

# 《安装部署 Jenkins》

## 安装 Jenkins

[Jenkins 离线安装 官方文档](#)

如需获取目标系统的 Jenkins RPM 或 WAR 离线安装包可前往 [Jenkins mirrors](#) 站点获取『推荐获取 LTS Releases 发行版』

例如本文使用的 Jenkins RPM 软件包下载链接 <https://archives.jenkins.io/redhat-stable/jenkins-2.426.2-1.1.noarch.rpm>

如您需要 Jenkins WAR 包,可前通过以下资源地址下载 <https://archives.jenkins.io/war-stable/2.426.2/jenkins.war>

### 1.配置 Jenkins YUM 源

```
#=====
#####
# 下载 Jenkins YUM 镜像文件到 /etc/yum.repos.d/ 目录下『离线模式安装请查阅 Jenkins 官方文档』
wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo
#-----
-----#
# 导入 Jenkins RPM 安装包校验证书
rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
#-----
-----#
# 升级系统软件包『可选(若需要执行以下命令,建议执行完后重启您的系统)』
yum upgrade -y
#=====
#####
```

### 2.部署 Jenkins

```
#=====
=====#
# 安装 Jenkins RPM
yum install jenkins -y
#-----
-----#
# 重新加载 systemd 管理器配置
systemctl daemon-reload
#-----
-----#
# 设置开机启动服务并立即启动 jenkins.service
systemctl --now enable jenkins.service
#=====
=====#
# 防火墙相关设置『可选(请根据实际情况配置您的防火墙,以允许访问 8080 端口)』
firewall-cmd --permanent --zone=public --add-port=8080/tcp
firewall-cmd --permanent --zone=public --add-service=http
# 使用 RPM 安装的方式默认会安装此规则『若采用其他安装方式安装 Jenkins 请参考官方文档配置您的防火墙规则』
firewall-cmd --permanent --zone=public --add-service=jenkins
# 重新加载防火墙并保留状态信息
firewall-cmd --reload
# 查看防火墙规则设置是否生效
firewall-cmd --permanent --zone=public --list-all
#=====
=====#
```

c43a88c4c6c2ba8eabf474d7555fdc0803dc93a0

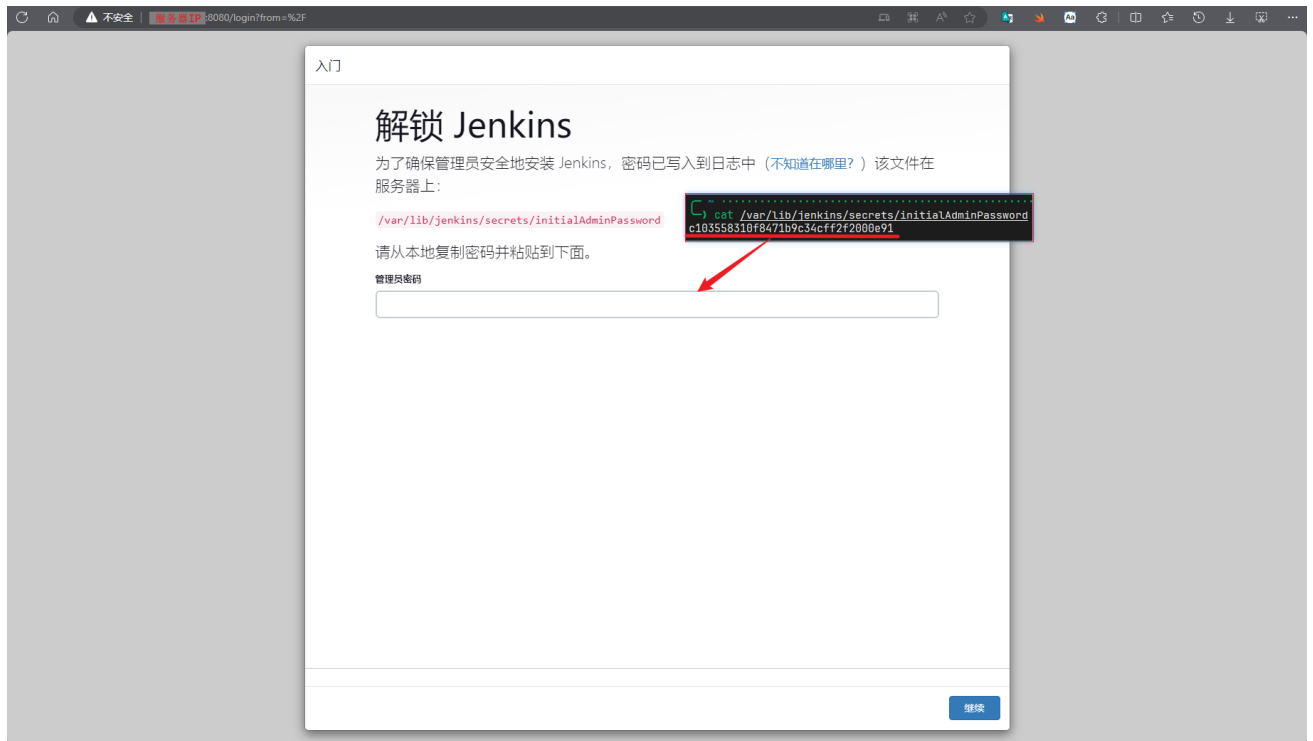
### 3.Jenkins 初始化设置

在浏览器端口键入以下地址访问 Jenkins 服务,并根据提示进行 Jenkins 的初始配置 <http://<服务器IP>:8080>

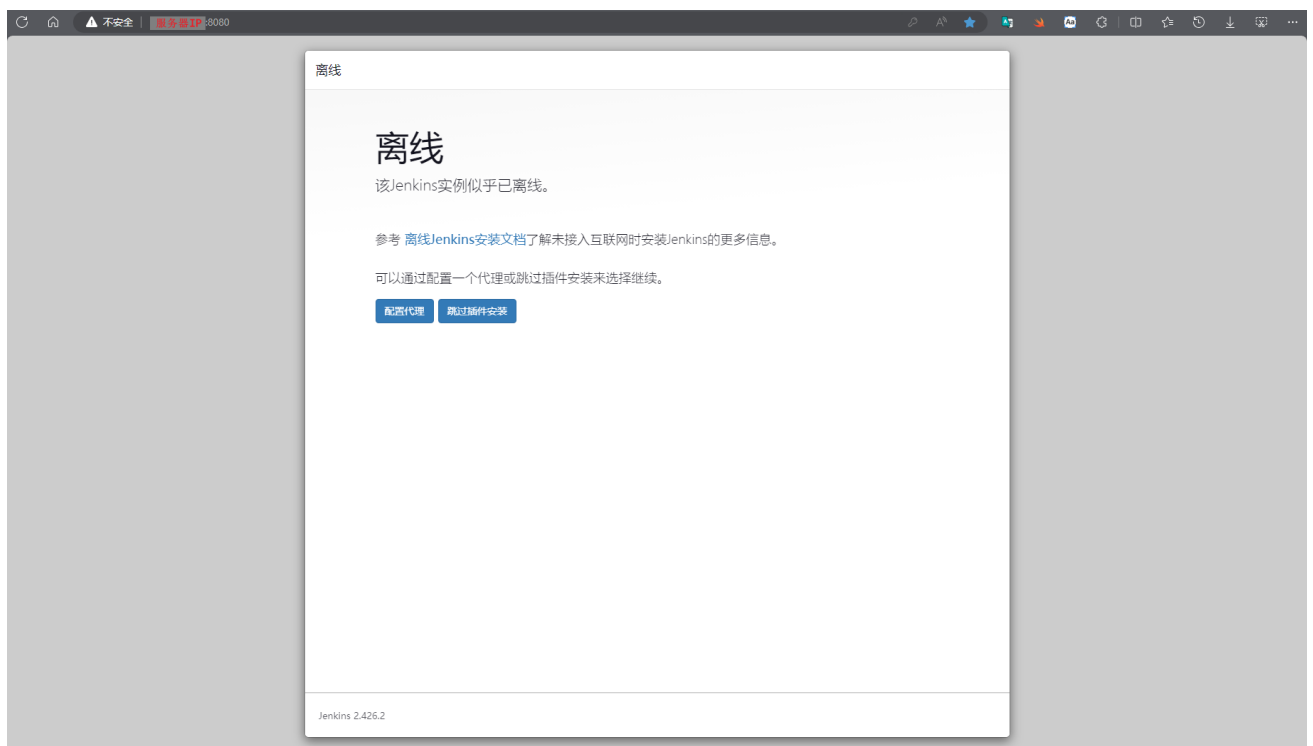
- 根据提示获取初始密码并登录

```
#=====
=====#
# 获取初始密码参考命令
cat /var/lib/jenkins/secrets/initialAdminPassword
#=====
=====#
```

