表前缀。前缀名只能为数字、字母加下划线。
 */
$table_prefix = 'wp_';

/**
 * 开发者专用：WordPress调试模式。
 *
 * 将这个值改为true，WordPress将显示所有用于开发的提示。
 * 强烈建议插件开发者在开发环境中启用WP_DEBUG。
 *
 * 要获取其他能用于调试的信息，请访问Codex。
 *
 * @link https://codex.wordpress.org/Debugging_in_WordPress
 */
define('WP_DEBUG', false);

/* 好了！请不要再继续编辑。请保存本文件。使用愉快！ */

/** WordPress目录的绝对路径。 */
if ( ! defined( 'ABSPATH' ) ) {
        define( 'ABSPATH', dirname( __FILE__ ) . '/' );
}

/** 设置WordPress变量和包含文件。 */
require_once( ABSPATH . 'wp-settings.php' );
```

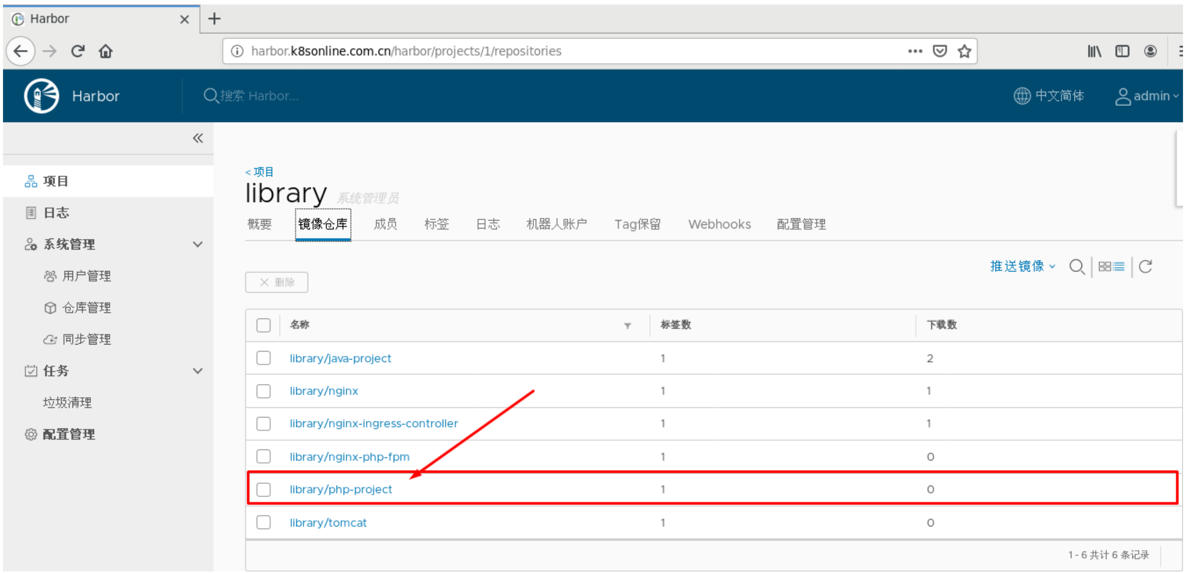
构建项目镜像

```
[root@harborserver php-project]# docker build -t harbor.k8sonline.com.cn/library/php-project:v1 .

Sending build context to Docker daemon  47.73MB
Step 1/4 : FROM harbor.k8sonline.com.cn/library/nginx-php-fpm:latest
 ---> bb49f1636b1a
```

```
Step 2/4 : LABEL maintainer "k8sonline
<admin@k8sonline.com.cn>"
 ---> Running in 180c58c1d178
Removing intermediate container 180c58c1d178
 ---> db07b3348cb5
Step 3/4 : RUN rm -rf /usr/share/nginx/html/*
 ---> Running in ddc6c85a62b6
Removing intermediate container ddc6c85a62b6
 ---> 28feee77d7a5
Step 4/4 : ADD wordpress/* /usr/share/nginx/html/
 ---> 4cb4c0e22e2f
Successfully built 4cb4c0e22e2f
Successfully tagged harbor.k8sonline.com.cn/library/php-
project:v1
```

上传已生成的php代码镜像

