


企业级Git Server服务的部署与应用

2017-08-15 10:13

阅读 4.4k

评论 0

 Python运维圈

开源的普及，让使用Git来进行版本控制管理的开发者、团队、企业也越来越多。

市场上，无论国内外，提供的免费的Git托管服务商也越来越多。Github当然首当其冲，还有Gitlab、Bitbucket等。国内基于Git的代码托管也非常多，如果是面向企业，收费其实都不低。最重要的，也不是收费问题，源代码对于企业来说是机密等级非常高的东东，放在自己的服务器上安全等级相对来说要高一点。所以企业就需要有自己内部的Git服务解决方案了。

今天来说说企业级的Git应用部署。

1安装Git

当然先说Git的安装，这里都以CentOS 6.x 64位操作系统为例。

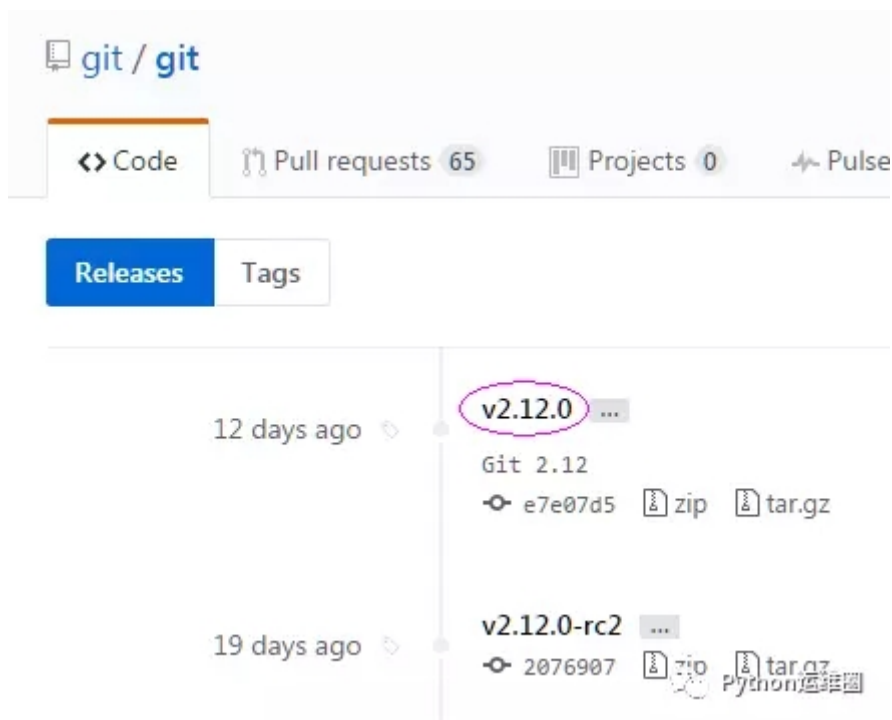
首先去下载Git的源代码，当然从下面的网址去下载：

<https://github.com/git/git/releases>

或者：

<https://www.kernel.org/pub/software/scm/git>

目前可以看到Git的最新版已经发布到了2.12，我们就以这个Releases版本为例（懂C和C++的童鞋可以去折腾下Git的开发版）。



Git软件版本的迭代更新还是发布的比较频繁的。下载了Git源码（从Github下载可能还是需要花费一些时间，原因你懂的），先别急着解压，先来另一件事。

2安装系统依赖包

直接使用yum进行安装：

```
yum install perl-ExtUtils-MakeMaker expat expat-devel xmlto perl-ExtUtils-MakeMakerasciidoc
```

如果有些软件提示找不到的，给CentOS装上epel的Yum源，即可安装成功。

3配置编译

拿到源代码之后就行就是下面的三步了：

- 解压
- 编译
- 安装

这里重点说说编译配置，来看看配置参数：

```
./configure --prefix=/usr/local/git \  
--with-gitconfig=/usr/local/git/etc/gitconfig \  
--with-gitattributes=/usr/local/git/etc/gitattributes \  
--with-editor=/usr/bin/vim \
```

```
-with-expat \  
  
-with-shell=/bin/bash \  
  
-with-perl=/usr/bin/perl \  
  
-with-zlib=/usr \  
  
-with-curl \  
  
-with-libpcre \  
  
-with-openssl \  
  
-with-iconv
```

这里我们把Git安装到/usr/local/git目录下面，如果有Python环境的也可以通过--with-python指定Python解释器。

在执行完configure配置命令后，编译有两种方式：

常规编译：

```
make all -j 2  
  
make install
```

这样安装，不会编译man文档，编译的时间比较快。

带man文档的安装方式：

```
wget ftp://rpmfind.net/linux/dag/redhat/el6/en/x86_64/extras/RPMS/asciidoc-8.6.9-1.el6.rf.xnoarch.rpm  
  
wget ftp://rpmfind.net/linux/dag/redhat/el6/en/x86_64/dag/RPMS/docbook2x-0.8.8-1.el6.rf.x86_64.rpm  
  
yum localinstall asciidoc-8.6.9-1.el6.rf.xnoarch.rpm docbook2x-0.8.8-1.el6.rf.x86_64.rpm  
  
yum install xmlto texinfo  
  
ln -s /usr/bin/db2x_docbook2texi /usr/bin/docbook2x-texi  
  
make -j 2 all doc info  
  
make install install-gitweb install-doc install-man install-html install-info
```

