[**nginx安装及编译参数详解**](http://www.cnblogs.com/HKUI/p/5225895.html)

**1.centos下Yum安装 Nginx  
yum list|grep nginx 发现没有可用的结果  
通过创建下面的文件在系统中添加nginx仓库的yum配置  
vi /etc/yum.repos.d/nginx.repo  
[nginx]  
name=nginx repo  
baseurl=http://nginx.org/packages/centos/6/$basearch/  
gpgcheck=0  
enabled=1  
wq!  
然后  
yum list|grep nginx  
[root@localhost110 nginx-1.8.0]# yum list|grep nginx  
nginx.x86\_64 1.8.1-1.el6.ngx nginx  
nginx-debug.x86\_64 1.8.0-1.el6.ngx nginx  
nginx-debuginfo.x86\_64 1.8.1-1.el6.ngx nginx  
nginx-nr-agent.noarch 2.0.0-9.el6.ngx nginx  
就可以用yum install nginx来安装了**

**2.源码方式安装  
Nginx代码提供2个独立的下载分支,标准版和开发版  
开发分支是一个处于积极开发状态的版本,都可以用于生产环境，主要区别在于对第三方模块的支持  
在开发版本内部的api可能会发生改变,而标准版的却保持不变  
因此,为了向下兼容第三方模块,在标准版中第三方模块都可以有效使用  
下载源码包,解压 编译安装即可**

**编译通用配置选项见下表**

|  |  |
| --- | --- |
| 选项 | 解释说明 |
| --prefix=<path> | Nginx安装的根路径,所有其它路径都要依赖该选项 |
| --sbin-path=<path> | 指定nginx二进制文件的路径,没指定的话 这个路径依赖--prefix选项 |
| --conf-path=<path> | 如果在命令行未指定配置文件,那么将会通过这里指定的路径,nginx将会去那里查找他的配置文件 |
| --error-log-path=<path> | 错误文件路径,nginx写入错误日志文件地址,除非有其他配置 |
| --pid-path=<path> | nginx master进程pid写入的文件位置,通常在var/run下 |
| --lock-path=<path> | 共享存储器互斥锁文件路径 |
| --user=<user> | worker进程运行的用户 |
| --group=<group> | worker进程运行的组 |
| --with-file-aio. | 为freeBSD4.3+和linux2.6.22+系统启用异步io |
| --width-debug | 启用调试日志,生产环境不推荐 |

**nginx -V  查看编译参数**

**[复制代码](javascript:void(0);)**

**nginx version: nginx/1.8.1**

**built by gcc 4.4.7 20120313 (Red Hat 4.4.7-16) (GCC)**

**built with OpenSSL 1.0.1e-fips 11 Feb 2013**

**TLS SNI support enabled**

**configure arguments: --prefix=/etc/nginx --sbin-path=/usr/sbin/nginx --conf-path=/etc/nginx/nginx.conf --error-log-path=/var/log/nginx/error.log --http-log-path=/var/log/nginx/access.log --pid-path=/var/run/nginx.pid --lock-path=/var/run/nginx.lock --http-client-body-temp-path=/var/cache/nginx/client\_temp --http-proxy-temp-path=/var/cache/nginx/proxy\_temp --http-fastcgi-temp-path=/var/cache/nginx/fastcgi\_temp --http-uwsgi-temp-path=/var/cache/nginx/uwsgi\_temp --http-scgi-temp-path=/var/cache/nginx/scgi\_temp --user=nginx --group=nginx --with-http\_ssl\_module --with-http\_realip\_module --with-http\_addition\_module --with-http\_sub\_module --with-http\_dav\_module --with-http\_flv\_module --with-http\_mp4\_module --with-http\_gunzip\_module --with-http\_gzip\_static\_module --with-http\_random\_index\_module --with-http\_secure\_link\_module --with-http\_stub\_status\_module --with-http\_auth\_request\_module --with-mail --with-mail\_ssl\_module --with-file-aio --with-ipv6 --with-http\_spdy\_module --with-cc-opt='-O2 -g -pipe -Wall -Wp,-D\_FORTIFY\_SOURCE=2 -fexceptions -fstack-protector --param=ssp-buffer-size=4 -m64 -mtune=generic'**

