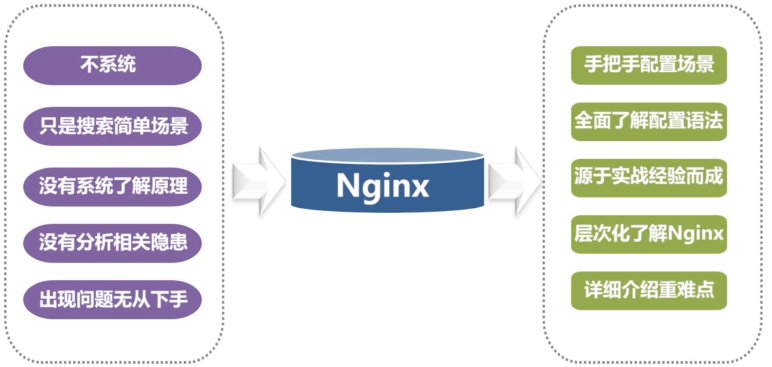
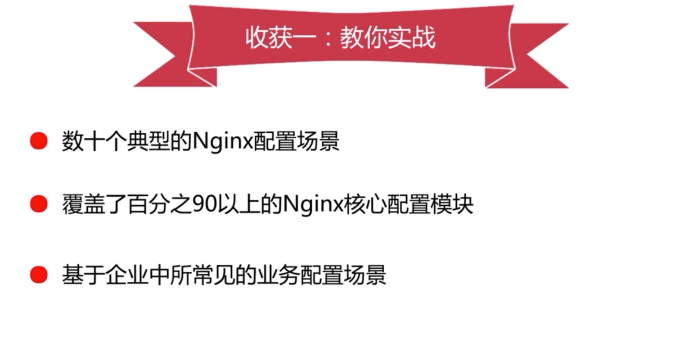
# Nginx从入门到实践

