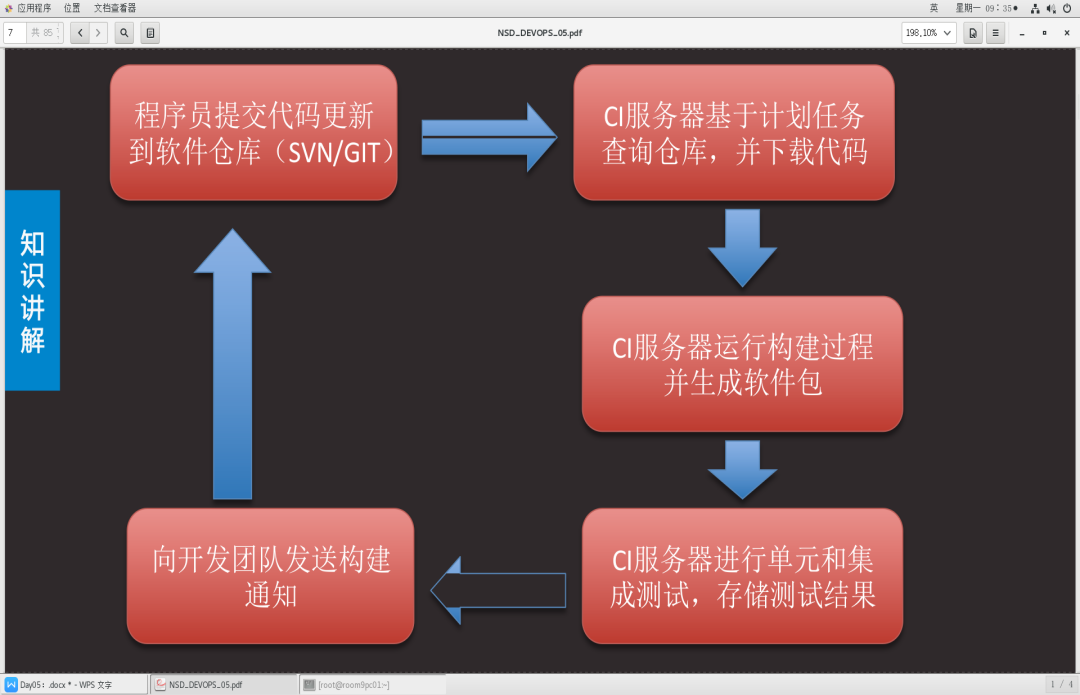
一、Jenkins简介

**1、CI流程图**

为了解决开发、测试、运维部署环境的不一致，将生产环境做成docker镜像。

程序员把源码上传github，每一组服务器群由唯一的分发服务器(Jenkins)下载，再批量分发。

用maven将java项目打包成war包，部署到tomcat服务器



CI/CD：持续集成/持续交付（部署）

Jenkins常用于构建和测试各种项目，主要功能是监视重复工作的执行。

1. **安装Jenkins**

1）环境准备：

gitlab 192.168.1.1

jenkins 192.168.1.3（需要能上网）

1. 安装jenkins

[root@jenkins ~]# yum -y install java-1.8.0-openjdk

[root@jenkins ~]# rpm -ivh jenkins-2.121.1-1.1.noarch.rpm

[root@jenkins ~]# systemctl start|enable jenkins

浏览器访问http://192.168.1.3:8080，不安装任何插件

1. 通过国内镜像站点安装插件

https://blog.csdn.net/ezreal\_tao/article/details/82528185获取升级站点

把jenkins首页--系统管理--管理插件--高级--修改"升级站点"

在"可选插件"中找到'Git Parameter'，直接安装

#Git Parameter插件可以让Jenkins使用git的tag

二、准备git仓库

**1、本地仓库**

[root@room8pc16 ~]# git init /tmp/myweb

[root@room8pc16 myweb]# echo '<h1>my web</h1>' > index.html

[root@room8pc16 myweb]# git add .

[root@room8pc16 myweb]# git commit -m "myweb site init"

[root@room8pc16 myweb]# git tag 1.0 #把当前状态标记为1.0版本

[root@room8pc16 myweb]# echo '<h2>new line</h2>' >> index.html

[root@room8pc16 myweb]# git add .

[root@room8pc16 myweb]# git commit -m "mysite version 2.0"

[root@room8pc16 myweb]# git tag 2.0 #把当前状态标记为2.0版本

[root@room8pc16 myweb]# git tag #查看所有的版本

[root@room8pc16 myweb]# git checkout 1.0 #本地切换到V1.0

[root@room8pc16 myweb]# cat index.html

<h1>my web site</h1>

[root@room8pc16 myweb]# git checkout master #本地切换回最新版本

1. **远程仓库**

参考1：阶段5 -> 1.PYTHON1 -> Day01 -> 三、git的使用，4~5

参考2：PPT（远程仓库） 创建群组->创项目->创用户->上传代码->查看项目

1. jenkins应用

**1、构建项目（CI）**

1）新建任务 -> 输入名称 -> 构建一个自由风格的软件项目 -> 确定

2）勾选"参数化构建过程" -> 定义形参名'Name'=webver

3）源码管理勾选"Git" -> "Repository URL"填写gitlab服务器项目地址

-> "Branch Specifier"填写$webver

-> "Additional Behaviours" 选择Check out to a sub-dir 填写子目录名为mysitesite\_$webver

#注意：jenkins服务器需要安装git命令

#构建项目默认保存目录：/var/lib/jenkins/workspace

4）jenkins安装httpd，用于存放gitlab下载的代码，以便分发

5）构建 -> 增加构建步骤选择"执行shell"

