## **1. 服务器部署步骤**

1. 购买自己的域名
2. 域名备案
3. 购买服务器
4. 配置服务器应用环境
5. 安装配置服务器
6. 项目远程部署和发布与更新

### **1.2 购买域名**

* [腾讯云](https://dnspod.cloud.tencent.com/)
* [阿里云](https://wanwang.aliyun.com/)
* [百度云](https://cloud.baidu.com/product/bcd.html)
* [爱名网](https://www.22.cn/)
* [godaddy](https://sg.godaddy.com/)

### **1.3 云主机**

* [阿里云 ECS](https://www.aliyun.com/)
* [亚马逊 AWS](https://aws.amazon.com/cn)
* [百度云](https://cloud.baidu.com/)

### **1.4 购买阿里云**

* [选择配置](https://ecs-buy.aliyun.com/wizard/" \l "/postpay/cn-beijing)
* 镜像 Ubuntu 16.04 64位

### **1.5 备案**

* [阿里云备案](https://beian.aliyun.com/)
* [备案服务号管理](https://bsn.console.aliyun.com/" \l "/bsnManagement)

## **2. 服务器**

* [Xshell4](http://img.zhufengpeixun.cn/Xshell4.zip)
* 使用git bash而非git cmd

### **2.1 连接服务器**

ssh root@47.104.75.43

### **2.2 创建用户**

adduser zhufeng

### **2.3 赋予权限**

gpasswd命令是Linux下工作组文件/etc/group和/etc/gshadow管理工具。

* -a: 添加用户到组
* -d: 从组删除用户

gpasswd -a zhufeng sudo

### **2.4 添加sudo权限**

* Linux用户配置sudo权限visudo,如果你用visudo来编辑这个文件，那么它会帮你自动做很多事情，比如说语法检查，加锁防止别人同时修改这个文件等等

sudo visudo

vi /etc/sudoers

visudo其实是打开/etc/sudoers

增加以下内容

# User privilege specification

zhufeng ALL=(ALL:ALL) ALL

* 1 "From ALL hosts", zhufeng 从任何机器登录，都可以应用接下来的规则
* 2 "Run As ALL User", zhufeng"可以以任何用户的身份运行一些命令
* 3 "Run As All Groups", zhufeng"可以以任何用户组的身份运行一些命令
* 4 前面的规定适用于任何命令

zhufeng这个用户可以从任何机器登录，以任何用户和用户组的身份运行任何命令。 保存并退出

### **2.5 SSH无密码登录**

ssh 公钥认证是ssh认证的方式之一。通过公钥认证可实现ssh免密码登陆，git的ssh方式也是通过公钥进行认证的。

#### **2.5.1 本地生成公钥和私钥**

ssh-keygen --help

cd ~/.ssh

ssh-keygen -t rsa -b 4096

* -t 指定加密方式
* -b 字节数

#### **2.5.2 开启ssh代理**

eval "$(ssh-agent -s)"

ssh-add ~/.ssh/id\_rsa

#### **2.5.3 服务器配置**

ssh-keygen -t rsa -b 4096eval "$(ssh-agent -s)"

ssh-add ~/.ssh/id\_rsa

#### **2.5.4 把本地的公钥上传到服务器授权文件中**

vi ~/.ssh/authorized\_keys

chmod 600 ~/.ssh/authorized\_keys

service ssh restart

### **2.6 安装软件**

apt-**get** update

apt-**get** install wget curl git -y

### **2.7 安装node**

* [nvm](https://github.com/creationix/nvm/blob/master/README.md)

wget -qO- https://raw.githubusercontent.com/creationix/nvm/v0.33.11/install.sh | bash

. /root/.bashrc

nvm install stable

node -v

npm i cnpm -g

npm i nrm -g

### **2.8 编写node程序**

**var** http = require('http');

**const** hostname = '127.0.0.1';**const** port = 3000;

**const** server = http.createServer((req, res) => {

res.statusCode = 200;

res.setHeader('Content-Type', 'text/plain');

res.end('Hello World\n');

});

server.listen(port, () => {

console.log(`Server running at http://${hostname}:${port}/`);

});

### **2.9 启动程序**

强大的进程管理器，进程异常退出时pm2会尝试重启

cnpm install pm2 -g

用pm2启动node |命令|用途| |:----|:----| |pm2 start blog.js --name "blog"|启动应用| |pm2 list|查看所有应用| |pm2 restart crawl|重启应用| |pm2 stop crawl|停止应用| |pm2 delete crawl|删除应用|

pm2 start blog.js --name "blog"

### **2.10 nginx**

* Nginx是一个高性能的HTTP和反向代理服务器

#### **2.10.1 安装**

apt-**get** install nginx -y

#### **2.10.2 nginx命令**

| **名称** | **命令** |
| --- | --- |
| 启动nginx | nginx -c /etc/nginx/nginx.conf |
| 关闭 nginx | nginx -s stop |
| 重读配置文件 | nginx -s reload / kill -HUP nginx |
| 常用命令 | service nginx {start | stop | status | restart | reload | configtest | } |