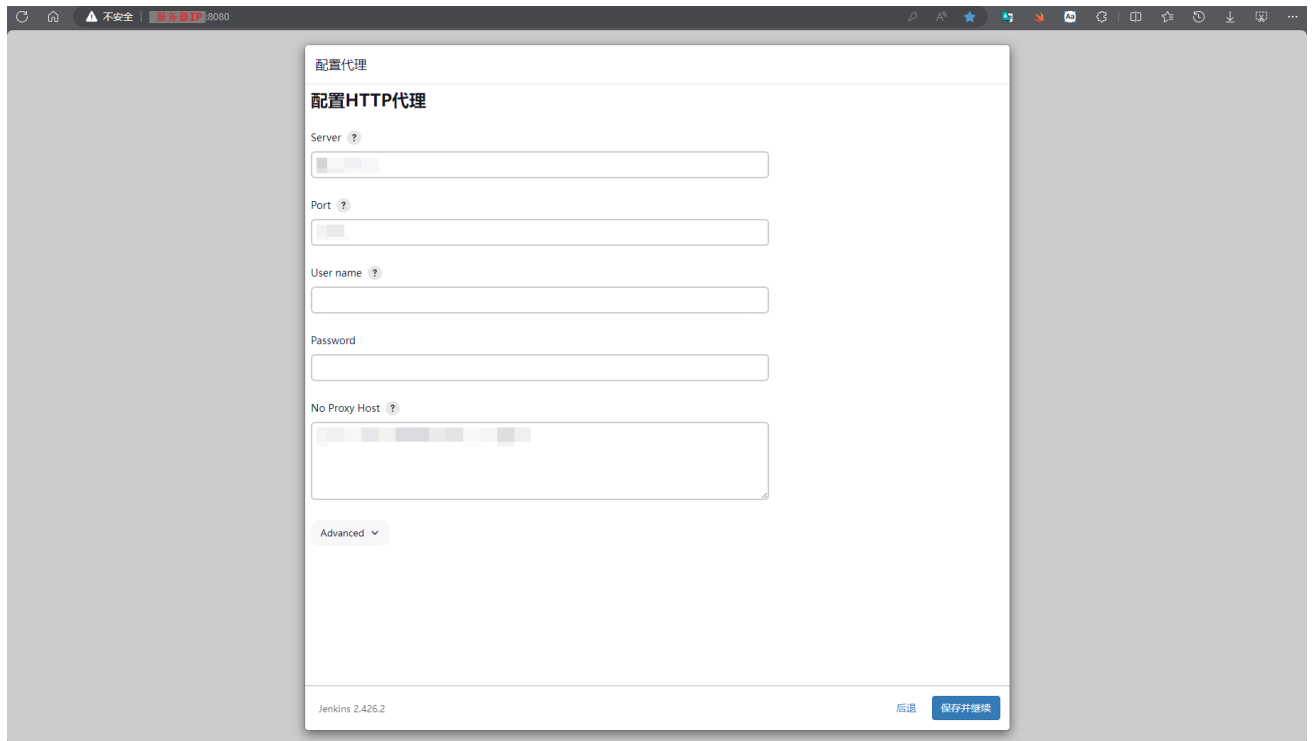
填写初始密码,点击 **继续** 按钮



- 配置代理『【可选】请根据您的网络环境进行设置,如需离线使用可点击 **跳过插件安装** 暂时跳过此步骤 (有关离线安装请查阅 [Jenkins离线安装官方文档](#))』



- 点击 **配置代理** 按钮进入下图界面配置代理,配置完成后点击 **保存并继续** 按钮



配置代理

配置HTTP代理

Server ?

Port ?

User name ?

Password

No Proxy Host ?