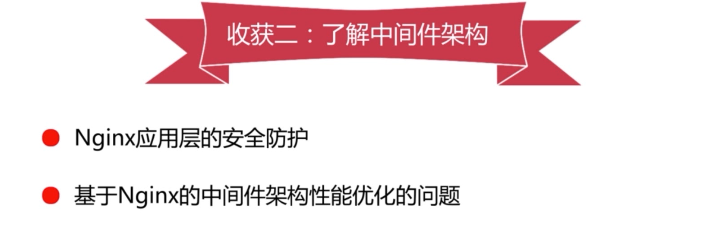
## 第1节基于Nginx的中间件架构

### 1.1介绍



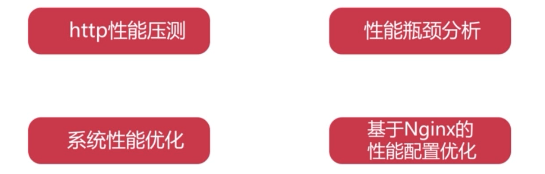










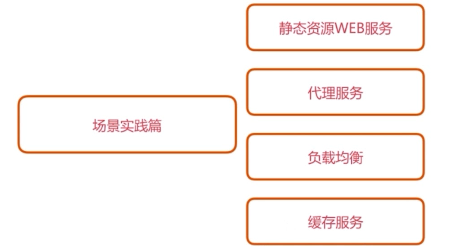




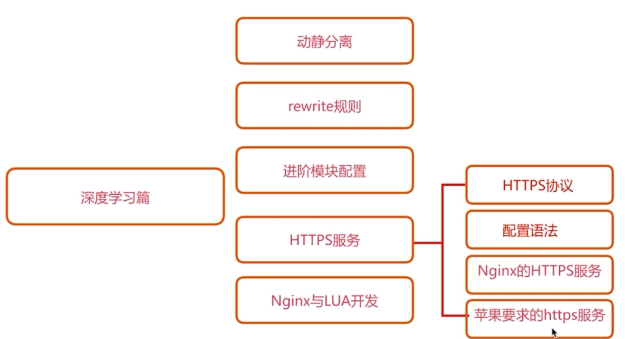


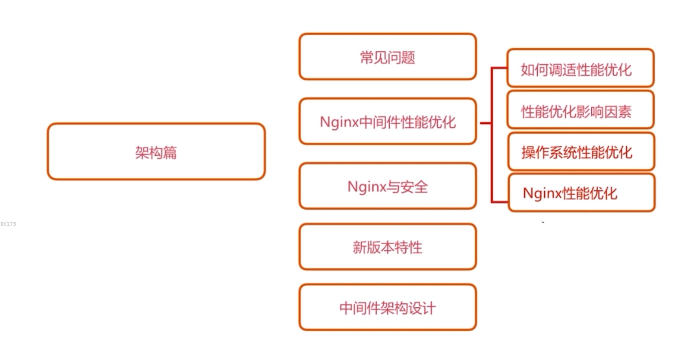


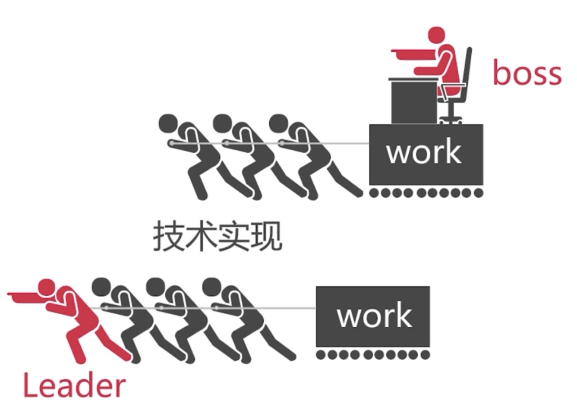










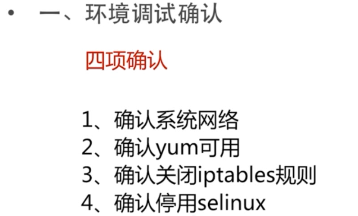




### 1.2环境准备



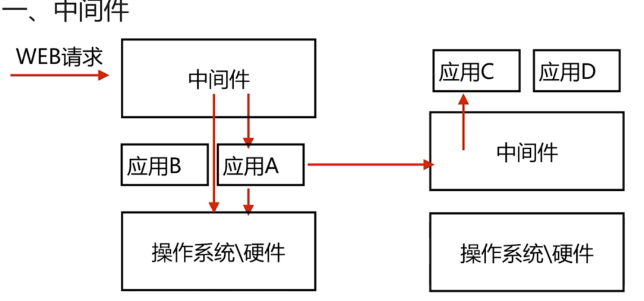


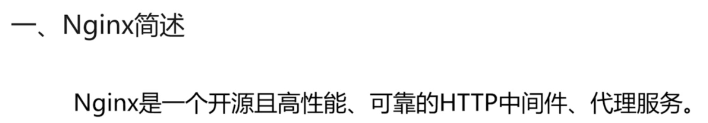


|  |
| --- |
| # 查看是否安装gcc  [root@s200 /opt]#yum list | grep gcc  # 查看防火墙状态是否关闭，并关闭  [root@s200 /opt]#iptables -L  Chain INPUT (policy ACCEPT)  target prot opt source destination  Chain FORWARD (policy ACCEPT)  target prot opt source destination  Chain OUTPUT (policy ACCEPT)  target prot opt source destination  [root@s200 /opt]#iptables -F  [root@s200 /opt]#iptables -t nat -L  Chain PREROUTING (policy ACCEPT)  target prot opt source destination  Chain INPUT (policy ACCEPT)  target prot opt source destination  Chain OUTPUT (policy ACCEPT)  target prot opt source destination  Chain POSTROUTING (policy ACCEPT)  target prot opt source destination  # 查看SELinux是否关闭  [root@s200 /opt]#iptables -t nat -F  [root@s200 /opt]#getenforce  Enforcing  [root@s200 /opt]#setenforce 0  [root@s200 /opt]#getenforce  Permissive  [root@s200 /opt]#vi /etc/selinux/config  [root@s200 /opt]#more /etc/selinux/config  # This file controls the state of SELinux on the system.  # SELINUX= can take one of these three values:  # enforcing - SELinux security policy is enforced.  # permissive - SELinux prints warnings instead of enforcing.  # disabled - No SELinux policy is loaded.  SELINUX=disabled  # SELINUXTYPE= can take one of three two values:  # targeted - Targeted processes are protected,  # minimum - Modification of targeted policy. Only selected processes are protec  ted.  # mls - Multi Level Security protection.  SELINUXTYPE=targeted  [root@s200 /opt]#reboot  [root@s200 /root]#getenforce  Disabled  # 安装gcc、gcc-c++、autoconf、pcre、pcre-devel、make、automake  [root@s200 /root]#yum -y install gcc gcc-c++ autoconf pcre pcre-devel make automake  # 安装wget，httpd-tools，vim  [root@s200 /root]#yum -y install wget httpd-tools vim  [root@s200 /root]#cd /opt/  [root@s200 /root]#mkdir nginx  [root@s200 /root]#cd nginx/  [root@s200 /root]#mkdir app backup download logs work |