**[复制代码](javascript:void(0);)**

**即为**

**[复制代码](javascript:void(0);)**

**--prefix=/etc/nginx**

**--sbin-path=/usr/sbin/nginx**

**--conf-path=/etc/nginx/nginx.conf**

**--error-log-path=/var/log/nginx/error.log**

**--http-log-path=/var/log/nginx/access.log**

**--pid-path=/var/run/nginx.pid**

**--lock-path=/var/run/nginx.lock**

**--http-client-body-temp-path=/var/cache/nginx/client\_temp**

**--http-proxy-temp-path=/var/cache/nginx/proxy\_temp**

**--http-fastcgi-temp-path=/var/cache/nginx/fastcgi\_temp**

**--http-uwsgi-temp-path=/var/cache/nginx/uwsgi\_temp**

**--http-scgi-temp-path=/var/cache/nginx/scgi\_temp**

**--user=nginx**

**--group=nginx**

**--with-http\_ssl\_module**

**--with-http\_realip\_module**

**--with-http\_addition\_module**

**--with-http\_sub\_module**

**--with-http\_dav\_module**

**--with-http\_flv\_module**

**--with-http\_mp4\_module**

**--with-http\_gunzip\_module**

**--with-http\_gzip\_static\_module**

**--with-http\_random\_index\_module**

**--with-http\_secure\_link\_module**

**--with-http\_stub\_status\_module**

**--with-http\_auth\_request\_module**

**--with-mail**

**--with-mail\_ssl\_module**

**--with-file-aio --with-ipv6**

**--with-http\_spdy\_module**

**--with-cc-opt='-O2 -g -pipe -Wall -Wp,-D\_FORTIFY\_SOURCE=2 -fexceptions -fstack-protector --param=ssp-buffer-size=4 -m64 -mtune=generic'**

**[复制代码](javascript:void(0);)**

**优化编译参数 见下表**

|  |  |
| --- | --- |
| 选项 | 说明 |
| --with-cc=<path> | 如果想设置一个不在默认path下的c编译器 |
| --with-cpp=<path> | 设置c预处理器的相对路径 |
| --with-cc-opt=<options> | 指定必要的include文件路径,可能d(-I<path>)指出,也可能是优化(-O4)和指定一个64位构建    (完全看不懂啊) |
| --with-ld-opt=<options> | 包含连接库的路径和运行路径 |
| -with-cpu-opt=<cpu> | 通过该选项为特定cpu构建nginx |

**邮件模块独有的配置选项**

|  |  |
| --- | --- |
| 选项 | 说明 |
| --with-mail | 激活POP3/IMAP4/SMTP代理模块,默认未激活 |
| --with-mail\_ssl\_module | 允许ngx\_mail\_ssl\_module模块  这个模块使得POP3／IMAP／SMTP可以使用SSL／TLS.配置已经定义了HTTP SSL模块，但是不支持客户端证书检测 |
| --without-mail\_pop3\_module | 启用mail模块后,单独禁用pop3模块 |
| --without-mail\_imap\_module | 启用mail模块后,单独禁用imap模块 |
| --without-mail\_smtp\_module | 启用mail模块后,单独禁用smtp模块 |
| --without-http | 完全禁用http模块,如果只想支持mall,可以使用此项设置 |
| --with-openssl=DIR | 设定OpenSSL库文件路径 |