Advanced ▾

Jenkins 2.426.2

后退 保存并继续

- 安装推荐的插件『【可选】如您的网络不可用,可跳过以下步骤。点击 **选择插件来安装** 根据提示,跳过安装』



新手入门

自定义Jenkins

插件通过附加特性来扩展Jenkins以满足不同的需求。

安装推荐的插件

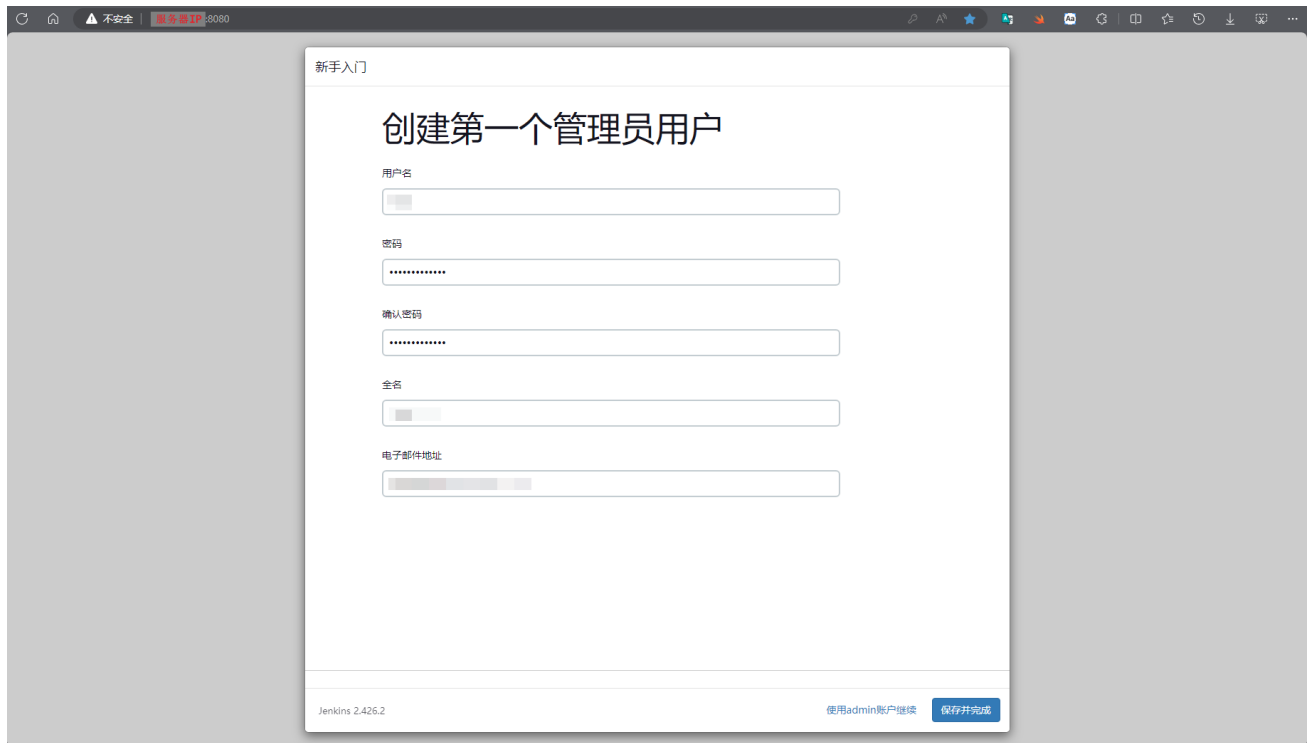
安装Jenkins社区推荐的插件。

选择插件来安装

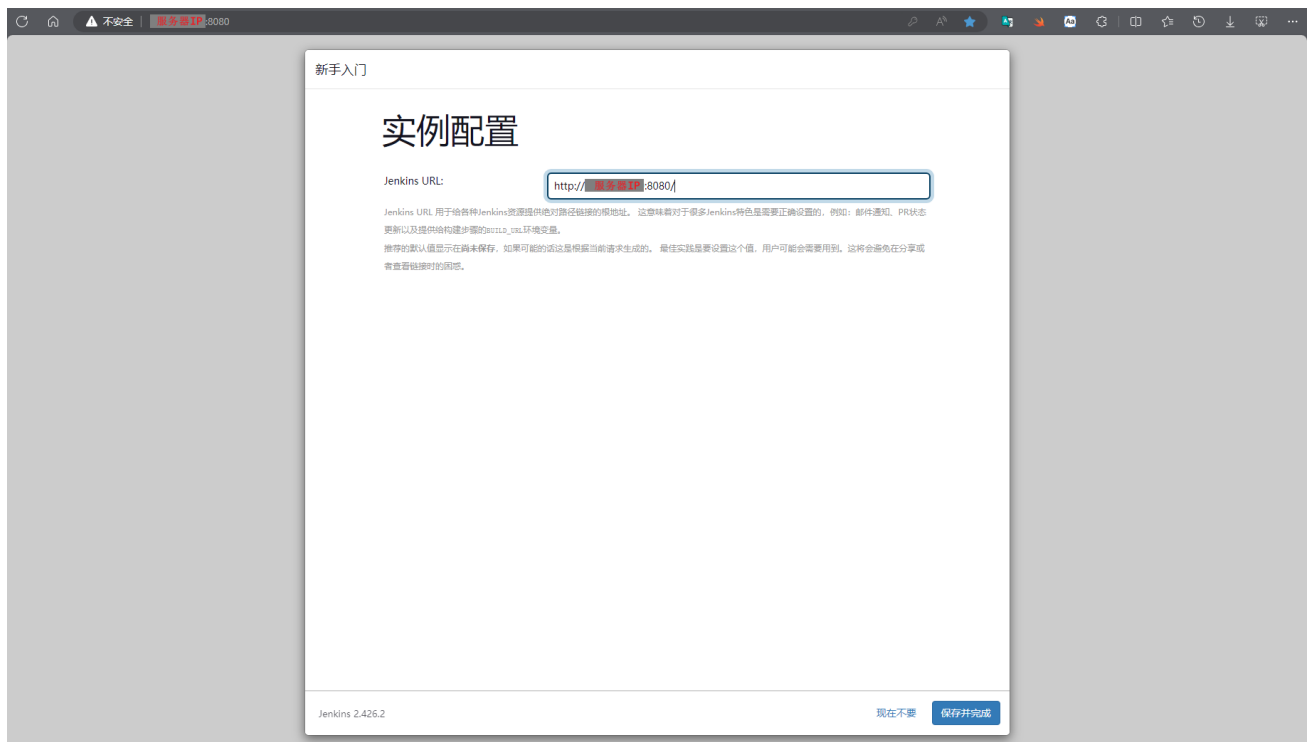
选择并安装最适合的插件。

Jenkins 2.426.2

- 配置管理员用户登录信息『请根据自己的实际需求填写信息,填写完成后请点击 **保存并完成按钮**』



- Jenkins 实例配置『请根据自己的实际需求填写,此处使用默认设置.填写完成后请点击 **保存并完成** 按钮』



- Jenkins安装已完成『请点击 **开始使用Jenkins** 按钮』



#### 4.Jenkins 基础插件安装『插件的离线安装方式请访问插件主页下载符合目标要求的插件安装包 (.hpi)』

有关 Jenkins 插件管理请阅读以下文章：[管理插件 \(jenkins.io\)](https://jenkins.io/doc/book/managing-plugins/)

- 安装 **Blue Ocean** 插件『可选 (以更直观的的方式查看 pipeline 状态)』

Dashboard > 系统管理 > 插件管理

### Plugins

Available plugins

Search: Blue Ocean

安装 名称 ↓ Released

<input checked="" type="checkbox"/>	<b>Blue Ocean</b> 1.27.9 外部工具集成 用户界面 BlueOcean Aggregator	2月2天 ago
<input type="checkbox"/>	<b>Common API for Blue Ocean</b> 1.27.9 外部工具集成 用户界面 This plugin is a part of Blue Ocean UI	2月2天 ago
<input type="checkbox"/>	<b>REST API for Blue Ocean</b> 1.27.9 外部工具集成 用户界面 This plugin is a part of Blue Ocean UI	2月2天 ago
<input type="checkbox"/>	<b>Design Language</b> 1.27.9 Jenkins Design Language Plugin. This plugin is a part of the Blue Ocean Plugin set.	2月2天 ago
<input type="checkbox"/>	<b>Blue Ocean Core JS</b> 1.27.9 Blue Ocean Core JS Plugin. This plugin is a part of the Blue Ocean Plugin set.	2月2天 ago
<input type="checkbox"/>	<b>Web for Blue Ocean</b> 1.27.9 外部工具集成 用户界面 Blue Ocean core	2月2天 ago
<input type="checkbox"/>	<b>JWT for Blue Ocean</b> 1.27.9 外部工具集成 用户界面 BlueOcean JWT plugin: Enables JWT based BlueOcean API authentication	2月2天 ago
<input type="checkbox"/>	<b>Pipeline SCM API for Blue Ocean</b> 1.27.9 This plugin is a part of BlueOcean Plugin	2月2天 ago
<input type="checkbox"/>	<b>REST Implementation for Blue Ocean</b> 1.27.9 外部工具集成 用户界面	2月2天 ago