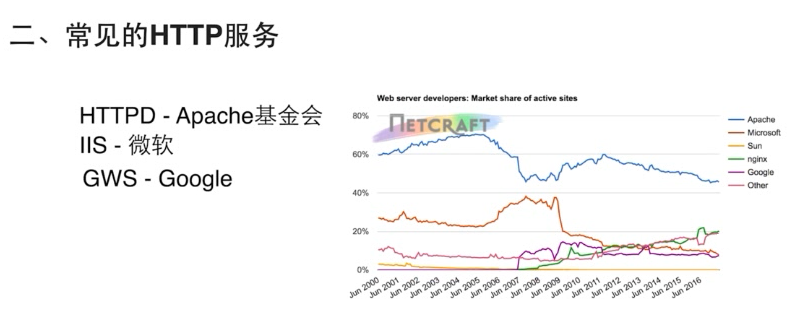
## 第2节基础篇

### 2.1什么是Nginx





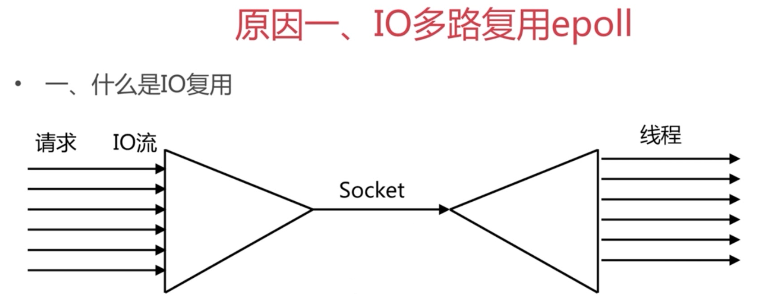
### 2.2常见的HTTP服务

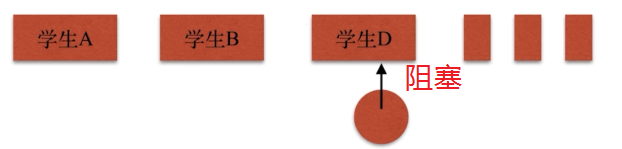


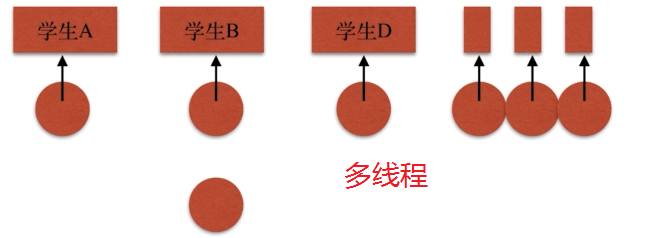
### 2.3Nginx特性

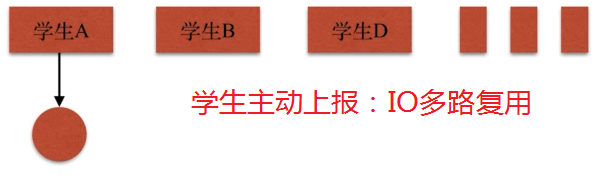
#### 2.4.1实现优点1：I/O多路复用

为什么选择Nginx







