Website deploy on CentOS 8 - Huawei Cloud

Website deploy on CentOS 8 - Huawei Cloud

- 系统配置检查
- 开发环境准备
  - 安装 Nginx 1.20
    - 上传 Nginx bin 文件
    - 安装相关依赖
    - 默认配置启动 Nginx
    - 配置 Nginx
    - 无人值守的 nohup 使用
  - 安装 Git 2.35
    - 上传 Git bin 文件
    - 安装依赖
    - 手动安装 Git
  - 配置 Python 3.6.8
    - 直接 pip 安装包
    - 尝试启动后端 Django 项目
  - 安装 npm
    - yum 安装不可行
    - 手动上传 bin
    - 更新环境、换源、安装相关依赖
    - 尝试启动 vue2 项目
- 试运行 NB-vue
  - Logs
    - 4.17 配置更新

系统配置检查

检查当前华为云 CentOS 8 是否开启公网访问（ip无法直接访问）：

```
cat /etc/centos-release
ifconifg
vi /etc/sysconfig/network-scripts/ifcfg-eth0
vi /etc/sysconfig/network-scripts/ifcfg-eth1
vi /etc/sysconfig/network-scripts/ifcfg-eth2
vi /etc/sysconfig/network-scripts/ifcfg-eth3
yum update
```

经过检查已经开启，另外有安装 gcc 等简单工具链，由于体积原因采取最小、最简洁安装，此时需要检查各项服务是否开启、各项配置是否设置。

在公网服务器的配置当中，**最重要的当属于 Nginx 反向代理的设置**，这部分的内容在下面作为说明文档进行详细记录。

开发环境准备

安装 Nginx 1.20

由于 yum 的各种原因，只能使用下载的 `.tar.gz` 压缩包，并手动 build 出 Nginx 的二进制文件。

上传 Nginx bin 文件

在 Nginx 官网下载 `nginx-1.20.2.tar.gz` 后上传到 root 用户 `~/` 目录下：

```
scp ./nginx-1.20.2.tar.gz root@122.9.32.180:~/
```

解压到 `/usr/local/` 目录下：

```
tar -zxvf  nginx-1.20.2.tar.gz
mv ./nginx-1.20.2 /usr/local/
```

安装相关依赖

安装 Nginx 所需要的 pcre 依赖：

```
yum install -y pcre pcre-devel
```

此时发现报错，原因是 CentOS 8 在 2021.12.31停止了服务，会出现如下报错：

```
Repository extras is listed more than once in the configuration
Repository centosplus is listed more than once in the
configuration
Repository PowerTools is listed more than once in the
configuration
Repository AppStream is listed more than once in the
configuration
CentOS-8 - AppStream
                                    3.3 kB/s | 394  B     00:00
Errors during downloading metadata for repository 'AppStream':
  - Status code: 404 for https://repo.huaweicloud.com/centos-
vault/centos-
vault/centos/8/AppStream/x86_64/os/repodata/repomd.xml (IP:
111.206.179.20)
Error: Failed to download metadata for repo 'AppStream': Cannot
download repomd.xml: Cannot download repodata/repomd.xml: All
mirrors were tried
```

此时需要把默认源的 cache 文件转移到新的目录，利用中国镜像 CentOS 源重新缓存所有的软件包地址：

```
cd /etc/yum.repos.d/
mkdir bak
mv *.repo bak
wget -O /etc/yum.repos.d/CentOS-Base.repo
https://mirrors.aliyun.com/repo/Centos-vault-8.5.2111.repo
yum clean all
yum makecache
```

更新官方源，防止国内源无法使用

```

```

```
[root@2022-buaa-bj-8 yum.repos.d]# vi CentOS-Linux-BaseOS.repo

[baseos]
name=CentOS Linux $releasever - BaseOS
#mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=BaseOS&infra=$infra
#baseurl=http://mirror.centos.org/$contentdir/$releasever/BaseOS/$basearch/os/
baseurl=https://vault.centos.org/centos/$releasever/BaseOS/$basearch/os/
gpgcheck=1
enabled=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-centosofficial


[root@2022-buaa-bj-8 yum.repos.d]# vi CentOS-Linux-AppStream.repo

[appstream]
name=CentOS Linux $releasever - AppStream
#mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=AppStream&infra=$infra
#baseurl=http://mirror.centos.org/$contentdir/$releasever/AppStream/$basearch/os/
baseurl=https://vault.centos.org/centos/$releasever/AppStream/$basearch/os/
gpgcheck=1
enabled=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-centosofficial
```

下一步正常安装所有需要的软件包，顺便升级一下系统的一些小组件：

```
yum install -y pcre pcre-devel zlib zlib-devel
yum update
```

默认配置启动 Nginx

进行安装配置，首先使用默认配置：

```
./configure
make
make install
cd ../nginx/sbin/
./nginx
```