```
[root@harborserver php-project]# docker push
harbor.k8sonline.com.cn/library/php-project:v1
```

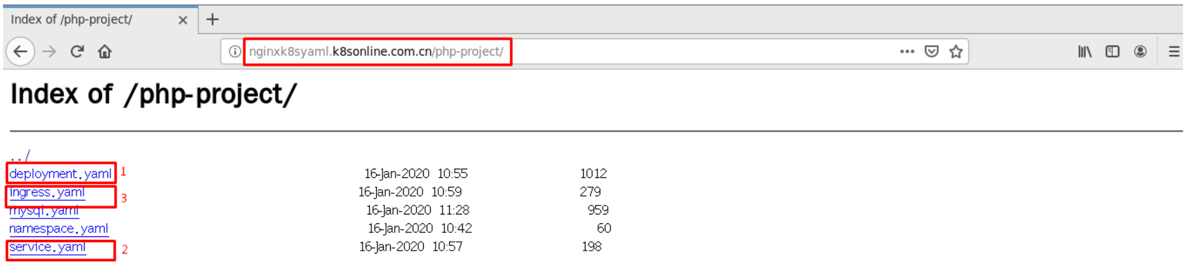


library

	名称	标签数	下载数
□	library/java-project	1	2
□	library/nginx	1	1
□	library/nginx-ingress-controller	1	1
□	library/nginx-php-fpm	1	0
□	library/php-project	1	0
□	library/tomcat	1	0

1 - 6 共计 6 条记录

四、项目部署



Index of /php-project/

```
../
deployment.yaml                16-Jan-2020 10:55      1012
ingress.yaml                   16-Jan-2020 10:59       279
mysql.yaml                     16-Jan-2020 11:28       959
namespace.yaml                 16-Jan-2020 10:42        60
service.yaml                   16-Jan-2020 10:57       198
```

部署php-project

```
[root@master1 ~]# kubectl apply -f
http://nginxk8syaml.k8sonline.com.cn/php-
project/deployment.yaml
deployment.apps/php-project created
```

部署php-project service

```
[root@master1 ~]# kubectl apply -f
http://nginxk8syaml.k8sonline.com.cn/php-project/service.yaml
service/php-project created
```

部署php-project ingress

```
[root@master1 ~]# kubectl apply -f
http://nginxk8syaml.k8sonline.com.cn/php-project/ingress.yaml
ingress.extensions/php-project created
```

验证创建的资源

```
[root@master1 ~]# kubectl get deployment.apps -n phpproject
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
php-project    2/2     2            2           2m

[root@master1 ~]# kubectl get svc -n phpproject
NAME           TYPE        CLUSTER-IP      EXTERNAL-IP
PORT(S)    AGE
php-project    ClusterIP   10.96.192.149   <none>
 80/TCP     109s
phpmysql       ClusterIP   None            <none>
 3306/TCP   36m

[root@master1 ~]# kubectl get endpoints -n phpproject
NAME           ENDPOINTS                             AGE
php-project    172.16.123.28:80,172.16.215.23:80     77s
phpmysql       172.16.215.21:3306                    50m


[root@master1 ~]# kubectl get pods -n phpproject
NAME                           READY   STATUS    RESTARTS
AGE
db-0                           1/1     Running   0
 36m
php-project-5688cb5cc9-nxcmv   1/1     Running   0
 2m48s
php-project-5688cb5cc9-xjx55   1/1     Running   0
 2m48s

[root@master1 ~]# kubectl get ingress -n phpproject
NAME           HOSTS                       ADDRESS   PORTS   AGE
```

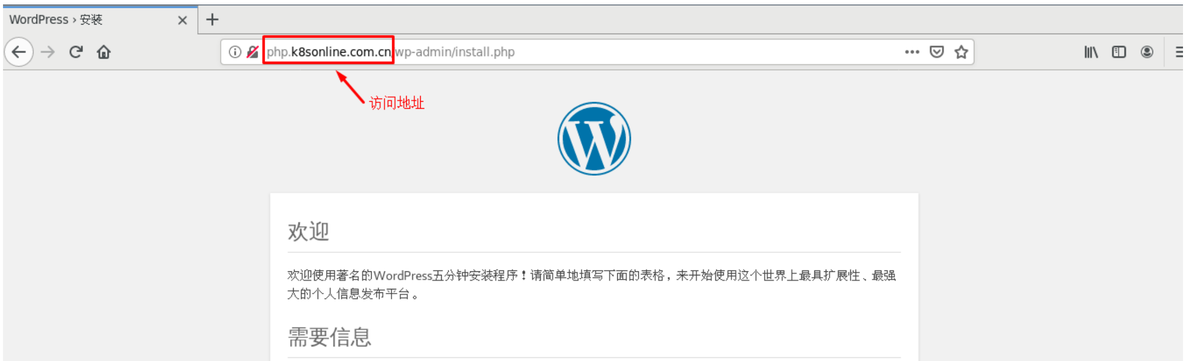
```
php-project   php.k8sonline.com.cn                 80      2m20s