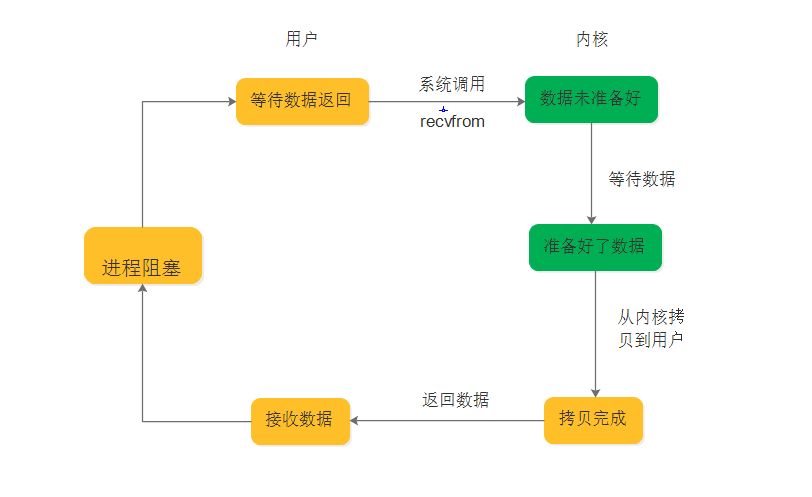


·五种IO模型

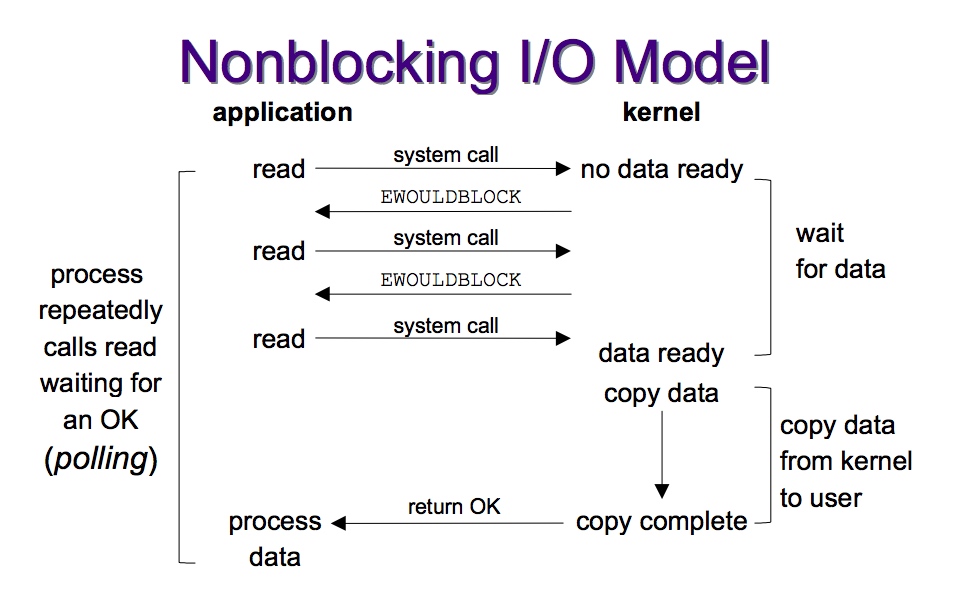
<https://www.jianshu.com/p/6a6845464770>

首先引用[levin](https://www.zhihu.com/people/levin-43-90" \t "_blank)的回答让我们理清楚**五种IO模型**

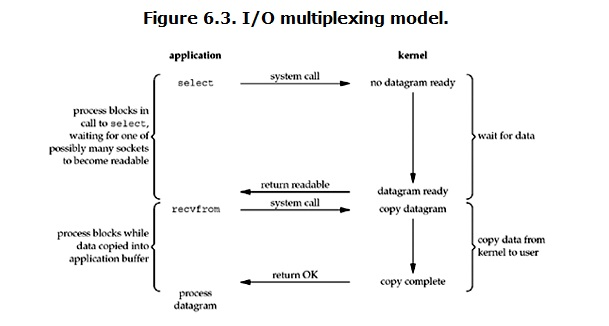
1. **阻塞I/O模型**  
   老李去火车站买票，排队三天买到一张退票。  
   耗费：在车站吃喝拉撒睡 3天，其他事一件没干。