这个时候应该已经启动了 Nginx 服务，可以使用下列命令行查看是否已经启动：

```
ps -ef | grep nginx
```

获得以下结果，证明已经启动：

```
root       21135       1  0 12:16 ?        00:00:00 nginx: master process ./nginx
nobody     21136   21135  0 12:16 ?        00:00:00 nginx: worker process
root       21203   21170  0 12:28 pts/0    00:00:00 grep --color=auto nginx
```

此时直接从公网中访问服务器的 ip，可以看到如下界面：

将 Nginx 加入系统 root 环境变量：

```
vim /etc/profile
```

在打开的文件尾部加入 NGINX_HOME 变量：

```
export NGINX_HOME=/usr/local/nginx
export PATH=$NGINX_HOME/sbin:$PATH
```

刷新变量，系统变量 Path 被添加：

```
source /etc/profile
```

配置 Nginx

```
cd /usr/local/nginx/conf
vim nginx.conf
```

需要哪个server，你就加一个，监听和host要同步加。可以在 location 关键字内，添加一个键 `proxy_pass` ，值可以写任何主机地址，例如：`127.0.0.1:xxxx`，`localhost:xxxx`。

有多种方式启动前后端，不只是靠 Nginx 代理。例如在 Django 后端直接启动 vue，或者 vue、Django 同时启动，但在 config 中手写路由。

修改好了以后，不要忘记刷新 Nginx 代理：

```
nginx -s reload
```

在 Nginx 启动之前，要启动所有的被反向代理服务，且挂在后台。此处给一个例子（但注意，这个例子用于测试，不用于正式服务）：

```
python ./backend/src/mysite/manage.py runserver (8000) &

cd ./frontend/
npm run dev &

```

此时，你就会获得一个在后台运行的服务器，但不代表可以一直在后台运行。经过测试，一旦退出登录，由任何角色用户启动的进程貌似会被挂起（暂停）。因此请使用 无人值守的 nohup 使用 配置服务方式进行部署。

每一次更新代码的时候，Django 后端不需要重新启动，vue 前端建议重新启动。

当你发现新的服务启动在了非监听端口上，你需要 **强制 kill 掉后台的进程**，(例如强制关闭 npm 有关进程)：

```
kill -9 $(pidof npm)
# or
kill -9 $(ps -w | grep npm | awk '$0 !~/grep/ {print $1}')

# 还可以先 top，浏览后锁定 PID，直接 kill 掉
```

无人值守的 nohup 使用

无人值守的设置，即将服务器的后端一直放在 CentOS 的后台运行。我们可以使用以下样例指令进行服务部署：

```
yum install coreutils # 安装 nohup
nohup python3 backend/src/mysite/manage.py runserver 8000 &
```

注意，nohup 无人值守会将记录下的 output 重定向到命令的启动目录，例如示例中的指令会在 backend 的同级目录下产生 `nohup.out` 文件。

这样一来，每一次在遇到前后端的任何问题时，你可以选择打印 `nohup.out` 文件的倒数几行进行简单的调试。但依然不推荐在服务器上直接 debug，**即请在开发端进行充分测试后，再上传到服务器，谢谢！**

如果对 `nohup` 有任何问题，可以使用 `nohup --help` 查看帮助。

安装 Git 2.35

与上面原因相同，需要手动 build 出 git 的二进制文件。

上传 Git bin 文件

在 Nginx 官网下载 `nginx-1.20.2.tar.gz` 后上传到 root 用户 `~/` 目录下：

```
scp ./git.tar.gz root@122.9.32.180:~/
```

解压到 `/usr/local/` 目录下：

```
tar -zxvf git-2.35.3.tar.gz
mv ./git-2.35.3 /usr/local/
```

安装依赖

一定记得安装依赖，要不然没有办法和 remote 链接：

```
yum install -y libcurl-devel
```

手动安装 Git

分为：配置安装路径，编译，安装 三步。

```
cd /usr/local/git-2.35.3
./configure --prefix=/usr/local/git
make
make install
```

将 Git 加入系统 root 环境变量：

```
vim /etc/profile
```

在打开的文件尾部加入 GIT_HOME 变量：

```
export GIT_HOME=/usr/local/git
export PATH=$GIT_HOME/bin:$PATH
```

刷新变量，系统变量 Path 被添加：

```
source /etc/profile
```

很遗憾，CentOS 8 华为云服务器无法访问到 `github.com`，这时候我们找到了一个解决方法：在 gitee.com 上注册账号，并开设 github.com 的镜像仓库。每一次需要更新代码的时候，首先从 gitee.com 上手动同步 github.com 上的代码，再在服务器上进行 `git pull` 操作，这样应该是部署代码更新最为直接的方式，且对于 Django 后端来说，不需要重新启动服务，即只需要一条同步代码的指令。

其余的方式只能通过本地的 **SFTP** 文件服务，或 `scp` 指令将本地的所有文件先打包成特定格式 (.zip, .tar, .tar.gz等)，再一起发上去，这就很鸡肋。

配置 Python 3.6.8

CentOS 8 最新安装的版本为 **Python 3.6.8**，不影响使用，不过习惯要改下，`python` 指令为 `python3`，`pip` 指令为 `pip3`。