deploy\_dir=/var/www/html/deploy

cp -r mysite\_$webver ${deploy\_dir}/packages

cd ${deploy\_dir}/packages

rm -rf mysite\_$webver/.git

#将构建的项目拷贝到web目录下，删除.git版本库文件

tar -czf mysite\_${webver}.tar.gz mysite\_$webver

md5sum mysite\_${webver}.tar.gz | awk '{print $1}' > mysite\_${webver}.tar.gz.md5

rm -rf mysite\_${webver}

#创建压缩包、md5文件，然后删除源目录

[ -f ${deploy\_dir}/live\_version ] && cat ${deploy\_dir}/live\_version > ${deploy\_dir}/last\_version

echo $webver > ${deploy\_dir}/live\_version

#创建live\_version、last\_version说明文档

**2、自动化部署（CD）**

**1）需求分析**

-目录规划：

/var/www/download存储下载的应用

/var/www/deploy存储解压的应用、版本说明

/var/www/html/current软连接，指向需要部署的应用版本

-下载应用：

sys.argv[1] = 'live' / 'last' / '--version'

'live'下载当前版本、'last'下载前一个版本、'--version'显示本地与服务器版本

-校验文件：

将下载的tar包进行md5计算

将md5值与服务器发布的md5比较，无误则执行解压

-发布应用：

根据指定的版本，创建/var/www/html/current连接

**2）Python框架搭建**

def get\_version():

#打印本地版本、服务器live/last版本

def has\_update(version\_url, version\_fname):

#查看本地版本是否为正确版本

#是则返回None，否则返回需要部署的版本号

def get\_md5(apptar\_fname):

#返回下载的tar包md5计算值

def deploy(apptar\_fname, deploy\_path):

#将tar包解压到deploy\_path

#重新指定网站软连接

**3）Python源码**

#!/usr/bin/env python3

"网站应用自动化版本控制"

#本脚本分发到所有web服务器，计划任务自动执行

import sys

import os

import requests

import wget

import hashlib

import tarfile

import shutil

def get\_version():

#打印本地版本、服务器live/last版本

livever\_url = 'http://192.168.1.30/deploy/live\_version'

lastver\_url = 'http://192.168.1.30/deploy/last\_version'

version\_fname = '/var/www/deploy/live\_version'

if not os.path.exists(version\_fname):

local\_version = 'None'

else:

with open(version\_fname) as fobj:

local\_version = fobj.read().strip()

r = requests.get(livever\_url)

web\_livever = r.text.strip()

r = requests.get(lastver\_url)

web\_lastver = r.text.strip()

print('本地版本:%s\n服务器live版本:%s\n服务器last版本:%s' % (local\_version, web\_livever, web\_lastver))

def has\_update(version\_url, version\_fname):

#查看本地版本是否为正确版本

#是则返回None，否则返回需要部署的版本号

r = requests.get(version\_url)

web\_version = r.text.strip()

if not os.path.exists(version\_fname):

return web\_version

with open(version\_fname) as fobj:

local\_version = fobj.read().strip()

if local\_version != web\_version:

return web\_version

else:

return None

def get\_md5(apptar\_fname):

#返回下载的tar包md5计算值

m = hashlib.md5()

with open(apptar\_fname, 'rb') as fobj:

while True:

data = fobj.read(4096)

if not data:

break

m.update(data)

return m.hexdigest()

def deploy(apptar\_fname, deploy\_path):

#将tar包解压到deploy\_path

#重新指定网站current软连接

app\_path = os.path.join(deploy\_path, apptar\_fname.split('/')[-1].replace('.tar.gz', ''))

print(app\_path)

if os.path.exists(app\_path):

shutil.rmtree(app\_path)

os.chdir(deploy\_path)

tar = tarfile.open(apptar\_fname, 'r:gz')

tar.extractall()

tar.close()

link\_path = '/var/www/html/current'

if os.path.islink(link\_path):

os.unlink(link\_path)

os.symlink(app\_path, link\_path)

if \_\_name\_\_ == '\_\_main\_\_':

#判断命令行输入python3 %s {live|last|--version}

if len(sys.argv) != 2 or sys.argv[1] not in ['live', 'last', '--version']:

print("Usage: python3 %s {live|last|--version}" % sys.argv[0])

exit(1)

download\_path = '/var/www/download'

deploy\_path = '/var/www/deploy'

#如果输入--version，返回版本信息

input\_ver = sys.argv[1].lower()

if input\_ver == '--version':

get\_version()

exit()

#如果输入live/last，判断版本是否正确

#不正确则下载并覆盖到download\_path

version\_url = 'http://192.168.1.30/deploy/%s\_version' % input\_ver

version\_fname = deploy\_path + '/live\_version'

is\_version = has\_update(version\_url, version\_fname)

if is\_version == None:

get\_version()

print('\033[5;31m不需要更新!\033[0m')

exit()

if os.path.exists(version\_fname):

os.remove(version\_fname)

wget.download(version\_url, version\_fname)

app\_url = 'http://192.168.1.30/deploy/packages/mysite\_%s.tar.gz' % is\_version

apptar\_fname = os.path.join(download\_path, app\_url.split('/')[-1])

if os.path.exists(apptar\_fname):

os.remove(apptar\_fname)

wget.download(app\_url, apptar\_fname)

#对下载的tar包进行md5校验

#检验成功，执行覆盖安装与连接

local\_md5 = get\_md5(apptar\_fname)

md5\_url = app\_url + '.md5'

r\_md5 = requests.get(md5\_url)

if local\_md5 != r\_md5.text.strip():

print('md5校验失败')

exit(1)

deploy(apptar\_fname, deploy\_path)