2.**非阻塞I/O模型**  
老李去火车站买票，隔12小时去火车站问有没有退票，三天后买到一张票。耗费：往返车站6次，路上6小时，其他时间做了好多事。

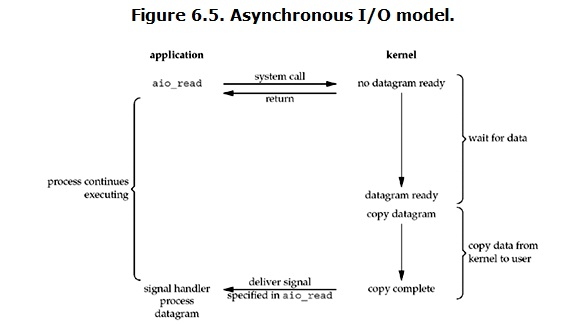


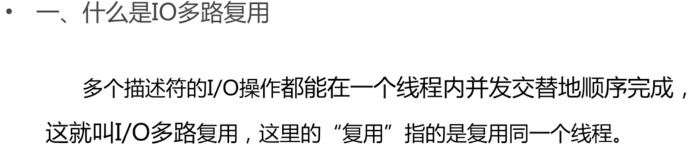
3.**I/O复用模型**  
[1.select/poll](https://link.zhihu.com/?target=http://1.select/poll" \t "_blank)  
老李去火车站买票，委托黄牛，然后每隔6小时电话黄牛询问，黄牛三天内买到票，然后老李去火车站交钱领票。  
耗费：往返车站2次，路上2小时，黄牛手续费100元，打电话17次  
2.epoll  
老李去火车站买票，委托黄牛，黄牛买到后即通知老李去领，然后老李去火车站交钱领票。  
耗费：往返车站2次，路上2小时，黄牛手续费100元，无需打电话

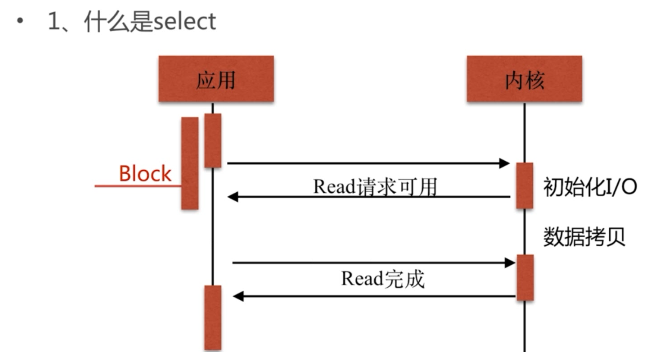


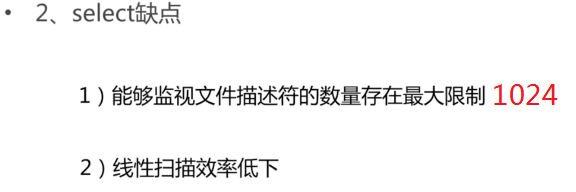
4.**信号驱动I/O模型**  
老李去火车站买票，给售票员留下电话，有票后，售票员电话通知老李，然后老李去火车站交钱领票。  
耗费：往返车站2次，路上2小时，免黄牛费100元，无需打电话

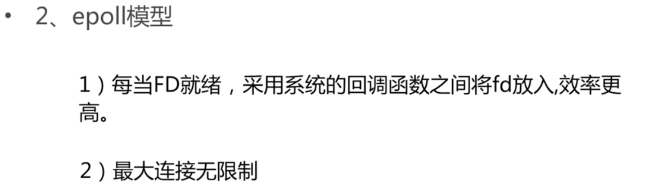
5.**异步I/O模型**  
老李去火车站买票，给售票员留下电话，有票后，售票员电话通知老李并快递送票上门。  
耗费：往返车站1次，路上1小时，免黄牛费100元，无需打电话



