

Nginx 的介绍和安装

作者: [xf616510229](#)

原文链接: <https://ld246.com/article/1556245234677>

来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

Nginx的介绍和安装

前言，本想着一篇文章写完Nginx所有相关知识点，但是写着写着发现太多了，迫不得已拆分为多篇章。我会尽量保证文章不是很散，利于查找，与此同时，我也会提供一个目录对文章进行归类。

什么是nginx

Nginx是一个开源、高性能、可靠的HTTP中间件、代理服务。

关于正向代理与反向代理

正向代理代理的客户端，而反向代理代理的则是服务端。比如，如果挂vpn代理访问google就是正向代理。

常见的web服务器

Apache、Nginx、Lighttpd、Tomcat、Jetty、Weblogic、IIS、Jboss、Netty、Glassfish、Resin、Websphere

Apache、Nginx、Lighttpd服务器，都是静态web服务器，在Nginx之前，绝大部分企业，都采用Apache使用Apache，Lighttpd则比较轻量级。Jetty、Tomcat则是动态的Http服务器。

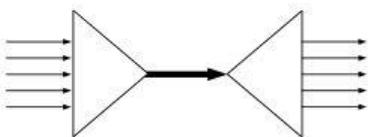
nginx的优势

IO多路复用 epoll

什么是IO多路复用

[知乎-罗志宇 IO 多路复用是什么意思?](#)

I/O就是指的网络I/O，多路指多个TCP连接(或多个Channel)，复用指复用一个或少量线程。串来理解就是很多个网络I/O复用一个或少量的线程来处理这些连接。如下图：



I/O复用 解决的是并发性请求的问题。**处理多个并发请求，要产生多个I/O流来进行系统内核数据的取。**常用的两种处理方式是串行，前一个阻塞，后面无法继续进行处理、并行处理请求 - 实现最大并和吞吐。

普通版：

老师给一个班学生出题，并且老师不停挨个询问学生有没有做完试题，如果有学生做完试题，则解答，这种方式采用的是串行的处理方式。

强化版：

老师分身，创建多个线程老师，分别处理每个学生做完的试题，这种方式采用的是多线程的处理方式但是资源分配上、上下文切换会出现额外资源消耗。

究极版：

真正的I/O多路复用

学生（网络请求-请求数据分组到达）主动上报自己做题的情况，复用的是老师处理学生做题情况的程

多路复用的三种方式：select、poll、epoll

select 模型：

即系统发出select系统调用，等待内核主动将可用的文件描述符(fd)信息发送给应用一端，fd未准备，应用会block住socket请求，当fd就绪后，select 会遍历维护的文件描述符发现可用的文件描述符。

缺点：采用线性遍历的方式获取可用的fd文件描述符，可维护文件描述符大小有限制为1024

epoll 模型：

每当fd就绪，系统采用回调函数将fd放入就绪列表，效率非常高。

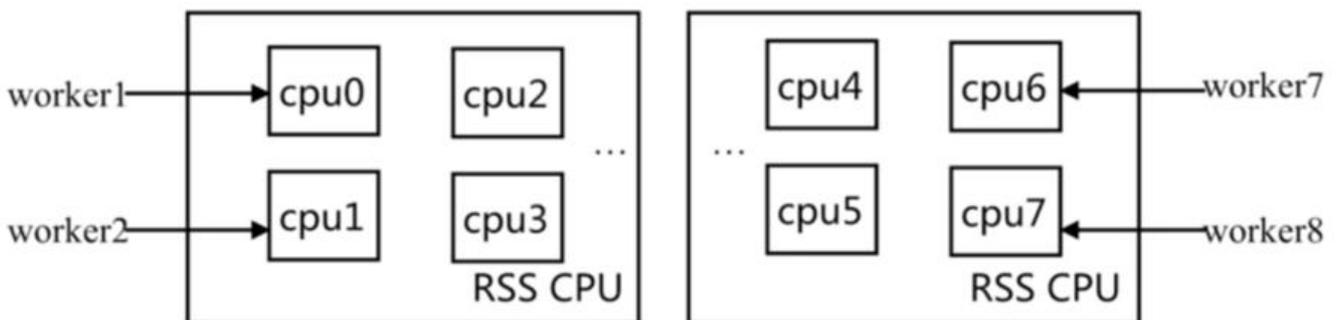
优点：每当fd就绪，系统采用回调函数将fd放入就绪列表，效率非常高 最大连接数没有限制