[root@master1 ~]# kubectl describe ingress php-project -n phpproject
Name:             php-project
Namespace:        phpproject
Address:
Default backend:  default-http-backend:80 (<none>)
Rules:
  Host                  Path  Backends
  ----                  ----  --------
  php.k8sonline.com.cn
                        /   php-project:80 (<none>)
Annotations:
  kubectl.kubernetes.io/last-applied-configuration:
 {"apiVersion":"extensions/v1beta1","kind":"Ingress","metadata":{"annotations":{},"name":"php-project","namespace":"phpproject"},"spec":{"rules":[{"host":"php.k8sonline.com.cn","http":{"paths":[{"backend":{"serviceName":"php-project","servicePort":80},"path":"/"}]}}]}}

Events:
  Type    Reason  Age    From                      Message
  ----    ------  ----   ----                      -------
  Normal  CREATE  2m56s  nginx-ingress-controller  Ingress phpproject/php-project
```

五、php项目访问

> 在物理主机上访问

```
[root@master1 ~]# kubectl get pods -n ingress-nginx -o wide
NAME                                        READY   STATUS
 RESTARTS   AGE   IP                NODE    NOMINATED NODE
READINESS GATES
nginx-ingress-controller-5c6985f9cc-snqtt   1/1     Running
7           14h   192.168.122.102   work2   <none>
<none>
```

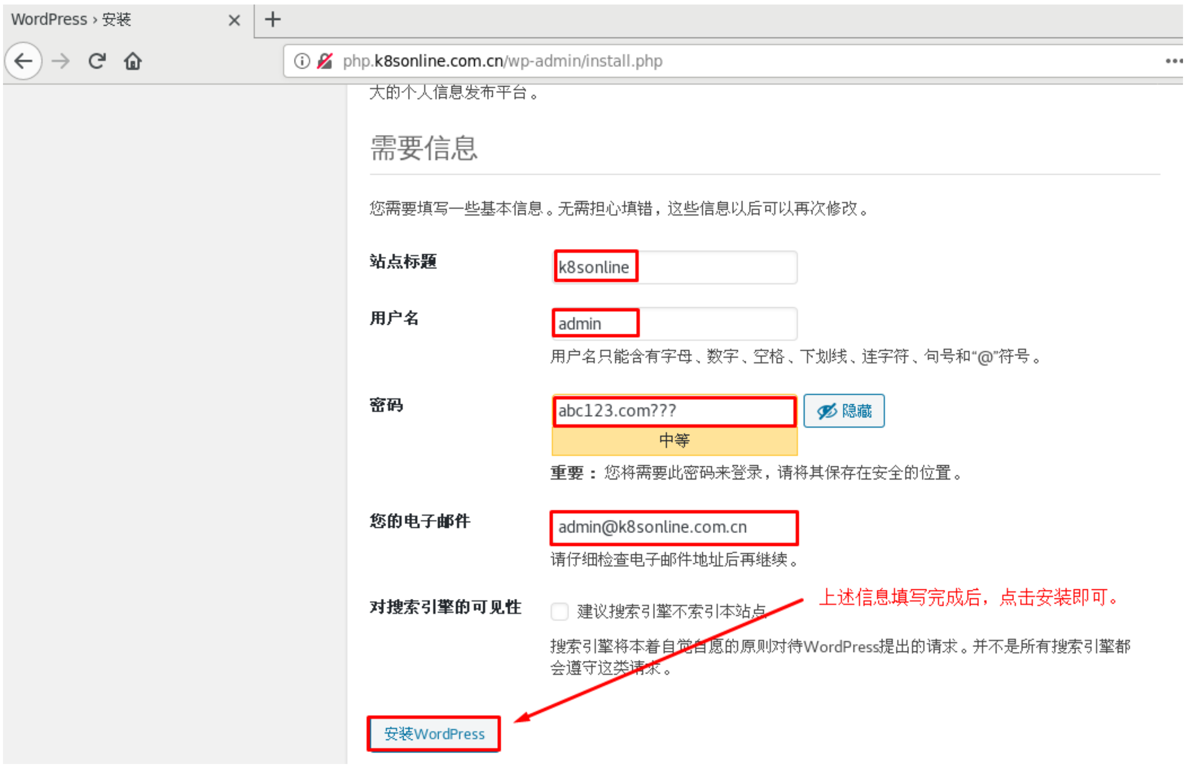
```
[root@bogon ~]# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4
localhost4.localdomain4
::1         localhost localhost.localdomain localhost6
localhost6.localdomain6
192.168.122.102 php.k8sonline.com.cn
```



欢迎

欢迎使用著名的WordPress五分钟安装程序！请简单地填写下面的表格，来开始使用这个世界上最具扩展性、最强大的个人信息发布平台。

需要信息



WordPress › 安装

php.k8sonline.com.cn/wp-admin/install.php

大的个人信息发布平台。

需要信息

您需要填写一些基本信息。无需担心填错，这些信息以后可以再次修改。

站点标题 k8sonline

用户名 admin

用户名只能含有字母、数字、空格、下划线、连字符、句号和“@”符号。

密码 abc123.com??? 隐藏

中等

重要：您将需要此密码来登录，请将其保存在安全的位置。

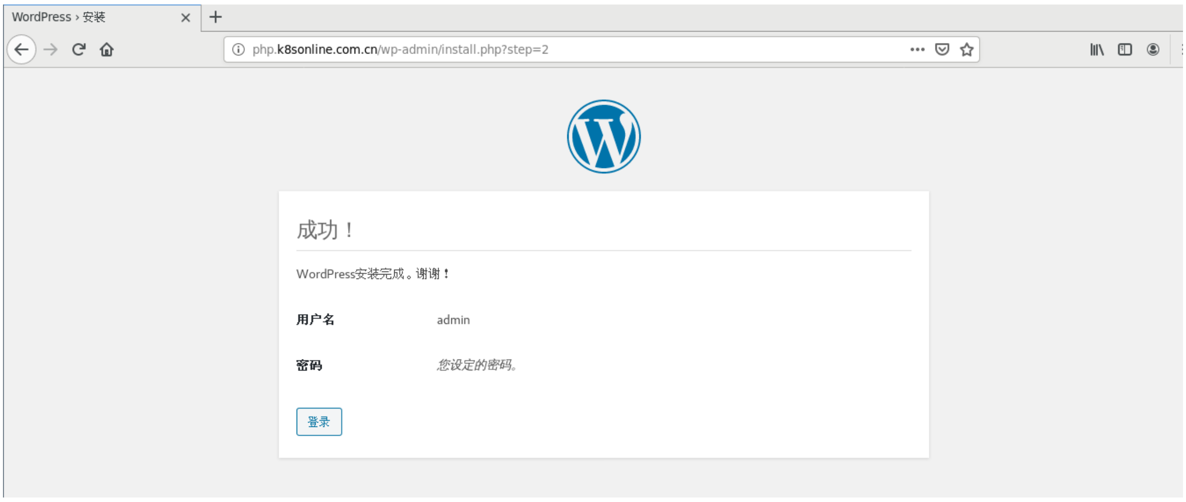
您的电子邮件 admin@k8sonline.com.cn

请仔细检查电子邮件地址后再继续。

对搜索引擎的可见性 □ 建议搜索引擎不索引本站点

搜索引擎将本着自觉自愿的原则对待WordPress提出的请求。并不是所有搜索引擎都会遵守这类请求。

上述信息填写完成后，点击安装即可。



WordPress › 安装

php.k8sonline.com.cn/wp-admin/install.php?step=2

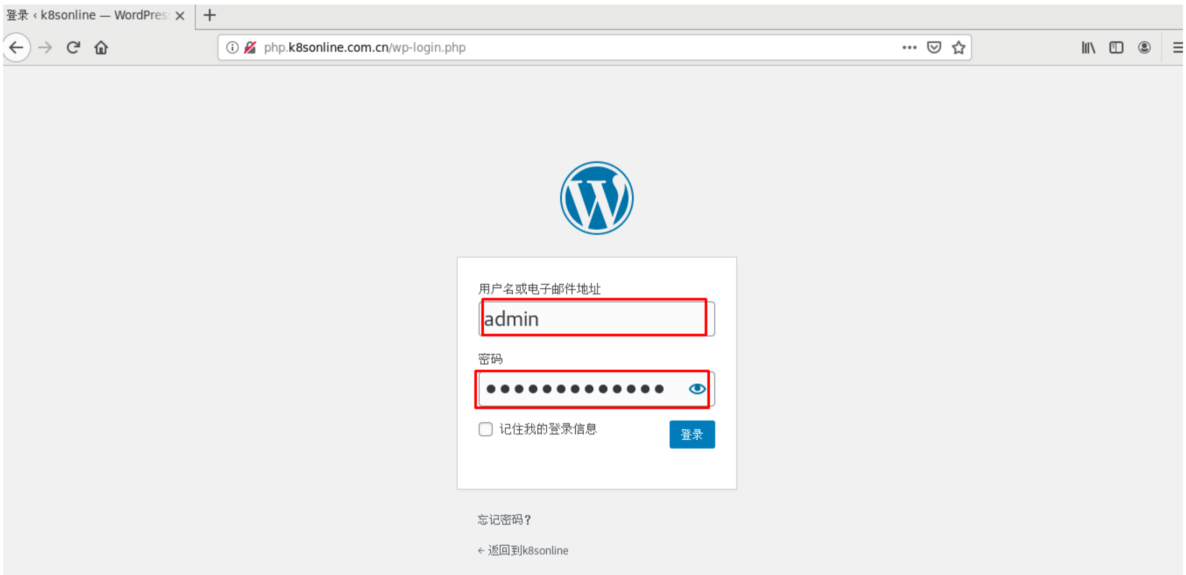
成功！

WordPress安装完成。谢谢！

用户名 admin

密码 *您设定的密码。*