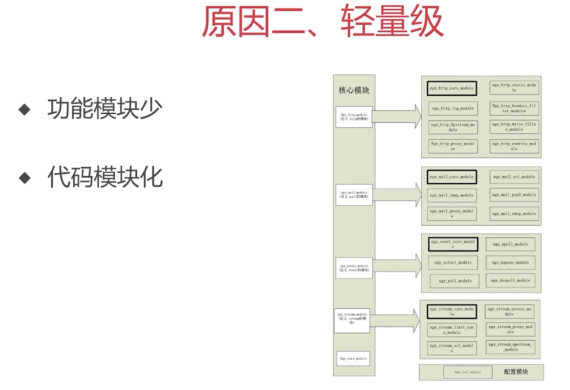




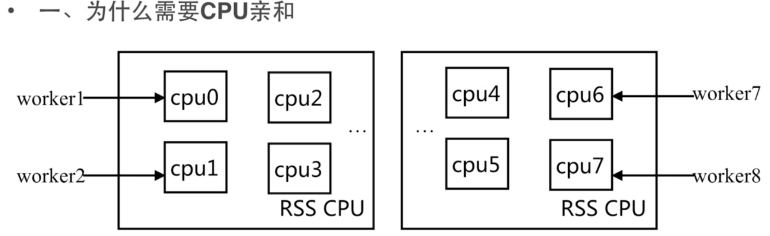


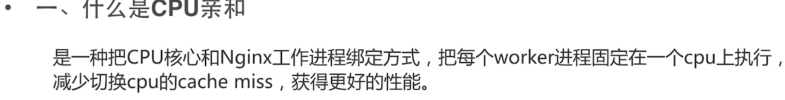


#### 2.4.2实现优点2：轻量级

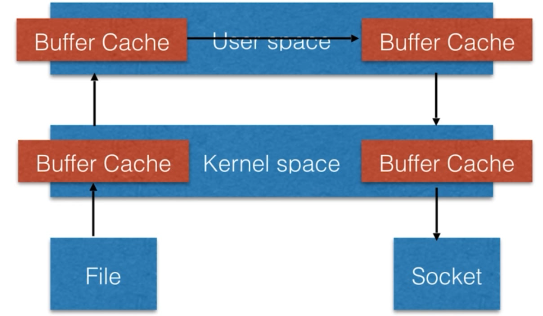


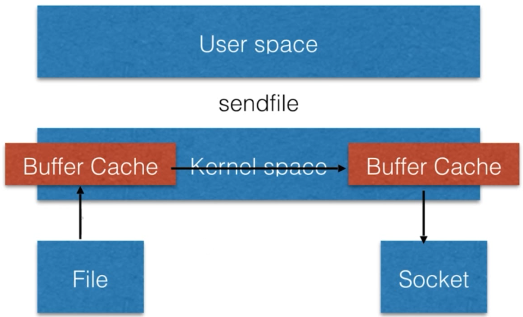
#### 2.4.3实现优点3：COU亲和（affinity）



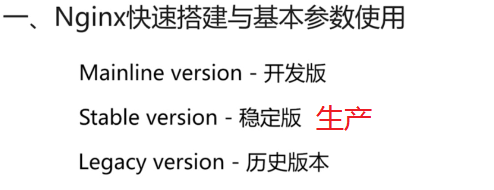


#### 2.4.4实现优点4：sendfile





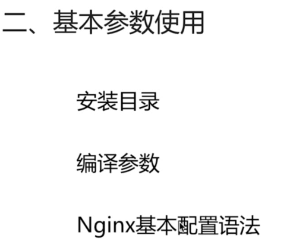
### 2.5快速安装



<http://nginx.org/en/linux_packages.html#stable>

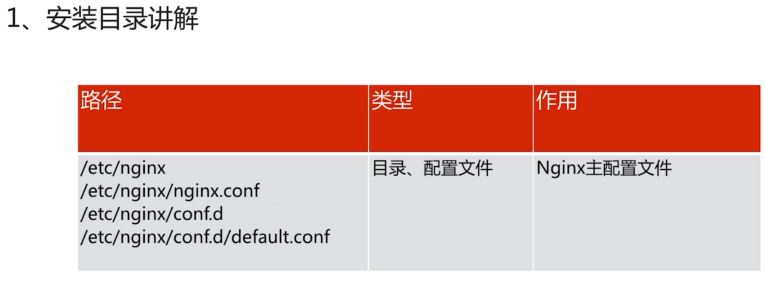
|  |
| --- |
| [root@s200 /opt/nginx]#vim /etc/yum.repos.d/nginx.repo  [root@s200 /opt/nginx]#cat /etc/yum.repos.d/nginx.repo  [nginx]  name=nginx repo  baseurl=http://nginx.org/packages/centos/7/$basearch/  gpgcheck=0  enabled=1  [root@s200 /opt/nginx]#yum list | grep nginx  nginx.x86\_64 1:1.14.2-1.el7\_4.ngx nginx  nginx-debug.x86\_64 1:1.8.0-1.el7.ngx nginx  nginx-debuginfo.x86\_64 1:1.14.2-1.el7\_4.ngx nginx  nginx-module-geoip.x86\_64 1:1.14.2-1.el7\_4.ngx nginx  nginx-module-geoip-debuginfo.x86\_64 1:1.14.2-1.el7\_4.ngx nginx  nginx-module-image-filter.x86\_64 1:1.14.2-1.el7\_4.ngx nginx  nginx-module-image-filter-debuginfo.x86\_64 1:1.14.2-1.el7\_4.ngx nginx  nginx-module-njs.x86\_64 1:1.14.2.0.2.7-1.el7\_4.ngx nginx  nginx-module-njs-debuginfo.x86\_64 1:1.14.2.0.2.7-1.el7\_4.ngx nginx  nginx-module-perl.x86\_64 1:1.14.2-1.el7\_4.ngx nginx  nginx-module-perl-debuginfo.x86\_64 1:1.14.2-1.el7\_4.ngx nginx  nginx-module-xslt.x86\_64 1:1.14.2-1.el7\_4.ngx nginx  nginx-module-xslt-debuginfo.x86\_64 1:1.14.2-1.el7\_4.ngx nginx  nginx-nr-agent.noarch 2.0.0-12.el7.ngx nginx  pcp-pmda-nginx.x86\_64 4.1.0-5.el7\_6 updates  [root@s200 /opt/nginx]#yum install nginx |