内核（kernel）利用文件描述符（file descriptor）来访问文件。文件描述符是非负整数。打开现存文件或新建文件时，内核会返回一个文件描述符。读写文件也需要使用文件描述符来指定待读写的文件。

轻量级

- 初始功能模块少，可手动添加移除模块
- 代码模块化，便于代码拓展

CPU亲和



现在的CPU大多都是多核的，而Nginx由多个worker进程同时工作，每个worker进程都与固定的一个PU核心绑定，减少切换CPU的cache miss(缓存未命中，查找地址不在Cache中)，从而获取更好的性能。

CPU Cache：

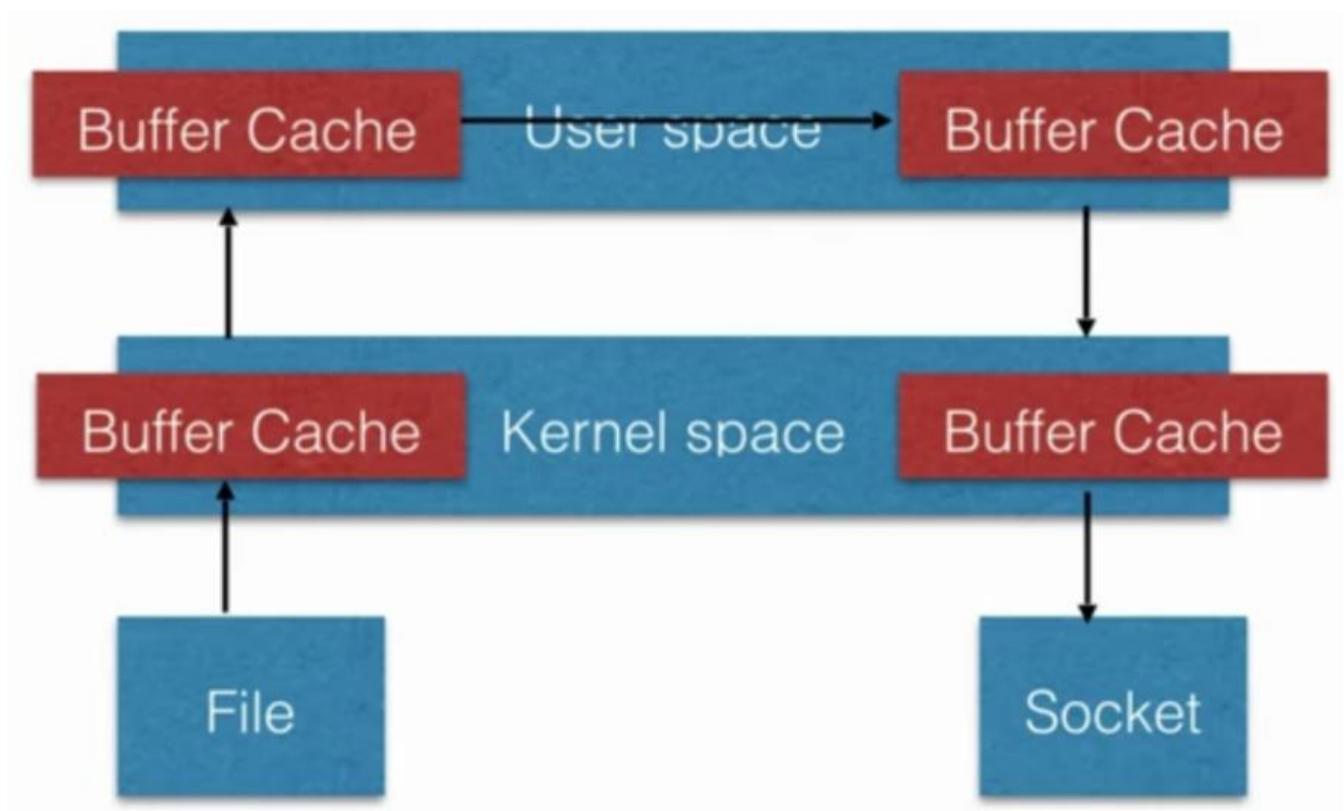
- Cpu Cache是一个快速缓存，访问速度比内存快很多，但是空间小，只能缓存部分内存数据
- 按照功能划分，可以分为：**指令缓存**(code cache)、**数据缓存**(data cache)、**TLB缓存**(translation lookaside buffer，加速虚拟地址转物理地址)
- 按照速度划分，大部分主流CPU都有二级甚至三级缓存：L1、L2、L3

