

基于jenkins + gitlab实现mave项目自动发布

一、环境描述

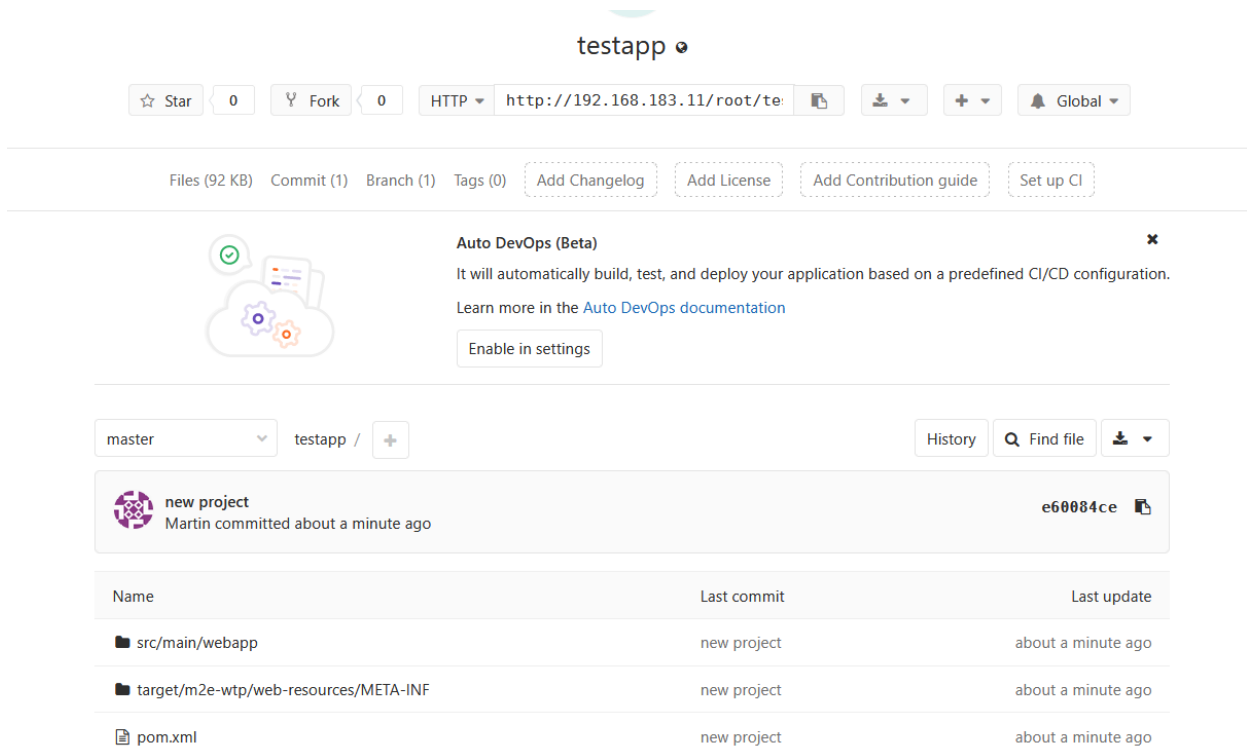
```
1 192.168.183.10 jenkins.linux.com
2 192.168.183.11 gitlab.linux.com
3 192.168.183.12 tomcat server
```

二、安装gitlab，上传项目测试代码

1、安装部署gitlab

```
1 [root@gitlab ~]# yum install -y gitlab-ce-10.1.5-ce.0.el7.x86_64.rpm
2
3 [root@gitlab ~]# grep -i "external_url" /etc/gitlab/gitlab.rb
4 external_url 'http://192.168.183.11'
5
6 [root@gitlab ~]# gitlab-ctl reconfigure
```

2、上传测试代码



The screenshot shows the GitLab web interface for a project named 'testapp'. At the top, there's a header with the project name 'testapp' and a search icon. Below the header, there are buttons for 'Star', 'Fork', and 'HTTP' (with a dropdown arrow). The 'Star' button shows 0 stars, and the 'Fork' button shows 0 forks. The 'HTTP' button shows the URL 'http://192.168.183.11/root/te'. There are also buttons for 'Add Changelog', 'Add License', 'Add Contribution guide', and 'Set up CI'. Below the header, there's a section for 'Auto DevOps (Beta)' with a description: 'It will automatically build, test, and deploy your application based on a predefined CI/CD configuration.' and a link to 'Learn more in the Auto DevOps documentation'. There is an 'Enable in settings' button. Below this, there's a section for the 'master' branch, showing the commit 'new project' by 'Martin' committed 'about a minute ago'. The commit hash is 'e60084ce'. Below this, there's a table with columns 'Name', 'Last commit', and 'Last update'. The table lists three items: 'src/main/webapp', 'target/m2e-wtp/web-resources/META-INF', and 'pom.xml', all with 'new project' as the last commit and 'about a minute ago' as the last update.

