

Jenkins+maven+git部署环境

1. 环境准备.

1.1 需要在linux环境上安装, 部署相应的linux环境
(jdk, 防火墙, 等等), 自己百度解决哈.

1.2 软件准备工作

这里是我用到的相关软件

maven tomcat 和 jenkins的war包及jenkins的扩展插件

附上jenkins的下载插件的地址(有些下载不了只能手动下载后上传了)

<http://mirror.xmission.com/jenkins/plugins/>

下面我将相关软件上传到我的linux机器上了.

```
[root@solr2 jenkins]# ll
total 80924
-rwxrwxrwx. 1 root root 8850470 Aug 24 2015 apache-tomcat-7.0.63.tar.gz
-rwxrwxrwx. 1 root root 68479320 May 25 06:31 jenkins.tar
-rwxrwxrwx. 1 root root 5529765 May 25 08:24 maven.tar.gz
[root@solr2 jenkins]# pwd
/root/jenkins
[root@solr2 jenkins]#
```

<http://blog.csdn.net/>

2. 搭建 相关服务

2.1 搭建maven环境

解压 文件

```
tar -zxvf maven.tar.gz
```

进入文件内部 查看文件路径

```
[root@solr2 maven]# pwd
/root/jenkins/maven
[root@solr2 maven]#
```

<http://blog.csdn.net/>

编辑maven的环境变量

```
vim /etc/profile
```

到文件最底部加入maven环境变量的如下内容:

```

else
    umask 022
fi

for i in /etc/profile.d/*.sh ; do
    if [ -r "$i" ]; then
        if [ "${-#*i}" != "$-" ]; then
            . "$i"
        else
            . "$i" >/dev/null 2>&1
        fi
    fi
done

unset i
unset -f pathmunge
JAVA_HOME=/usr/local/java/jdk1.8.0_60
export PATH=$JAVA_HOME/bin:$PATH

#maven
export MAVEN_HOME=/root/jenkins/maven
export PATH=$MAVEN_HOME/bin:$PATH
http://blog.csdn.net/

```

2.2 搭建jenkins的相关服务

解压tomcat 修改一个不会被占用的端口, 设置一下tomcat的url编码格式为UTF-8

```

    maxThreads="150" minSpareThreads="4"/>
-->

<!-- A "Connector" represents an endpoint by which requests are received
and responses are returned. Documentation at :
Java HTTP Connector: /docs/config/http.html (blocking & non-blocking)
Java AJP Connector: /docs/config/ajp.html
APR (HTTP/AJP) Connector: /docs/apr.html
Define a non-SSL HTTP/1.1 Connector on port 8080
-->
<Connector port="9800" protocol="HTTP/1.1"
    connectionTimeout="20000"
    redirectPort="9804"
    URIEncoding="UTF-8"/>
<!-- A "Connector" using the shared thread pool-->
<!--
<Connector executor="tomcatThreadPool"
    port="8080" protocol="HTTP/1.1"
    connectionTimeout="20000"
    redirectPort="8443" />
-->

```

<http://blog.csdn.net/>

将jenkins的包解压, 移动到tomcat的webapps中

```

root@solr2 webapps]# ll
total 4
lrwxrwxrwx. 10 root root 4096 May 25 03:16 jenkins
root@solr2 webapps]# pwd
/root/jenkins/tomcat-9800/webapps
root@solr2 webapps]#

```

<http://blog.csdn.net/>

在 profile中配置 jenkins_home

```
unset i
unset -f pathmunge
JAVA_HOME=/usr/local/java/jdk1.8.0_60
export PATH=$JAVA_HOME/bin:$PATH

#maven
export MAVEN_HOME=/root/jenkins/maven
export PATH=$MAVEN_HOME/bin:$PATH

#jenkins
export JENKINS_HOME=/root/jenkins/tomcat-9800/webapps/jenkins
```

重新加载一下配置文件

```
source /etc/profile
```

启动tomcat 查看日志输出文件 catalina.out,复制下来一个密码数字

```
[root@solr2 tomcat-9800]# ll
total 116
drwxr-xr-x. 2 root root 4096 May 29 07:54 bin
drwxr-xr-x. 3 root root 4096 May 29 08:07 conf
drwxr-xr-x. 2 root root 4096 May 29 07:54 lib
-rw-r--r--. 1 root root 56846 Jun 30 2015 LICENSE
drwxr-xr-x. 2 root root 4096 May 29 08:07 logs
-rw-r--r--. 1 root root 1239 Jun 30 2015 NOTICE
-rw-r--r--. 1 root root 8965 Jun 30 2015 RELEASE-NOTES
-rw-r--r--. 1 root root 16204 Jun 30 2015 RUNNING.txt
drwxr-xr-x. 3 root root 4096 May 29 08:08 temp
drwxr-xr-x. 3 root root 4096 May 29 08:01 webapps
drwxr-xr-x. 3 root root 4096 May 29 08:07 work
[root@solr2 tomcat-9800]# cat logs/catalina.out
```