sendfile文件处理机制

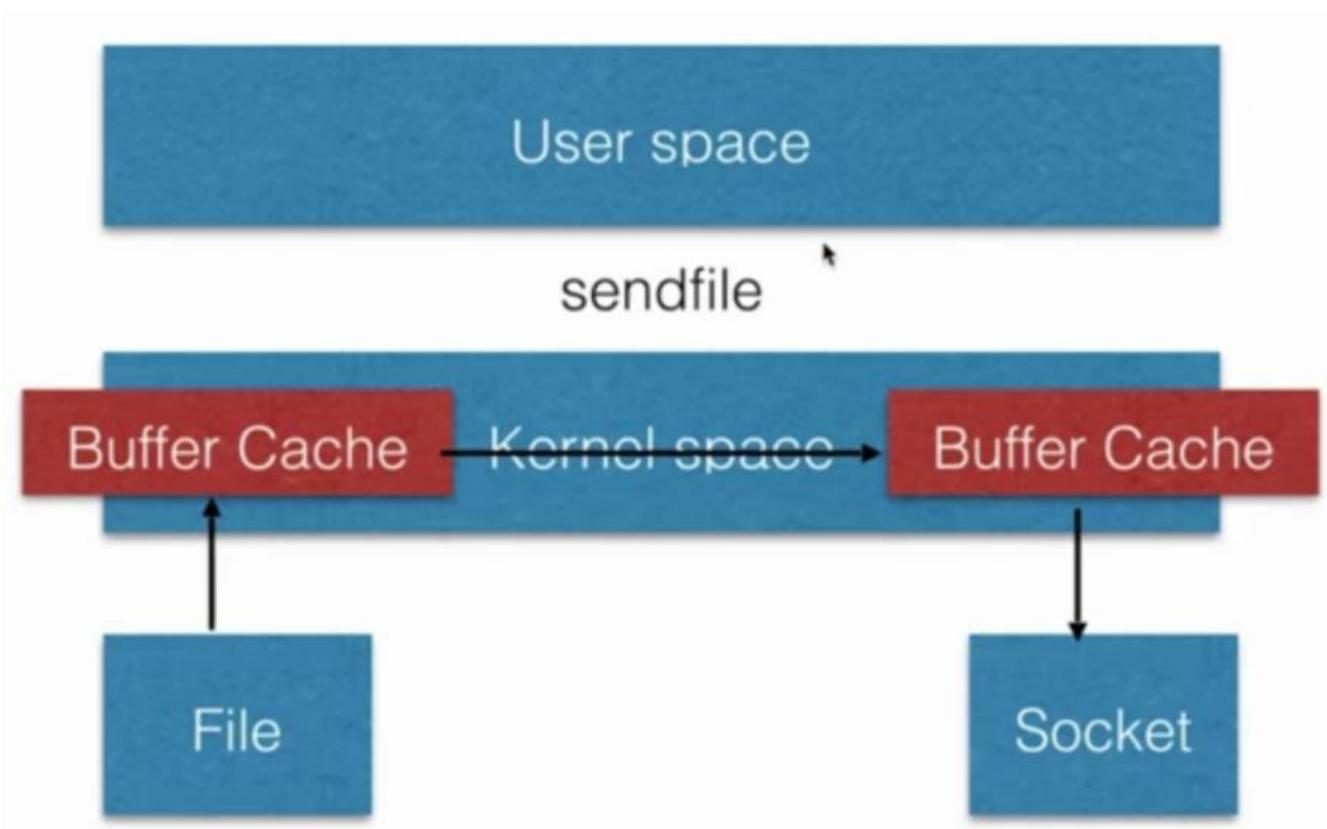
Nginx采用sendfile处理机制，所以静态文件的处理相当快。

Sendfile (零拷贝)：

普通的静态资源处理方式需要将文件加载到内核空间(Kernel Space)中，再拷贝到用户空间(User Space)中，这个过程发生了多次切换——



而静态文件无需用户空间，所以使用sendfile零拷贝的方式更为合适：



安装nginx

nginx的版本

- Mainline version 开发版
- Stable version 稳定版
- Legacy version 历史版本

下载地址

下载地址: <http://nginx.org/en/download.html>

在此页面上共有如下几个可以下载选项:

- Mainline version
- Stable version
- Legacy versions
- Source Code 源代码
- Pre-Built Packages