**对于典型的mail代理,nginx配置可为**

**./configure --with-mail  --with-mail\_ssl\_module  --with-openssl=${BUILD\_DIR}/openssl-1.0.1c**

**指定路径的配置选项**

**下边显示了http模块有效的配置选项，从激活perl模块到临时目录的位置**

|  |  |
| --- | --- |
| 选项 | 说明 |
| --without-http-cache | 在使用upstream模块时,nginx能够配置本地缓存内容,此选项可禁用缓存 |
| --with-http\_perl\_module | 这个模块允许nginx使用SSI调用perl或直接执行perl(使用会降低性能) |
| --with-perl\_modules\_path=PATH | 设置perl模块路径(perl解析器路径) |
| --with-perl\_modules\_path=PATH | 设置perl模块路径 |
| --http-log-path=PATH | 设置access log文件路径 |
| --http-client-body-temp-path=PATH | 设置客户端请求临时文件路径，如果WebDAV启用,推荐设置该路径为同一文件系统上的目录作为最终的目的地 |
| --http-proxy-temp-path=PATH | 代理设置后,设置http proxy临时文件路径 |
| --http-fastcgi-temp-path=PATH | 设置http fastcgi临时文件路径 |
| --http-fastcgi-temp-path=PATH | set path to store http fastcgi temporary files |
| --http-scgi-temp-path=PATH | set path to store http scgi temporary files |

**使用各种模块**

**在nginx发布的版本中,除了http和mail模块之外,还有一些其他模块在默认安装时并没有被安装**

**可通过以下配置来启用相应的选项**

**http模块配置选项**

|  |  |
| --- | --- |
| 选项 | 说明 |
| --with-http\_ssl\_module | 如果需要对流量加密.可使用此选项,在urls中开始部分将会是https(需要openssl库) |
| --with-http\_realip\_module | 允许ngx\_http\_realip\_module模块(mod\_rpaf)  此模块支持显示真实来源IP地址，主要用于NGINX做前端负载均衡服务器使用,  如果你的nginx在七层负载均衡器或者其它设备之后,它们将Http头中的客户端ip地址传递,这时需要启用此模块,在多个客户处于一个ip地址的情况下使用 |
| --with-http\_addition\_module | 作为一个输出过滤器,使你能够在请求经过一个location前或后时在该location本身添加内容 |
| --with-http\_xslt\_module | 这个模块是一个过滤器，它可以通过XSLT模板转换XML应答 |
| --with-http\_image\_filter\_module | 图像过滤器,在将图像投递到客户之前进行处理(需要libgd库) |
| --with-http\_geoip\_module | 使用该模块,能够设置设置个中变量以便在配置区段中使用 |
| --with-http\_sub\_module | 允许ngx\_http\_sub\_module模块  这个模块可以能够在nginx的应答中搜索并替换文本 |
| --with-http\_dav\_module | 允许ngx\_http\_dav\_module模块(mod\_dav)  为文件和目录指定权限，限制不同类型的用户对于页面有不同的操作权限 |
| --with-http\_flv\_module | 允许ngx\_http\_flv\_module模块(mod\_flvx)  这个模块支持对FLV（flash）文件的拖动播放 |
| --with-http\_mp4\_module | 支持H.264/AAC文件为伪流媒体 |
| -with-http\_gzip\_static\_module | 允许ngx\_http\_gzip\_static\_module模块(mod\_dflate)  这个模块在一个预压缩文件传送到开启Gzip压缩的客户端之前检查是否已经存在以“.gz”结尾的压缩文件，这样可以防止文件被重复压缩 |
| --with-http\_gunzip\_module | 对于不支持gzip编码的客户,该模块用于为客户解压缩预压缩内容 |
| --with-http\_random\_index\_module | 允许ngx\_http\_random\_index\_module模块(mod\_autoindex)，从目录中选择一个随机主页 |
| --with-http\_secure\_link\_module | 该模块提供一种机制,它会将一个哈希值链接到一个url中,因此,只有那些使用正确的密码能够计算链接 |
| --with-http\_stub\_status\_module | 这个模块可以取得一些nginx的运行状态，如果是工业状况，可以直接取消  输出的状态信息科使用RRDtool或类似的工具绘制成图 |

**正如上表所列,这些模块都是建立在http模块基础之上的,它们提供了额外的功能。在编译时启用这些模块根本不会影响到运行性能,以后再配置使用这些模块时性能会产生影响**

**对于网络加速代理,就配置选项,建议以下配置**

**./configure --with-http\_ssl\_module --with-http\_realip\_module --with-http\_geoip\_module --with-http\_stub\_status\_module  --with-openssl=${BUILD\_DIR}/openssl-1.0.1c**

**及web服务器**

**./configure --with-http\_stub\_status\_module**