直接 pip 安装包

```
pip3 install -r ~/Project/backend/requirements.txt
```

注意，没有启用虚拟环境。

尝试启动后端 Django 项目

Django 后端项目默认启用 8000 端口运行。

```
python3 ~/Project/backend/src/mysite/manage.py runserver (8000)
```

经过测试，没有产生任何问题。

安装 npm

直接安装 centOS 8 最新的 node.js 10.24.0-1 发现版本不够，只能手动下载。

yum 安装不可行

```
yum install npm
npm -v
npm install npm -g
```

！但你发现依然不行，在安装 `@vue/cli@5.0.4` 时候提示如下信息：

```
You are using Node v10.24.0, but this version of @vue/cli
requires Node ^12.0.0 || >= 14.0.0.
Please upgrade your Node version.
```

手动上传 bin

无奈之下只能再次去官网下载 bin 文件，并上传到服务器解压使用：

```
scp ./node-v16.14.2-linux-x64.tar.xz root@122.9.32.180:~/
```

CentOS 端进行两步解压获得 node-v16.14.2 安装目录，并移动到统一安装目录：

```
xz -d node-v16.14.2-linux-x64.tar.xz
tar xvf node-v16.14.2-linux-x64.tar
mv node-v16.14.2-linux-x64/ /usr/local/
```

将 Git 加入系统 root 环境变量：

```
vim /etc/profile
```

在打开的文件尾部加入 NODE_HOME 变量：

```
export NODE_HOME=/usr/local/node-v16.14.2-linux-x64
export PATH=$NODE_HOME/bin:$PATH
```

刷新变量，系统变量 Path 被添加：

```
source /etc/profile
```

更新环境、换源、安装相关依赖

```
npm install -g npm@8.7.0
npm set registry https://registry.npm.taobao.org/
npm install -g @vue/cli
```

查询 vue 脚手架工具版本，正常结果应显示为 `@vue/cli 5.0.4`：

```
vue --version
```

尝试启动 vue2 项目

尝试在 `~/Project/frontend` 目录下使用 vue2 工具进行 build 操作：

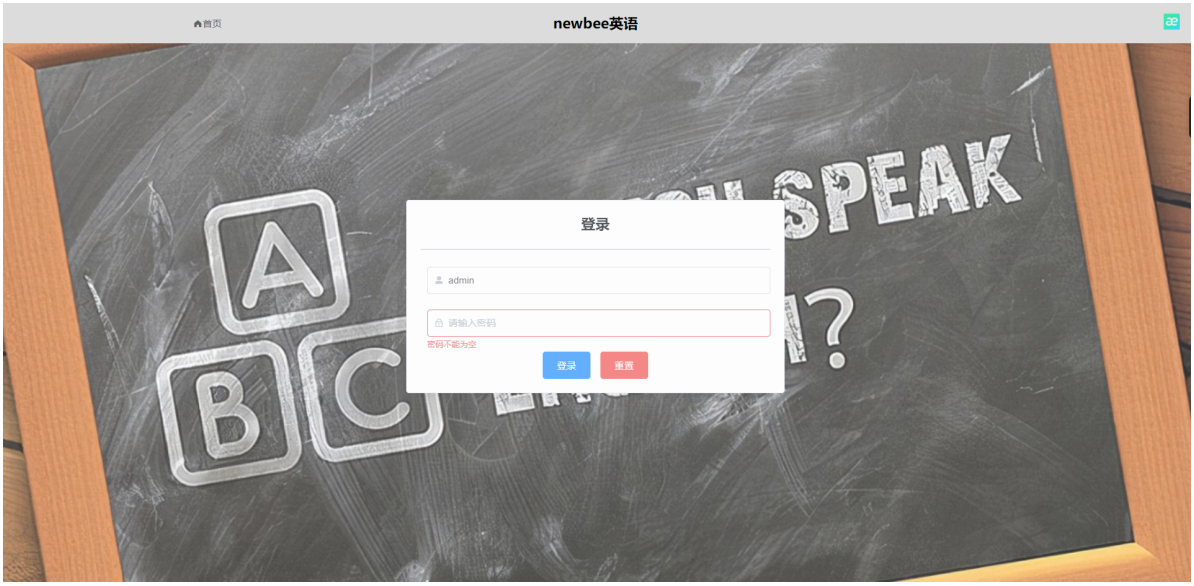
```
npm install
npm run build  # can skip
npm run dev
```

经过测试，可以正常运行前端！正常作为服务运行的时候，还请参阅 配置 Nginx。

试运行 NB-vue

至此，一个华为云 server 可以正常提供公网访问服务，下面是示例：

网址：NB-vue



直接监听 8000 端口，依靠 vue 和 django 的路由，不需要 Nginx 再写代理单独配置前后端的联系。

Logs

4.17 配置更新

在最新的服务器配置中，我们更改了前后端联合启动的方式，由 Django 后端启动代替双服务启动。因此 Nginx 启动服务中，我只监听了 8000 一个端口即可完成代理工作。

新增了使用 nohup 无人值守的服务配置，具体配置 **请参考** Nohup 无人值守 。