除此之外, 还有如下几个选项:

| | | | | |
|------------------------------|------------------------------|---------------------|--------------------------------------|---------------------|
| CHANGES-1.12 | nginx-1.12.2 | pgp | nginx/Windows-1.12.2 | pgp |
| (版本变更内容)10 | nginx-1.10.3 | pgp | nginx/Windows-1.10.3 | pgp |
| CHANGES-1.8 | nginx-1.8.1 | pgp | nginx/Windows-1.8.1 | pgp |
| CHANGES-1.6 | nginx-1.6.3 | pgp | nginx/Windows-1.6.3 | pgp |
| CHANGES-1.4 | nginx-1.4.7 | pgp | nginx/Windows-1.4.7 | pgp |
| CHANGES-1.2 | nginx-1.2.9 | pgp | nginx/Windows-1.2.9 | pgp |

↓ ↑ ↑ ↑
(版本变更内容)10 下载nginx-linux版本 windows版本
校验工具，第三方下载的nginx可能安全代码被篡改

使用yum安装

具体请参考官方文档: http://nginx.org/en/linux_packages.html

```
# 新建nginx yum源
vim /etc/yum.repos.d/nginx.repo

# 加入如下配置
[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
```

配置完成后, 可以通过 `yum list | grep nginx` 查看可安装的包。使用如下命令安装nginx:

```
yum install nginx
```

Ubuntu下安装

```
sudo apt-get install nginx
```

编译安装

确认gcc、g++开发类库是否装好:

```
# ubuntu
sudo apt-get install build-essential
sudo apt-get install libtool

# centos
sudo yum -y install gcc automake autoconf libtool make
sudo yum install gcc gcc-c++
```

安装PCRE正则表达式库:

```
# centos
sudo yum install pcre-devel

# ubuntu
sudo apt-get install libpcre3 libpcre3-dev
```

安装openssl依赖:

```
# ubuntu
sudo apt-get install openssl
sudo apt-get install libssl-dev

# centos
yum -y install openssl openssl-devel
```

安装zlib依赖:

```
# ubuntu
sudo apt-get install zlib1g-dev

# centos
sudo yum install zlib-devel
```

安装nginx:

```
cd /usr/local/src
wget http://nginx.org/download/nginx-1.3.9.tar.gz
tar -zxvf nginx-1.3.9.tar.gz
cd nginx-1.3.9
./configure
make
make install
```

编译参数

路径可参考nginx目录结构一节

在编译nginx时, 即执行 `./configure` 命令时, 可以指定一些参数为对nginx进行定制, 比如指定nginx块的位置:

```
./configure --modules-path=/usr/lib64/nginx/modules
```

如果你已经安装了nginx, 可以通过命令 `nginx -V` 查看支持的参数:

```
[root@Feathers nginx]# nginx -V
nginx version: nginx/1.14.2
built by gcc 4.8.5 20150623 (Red Hat 4.8.5-28) (GCC)
built with OpenSSL 1.0.2k-fips 26 Jan 2017
TLS SNI support enabled
configure arguments:
# 指定安装目录、各种文件路径
--prefix=/etc/nginx
```

```
--sbin-path=/usr/sbin/nginx # 执行文件路径
--modules-path=/usr/lib64/nginx/modules # 模块路径
--conf-path=/etc/nginx/nginx.conf # 配置文件路径
--error-log-path=/var/log/nginx/error.log # 错误日志路径
--http-log-path=/var/log/nginx/access.log # 访问日志路径
--pid-path=/var/run/nginx.pid # pid文件路径
--lock-path=/var/run/nginx.lock # nginx锁路径

# 执行对应模块时，nginx所保留的临时性文件
--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

# nginx进程启动的用户和组
--user=nginx
--group=nginx

# 下面以with开头的时编译时添加的模块，如果你要的模块存在在这下面，说明模块已经安装
--with-compat --with-file-aio --with-threads --with-http_addition_module --with-http_auth_r
quest_module --with-http_dav_module --with-http_flv_module --with-http_gunzip_module -
with-http_gzip_static_module --with-http_mp4_module --with-http_random_index_module --
with-http_realip_module --with-http_secure_link_module --with-http_slice_module --with-http_
sl_module --with-http_stub_status_module --with-http_sub_module --with-http_v2_module -
with-mail --with-mail_ssl_module --with-stream --with-stream_realip_module --with-stream_s
l_module --with-stream_ssl_preread_module

# 设置额外的参数将被添加到CFLAGS变量
--with-cc-opt='-O2 -g -pipe -Wall -Wp,-D_FORTIFY_SOURCE=2 -fexceptions -fstack-protecto
-strong --param=ssp-buffer-size=4 -grecord-gcc-switches -m64 -mtune=generic -fPIC'

# 设置附加参数，链接系统库
--with-ld-opt='-Wl,-z,relro -Wl,-z,now -pie'
```