Dashboard > 系统管理 > 插件管理

Download progress

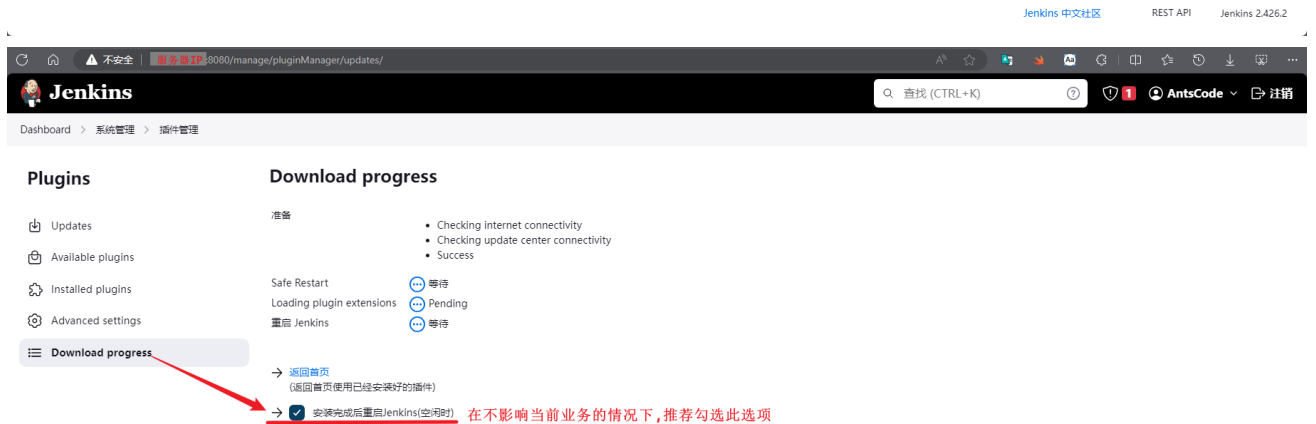
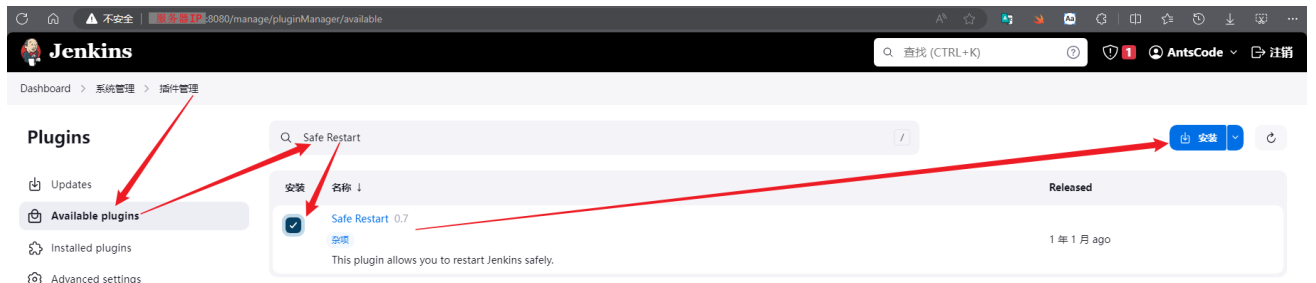
Blue Ocean Core JS	完成
Common API for Blue Ocean	完成
REST API for Blue Ocean	完成
Pub-Sub "light" Bus	完成
Pipeline SCM API for Blue Ocean	完成
HTML Publisher	完成
Web for Blue Ocean	完成
JWT for Blue Ocean	完成
Favorite	完成
REST Implementation for Blue Ocean	完成
Pipeline Implementation for Blue Ocean	完成
GitHub Pipeline for Blue Ocean	完成
Git Pipeline for Blue Ocean	等待
Config API for Blue Ocean	等待
Authentication Tokens API	等待
Joda Time API	等待
Handy Uri Templates 2x API	等待
Bitbucket Branch Source	等待
Bitbucket Pipeline for Blue Ocean	等待
Dashboard for Blue Ocean	等待
Personalization for Blue Ocean	等待
Display URL for Blue Ocean	等待
Server Sent Events (SSE) Gateway	等待
Events API for Blue Ocean	等待
Blue Ocean Pipeline Editor	等待
i18n for Blue Ocean	等待
Blue Ocean	等待
Loading plugin extensions	Pending
重启 Jenkins	等待

→ 返回首页  
(返回首页使用已经安装好的插件)

→ ☒ 安装完成后重启Jenkins(空闲时)

