

Jenkins

部署

1.配置基础环境，安装docker

```
1 yum install -y yum-utils device-mapper-persistent-data lvm2
2 yum-config-manager --add-repo https://mirrors.aliyun.com/docker-
  ce/linux/centos/docker-ce.repo
3 sed -i 's+download.docker.com+mirrors.aliyun.com/docker-ce+'
  /etc/yum.repos.d/docker-ce.repo
4 yum makecache fast
5
6 selinuxdefcon 0
7 sed -i 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/selinux/config
8 if egrep "7.[0-9]" /etc/redhat-release &>/dev/null; then
9     systemctl stop firewalld
10    systemctl disable firewalld
11 fi
12 yum install -y iptables-services vim lrzsz zip wget net-tools unzip wget
13 systemctl enable iptables --now
14
15 yum -y install docker-ce
16 systemctl start docker &&systemctl enable docker
17
18 mkdir -p /etc/docker
19 tee /etc/docker/daemon.json <<- 'EOF'
20 {
21     "registry-mirrors": ["https://zd29wsn0.mirror.aliyuncs.com"]
22 }
23 EOF
24 systemctl daemon-reload
25 systemctl restart
26
27 if ! grep HISTTIMEFORMAT /etc/bashrc; then
28     echo 'export HISTTIMEFORMAT="%F %T `whoami` "' >> /etc/bashrc
29 fi
30 if ! grep "* soft nofile 65535" /etc/security/limits.conf &>/dev/null; then
31     cat >> /etc/security/limits.conf << EOF
32     * soft nofile 65535
33     * hard nofile 65535
34 EOF
35 fi
36 cat >> /etc/sysctl.conf << EOF
37 net.ipv4.tcp_syncookies = 1
38 net.ipv4.tcp_max_tw_buckets = 20480
39 net.ipv4.tcp_max_syn_backlog = 20480
40 net.core.netdev_max_backlog = 262144
41 net.ipv4.tcp_fin_timeout = 20
42 EOF
43 echo "0" > /proc/sys/vm/swappiness
44 sed -i '/SELINUX/{s/permissive/disabled/}' /etc/selinux/config
45 setenforce 0
```

2.安装docker-compose

```
1 curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-  
compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose  
2 chmod +x /usr/local/bin/docker-compose  
3 ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose  
4 docker-compose --version
```

3.编写jenkins的compose.yaml文件

```
1 cat >jenkins-compose.yaml <<EOF  
2 version: '3.8'  
3 services:  
4   jenkins:  
5     # lts 表示长期支持版本。  
6     # image: jenkins  
7     image: jenkins/jenkins:lts  
8     # web 端口为8080,代理端口为50000  
9     ports:  
10      - "8080:8080" # Jenkins web interface  
11      - "50000:50000" # Jenkins agents  
12     container_name: jenkins  
13     volumes:  
14       - jenkins_data:/var/jenkins_home # Jenkins data volume  
15       - jenkins_backup:/var/jenkins_backup # Jenkins backup volume  
16       - tools_path:/opt/tools # 后期存放一些插件环境  
17     # 设置重启规则,除非手工停止,否则一直保持启动状态  
18     restart: unless-stopped # Auto-restart unless manually stopped  
19  
20 volumes:  
21   jenkins_data:  
22   jenkins_backup:  
23   tools_path:  
24 EOF
```

3.1) 语法解释

```
1 # 声明 volumes 的好处 将数据卷的声明与服务的配置分开,使得文件结构更加清晰,方便阅读和  
   维护。拥有可重用性  
2  
3 # 如果要查看有哪些 volume,可以适用这个命令  
4 [root@hecs-131633 ~]# docker volume ls  
5 DRIVER      VOLUME NAME  
6 local       composes_jenkins_backup  
7 local       composes_jenkins_data  
8  
9 # 查看 volume 详细信息  
10 docker volume inspect composes_jenkins_data  
11 [  
12   {  
13     "CreatedAt": "2024-04-17T15:30:44+08:00",  
14     "Driver": "local",  
15     "Labels": {  
16       "com.docker.compose.project": "composes",  
17       "com.docker.compose.version": "1.29.2",  
18       "com.docker.compose.volume": "jenkins_data"  
19     },  
20   },  
21 ]
```

```

20 |         "Mountpoint": "/var/lib/docker/volumes/composes_jenkins_data/_data",
21 |         "Name": "composes_jenkins_data",
22 |         "Options": null,
23 |         "Scope": "local"
24 |     }
25 | ]