#### **2.10.3 nginx配置**

cd /etc/nginx/sites-enabled

vi blog.conf

upstream blog{

server 127.0.0.1:3000;

}

server {

listen 80;

server\_name 127.0.0.1;

location / {

proxy\_pass http://blog;

}

}

#### **2.10.4 重启nginx**

nginx -s reload

## **3. Docker 是什么**

* Docker 是一个开源的应用容器引擎，让开发者可以打包他们的应用以及依赖包到一个可移植的镜像中，然后发布到任何流行的 Linux或Windows 机器上，也可以实现虚拟化。容器是完全使用沙箱机制，相互之间不会有任何接口

### **3.1 docker安装**

* docker分为企业版(EE)和社区版(CE)
* [docker-ce](https://docs.docker.com/install/linux/docker-ce/centos/)
* [hub.docker](https://hub.docker.com/)

### **3.2 安装**

yum install -y yum-utils device-mapper-persistent-data lvm2

yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo

yum install docker-ce docker-ce-cli containerd.io

### **3.3 启动**

systemctl start docker

### **3.4 查看docker版本**

$ docker version

$ docker info

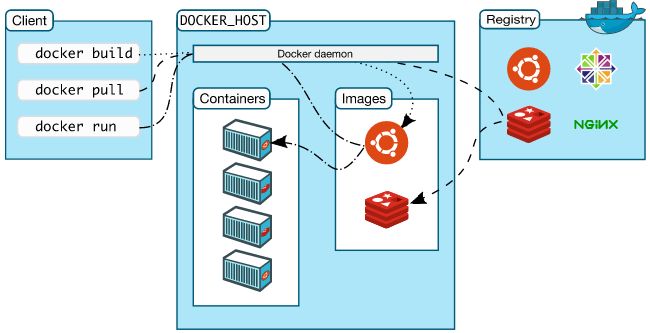
### **3.5 卸载**

docker info

yum remove docker

rm -rf /**var**/lib/docker

### **3.6 Docker架构**



### **3.7 阿里云加速**

* [镜像仓库](https://dev.aliyun.com/search.html)
* [镜像加速器](https://cr.console.aliyun.com/cn-hangzhou/instances/mirrors)

mkdir -p /etc/docker

tee /etc/docker/daemon.json <<-'EOF'

{

"registry-mirrors": ["https://fwvjnv59.mirror.aliyuncs.com"]

}

EOF

# 重载所有修改过的配置文件

systemctl daemon-reload

systemctl restart docker

启动服务

docker run ubuntu /bin/echo "Hello world"

* docker: Docker 的二进制执行文件
* run:与前面的 docker 组合来运行一个容器
* ubuntu指定要运行的镜像，Docker首先从本地主机上查找镜像是否存在，如果不存在，Docker 就会从镜像仓库 Docker Hub 下载公共镜像
* /bin/echo "Hello world": 在启动的容器里执行的命令

### **3.8 启动node服务**

#### **3.8.1 Dockerfile**

FROM node

COPY ./app /app

WORKDIR /app

RUN npm install

EXPOSE 3000

* FROM 表示该镜像继承的镜像 :表示标签
* COPY 是将当前目录下的app目录下面的文件都拷贝到image里的/app目录中
* WORKDIR 指定工作路径，类似于执行 cd 命令
* RUN npm install 在/app目录下安装依赖，安装后的依赖也会打包到image目录中
* EXPOSE 暴露3000端口，允许外部连接这个端口

#### **3.8.2 创建image**

docker build -t zhufengblog .

* -t用来指定image镜像的名称，后面还可以加冒号指定标签，如果不指定默认就是latest
* . 表示Dockerfile文件的所有路径,.就表示当前路径

#### **3.8.3 使用新的镜像运行容器**

docker container run -p 3333:3000 -it zhufengblog /bin/bash

npm start

* -p 参数是将容器的3000端口映射为本机的3333端口
* -it 参数是将容器的shell容器映射为当前的shell,在本机容器中执行的命令都会发送到容器当中执行
* zhufengblog image的名称
* /bin/bash 容器启动后执行的第一个命令,这里是启动了bash容器以便执行脚本

#### **3.8.4 CMD**

Dockerfile

+ CMD npm start

重新制作镜像

docker build -t zhufengblog .

* RUN命令在 image 文件的构建阶段执行，执行结果都会打包进入 image 文件
* CMD命令则是在容器启动后执行
* 一个 Dockerfile 可以包含多个RUN命令，但是只能有一个CMD命令
* 指定了CMD命令以后，docker container run命令就不能附加命令了(比如前面的/bin/bash),否则它会覆盖CMD命令