注意这里，安装man文档时，需要解决几个依赖，否则会报错。这大概就是一个比较完整的编译安装，其实官方也提供man文档的下载，可以不自行编译。

<https://www.kernel.org/pub/software/scm/git/git-manpages-2.12.0.tar.gz>
<https://www.kernel.org/pub/software/scm/git/git-htmldocs-2.12.0.tar.gz>

4Git初始配置

完成了Git的编译安装后，需要进行一些初始设置。如果不设置，会出现一些问题。

```
ln -s /usr/local/git/bin/git /usr/bin/

ln -s /usr/local/lib/libcharset.so.1 /lib/libcharset.so.1

ldconfig

mkdir /usr/local/git/etc

git --version
```

这里有一个动态库的链接配置，需要留意。执行git --version的时候，就可以看到显示为2.12的版本。到这里，git就正式可用了，但是还是作为客户端来使用的。

5整合Apache

安装了Git之后，就需要整合Apache，来提供基于http的Git服务。Apache服务器可以使用Yum来安装，这里不多赘述。整合Apache配置：

```
<VirtualHost *:80>

    ServerName git.test.com

    ServerAdmin git@test.com

    SetEnv GIT_PROJECT_ROOT /data/git

    SetEnv GIT_HTTP_EXPORT_ALL

    ScriptAlias /usr/local/git/libexec/git-core/git-http-backend/

    <Location />

        AuthType Basic

        AuthName "Test Git Repository"

        AuthUserFile /etc/httpd/conf.d/git.auth

        Require valid-user

    </Location>
```

```
</VirtualHost>
```

注意加粗部分的配置，这一句“/usr/local/git/libexec/git-core/git-http-backend/”就是用来整合Git和Apache的。

另外git.auth是用来做登录授权配置的，配置这授权文件使用htpasswd命令进行配置。配置了git.auth之后，启动Apache，就可以使用http来提供Git服务了。

6创建第一个版本库

我们来创建第一个版本库：

```
cd /data/git  
  
git init -bare TestPorject.git
```

这里还需要注意一个问题，需要把版本库所在目录授权给Apache的运行用户，否则会出现无法进行git push的操作。这是Linux层的文件权限控制问题。

7克隆与提交版本库

这里就属于客户端的操作了。在创建好了上面的第一个版本库后，就可以在客户端进行Git操作了：

```
git clone http://git.test.com/TestPoject.git
```

在Clone的过程中，需要输入在git.auth文件中配置的用户名和密码，这点需要注意。

完成Clone操作了，就看可以在客户端完全使用git来管理你的代码了。

这样把Git部署在自己的服务器上，就完全不限仓库的数量和容量问题，想创建多少就多少。

8配置基础管理Web端

我们在配置好了上面的http服务后，可以使用git clone http://git.test.com/project.git进行代码的克隆，这里虽然提供了http服务，但还不能在浏览器里访问，要实现这一Web基础功能，还需要对Apache进行下一步的配置：

```
<VirtualHost *:81>
```

```
ServerName git.test.com
```

```
ServerAdmin git@test.com
```

```
<Location />
```

```
AuthType Basic

AuthName "Test Git Repository"

AuthUserFile /etc/httpd/conf.d/git.auth

Require valid-user

</Location>

DocumentRoot /var/www/gitweb

<Directory /var/www/gitweb>

Options +ExecCGI +FollowSymLinks +SymLinksIfOwnerMatch

AllowOverride All

order allow,deny

Allow from all

AddHandler cgi-script cgi

DirectoryIndex gitweb.cgi

</Directory>

</VirtualHost>
```

注意这里使用的81端口，不能再用80端了。

上面有一个/var/www/gitweb的目录，这个目录需要放置一些脚本文件：

```
cp -rap /usr/local/git/share/gitweb/* /var/www/gitweb/
```