nginx的目录结构

如果是yum 安装，安装后的文件分散于各个目录，可以使用命令`rpm -ql nginx`查看所有安装后的文件

安装nginx后目录结构如下：

```
# 执行命令，比如启动关闭管理
/usr/sbin/nginx
/usr/sbin/nginx-debug

# 主配置文件
/etc/nginx
/etc/nginx/nginx.conf
/etc/nginx/conf.d
/etc/nginx/conf.d/default.conf

# 日志路径
/var/log/nginx
```

```
# nginx日志轮转配置, 用于logrotate服务(linux系统服务)的日志切割
/etc/logrotate.d/nginx

# cgi配置、fastcgi配置
/etc/nginx/fastcgi_params
/etc/nginx/scgi_params
/etc/nginx/uwsgi_params

# 编码转换映射转化文件 (不常用)
/etc/nginx/koi-utf
/etc/nginx/koi-win
/etc/nginx/win-utf

# 设置http协议的Content-Type与扩展名的对应关系
/etc/nginx/mime.types

# 新版centos采用配置方式实现守护进程的管理 (systemd) , 下面的配置文件是nginx对此的支持
/usr/lib/systemd/system/nginx-debug.service
/usr/lib/systemd/system/nginx.service
/etc/sysconfig/nginx
/etc/sysconfig/nginx-debug

# 模块目录, 模块以及模块依赖会放到下面的文件夹中
/etc/nginx/modules
/usr/lib64/nginx # 目录
/usr/lib64/nginx/modules

# 帮助手册, 可以使用man命令查看
/usr/share/doc/nginx-1.14.2
/usr/share/doc/nginx-1.14.2/COPYRIGHT
/usr/share/man/man8/nginx.8.gz

# nginx 缓存目录, nginx cache的默认缓存路径
/var/cache/nginx

# nginx静态资源默认存储路径
/usr/share/nginx
/usr/share/nginx/html
/usr/share/nginx/html/50x.html
/usr/share/nginx/html/index.html

/usr/libexec/initscripts/legacy-actions/nginx
/usr/libexec/initscripts/legacy-actions/nginx/check-reload
/usr/libexec/initscripts/legacy-actions/nginx/upgrade
```

启动停止重启nginx

```
# 快速停止nginx, 不管有没有正在处理的请求
nginx -s stop
# 优雅退出, 退出前完成已经接受的请求
nginx -s quit

# 使用发送信号的方式停止nginx
# 快速停止nginx, 此方式等同于 kill -9
```

```
kill -TERM 进程id
```

```
# 优雅退出
```

```
kill -QUIT 进程id
```

```
# 启动nginx
```

```
nginx -c 配置文件地址
```

```
nginx -c /usr/local/nginx/conf/nginx.conf
```

```
# 验证nginx配置文件是否正确，如果显示nginx.conf syntax is ok nginx.conf test is successful 则  
明配置文件正确
```

```
nginx -t
```

```
# 启动之前校验nginx配置文件的正确性
```

```
nginx -t -c /usr/local/nginx/conf/nginx.conf
```

```
# 重新加载nginx.conf
```

```
nginx -s reload
```

```
# 重启nginx服务
```

```
kill -HUP 2255
```

关于Nginx模块

nginx的模块分为两种：

- nginx官方模块
- 第三方模块

查看nginx编译时自带的模块：

nginx -V，编译参数中以with开头的就是编译时nginx已经带着的模块

```
--with-compat
```

```
--with-file-aio
```

```
--with-threads
```

```
--with-http_addition_module
```

```
--with-http_auth_request_module
```

```
--with-http_dav_module
```

```
--with-http_flv_module
```

```
--with-http_gunzip_module
```

```
--with-http_gzip_static_module
```

```
--with-http_mp4_module
```

```
--with-http_random_index_module
```

```
--with-http_realip_module
```

```
--with-http_secure_link_module
```

```
--with-http_slice_module
```

```
--with-http_ssl_module
```

```
--with-http_stub_status_module
```

```
--with-http_sub_module
```

```
--with-http_v2_module
```

```
--with-mail
```

```
--with-mail_ssl_module
```

```
--with-stream
```

```
--with-stream_realip_module
```

```
--with-stream_ssl_module  
--with-stream_ssl_preread_module
```