### 2.6Nginx的目录和配置语法



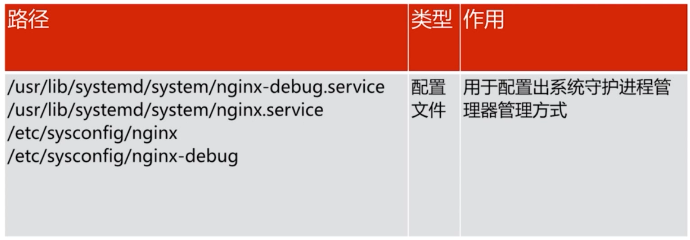
#### 2.6.1Nginx的安装目录











centos7.2采用systemctl 而不是init.d







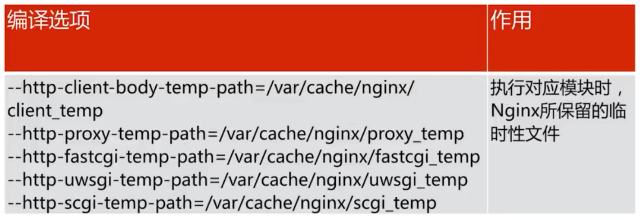




#### 2.6.2安装编译参数

nginx -V











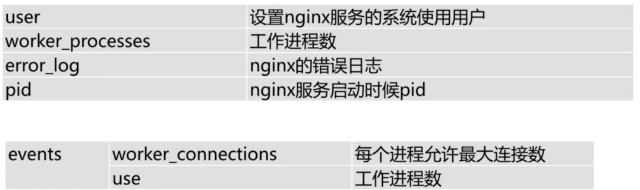
#### 2.6.3Nginx默认配置语法

·主配置文件nginx.conf

|  |
| --- |
| [root@s200 /usr/share/nginx/html]#cat /etc/nginx/nginx.conf | grep -v "^$" | grep -v "#"  user nginx;  worker\_processes 1;  error\_log /var/log/nginx/error.log warn;  pid /var/run/nginx.pid;  events {  worker\_connections 1024;  }  http {  include /etc/nginx/mime.types;  default\_type application/octet-stream;  log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '  '$status $body\_bytes\_sent "$http\_referer" '  '"$http\_user\_agent" "$http\_x\_forwarded\_for"';  access\_log /var/log/nginx/access.log main;  sendfile on;  keepalive\_timeout 65;  include /etc/nginx/conf.d/\*.conf;  } |

·从配置文件/etc/nginx/conf.d/\*.conf

|  |
| --- |
| server {  listen 80;  server\_name localhost;  location / {  root /usr/share/nginx/html;  index index.html index.htm;  }  error\_page 500 502 503 504 /50x.html;  location = /50x.html {  root /usr/share/nginx/html;  }  } |





2.6.4默认配置与默认站点启动

/etc/nginx/nginx.conf

|  |
| --- |
| user nginx;  worker\_processes 1;  error\_log /var/log/nginx/error.log warn;  pid /var/run/nginx.pid;  events {  worker\_connections 1024;  }  http {  include /etc/nginx/mime.types;  default\_type application/octet-stream;  log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '  '$status $body\_bytes\_sent "$http\_referer" '  '"$http\_user\_agent" "$http\_x\_forwarded\_for"';  access\_log /var/log/nginx/access.log main;  sendfile on;  #tcp\_nopush on;  keepalive\_timeout 65;  #gzip on;  include /etc/nginx/conf.d/\*.conf;  } |