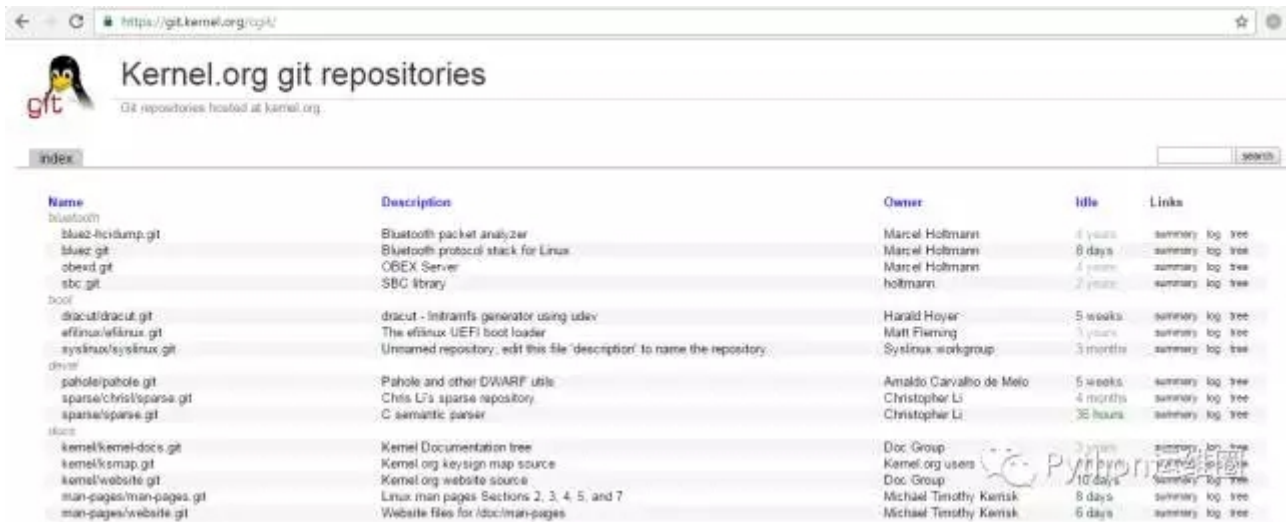
然后需要配置/var/www/gitweb/gitweb.cgi这个文件：

```
our $projectroot = "/data/git";

our $home_link_str = "Projects";
```

这里\$projectroot就需要改成版本库的根目录，这里使用的是/data/git目录。

重启下Apache，然后就可以在浏览器上访问：<http://git.test.com:81/>，这里也是需要输入用户名和密码（也即里git.auth文件里配置的）。



Name	Description	Owner	Title	Links
bluetooth				
bluetooth-hcidump.git	Bluetooth packet analyzer	Marcel Holtmann	4 years	summary log tree
bluetooth.git	Bluetooth protocol stack for Linux	Marcel Holtmann	8 days	summary log tree
obexd.git	OBEX Server	Marcel Holtmann	4 years	summary log tree
obex.git	OBEX library	Holtmann	2 years	summary log tree
boot				
dracut/dracut.git	dracut - initramfs generator using udev	Harald Hoyer	5 weeks	summary log tree
efi/linux/efi/linux.git	The efi/linux UEFI boot loader	Matt Fleming	3 years	summary log tree
syslinux/syslinux.git	Unnamed repository, edit this file 'description' to name the repository	Syslinux workgroup	3 months	summary log tree
driver				
pahole/pahole.git	Pahole and other DWARF utils	Amalio Carvalho de Melo	5 weeks	summary log tree
sparse/chris/sparse.git	Chris Li's sparse repository	Christopher Li	4 months	summary log tree
sparse/sparse.git	C semantic parser	Christopher Li	35 hours	summary log tree
kernel				
kernel/kernel-docs.git	Kernel Documentation tree	Doc Group	3 years	summary log tree
kernel/ksmap.git	Kernel.org key/sign map source	Kernel.org users	3 years	summary log tree
kernel/website.git	Kernel.org website source	Doc Group	10 days	summary log tree
man-pages/man-pages.git	Linux man pages Sections 2, 3, 4, 5, and 7	Michael Timothy Kemsk	8 days	summary log tree
man-pages/website.git	Website files for /doc/man-pages	Michael Timothy Kemsk	6 days	summary log tree

GitWeb的U效果就如上图所示，这里我们使用<https://git.kernel.org/cgi/>来演示。

9 仓库的备份

服务已经部署好了，并且运行起来了。但还有个很重要的问题需要引起注意，那就是：

备份，备份，备份！！

得说三遍才行！！

还记得今年年初**Gitlab丢数据的故障**，这可是国际级的版本托管且比较有影响力的故障事件了。

备份可以使用Shell脚本，设置定时任务来实现备份。主要注意备份下面几个文件或目录：

- /data/git
- /etc/httpd/conf.d/git.auth
- /etc/httpd/conf.d/git.conf

其中/data/git目录就是仓库的根目录了，另外两个文件就是与Apache进行整合的配置文件和Git访问的用户权限控制文件。

备份的方式注意一定要进行异地备份，不要备份在本机，这可是常识性问题了。

到这里，就可以正式启用Git服务来给自己的企业做代码托管了。

10 第三方Git Server的解决方案

这里推荐几款市场上比较火的第三方的开源的Git Server解决方案：

- Gogs
- Gitlab CC

Gogs是基于Go语言开发的，Gitlab基于Ruby开发的。Gitlab比Gogs更重量级、更加麻烦一点，但网上也有一些一键安装的Gitlab的脚本，基本可以成功安装。Gogs来说相对轻量级一点，部署也比较容易一点，同时具备了Github的大部分功能。

关于Gogs和Gitlab的具体部署，这里就不多说了，官方都有提供比较详细的教程和文档，可以参照官方的说明来部署。其实不管是Gogs或者Gitlab，或者自己部署Git http服务，但离不开底层的Git软件。

文章来自微信公众号：Python运维圈