Name	Last commit	Last update
src/main/webapp	new project	about a minute ago
target/m2e-wtp/web-resources/META-INF	new project	about a minute ago
pom.xml	new project	about a minute ago

三、安装配置jenkins

1、安装jdk1.8.0

```
1 [root@jenkins ~]# tar xf jdk-8u91-linux-x64.tar.gz -C /usr/local/
2 [root@jenkins ~]# vim /etc/profile
3 export JAVA_HOME=/usr/local/jdk1.8.0_91
4 export PATH=$PATH:$JAVA_HOME/bin
5
6 [root@jenkins ~]# source /etc/profile
7
8 [root@jenkins ~]# java -version
9 java version "1.8.0_91"
10 Java(TM) SE Runtime Environment (build 1.8.0_91-b14)
11 Java HotSpot(TM) 64-Bit Server VM (build 25.91-b14, mixed mode)
```

2、安装maven编译工具

```
1 [root@jenkins ~]# tar xf apache-maven-3.6.3-bin.tar.gz -C /usr/local/
2
3 [root@jenkins ~]# vim /etc/profile
4 export PATH=$PATH:$JAVA_HOME/bin:/usr/local/apache-maven-3.6.3/bin/
5 [root@jenkins ~]# source /etc/profile
6
7 [root@jenkins ~]# mvn -version
8 Apache Maven 3.6.3 (cecedd343002696d0abb50b32b541b8a6ba2883f)
9 Maven home: /usr/local/apache-maven-3.6.3
10 Java version: 1.8.0_91, vendor: Oracle Corporation, runtime: /usr/local/
    jdk1.8.0_91/jre
11 Default locale: en_US, platform encoding: UTF-8
12 OS name: "linux", version: "3.10.0-693.el7.x86_64", arch: "amd64", famil
    y: "unix"
```

3、安装git客户端工具

```
1 [root@jenkins ~]# yum install -y git
```

4、安装jenkins

```
1 [root@jenkins ~]# rpm -ivh jenkins-2.277-1.1.noarch.rpm
2 warning: jenkins-2.277-1.1.noarch.rpm: Header V4 RSA/SHA512 Signature, ke
    y ID 45f2c3d5: NOKEY
3 Preparing... ##### [100%]
4 Updating / installing...
5 1:jenkins-2.277-1.1 ##### [100%]
```

5、修改jenkins启动脚本，添加java命令路径

```
1 [root@jenkins ~]# vim /etc/init.d/jenkins
2 candidates="
3 /etc/alternatives/java
4 /usr/lib/jvm/java-1.8.0/bin/java
5 /usr/lib/jvm/jre-1.8.0/bin/java
6 /usr/lib/jvm/java-11.0/bin/java
7 /usr/lib/jvm/jre-11.0/bin/java
8 /usr/lib/jvm/java-11-openjdk-amd64
9 /usr/bin/java
10 /usr/local/jdk1.8.0_91/bin/java
11 "
12 [root@jenkins ~]# systemctl daemon-reload
13 [root@jenkins ~]# systemctl start jenkins
14 [root@jenkins ~]# systemctl enable jenkins
15
16 [root@jenkins ~]# chkconfig jenkins on
17 [root@jenkins ~]# netstat -antp | grep 8080
18 tcp6 0 0 :::8080 :::* LISTEN 1451/java
```

6、浏览器访问jenkins管理界面，进行初始化设置

```
1 http://192.168.183.10:8080
```

解锁 Jenkins

为了确保管理员安全地安装 Jenkins，密码已写入到日志中（[不知道在哪里?](#)）该文件在服务器上：

```
/var/lib/jenkins/secrets/initialAdminPassword
```

请从本地复制密码并粘贴到下面。

管理员密码



离线

离线

该Jenkins实例似乎已离线。

参考 [离线Jenkins安装文档](#) 了解未接入互联网时安装Jenkins的更多信息。

可以通过配置一个代理或跳过插件安装来选择继续。

[配置代理](#)[跳过插件安装](#)