在不影响当前业务的情况下,推荐勾选此选项

## • 安装 Safe Restart 插件『可选 (安全重启 Jenkins)』



# 将各个执行机添加至Jenkins集群

## 1.凭证设置


添加凭据域



凭据

类型	提供者	存储 ↓	域	唯一标识	名称
----	-----	------	---	------	----

Stores scoped to Jenkins

提供者	存储 ↓	域
 System		全局

图标: 小 中 大

添加域

域名和描述按需输入，便于识别和管理即可

New domain

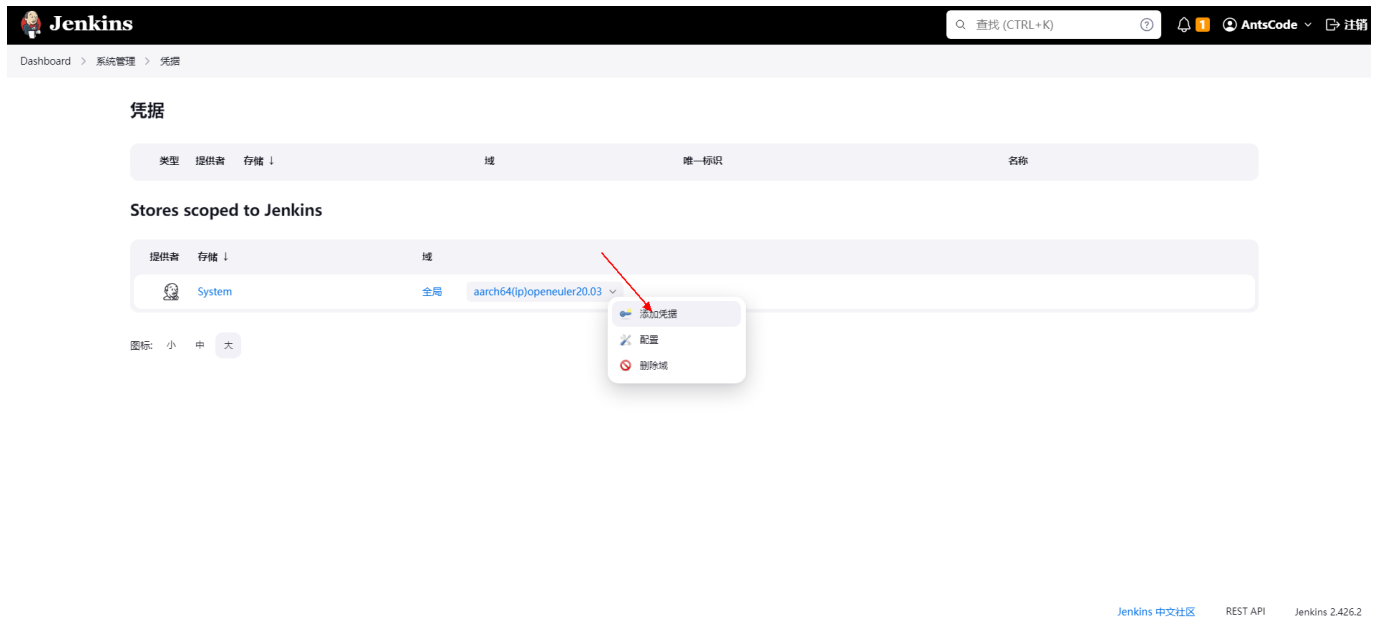
域名 ?  
aarch64(p)openeuler20.03

描述 ?  
builder

规范 ?  
新增

Create

添加凭据到凭据域下



## 在 DevKit 凭据域下添加 aarch64 Jenkins 工作节点 SSH 凭据

```
#=====
#####
# 在安装了 Jenkins 的环境上生成工作节点 SSH 免密登录证书『请根据实际需求设置 SSH key
phrases』
ssh-keygen -b 4096 -C "<邮件地址或其他标签>" -f ~/.ssh/id_ed25519_<推荐按照 *_*_*_* 格式
填写目标服务器IP,便于管理 KEY> -t ed25519
#-----
-----#
# 在安装了 Jenkins 的环境上将生成的证书的公钥上传至目标服务器『请根据提示输入目标服务器对应
账户密码』
ssh-copy-id -i ~/.ssh/id_ed25519_<推荐按照 *_*_*_* 格式填写目标服务器IP,便于管理
KEY>.pub root@<目标服务器IP>
#-----
-----#
# 删除已知主机名文件中属于指定主机名的所有密钥
ssh-keygen -R <目标服务器IP>
#-----
-----#
# 使用 SSH key 测试连接目标主机『如果您设置了 SSH key phrases,请在连接目标主机时根据提示
输入证书密码』
ssh -o IdentitiesOnly=yes -o PasswordAuthentication=no -i ~/.ssh/id_ed25519_<推荐按照
*_*_*_* 格式填写目标服务器IP,便于管理 KEY> -l root -p 22 <目标服务器IP>
#-----
-----#
# 查看 SSH key 私钥
cat ~/.ssh/id_ed25519_<推荐按照 *_*_*_* 格式填写目标服务器IP,便于管理 KEY>
#=====
#####
```

不安全

8080/manage/credentials/store/system/domain/DevKit/newCredentials

🔍 查找 (CTRL+K)

AntsCode 注册

Dashboard > 系统管理 > 凭据 > 系统 > DevKit >

New credentials

类型SSH Username with private key

范围全局 (Jenkins, nodes, items, all child items, etc.)

ID

描述aarch64 worker node

Usernameroot

☒ Treat username as secret

Private KeyEnter directly

Key

在下面输入新秘文

通过 SSH key 加密连接目标主机的私钥  
-----END OPENSSH PRIVATE KEY-----

Passphrase如果您的 SSH key 设置了 passphrases 则需要填写此选项

Create

Jenkins 中文社区

REST API

Jenkins 2.426.2

## 2. 工作节点设置

Nodes

Clouds

构建队列

构建执行状态

节点列表

+ New Node

Node Monitoring

🔄

S	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 (aarch64)	已同步	55.91 GB	7.87 GB	7.18 GB	0ms
	获取到的数据	46 分	46 分	46 分	46 分	46 分	46 分

队列中没有构建任务

1 空闲

2 空闲

### New node

节点名称

WorkerNode\_aarch64

Type

🔵 固定节点

添加一个普通、固定的节点到Jenkins。之所以叫做“固定”，是因为Jenkins没给这种节点提供更高级的集成方式，例如：动态配置。没有其他类型能选择的话可以选择该类型；例如，你在添加不受Jenkins管理的物理机、虚拟机等等。

Create

[Jenkins 中文社区](#) [REST API](#) [Jenkins 2.426.2](#)

配置项	配置说明
名字	与节点名称保持一致
描述	按需填写,便于管理标识和即可，如aarch64node(ip)openeuler22.03
Number of executors	默认为1
远程工作目录	/home/JenkinsWorkspace/
标签	流水线脚本中根据标签来选取执行机，可以打多个标签，用空格隔开，标签需要以kunpeng_为前缀，如kunpeng_scanner kunpeng_builder kunpeng_executor
用法	Only build jobs with label expressions matching this node
启动方式	Launch agents via SSH
主机	节点IP
Credentials	已添加的凭据
Host Key Verification Strategy	Known hosts file Verification Strategy
可用性	Keep this agent online as much as possible
节点属性(可选)	若需要配置环境变量可选择Environment variables

Dashboard > 系统管理 > 节点列表 >

WorkerNode\_aarch64

描述 ?

aarch64工作节点 (目标服务器IP) (目标主机OS NAME)

纯文本 预览

Number of executors ?

1

远端工作目录 ?

/home/jenkinsWorkspace/

标签 ?

kunpeng\_scanner kunpeng\_builder kunpeng\_executor

用法 ?

只允许运行绑定到这台机器的Job

启动方式 ?

Launch agents via SSH

主机 ?

目标主机IP

Credentials ?

aarch64 worker node (目标主机IP) (目标主机 OS NAME)

+添加

Host Key Verification Strategy ?

Known hosts file Verification Strategy

高级

可用性 ?

尽量保持代理在线

节点属性

☐ Disable deferred wipeout on this node ?