```

4.启动和关闭 jenkins 操作

```

1 | # 语法格式
2 | docker-compose -f compose.yaml 【文件路径，比如 /opt/composes/jenkins-
  | compose.yaml】 up -d【动作】
3 | # 用于首次启动服务。如果服务的容器尚未创建，up 命令会创建并启动容器
4 | docker-compose -f jenkins-compose.yaml up -d
5 |
6 | # 用于启动已经存在但是被停止的容器。它不会重新创建容器，也不会应用配置的更新
7 | docker-compose -f jenkins-compose.yaml start
8 |
9 | # 用于停止容器
10 | docker-compose -f jenkins-compose.yaml stop
11 |
12 | # 查看 docker-compose 中管理容器的状态
13 | docker-compose -f jenkins-compose.yaml ps
14 |
  | Name                                Command                                State
  |
  | Ports
  | -----
15 |
  | -----
16 | composes_jenkins_1  /usr/bin/tini -- /usr/loca ...      Up
  | 0.0.0.0:50000->50000/tcp, :::50000->50000/tcp
17 |
  | 0.0.0.0:8080->8080/tcp, :::8080->8080/tcp

```

5.访问jenkins web

访问地址是

http://ip:8080

首次访问jenkins web界面时，需要输入密码。获取密码的指令如下：

```

1 | docker exec -it jenkins cat /var/jenkins_home/secrets/initialAdminPassword
2 | 6cf1fcda25de433cbac895563b4aad0d

```

6.下载 Git 和 Maven

使用自动化流水线工具需要再 jenkins 服务器上安装 这两个插件，以用来拉取代码和进行打包构建

```

1 | yum install -y git maven
2 | which git
3 | which git

```

配置维护

1.设置 Jenkins 插件代理地址

1.1) 通过 Jenkins 界面设置

1. 登录到您的 Jenkins 界面。
2. 导航到 **Manage Jenkins > Manage Plugins > Advanced**。
3. 在 **Updates** 部分，找到 **Update Site** 输入框。
4. 替换现有的更新站点 URL 为国内镜像站点的 URL。例如：

```
1 | https://mirrors.tuna.tsinghua.edu.cn/jenkins/updates/update-center.json
2 | http://mirror.esuni.jp/jenkins/updates/update-center.json
3 | https://mirrors.aliyun.com/jenkins/updates/update-center.json
```

1.2) 通过 Linux 终端修改配置文件设置

- 1.查看容器存储卷，找到关联的目录位置

```
1 | [root@localhost opt]# docker inspect jenkins|grep -A 15 Mounts
2 |     "Mounts": [
3 |         {
4 |             "Type": "volume",
5 |             "Name": "compose_jenkins_backup",
6 |             "Source":
7 |             "/var/lib/docker/volumes/compose_jenkins_backup/_data",
8 |             "Destination": "/var/jenkins_backup",
9 |             "Driver": "local",
10 |             "Mode": "rw",
11 |             "RW": true,
12 |             "Propagation": ""
13 |         },
14 |         {
15 |             "Type": "volume",
16 |             "Name": "compose_jenkins_data",
17 |             "Source": "/var/lib/docker/volumes/compose_jenkins_data/_data",
18 |             "Destination": "/var/jenkins_home",
```

- 2.修改配置 hudson.model.UpdateCenter.xml

```
1 | vim
2 | /var/lib/docker/volumes/compose_jenkins_data/_data/hudson.model.UpdateCenter.xml
3 | <?xml version='1.1' encoding='UTF-8'?>
4 | <sites>
5 |   <site>
6 |     <id>default</id>
7 |     <url> https://mirrors.tuna.tsinghua.edu.cn/jenkins/updates/update-
8 | center.json</url> #更换为清华源地址
9 |   </site>
10 | </sites>
```

- 3.保存文件并重启 Jenkins 服务来应用更改

```

1 # 通过compose.yaml文件停止 jenkins 容器
2 [root@localhost composes]# docker-compose -f jenkins-compose.yaml stop
3 Stopping jenkins ... done
4 [root@localhost composes]# docker-compose -f jenkins-compose.yaml start
5 Starting jenkins ... done
6 [root@localhost composes]# docker-compose -f jenkins-compose.yaml ps
7   Name                                Command                                State
8   -----                                -----                                -----
9 jenkins    /sbin/tini -- /usr/local/b ...    Up                                0.0.0.0:50000-
    >50000/tcp, :::50000->50000/tcp, 0.0.0.0:8080->8080/tcp, :::8080->8080/tcp