1 此处出现离线，是因为jenkins默认联网的地址是国外服务器，国内无法正常访问，可选择跳过插件安装，后续修改国内插件下载地址

创建第一个管理员用户

Username:	<input type="text" value="martin"/>
Password:	<input type="password" value="••••••••"/>
Confirm password:	<input type="password" value="••••••••"/>
Full name:	<input type="text" value="martin"/>

实例配置

Jenkins URL:

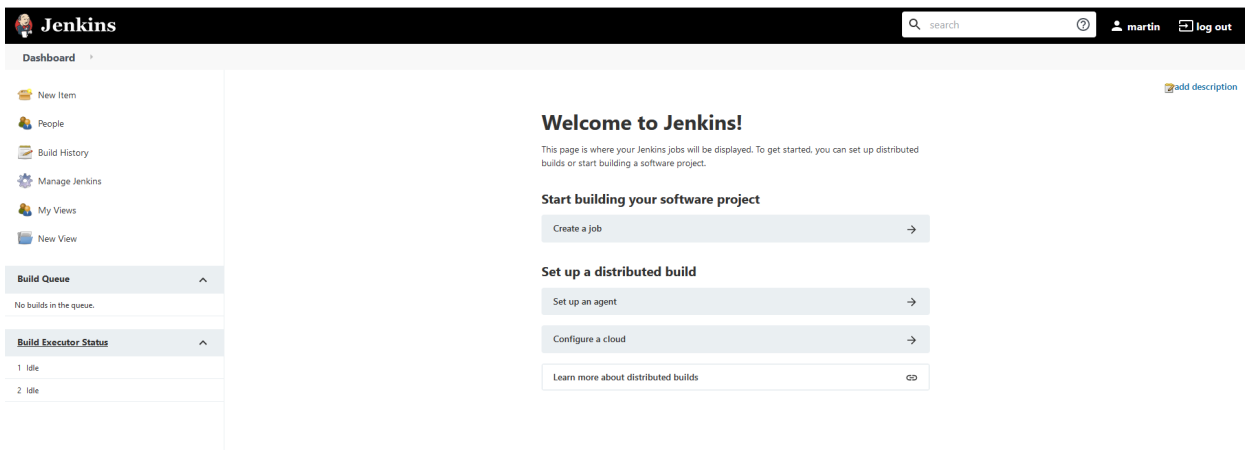
Jenkins URL 用于给各种Jenkins资源提供绝对路径链接的根地址。这意味着对于很多Jenkins特色是需要正确设置的，例如：邮件通知、PR状态更新以及提供给构建步骤的BUILD_URL环境变量。

推荐的默认值显示在尚未保存，如果可能的话这是根据当前请求生成的。最佳实践是要设置这个值，用户可能会需要用到。这将会避免在分享或者查看链接时的困惑。

Jenkins已就绪！

Jenkins安装已完成。

开始使用Jenkins



1 至此，jenkins安装完成!!!

四、修改jenkins插件下载地址为国内地址

Upload Plugin

You can upload an .hpi file to install a plugin from outside the central plugin repository.

File: 浏览... 未选择文件。

Upload

Update Site

URL

Submit

```
1 [root@jenkins ~]# systemctl restart jenkins
```

```

2
3 [root@jenkins ~]# sed -i 's/https:\\\\updates.jenkins.io\\/download/http
p:\\\\mirrors.tuna.tsinghua.edu.cn\\/jenkins/g' /var/lib/jenkins/updates/def
ault.json && sed -i 's/http:\\\\www.google.com/https:\\\\www.baidu.com/g' /
var/lib/jenkins/updates/default.json
4
5 [root@jenkins ~]# systemctl restart jenkins

```

五、安装插件

插件作用说明:

- 1 1. Maven Intergration 集成编译工具
- 2 2. Publish Over SSH 基于ssh协议向web服务器发布项目
- 3 3. **git**
- 4 4. gitlab

1、安装界面汉化插件

The screenshot shows the Jenkins Plugin Manager interface. The 'Available' tab is selected, and a search for 'cn' has been performed. The 'Localization: Chinese (Simplified)' plugin is highlighted with a red box. Below the plugin list, the 'Install without restart' button is also highlighted with a red box.