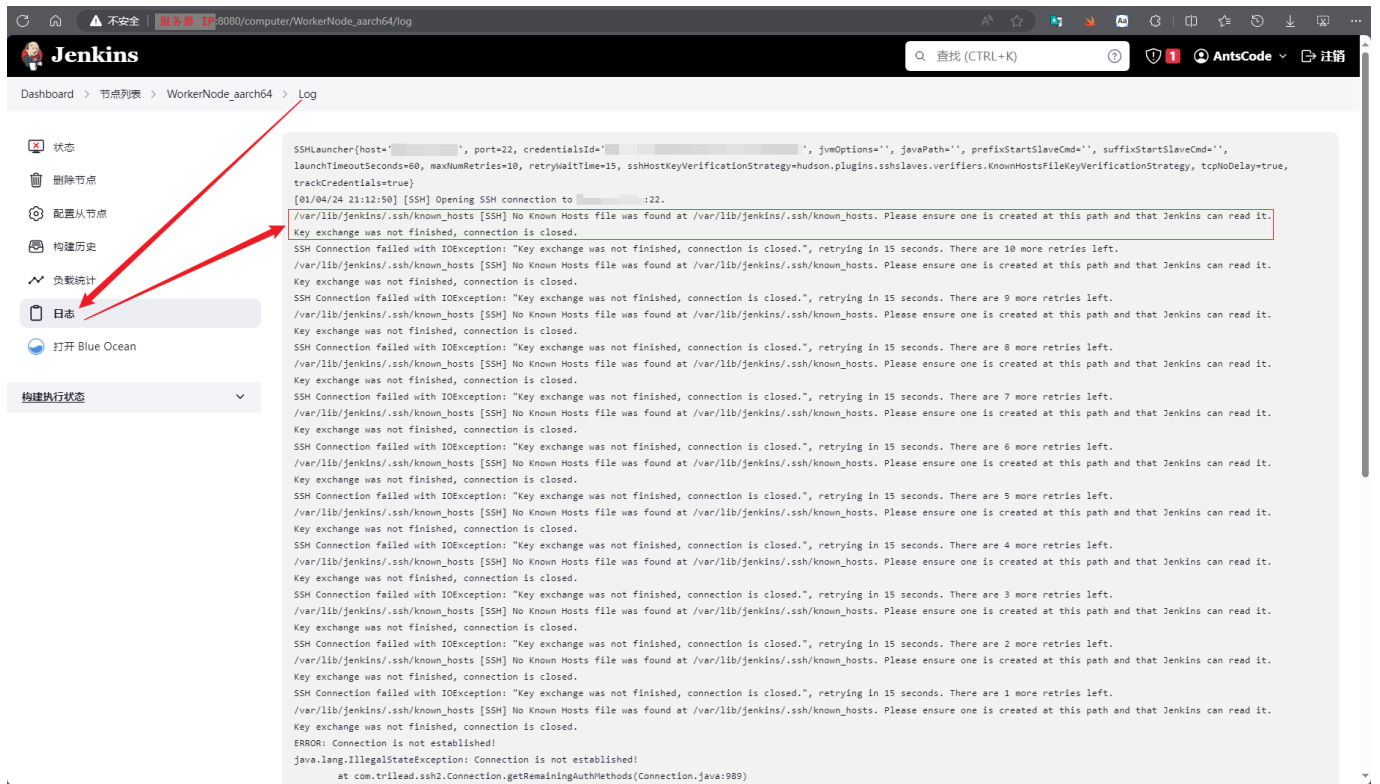
☐ 工具位置

☐ 环境变量

保存

Jenkins 中文社区 REST API Jenkins 2.426.2

### 3.FAQ



当 Jenkins 工作节点连接不上时,且查看日志如上图所示时请考虑通过以下解决方案解决此问题。

有关 Jenkins 工作节点连接错误的问题可参考以下文章中 poddingue 的解答:  
<https://community.jenkins.io/t/node-connection-error/6082>