```

2.安装插件

```

1 Blue Ocean                                https://plugins.jenkins.io/blueocean
2 Git                                        https://plugins.jenkins.io/git/
3 Build Pipeline                            https://plugins.jenkins.io/build-pipeline-plugin
4 Rebuilder                                https://plugins.jenkins.io/rebuild
5 Maven                                    https://plugins.jenkins.io/maven-plugin
6 Gradle
7 SSH Agent
8 SSH Plugin

```

2.1) Gradle 环境安装

1.下载 gradle 所需版本的文件

```

1 mkdir /opt/gradle
2 wget -P /opt/gradle https://services.gradle.org/distributions/gradle-7.2-bin.zip \
3 https://services.gradle.org/distributions/gradle-7.4.1-bin.zip \
4 https://services.gradle.org/distributions/gradle-7.5-bin.zip \
5 https://services.gradle.org/distributions/gradle-7.6-bin.zip

```

2查看 tools 数据卷的目录位置，将下载的 gradle文件 上传

```

1 [root@jenkins _data]# docker volume ls
2 DRIVER      VOLUME NAME
3 local      composes_tools_path
4 local      composes_jenkins_backup
5 local      composes_jenkins_data
6 [root@jenkins _data]# docker volume inspect composes_tools_path
7 [
8     {
9         "CreatedAt": "2024-04-21T22:56:50-04:00",
10        "Driver": "local",
11        "Labels": {
12            "com.docker.compose.project": "composes",
13            "com.docker.compose.version": "1.29.2",
14 [root@jenkins _data]# mv /opt/gradle/*
    /var/lib/docker/volumes/composes_tools_path/_data
15 [root@jenkins _data]# ll /var/lib/docker/volumes/composes_tools_path/_data
16 total 0
17 drwxr-xr-x. 5 root root 85 Feb  1 1980 gradle-7.2
18 drwxr-xr-x. 5 root root 85 Feb  1 1980 gradle-7.4.1
19 drwxr-xr-x. 5 root root 85 Feb  1 1980 gradle-7.5
20 drwxr-xr-x. 5 root root 85 Feb  1 1980 gradle-7.6

```

21 |

22 在 Jenkins web 界面上设置 gradle 的目录位置为 /opt/tools/gradle-7.2 即可

2.2) SDK 环境安装

1.下载SDK软件包 <https://dl.google.com/android/repository/sdk-tools-linux-4333796.zip>

2.解压并映射到 jenkins 容器内

```
1 [root@jenkins tools]# wget https://dl.google.com/android/repository/sdk-tools-  
linux-4333796.zip  
2 [root@jenkins tools]# cp sdk-tools-linux-4333796.zip  
/var/lib/docker/volumes/composes_tools_path/_data  
3 [root@jenkins _data]# unzip sdk-tools-linux-4333796.zip -d android-sdk  
4 [root@jenkins android-sdk]# cd android-sdk  
5 [root@jenkins android-sdk]# ls  
6 tools  
7 [root@jenkins android-sdk]# cd tools/bin  
8 [root@jenkins opt]# ./sdkmanager "build-tools;28.0.2" "platforms;android-27"  
"platform-tools" "ndk-bundle" "extras;android;m2repository"  
"extras;google;m2repository"  
"extras;m2repository;com;android;support;constraint;constraint-layout;1.0.2"  
"tools"  
9  
10 # 给jenkins 用户授权sdk  
11 [root@jenkins bin]# docker exec -it -uroot jenkins bash  
12 root@26cdd305980b:/# cd /opt/tools  
13 root@26cdd305980b:/opt/tools# chown -R jenkins:jenkins /opt/tools
```