<http://blog.csdn.net/>

Jenkins initial setup is required. An admin user has been created and a password generated. Please use the following password to proceed to installation:

727ff0a70781484fb4cedd2e6b73ca24 这个复制下来

This may also be found at: /root/jenkins/tomcat-9800/webapps/jenkins/secrets/initialAdminPassword

<http://blog.csdn.net/>

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

```
/root/jenkins/tomcat-9800/webapps/jenkins/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

刚刚复制下来的粘贴进去 点击继续

Continue

<http://blog.csdn.net/>

Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins

Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

点击这个 安装默认的插件

<http://blog.csdn.net/>

接下来进入到这个界面. 我们等待安装结束即可:

Getting Started

Getting Started



<input checked="" type="checkbox"/> Folders Plugin	OWASP Markup Formatter Plugin	build timeout plugin	<input type="checkbox"/> Credentials Binding Plugin	Folders Plugin ** JUnit Plugin
<input type="checkbox"/> Timestampers	Workspace Cleanup Plugin	<input type="checkbox"/> Ant Plugin	Gradle plugin	
Pipeline	GitHub Organization Folder Plugin	<input type="checkbox"/> Pipeline: Stage View Plugin	Git plugin	
Subversion Plug-in	<input type="checkbox"/> SSH Slaves plugin	<input type="checkbox"/> Matrix Authorization Strategy Plugin	PAM Authentication plugin	
LDAP Plugin	Email Extension Plugin	<input type="checkbox"/> Mailer Plugin		

<http://blog.csdn.net/>

最后进入jenkins, 界面如下, jenkins初步安装成功了:

可能会有很多没有安装成功的插件, 不必担心, 需要的我们再手动安装就行了

还有一个配置登录的帐号密码的界面, 配置一下就行了, 接下来是这个:

Getting Started

Create First Admin User

Invalid e-mail address

用户名:	<input type="text" value="root"/>
密码:	<input type="password" value="..."/>
确认密码:	<input type="password" value="..."/>
全名:	<input type="text" value="admin"/>
电子邮件地址:	<input type="text" value="songqinghuwork@126.com"/>

<http://blog.csdn.net/>



初步成功了！

3. 配置 自动化部署


3.1 插件的安装

我们配置的是 git + maven 方式的 自动化部署 所以git和maven的相关插件是必须的 还有一个ssh用于机器间的文件传送


我新建一个job 人品不好, 没有maven工程的项目, 我们需要安装一下maven 的插件

Enter an item name


» This field cannot be empty, please enter a valid name




构建一个自由风格的软件项目
这是Jenkins的主要功能.Jenkins将会结合任何SCM和任何构建系统来构建你的项目,甚至可以构建软件以外的东西。




Pipeline
Orchestrates long-running activities that can span multiple build slaves. Suitable for building pipelines (for example, for building a new version of a project).



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。



External Job
这个类型的任务允许你记录执行在外部Jenkins的任务,任务甚至运行在远程机器上.这可以让Jenkins作为你所构建的项目的一部分。



Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is only for displaying items, a folder can be used as long as they are in different folders.

OK

下面我们进入插件安装的页面：

Jenkins

新建

用户

任务历史

系统管理

My Views

Credentials

构建队列

队列中没有构建任务

构建执行状态

1 空闲

2 空闲

管理Jenkins

系统设置

全局设置&路径

Configure Global Security

Secure Jenkins; define who is allowed to access/use the system.

Configure Credentials

Configure the credential providers and types

Global Tool Configuration