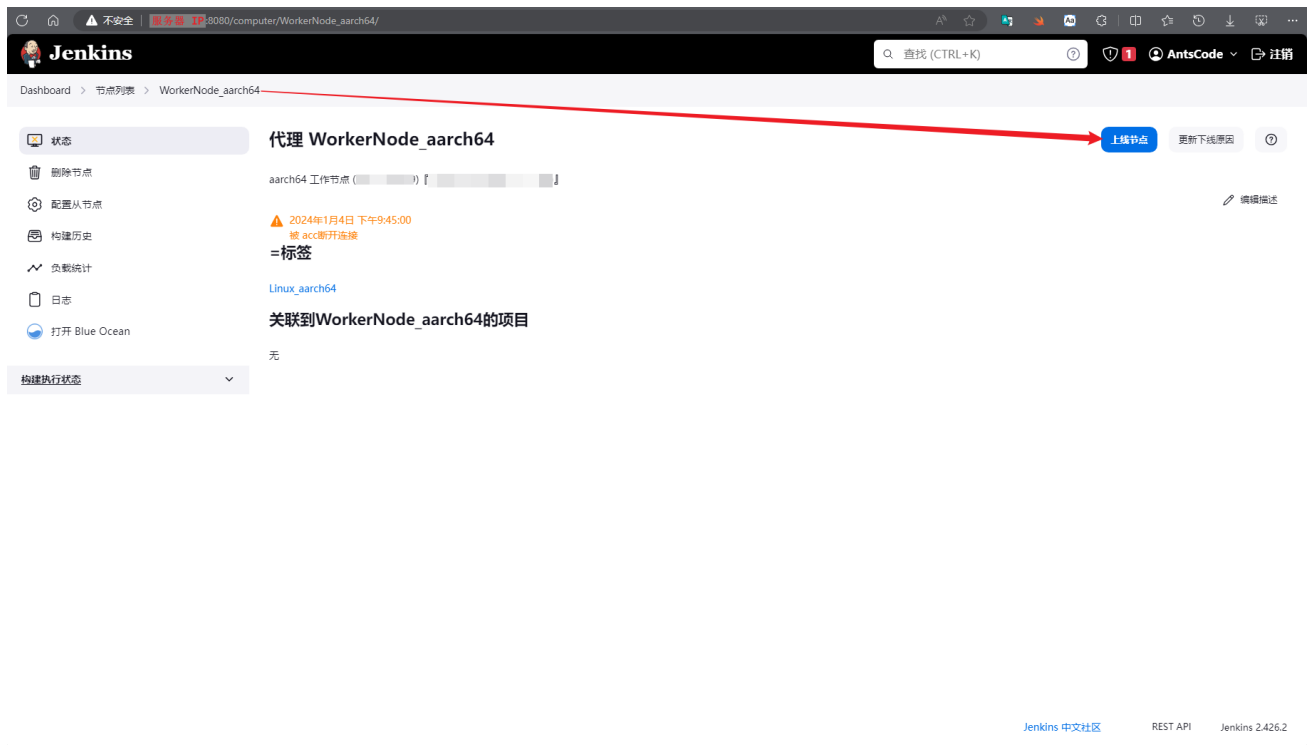
## • 临时断开节点



- 在安装 Jenkins 服务的设备上配置 `/var/lib/jenkins/.ssh/known_hosts`

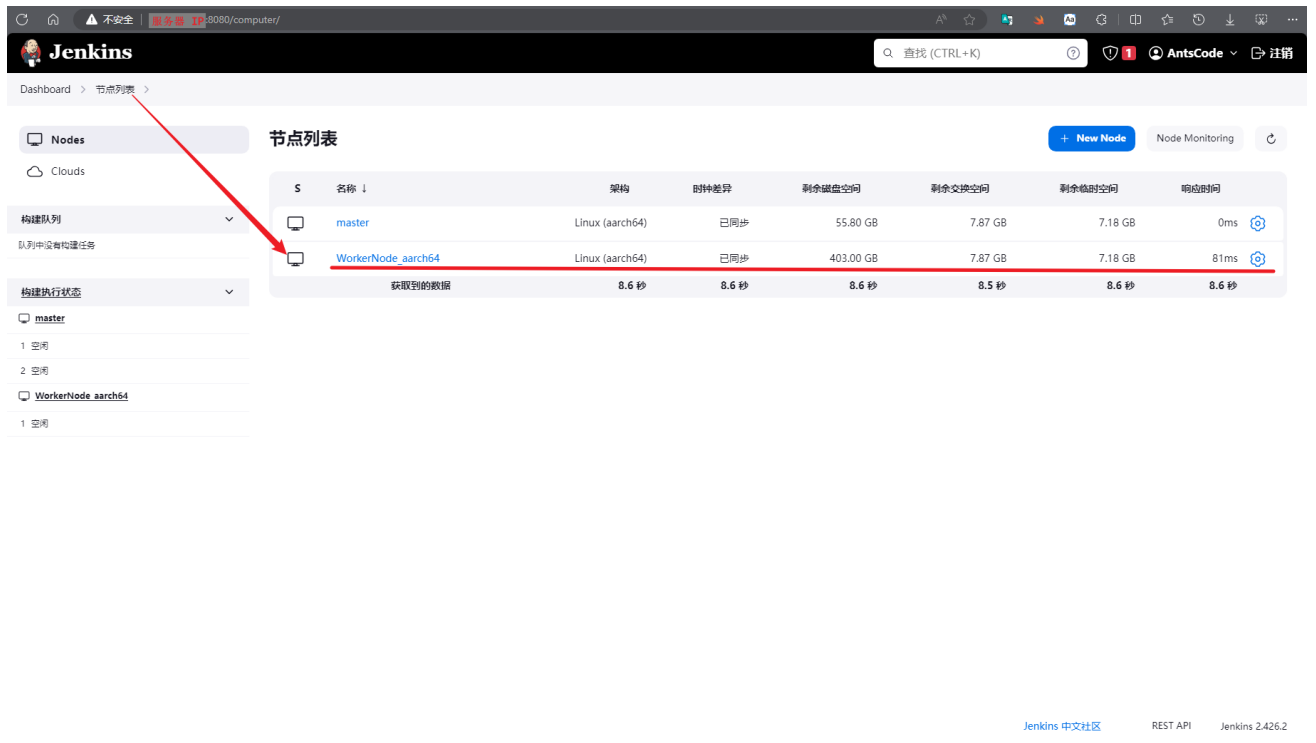
```
#=====
#####
# 创建目标 /var/lib/jenkins/.ssh 目录
mkdir -p /var/lib/jenkins/.ssh
#-----#
# 新建 known_hosts 文件
touch /var/lib/jenkins/.ssh/known_hosts
#-----#
# 修改 known_hosts 文件权限为 600
chmod 600 /var/lib/jenkins/.ssh/known_hosts
#-----#
# 将远程主机的 SSH 主机密钥添加到 known_hosts 文件中
ssh-keyscan <目标服务器IP> >> /var/lib/jenkins/.ssh/known_hosts
#-----#
# 修改文件夹下所有文件的所属用户及用户组为 jenkins
chown -R jenkins:jenkins /var/lib/jenkins/.ssh
#=====
=====#
```

- 重新连接节点



- 节点连接成功后如下图所示





### 3. 修改CSP策略从而保证html功能正常

#### 原因

jenkins中使用htmlpublisher来显示报告，由于默认的CSP策略限制，会导致html功能异常

#### 1. 修改方式

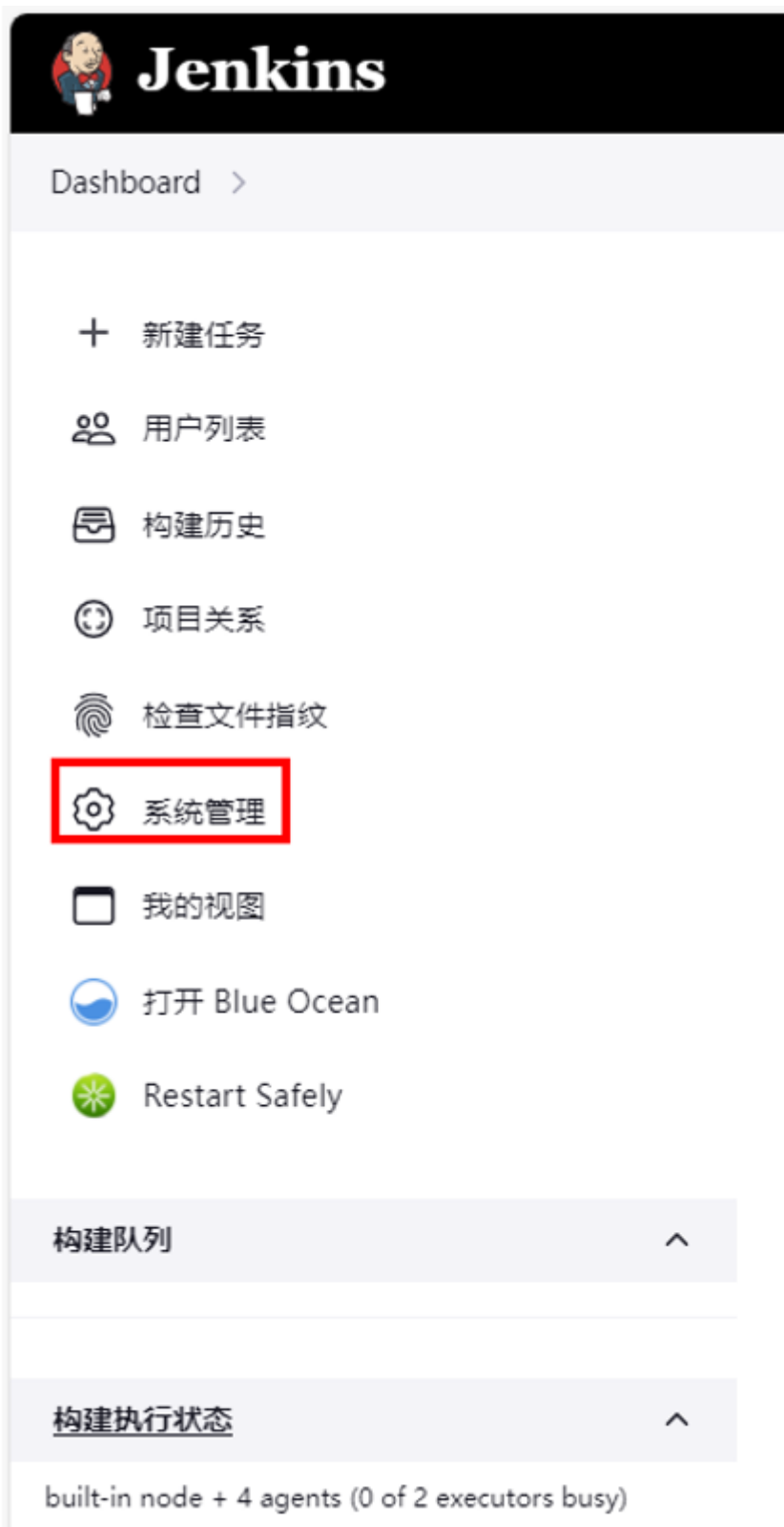
需要通过执行以下命令修改CSP策略

```
System.setProperty("hudson.model.DirectoryBrowserSupport.CSP","script-src 'self' 'unsafe-inline';style-src 'self' 'unsafe-inline';img-src 'self' data;");
```

上面的值为cli报告能正常显示的最小集，如果其他功能也需要修改该配置，可以取相应值的并集，也可以设置值为空字符串（但可能会有安全风险，不推荐）

#### 2. 手动临时修改（重启jenkins后失效，需要再次手动执行命令）





进入系统管理 -> 脚本命令行

状态信息



系统信息

显示系统环境信息以帮助解决问题。



系统日志

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。




关于Jenkins

查看版本以及证书信息。

问题排查

 **管理旧数据**  
从旧的、早期版本的插件中清理配置文件。

## 工具和动作

 **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时重  
新读取设置。

 **Jenkins 命令行接口**  
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。

 **Restart Safely**  
Restart once no jobs are running.

 **准备关机**  
停止执行新的构建任务以安全关闭计算机。

在输入框中添加命令后，点击运行即可（Result中的内容为修改前的值，可以再次点击运行查看是否修改成功）

### Script Console

Type in an arbitrary [Groovy script](#) and execute it on the server. Useful for trouble-shooting and diagnostics. Use the 'println' command to see the output (if you use `System.out`, it will go to the server's stdout, which is harder to see.) Example:

```
println(Jenkins.instance.pluginManager.plugins)
```