3.web视图中选择左边系统管理 点击系统配置，在全局属性中配置SDK

key:ANDROID_HOME

值:/opt/tools/android-sdk

Jenkins 更新

在有些时候版本太旧，很多插件已经不支持的时候。我们需要通过更新来解决，以避免程序太老，存在程序漏洞，对生产有一定的危险

可以从官网下载新版本的jenkins 软件包，上传到服务器上，通过使用dockerfile 来对镜像进行修改。

[官网升级地址](#)，有讲解操作步骤。

```
1 vim Dockerfile  
2 #使用 jenkins/jenkins:lts 镜像作为基础镜像  
3 FROM jenkins/jenkins:lts  
4  
5 USER root  
6 # 删除原有的 jenkins.war  
7 RUN rm /usr/share/jenkins/jenkins.war  
8  
9 # 将宿主机的 jenkins.war 文件复制到镜像中的相应位置  
10 COPY jenkins.war /usr/share/jenkins/jenkins.war  
11
```

```

12 # 为了保证 jenkins.war 文件在容器中拥有正确的权限
13 RUN chown jenkins:jenkins /usr/share/jenkins/jenkins.war
14 USER jenkins
15
16 # 暴露 Jenkins web 接口和代理端口
17 EXPOSE 8080 50000
18
19 # 当容器启动时运行 Jenkins
20 ENTRYPOINT ["/sbin/tini", "--", "/usr/local/bin/jenkins.sh"]

```

运行dockerfile，来修改镜像内容jenkins.war包

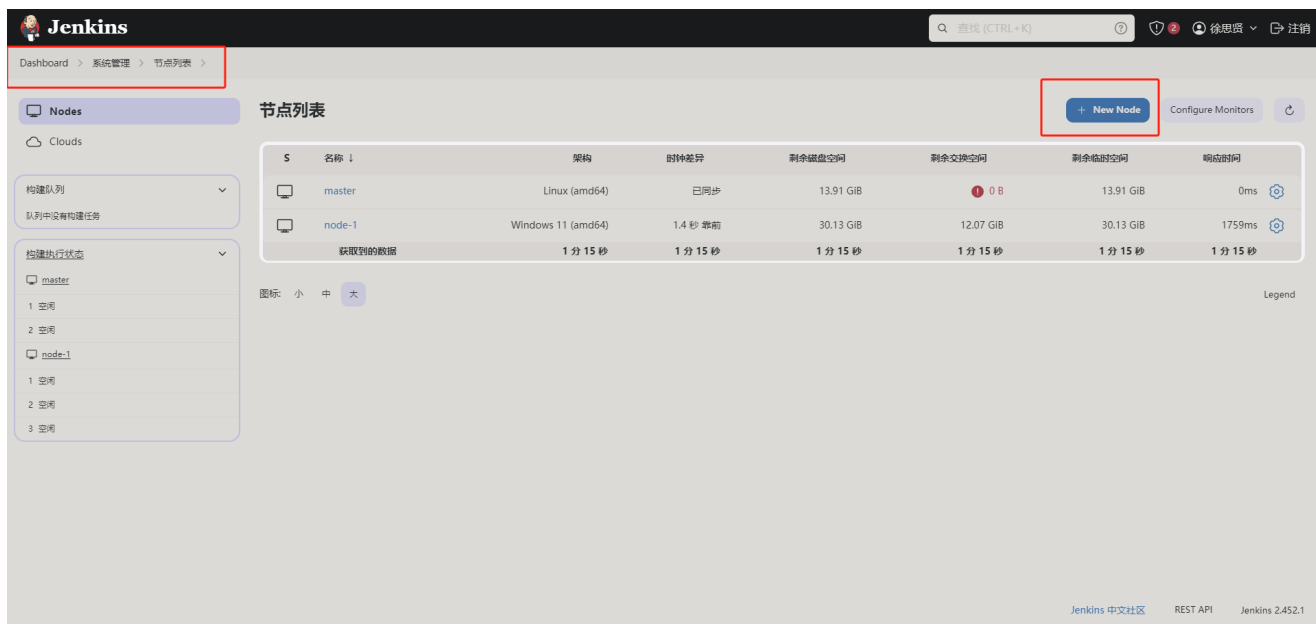
```

1 [root@jenkins opt]# docker build -t jenkins:latest .
2 # -t jenkins:latest . :这里 jenkins 是镜像的名称，latest是镜像的标签
3
4 [root@jenkins opt]# docker images
5 REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
6 jenkins              latest             77cdae6f12fd       3 minutes ago      628MB
7 jenkins/jenkins     lts                2a4bbe50c40b       2 years ago        441MB

```

jenkins添加代理

1、控制台界面打开系统管理-->节点列表-->新增节点



The screenshot shows the Jenkins web interface. The top navigation bar includes 'Dashboard', '系统管理', and '节点列表'. The 'Nodes' section is active, displaying a table of nodes. A red box highlights the '+ New Node' button in the top right corner of the nodes section. The table lists two nodes: 'master' (Linux (amd64)) and 'node-1' (Windows 11 (amd64)). The table columns include '名称', '架构', '时钟差异', '剩余磁盘空间', '剩余交换空间', '剩余临时空间', and '响应时间'.