Configure tools, their locations and automatic installers.

读取设置

放弃当前内存中所有的设置信息并从配置文件中重新读取 仅用于当您手动修改配置文件

管理插件

添加、删除、禁用或启用Jenkins功能扩展插件。

系统信息

显示系统环境信息以帮助解决问题。

System Log

系统日志从java.util.logging捕获Jenkins相关的日志信息。

负载统计

检查您的资源利用情况，看看是否需要更多的计算机来帮助您构建。

Jenkins CLI

从您命令行或脚本访问或管理您的Jenkins。

脚本命令行

执行用于管理或故障探测或诊断的任意脚本命令。

管理节点

添加、删除、控制和监视系统运行任务的节点。

About Jenkins

我们要安装的插件有：

GIT plugin （可能已经默认安装了） Publish Over SSH （远程Shell） [Maven Integration plugin](#)
（这一部比较重要）

git我这是安装成功了。 下载maven的安装时失败了,我们手动安装

GitHub plugin	失败 - 详细
GitHub Branch Source Plugin	失败 - 详细
GitHub Organization Folder Plugin	失败 - 详细
Pipeline: Stage View Plugin	完成
Git plugin	完成
MapDB API Plugin	完成
Subversion Plug-in	失败 - 详细
SSH Slaves plugin	完成
Matrix Authorization Strategy Plugin	完成
PAM Authentication plugin	完成
LDAP Plugin	完成
Email Extension Plugin	完成
Mailer Plugin	完成
Subversion Plug-in	完成
Maven Integration plugin	失败 - <pre> hudson.util.IOException2: Failed to download from http://updates.jenkins-ci.org/download http://ftp.tsukuba.wide.ad.jp/software/jenkins/plugins/maven-plugin/2.13/maven-plugin.h at hudson.model.UpdateCenter\$UpdateCenterConfiguration.download(UpdateCenter.j at hudson.model.UpdateCenter\$DownloadJob._run(UpdateCenter.java:1641) at hudson.model.UpdateCenter\$InstallationJob._run(UpdateCenter.java:1839) at hudson.model.UpdateCenter\$DownloadJob.run(UpdateCenter.java:1615) at java.util.concurrent.Executors\$RunnableAdapter.call(Executors.java:511) at java.util.concurrent.FutureTask.run(FutureTask.java:266) at hudson.remoting.AtmostOneThreadExecutor\$Worker.run(AtmostOneThreadExecutor. at java.lang.Thread.run(Thread.java:745) Caused by: java.io.IOException: Inconsistent file length: expected 10898926 but only g at hudson.model.UpdateCenter\$UpdateCenterConfiguration.download(UpdateCenter.j ... 7 more </pre>
Publish Over SSH	完成

<http://blog.>

3.2 基本配置的部署

插件已经基本准备好了, 下面我们还要在做一些基础的配置


要配置的有 jdk maven 和git 其中git需要在机器上安装

 新建

 用户

 任务历史

 系统管理

 My Views

 Credentials

构建队列

队列中没有构建任务

构建执行状态

1 空闲

2 空闲

管理Jenkins



系统设置

全局设置&路径



Configure Global Security

Secure Jenkins; define who is allowed to access/use the system.



Configure Credentials

Configure the credential providers and types



Global Tool Configuration

Configure tools, their locations and automatic installers.



读取设置

放弃当前内存中所有的设置信息并从配置文件中重新读取 仅用于当您手动修改配置文件



管理插件

添加、删除、禁用或启用Jenkins功能扩展插件。



系统信息

显示系统环境信息以帮助解决问题。



System Log

系统日志从java.util.logging捕获Jenkins相关的日志信息。



负载统计

检查您的资源利用情况，看看是否需要更多的计算机来帮助您构建。



Jenkins CLI

从您命令行或脚本访问或管理您的Jenkins。



脚本命令行

执行用于管理或故障探测或诊断的任意脚本命令。



管理节点

添加、删除、控制和监视系统运行任务的节点。



About Jenkins

See the version and license information.



Manage Old Data

Scrub configuration files to remove remnants from old plugins and earlier versions.

JDK

JDK 安装

JDK

别名

jdk

JAVA_HOME

/usr/local/java/jdk1.8.0_60

☐ 自动安装

新增 JDK

系统下 JDK 安装列表

Maven

Maven 安装

Maven

Name

maven

MAVEN_HOME

/root/jenkins/maven

☐ 自动安装

新增 Maven

系统下Maven 安装列表

我们再机器上执行命令安装git:

```
yum install git
```

安装完了 我的git在

```
/usr/bin/git
```

```
(2/3): perl-Error-0.17015-4.el6.noarch.rpm
(3/3): perl-Git-1.7.1-4.el6_7.1.noarch.rpm
-----
Total
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : 1:perl-Error-0.17015-4.el6.noarch
  Installing : git-1.7.1-4.el6_7.1.x86_64
  Installing : perl-Git-1.7.1-4.el6_7.1.noarch
  Verifying  : perl-Git-1.7.1-4.el6_7.1.noarch
  Verifying  : 1:perl-Error-0.17015-4.el6.noarch
  Verifying  : git-1.7.1-4.el6_7.1.x86_64

Installed:
  git.x86_64 0:1.7.1-4.el6_7.1

Dependency Installed:
  perl-Error.noarch 1:0.17015-4.el6

Complete!
[root@solr2 maven]# whereis git
git: /usr/bin/git /usr/share/man/man1/git.1.gz
[root@solr2 maven]#
```

将git配置上去:

Git

Git installations

Git

Name

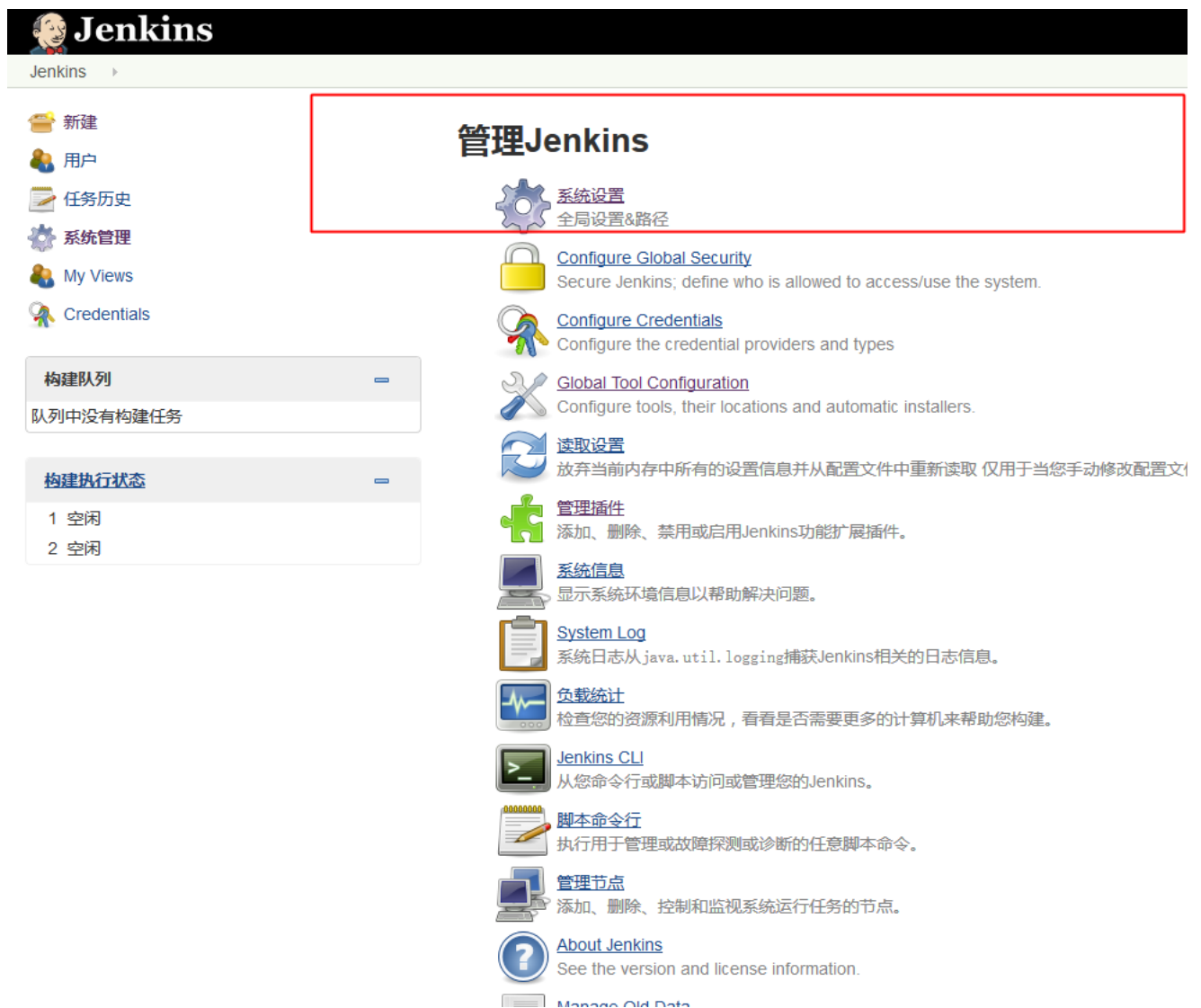
Default

Path to Git executable













/usr/bin/git

☐ 自动安装

再配置ssh相关的配置：



管理Jenkins

-  **系统设置**
全局设置&路径
-  **Configure Global Security**
Secure Jenkins; define who is allowed to access/use the system.
-  **Configure Credentials**
Configure the credential providers and types
-  **Global Tool Configuration**
Configure tools, their locations and automatic installers.
-  **读取设置**
放弃当前内存中所有的设置信息并从配置文件中重新读取 仅用于当您手动修改配置文
-  **管理插件**
添加、删除、禁用或启用Jenkins功能扩展插件。
-  **系统信息**
显示系统环境信息以帮助解决问题。
-  **System Log**
系统日志从java.util.logging捕获Jenkins相关的日志信息。
-  **负载统计**
检查您的资源利用情况，看看是否需要更多的计算机来帮助您构建。
-  **Jenkins CLI**
从您命令行或脚本访问或管理您的Jenkins。
-  **脚本命令行**
执行用于管理或故障探测或诊断的任意脚本命令。
-  **管理节点**
添加、删除、控制和监视系统运行任务的节点。
-  **About Jenkins**
See the version and license information.
-  **Manage Old Data**

公共配置：

Jenkins 与 应用服务器 关于 ssh 的连接：

(<https://blog.csdn.net/fengzhilin6773/article/details/78246777>)

Passphrase：密码（key的密码，如果你设置了）

Path to key：key文件（私钥）的路径

Key：将私钥复制到这个框中

Disable exec：禁止运行命令

☐ 通过发送测试邮件测试配置

Publish over SSH

Jenkins SSH Key

Passphrase

Path to key

Key

Disable exec

SSH Servers

☐

SSH Server

Name

solr1

Hostname

192.168.31.60

Username

root

Remote Directory

/

源码管理

☐ None

☒ Git

Repositories

Repository URL

Credentials

Branches to build

Branch Specifier (blank for 'any')

源码库浏览器

(自动)

Additional Behaviours

Build

Root POM:

Goals and options:

Post Steps

☐ 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.

Add pre-build step ▼

Add post-build step ▼

这里的文件夹可要创建好, 最后那个是执行的脚本

Send build artifacts over SSH

SSH Publishers

SSH Server

Name:

Transfers

Transfer Set

Source files:

Remove prefix:

Remote directory:

Exec command:

⚠️ Either Source files, Exec command or both must be supplied
 All of the transfer fields (except for Exec timeout) support substitution of [Jenkins environ](#)

Add Transfer Set

Add Server

这是我简单写的一个脚本, 你需要改成相应的操作就行了. 关闭tomcat, 备份, 清理, 自动解压, 启动tomcat
我这里只是演示能将war包传递过来, 并且移动重命名了.

sheel脚本:

```
#!/bin/bash
WAR_PATH="/home/simm/script/sit_hive_api"
TOMCAT_PATH="/home/simm/tomcat_sit_hive_api/"
TOMCATTHREAD=`ps aux | grep 'tomcat_sit_hive_api' | grep -v grep`
if test -z "$TOMCATTHREAD"
then
    echo "TOMCAT NOT START"
else
    THREADLIST=(${TOMCATTHREAD// / })
    PID=${THREADLIST[1]}
    kill -9 $PID
    sleep 2s
fi
echo "TOMCAT START"
cd ${TOMCAT_PATH}/webapps
rm -f ROOT.war
rm -rf ROOT
mv ${WAR_PATH}/hive-api.war ROOT.war
cd ${TOMCAT_PATH}/bin
nohup sh startup.sh
sleep 5s
```

看执行结果:

```
[JENKINS] Recording test results
[INFO]
[INFO] --- maven-war-plugin:2.6:war (default-war) @ hive-api ---
[INFO] Packaging webapp
[INFO] Assembling webapp [hive-api] in [/root/.jenkins/jobs/sit-hive-api/workspace/WebContent]
[INFO] Processing war project
[INFO] Webapp assembled in [2476 msecs]
[INFO] Building war: /root/.jenkins/jobs/sit-hive-api/workspace/target/hive-api.war
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 31.276 s
[INFO] Finished at: 2018-06-21T13:17:24+08:00
[INFO] Final Memory: 40M/383M
[INFO] -----
[JENKINS] Archiving /root/.jenkins/jobs/sit-hive-api/workspace/pom.xml to com.simm/hive-api/server/hive-api-server.pom
[JENKINS] Archiving /root/.jenkins/jobs/sit-hive-api/workspace/target/hive-api.war to com.simm/hive-api/server/hive-api-server.war
channel stopped
SSH: Connecting from host [iZwz999sw02ojy2d1g0v58Z]
SSH: Connecting with configuration [sit-hive] ...
SSH: EXEC: STDOUT/STDERR from command [/home/simm/script/sit_hive_api/auto_deploy.sh] ...
TOMCAT START
Tomcat started.
SSH: EXEC: completed after 7,605 ms
SSH: Disconnecting configuration [sit-hive] ...
SSH: Transferred 1 file(s)
Finished: SUCCESS
```

好, 到现在自动化部署就完成了. 祝你成功!