Nginx安装手册

快速开始，windows安装

进入Nginx下载页面

选择nginx/Windows-xxx.zip的安装包下载，xxx表示最新的版本

解压zip到你的程序安装目录

进入nginx-xxx目录，双击nginx.exe启动Nginx服务

非常简单快速，适合学习初用，快速搭建

linux安装

linux下面有两种安装方式，二进制安装、通过源码编译安装。

二进制安装

不同的平台都有支持，具体参考官网说明，这里以centos7为示例，使用yum进行安装。

```
# 检查更新yum依赖
sudo yum install yum-utils
```

添加yum的nginx仓库地址，（可以省略该步骤）

```
sudo vim /etc/yum.repos.d/nginx.repo
# 进入编辑模式，输入下面内容
[nginx-stable]
name=nginx stable repo
baseurl=http://nginx.org/packages/centos/$releasever/$basearch/
gpgcheck=1
enabled=1
gpgkey=https://nginx.org/keys/nginx_signing.key

[nginx-mainline]
name=nginx mainline repo
baseurl=http://nginx.org/packages/mainline/centos/$releasever/$basearch/
gpgcheck=1
enabled=0
gpgkey=https://nginx.org/keys/nginx_signing.key
# 保存即可
```

安装nginx，也可以直接使用该命令进行安装

```
sudo yum install nginx
```

通过源码编译安装

通过源码编译安装，能够集成一些默认没有安装的模块以及第三方插件。下面我们来编译一个携带ssl和echo模块的Nginx。

安装依赖

```
yum install -y gcc       # GCC编译器，用来编译C语言程序
yum install -y gcc-c++       # C++编译器，用来编译C++语言程序
yum install -y pcre pcre-devel       # Perl库兼容正则表达式，Nginx的HTTP模块要靠它来解析正则表达式
yum install -y zlib zlib-devel       # zlib库，用于对HTTP包的内容做gzip格式的压缩
yum install -y openssl openssl-devel     # OpenSSL开发库，用于Http的SSL协议，需要源码编译

# 统一执行前面的命令
sudo yum install -y gcc gcc-c++ pcre pcre-devel zlib zlib-devel openssl openssl-devel
```

Nginx需要通过openssl源码库来进行安装ssl协议，下载openssl源码，并解压

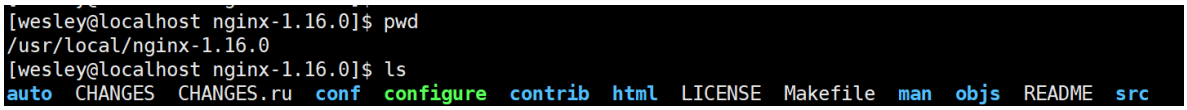
```
wget https://github.com/openssl/openssl/archive/OpenSSL_1_0_2k.tar.gz    # 下载openssl
tar -xzvf OpenSSL_1_0_2k.tar.gz # 解压openssl

wget https://github.com/openresty/echo-nginx-module/archive/v0.61.tar.gz
tar -xzvf v0.61.tar.gz
```

配置编译选项

编译安装，通过configure文件来操作，用于一些特殊需求、特殊依赖的安装方式，操作也不是那么难。可以参考官网，也可以参考configure的帮助文档。

先了解目录内容，进入Nginx目录，输入ls查看文件内容如下图，里面有一个configure可执行文件



在当前默认输入下面命令，就可以看到对应的帮助文档

```
./configure --help
```

这里列出几个常用的选项

```
--prefix= #指向安装目录
--sbin-path #指向（执行）程序文件（nginx）
--conf-path= #指向配置文件（nginx.conf）
--error-log-path= #指向错误日志目录
--pid-path= #指向 pid 文件（nginx.pid）
--lock-path= #指向 lock 文件（nginx.lock）（安装文件锁定，防止安装文件被别人利用，或自己误操作。）
--user= #指定程序运行时的非特权用户
--group= #指定程序运行时的非特权用户组
--builddir= #指向编译目录
--without-xxx   #禁用默认编译启用的xxx模块
--with-yyy  #启用默认不启用的yyy模块
```

--without/--with命令，它好比我们的maven资源管理，在父项目的pom中定义好了依赖的资源，子项目如果需要使用通过引用即可。

操作示例

```
sudo ./configure \
    --prefix=/usr/local/nginx \
    --sbin-path=/usr/local/nginx/nginx \
    --conf-path=/usr/local/nginx/conf/nginx.conf \
    --error-log-path=/usr/local/nginx/logs/error.log \
    --pid-path=/usr/local/nginx/pid/nginx.pid \
    --with-http_ssl_module \
    --with-openssl=/home/wesley/openssl-OpenSSL_1_0_2k \
    --add-module=/home/wesley/echo-nginx-module-0.61
```

configure命令做了大量的“幕后”工作，包括检测操作系统内核和已经安装的软件，参数的解析，中间目录的生成以及根据各种参数生成一些C源码文件、Makefile文件等。

执行编译

```
sudo make
```

make命令根据configure命令生成的Makefile文件编译Nginx工程，并生成目标文件、最终的二进制文件。

安装

```
sudo make install
```

make install命令根据configure执行时的参数将Nginx部署到指定的安装目录，包括相关目录的建立和二进制文件、配置文件的复制。