名称	架构	时钟差异	剩余磁盘空间	剩余交换空间	剩余临时空间	响应时间
master	Linux (amd64)	已同步	13.91 GiB	0 B	13.91 GiB	0ms
node-1	Windows 11 (amd64)	1.4 秒 靠前	30.13 GiB	12.07 GiB	30.13 GiB	1759ms

2.进行配置

名称 ?

node-1

描述 ?

这是银伟的windows 机器

纯文本 预览

Number of executors ?

3

远程工作目录 ?

C:\Users\yfa\Desktop\pack

标签 ?

windows

用法 ?

尽可能的使用这个节点

启动方式 ?

保存

标签 ?

windows

用法 ?

尽可能的使用这个节点

启动方式 ?

通过 SSH 启动代理

主机 ?

192.168.0.225

证书 ?

yfa/*****

+ 添加

主机密钥验证策略 ?

已知主机文件验证策略

高级

已编辑

可用性 ?

尽量保持代理在线

保持

节点属性

☐ Disable deferred wipeout on this node ?

☐ Disk Space Monitoring Thresholds

☒ 工具位置

工具位置列表 ?

名称

(Git) git

目录

D:\Git\Git-2.40.1-64-bit.exe

新增

☐ 环境变量

保存

注意：

如果使用windwos服务器作为从节点，需要确保从节点ssh端口是开放的。

```
1 # 首先检查 OpenSSH 服务有没有安装
2 Get-WindowsCapability -Online | ? Name -like 'OpenSSH*'
3
4 # 如果没有安装，则进行安装 OpenSSH
5 Add-WindowsCapability -Online -Name OpenSSH.Server
6
7 # 启动OpenSSH的SSH服务器服务
8 Start-Service sshd
9
10 # 设置OpenSSH服务自动启动
11 Set-Service -Name sshd -StartupType 'Automatic'
12
13 # 手动添加主机到已知主机文件
14 ssh -o StrictHostKeyChecking=ask yfa@192.168.0.225
```

VSFTP 安装

1.使用 yum 安装 vsftp 、 ftp。

```
1 yum install -y vsftpd ftp
2 systemctl enable vsftpd --now
3 iptables -I INPUT -p tcp --dport 40000:50000 -j ACCEPT
4 service iptables save
5 service iptables restart
```

2.配置优化

```
1 vim /etc/vsftpd/vsftpd.conf
2 systemctl restart vsftpd

1 # 启用或禁用匿名访问。出于安全考虑，通常建议设置为 NO
2 anonymous_enable=NO
3 # 允许本地系统用户登录
4 local_enable=YES
5 # 允许已登录用户上传文件（需要 local_enable=YES）
6 write_enable=YES
7 # 将用户限制在其主目录中，增加安全性。
8 chroot_local_user=YES
9 # 配合 chroot_local_user=YES 使用，允许用户写入其被限制的主目录。
10 allow_writeable_chroot=YES
11 # 允许/禁止匿名用户上传文件
12 anon_upload_enable=NO
13 # 允许/禁止匿名用户创建新目录
14 anon_mkdir_write_enable=NO
15 # 设置上传文件的默认权限。
16 file_open_mode=0666
17 # 设置本地用户上传文件时的 umask，进一步控制文件权限。
18 local_umask=022
19 # 当用户进入一个新目录时，将显示该目录中的 .message 文件的内容
20 dirmessage_enable=YES
21 # 启用上传和下载的日志记录功能，这将记录所有的文件传输信息。
22 xferlog_enable=YES
23 # 这将 FTP 数据连接的源端口设置为 20，这是 FTP 协议的标准数据端口。
24 connect_from_port_20=YES
25 # 被动模式配置
26 pasv_enable=YES
27 # 设置被动模式端口范围的最小值
28 pasv_min_port=40000
29 # 设置被动模式端口范围的最大值
30 pasv_max_port=50000
31 # 使用标准的 xferlog 日志文件格式。如果设置为 NO，vsftpd 将使用更详细的日志记录格式。
32 xferlog_std_format=YES
33 # 指定传输日志文件的位置
34 xferlog_file=/var/log/vsftpd.log
35 # 指定 PAM（可插拔认证模块）服务的名称，用于用户认证。这里指定 vsftpd 将使用
   /etc/pam.d/vsftpd 文件中的设置。
36 pam_service_name=vsftpd
37 # 启用用户列表功能，这样 vsftpd 将检查 /etc/vsftpd/user_list 文件（或者通过
   userlist_file 指定的文件），来决定是否允许用户登录。
38 userlist_enable=YES
39 # 启用 tcp_wrappers 支持，允许您使用主机访问控制文件（通常是 /etc/hosts.allow 和
   /etc/hosts.deny）来允许或拒绝服务
40 tcp_wrappers=YES
41 #####--性能优化--#####
42 # 关闭独立模式。如果设置为 YES，vsftpd 将作为独立服务启动，而不是由 xinetd 控制
43 listen=YES
44 # 禁用 IPV6 监听
45 listen_ipv6=NO
46 # 设置最大客户端连接数
47 max_clients=100
48 # 设置每个 IP 地址的最大连接数
49 max_per_ip=5
```