All the classes from all the plugins are visible. `jenkins *`, `jenkins.model *`, `hudson *`, and `hudson.model *` are pre-imported

```
1 System.setProperty("hudson.model.DirectoryBrowserSupport.CSP", "script-src 'self' 'unsafe-inline'; style-src 'self' 'unsafe-inline'; img-src 'self' data;");
```

运行


### Result

Result: script-src 'self' 'unsafe-inline'; style-src 'self' 'unsafe-inline'; img-src 'self' data;;





修改后查看cli报告即可(对修改前创建的报告同样有效，若仍然无效，可能等待1~2分钟后重试)

## 3. 永久自动修改（启动时自动执行）

1. 安装依赖 需要安装groovy、startup trigger插件 离线安装地址： groovy: [Groovy|Jenkins plugin](#) startup-trigger: [Startup Trigger|Jenkins plugin](#)

 Jenkins

Q 查找 (CTRL+K)

   AntsCode  注销

Dashboard > 系统管理 > 插件管理

Plugins

Updates

Available plugins

Installed plugins

Advanced settings

Advanced settings

代理设置

服务器 ?

端口 ?

用户名 ?

密码

不通过代理的主机 ?

高级

提交

部署插件

您可以从本地系统中选择一个插件文件，或者提供一个 URL 从中央插件仓库以外的位置安装插件。

文件

选择文件 未选择文件

## 2-1. 新建一个Job


输入一个任务名称

CSPInit

» 必填项

 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系统.

 构建一个maven项目

## 2-2. 选择在jenkins启动后执行

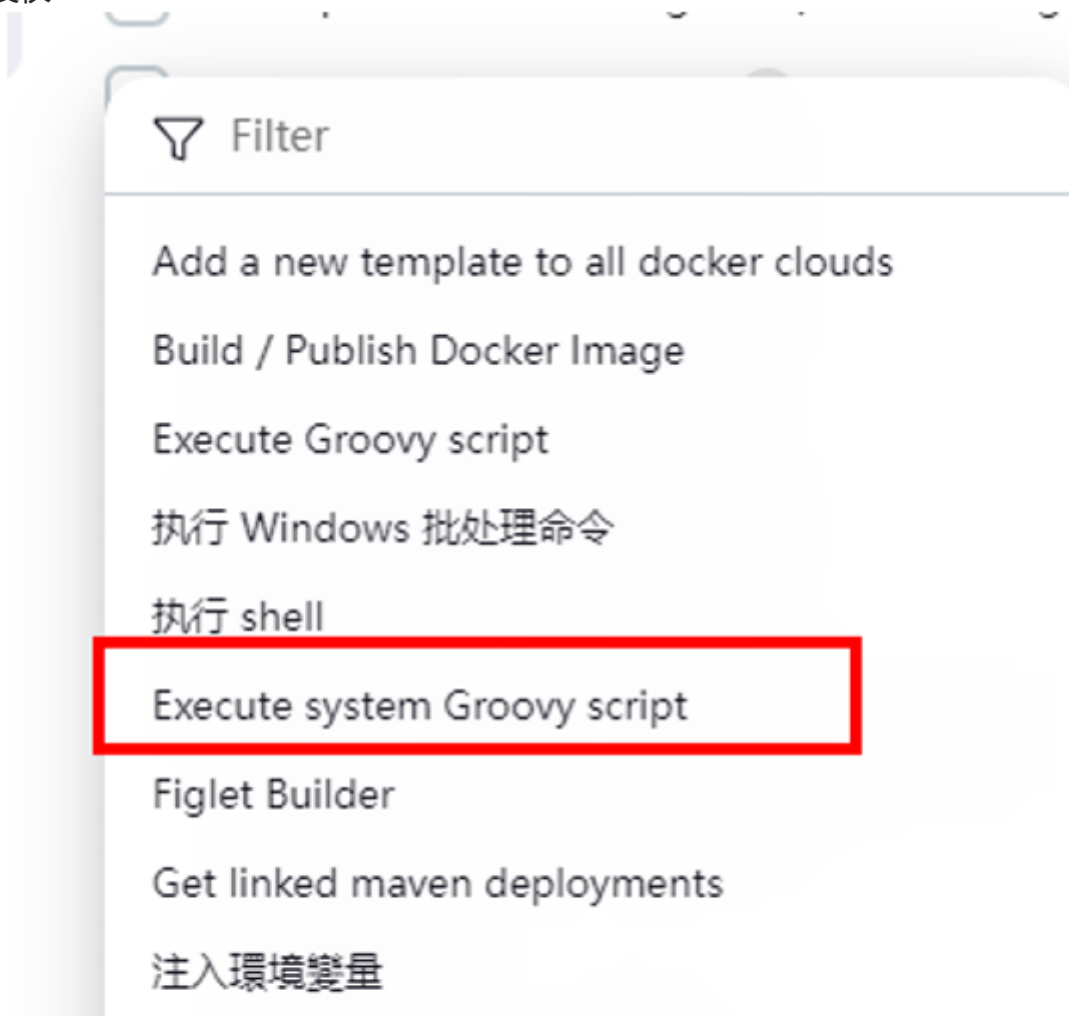
**构建触发器**

- ☐ 触发远程构建 (例如,使用脚本) ?
- ☐ 其他工程构建后触发 ?
- ☐ 定时构建 ?
- ☐ Build when a change is pushed to CodeHub. CodeHub webhook URL: http://10.175.118.214:8080/project/CSPLnit ?
- ☒ Build when job nodes start
  - Restricted node Label ?
  - Quiet period ?

高级 ▾

- ☐ Enable Artifactory trigger
- ☐ Generic Webhook Trigger ?
- ☐ GitHub hook trigger for GITScm polling ?
- ☐ 轮询 SCM ?

## 2-3. 构建步骤添加 Execute system Groovy script (注意不是Execute Groovy script), 填写脚本并授权



Invoke Ant

Invoke Artifactory Maven 3

Invoke Gradle script

调用顶层 Maven 目标

PortingAdvisor

Provide Configuration files

Run with timeout

Set build status to "pending" on GitHub commit

Start/Stop Docker Containers

增加构建步骤 ^

## Build Steps

≡ Execute system Groovy script ?

Groovy command

Groovy Script

System.setProperty("hudson.model.DirectoryBrowserSupport.CSP","script-src 'self' 'unsafe-inline';style-src 'self' 'unsafe-inline';img-src 'self' data:");

The script is already approved

☐ Use Groovy Sandbox ?

Additional classpath ?  
Add entry

高级 v

增加构建步骤 v

2-4. 点击保存即可，可以手动触发一次job，后续jenkins重启后会自动执行