Install	Name	Version	Released
<input checked="" type="checkbox"/>	Localization: Chinese (Simplified) <small>Localization</small> Jenkins Core 及其插件的简体中文语言包。由 Jenkins 中文社区 维护。	1.0.24	2 mo 17 days ago
<input type="checkbox"/>	IBM Content Navigator remote plug-in reloader <small>Miscellaneous</small> This plug-in reloads automatically and remotely a plug-in on an IBM Content Navigator server. The plug-in jar file needs to be already on the server but it will reload it without stopping the application. This is suitable for development platform, as well as for production platform where you don't want any interruption of service. Although the plug-in's services will be using the new version as soon as you run the task, user will have to reload/refresh their cache to use the last version of all JavaScript files, (or wait 24h hours, default Cache-Control time for ICN). This works for an application managed authentication so far, not if you're using an other mechanism (basic, form based, ...)	1.0	3 yr 11 mo ago

Buttons: **Install without restart**, Download now and install after restart, Update information obtained: 2 min 1 sec ago, Check now

2、安装maven Intergration插件

The screenshot shows the Jenkins interface with a list of plugins and their installation status. The 'Maven Integration' plugin is highlighted with a red box and shows a progress bar indicating it is 'Installing'.

Plugin Name	Status
JACKSON 2 API	完成
ECharts API	完成
Struts	完成
Display URL API	完成
Pipeline: Step API	完成
Checks API	完成
SCM API	完成
Pipeline: API	完成
JUnit	完成
Mailer	完成
Apache HttpComponents Client 4.x API	完成
Credentials	完成
SSH Credentials	完成
JSch dependency	完成
Maven Integration	安装中
Loading plugin extensions	Pending