3.使用方法

```
1 # 1、连接 FTP 服务器
2 [root@jenkins vsftpd]# ftp 192.168.72.129
3 Connected to 192.168.72.129 (192.168.72.129).
4 220 (vsFTPd 3.0.2)
5 Name (192.168.72.129:root): jenkinsftp      #输入用户名
6 331 Please specify the password.
7 Password:                               #输入密码
8 230 Login successful.
9 Remote system type is UNIX.
10 Using binary mode to transfer files.
11
12
13 #列出当前目录下的文件和文件夹。
14 ftp> ls
15 # 创建目录
16 ftp> mkdir project-pack
17 # 进入指定的目录。
18 ftp> cd project-pack
19 #显示当前工作目录的路径
20 ftp> pwd
21 257 "/home/jenkinsftp/project-pack"
22 # 上传文件，localfile 是本地文件的路径。
23 put localfile
24 # 下载文件，remotefile 是服务器上的文件路径。
25 get remotefile
26 # 退出 FTP 会话：
27 使用 quit 命令退出 FTP 会话。
```

4.创建ftpadmin 管理员账号

分配ftpadmin 账号超级权限，但是只能访问到/home，不能在访问上级

```
1 useradd ftpadmin
2 passwd ftpadmin
```

修改配置文件，添加 ftpadmin 权限

```
1 cat >> /etc/vsftpd/vsftpd.conf << EOF
2 user_config_dir=/etc/vsftpd/user_conf
3 EOF
```

为ftpadmin 用户创建一个配置文件

```
1 echo "local_root=/" > /etc/vsftpd/user_conf/ftpadmin
2 sudo setfacl -R -m u:ftpadmin:rxw /home/
```

5. WEB 界面上配置关联Jenkins

5.1) 打开系统管理，添加FTP的凭据信息

Publish over FTP

FTP Servers

FTP Server

Name ?

myftp

输入自定义名称

Hostname ?

192.168.72.129

FTP 服务器的IP地址

Username ?

jenkinsftp

FTP 的登陆账号

Password ?

已隐藏

FTP 账号的密码

修改密码

Remote Directory ?

/project-pack

设置你的ftp远程目录

高级

Test Configuration

保存

应用

5.2) 在项目中配置关联信息

Configure

- General
- 源码管理
- 构建触发器
- 构建环境
- Build Steps
- 构建后操作

构建后操作 2

Send build artifacts over FTP ?

FTP Publishers

FTP Server

Name ?
myftp 刚刚创建的凭据账号

Source files ?
/app/newBuild/*.apk 构建完成后，存放apk包的目录路径。这里项目名称默认不用添加即可

Remove prefix ?
/app/newBuild/ 会删除前缀，保留文件，美观

Remote directory ?
"CloudPad/yyyy-MM-dd" 上传到ftp的目录路径，我这里设置了个时间为目录，需要勾选目录是日期格式

All of the transfer fields support substitution of [Jenkins environment variables](#)

Exclude files ?

Pattern separator ?
[.]*

☐ No default excludes ?

☒ Make empty dirs ?

☐ Flatten files ?

☒ Remote directory is a date format ?

☐ Clean remote ?

☐ ASCII mode ?