```
查看NFS服务器数据库目录中是否有文件存在
[root@nfsserver ~]# ls /vdb/k8spublic/phpproject-mysql-data-
db-0-pvc-b8b3d6cf-7c06-4834-bf94-39d403bcf726/wordpress/
db.opt                    wp_options.frm    wp_termmeta.ibd
   wp_usermeta.frm
wp_commentmeta.frm  wp_options.ibd
wp_term_relationships.frm  wp_usermeta.ibd
wp_commentmeta.ibd  wp_postmeta.frm
 wp_term_relationships.ibd  wp_users.frm
wp_comments.frm     wp_postmeta.ibd  wp_terms.frm
  wp_users.ibd
wp_comments.ibd     wp_posts.frm     wp_terms.ibd
wp_links.frm        wp_posts.ibd     wp_term_taxonomy.frm
wp_links.ibd        wp_termmeta.frm  wp_term_taxonomy.ibd
```



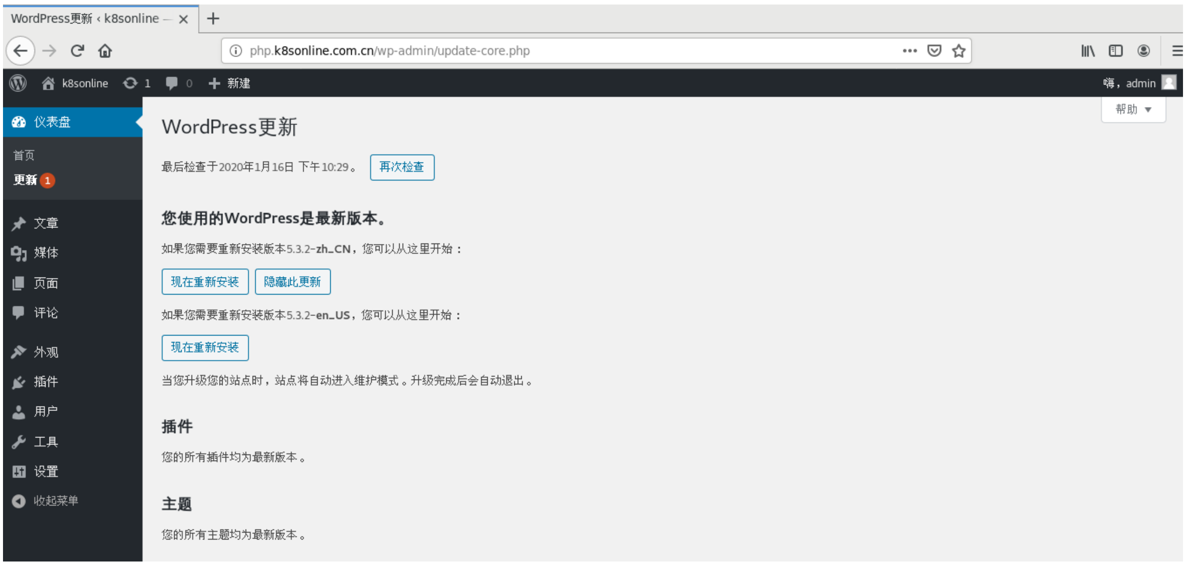
登录 ‹ k8sonline — WordPres

php.k8sonline.com.cn/wp-login.php

用户名或电子邮件地址: admin

密码: ●●●●●●●●●●●●●

□ 记住我的登录信息



WordPress更新

最后检查于2020年1月16日 下午10:29。 再次检查

您使用的WordPress是最新版本。

如果您需要重新安装版本5.3.2–**zh_CN**，您可以从这里开始：

现在重新安装 隐藏此更新

如果您需要重新安装版本5.3.2–**en_US**，您可以从这里开始：

现在重新安装

当您升级您的站点时，站点将自动进入维护模式。升级完成后会自动退出。

插件

您的所有插件均为最新版本。

主题

您的所有主题均为最新版本。