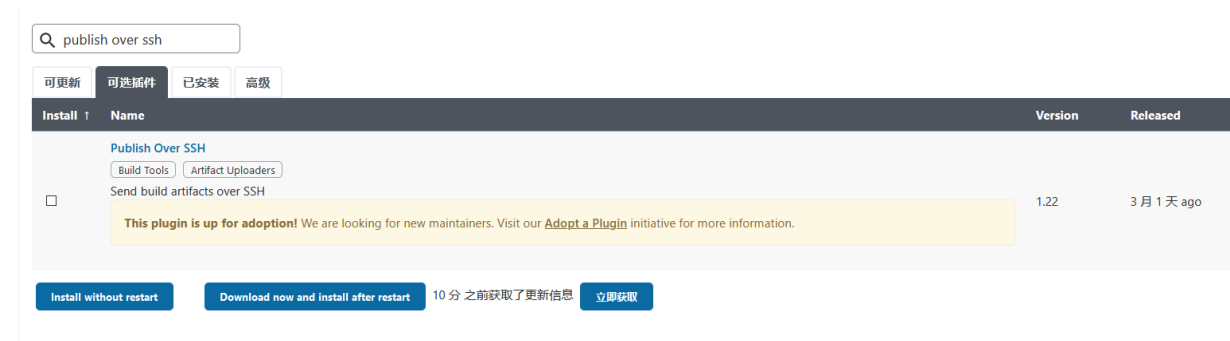
3、安装git插件



4、安装gitlab插件



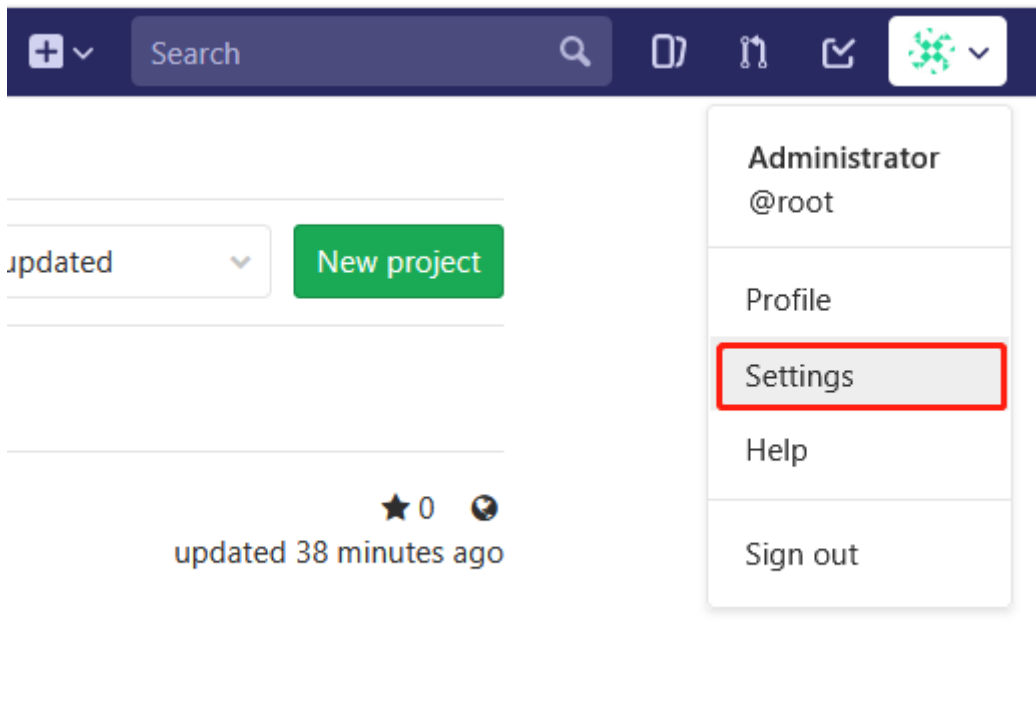
5、安装Publish over ssh插件



6、所有插件安装完成后，重启jenkins服务

六、配置jenkins连接gitlab

1、在gitlab上生成认证需要的令牌



User Settings > Access Tokens

Personal Access Tokens

You can generate a personal access token for each application you use that needs access to the GitLab API.

You can also use personal access tokens to authenticate against Git over HTTP. They are the only accepted password when you have Two-Factor Authentication (2FA) enabled.

Add a personal access token

Pick a name for the application, and we'll give you a unique personal access token.

Name

admin

Expires at

2021-02-27

Scopes

☒ **api** Access the authenticated user's API

Full access to GitLab as the user, including read/write on all their groups and projects

☐ **read_user** Read the authenticated user's personal information

Read-only access to the user's profile information, like username, public email and full name

Create personal access token

Active Personal Access Tokens (1)

Name	Created	Expires	Scopes	
admin	Feb 1, 2021	In 11 months	api	Revoke

Your New Personal Access Token

scsCU18bzjb9E8eKsn5p



Make sure you save it - you won't be able to access it again.

2、在系统设置中找到gitlab

Gitlab

☒ Enable authentication for '/project' end-point

GitLab connections

Connection name

gitlab_conn

A name for the connection

Gitlab host URL

http://192.168.183.11/

The complete URL to the Gitlab server (e.g. http://gitlab.mydomain.com)

Credentials

- 无 -

添加

API Token for Gitlab access required

API Token for accessing Gitlab

ERROR

高级...

Test Connection

删除

Jenkins 凭据提供者: Jenkins

添加凭据

Domain

全局凭据 (unrestricted)

类型

GitLab API token

范围

全局 (Jenkins, nodes, items, all child items, etc)

API token

.....

ID

描述

添加

取消

Gitlab

☒ Enable authentication for '/project' end-point

GitLab connections

Connection name

gitlab_conn

A name for the connection

Gitlab host URL

http://192.168.183.11/

The complete URL to the Gitlab server (e.g. http://gitlab.mydomain.com)

Credentials

GitLab API token

添加

Success

高级...

Test Connection

删除

新增

七、配置jenkins全局工具

1、配置jdk安装路径

JDK

JDK 安装

新增 JDK

JDK

别名

jdk1.8.0

JAVA_HOME

/usr/local/jdk1.8.0_91

☐ Install automatically

2、配置git客户端

Git

Git installations

Git

Name

Default

Path to Git executable

git

☐ Install automatically

3、配置maven工具

Maven

Maven 安装

新增 Maven



Maven

Name

maven3.6.3

MAVEN_HOME

/usr/local/apache-maven-3.6.3/

☐ Install automatically

八、创建任务编译项目

1、配置jenkins与web server间的免密ssh

```
1 [root@jenkins ~]# ssh-keygen -t rsa
2 [root@jenkins ~]# ssh-copy-id root@192.168.183.12
```

2、配置Publish over ssh插件

Publish over SSH

Jenkins SSH Key

Passphrase

Path to key

Key

```
Mp5cNRcdKxyHvFs6ZeZzgQgwoUv+Sa0Etv6UtZf0cWNbRJZhBLJORmySYBec9hW
L46pmR9O3jN84f8Xl16+bJfXPw5JcEXsukiReijQbidxk5e9/ooufhEjXhGu09m
muOV9QKBgCOOwpAjvzTCTneIFZoMExpJnu/A/d9t+1OVWjweSwmPKVmUb62un5
wr6Dsj+S80m0+B+Jys9wZysqoWCtDPVgVuL9YTOLHh/kLJnNkcI9Z0RJmN0nEd
RWGt9UfwiWVBba/7XwkP9CB/CNOT4feSCOOHQGs56qg6k5HbC0
-----END RSA PRIVATE KEY-----
```

☐ Disable exec

SSH Servers

新增

保存

应用

高级...

jenkins私钥

3、添加远程web服务器

Name

server_192.168.183.12

Hostname

192.168.183.12

Username

root

Remote Directory

/app/jenkins

Success

高级...

Test Configuration

删除

4、创建任务

输入一个任务名称

test01

» 必填项



Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this used for something other than software build.



构建一个maven项目

构建一个maven项目.Jenkins利用你的POM文件,这样可以大大减轻构建配置.



流水线

精心地组织一个可以长期运行在多个节点上的任务。适用于构建流水线（更加正式地应当称为工作流），增加或者组织难的任务类型。



构建一个多配置项目

适用于多配置项目,例如多环境测试,平台指定构建,等等.

确定

5、设置gitlab连接

General

源码管理

构建触发器

构建环境

Pre Steps

Build

Post Steps

构建设置

构建后操作

描述

test01 web

[Plain text] 预览

☐ Discard old builds

?

GitLab Connection

gitlab_conn

☐ This project is parameterized

?

☐ 关闭构建

?

☐ 在必要的时候开发构建

?

高级...

6、设置项目仓库地址

General

源码管理

构建触发器

构建环境

Pre Steps

Build

Post Steps

构建设置

构建后操作

☐ 无

☒ Git

Repositories

Repository URL

http://192.168.183.11/root/testapp.git

Credentials

root/*****

添加

高级...

Add Repository

7、设置项目构建时间

构建触发器

☐ Build whenever a SNAPSHOT dependency is built

?

☐ 触发远程构建 (例如,使用脚本)

?

☐ Build after other projects are built

?

☐ Build periodically

?

☐ Build when a change is pushed to GitLab. GitLab webhook URL: http://192.168.183.10:8080/project/test01

?

☒ Poll SCM

?

日程表

H/5 * * * *

Would last have run at 2021年2月2日 星期二 下午02时11分40秒 CST; would next run at 2021年2月2日 星期二 下午02时16分40秒 CST.

☐ 忽略钩子 post-commit

?

8、设置pom.xml文件位置，相对于任务的存储位置

Pre Steps

Add pre-build step ▾

Build

Root POM?

pom.xml

Goals and options?

clean package -Dmaven.test.skip=true

高级...

9、设置构建后执行的动作

General 源码管理 构建触发器 构建环境 Pre Steps Build **Post Steps** 构建设置 构建后操作

☒ Run only if build succeeds ☐ Run only if build succeeds or is unstable ☐ Run regardless of build result
Should the post-build steps run only for successful builds, etc.

Send files or execute commands over SSHX?

SSH Publishers

SSH Server

Name?

server_192.168.183.12

高级...

Transfers

Transfer Set

Source files?

target/*.war

Remove prefix?

target

Remote directory?

pro1

Exec command?

export PATH=\$PATH:/opt/jdk1.8.0_91/bin/
cd /opt/apache-tomcat-8.5.11/bin/

All of the transfer fields (except for Exec timeout) support substitution of [Jenkins environment variables](#)

执行命令参考

```
1 export PATH=$PATH:/opt/jdk1.8.0_91/bin/
2 cd /opt/apache-tomcat-8.5.11/bin/
3 ./shutdown.sh
4 /bin/cp -rf /app/jenkins/pro1/*.war cd /opt/apache-tomcat-8.5.11/webapps
5 ./startup.sh
```

九、测试

1、在jenkins服务器上查看构建的项目

```
1 [root@jenkins ~]# ls /var/lib/jenkins/workspace/test01
2 pom.xml src target
3 [root@jenkins ~]# ls /var/lib/jenkins/workspace/test01/target/
4 hello hello.war maven-archiver
```

2、查看发布到web server上的项目

```
1 [root@web_server webapps]# ls /app/jenkins/pro1/
2 hello.war
3
4 [root@web_server webapps]# ls /opt/apache-tomcat-8.5.11/webapps/
5 docs examples hello hello.war host-manager manager ROOT
```

3、客户端访问web server