/etc/nginx/conf.d/default.conf

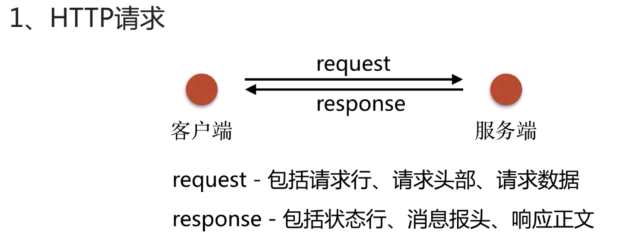
|  |
| --- |
| server {  listen 80;  server\_name localhost;  location / {  root /usr/share/nginx/html;  index index.html index.htm;  }  error\_page 500 502 503 504 404 /50x.html;  location = /50x.html {  root /usr/share/nginx/html;  }  } |

/usr/share/nginx/html/50x.html index.html

$ systemctl restart nginx.service

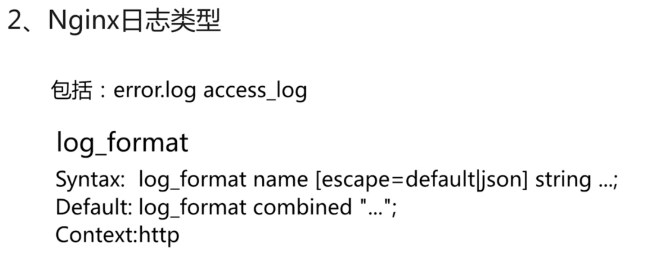
$ systemctl reload nginx.service

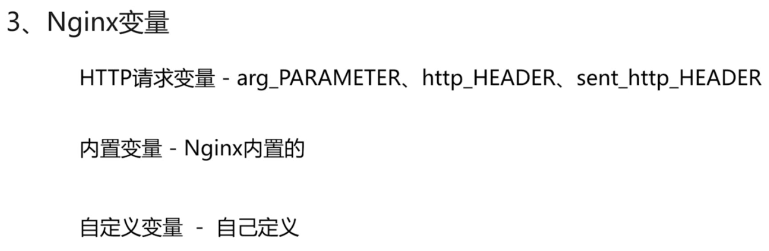
### 2.7HTTP请求



|  |
| --- |
| #curl s200  #curl s200:80/12  # 浏览器的开发者工具也可以看到  #curl -v s200 > /dev/null  % Total % Received % Xferd Average Speed Time Time Time Current  Dload Upload Total Spent Left Speed  0 0 0 0 0 0 0 0 --:--:-- --:--:-- --:--:-- 0\* About to connect() to s200 port 80 (#0)  \* Trying 192.168.0.200...  \* Connected to s200 (192.168.0.200) port 80 (#0)  > GET / HTTP/1.1  > User-Agent: curl/7.29.0  > Host: s200  > Accept: \*/\*  >  < HTTP/1.1 200 OK  < Server: nginx/1.14.2  < Date: Sat, 19 Jan 2019 14:43:06 GMT  < Content-Type: text/html  < Content-Length: 633  < Last-Modified: Sat, 19 Jan 2019 14:31:53 GMT  < Connection: keep-alive  < ETag: "5c4334d9-279"  < Accept-Ranges: bytes  <  { [data not shown]  100 633 100 633 0 0 93197 0 --:--:-- --:--:-- --:--:-- 103k  \* Connection #0 to host s200 left intact |

### 2.8Nginx日志





#curl -v s200 > /dev/null

...

> User-Agent: curl/7.29.0

...

转换为http\_access的日志格式，就需要在log\_format main 中添加’http\_user\_agent’，全小写，横杠变为下划线。

·Nginx的语法校验

#nginx -t -c /etc//nginx/nginx.conf

nginx: the configuration file /etc//nginx/nginx.conf syntax is ok

nginx: configuration file /etc//nginx/nginx.conf test is successful

·校验通过，重新加载配置文件

#nginx -s reload -c /etc/nginx/nginx.conf

·nginx内置变量

<http://nginx.org/en/docs/>

[Logging to syslog](http://nginx.org/en/docs/syslog.html)

<http://nginx.org/en/docs/syslog.html>

<http://nginx.org/en/docs/http/ngx_http_core_module.html#var_status>

|  |  |
| --- | --- |
| 变量名 | 含义 |
| $remote\_addr | client address |
| $remote\_user | user name supplied with the Basic authentication |
| $time\_local | local time in the Common Log Format (1.3.12, 1.2.7) |
| $request | full original request line |
| $status | response status (1.3.2, 1.2.2) |
| $body\_bytes\_sent | number of bytes sent to a client, not counting the response header; this variable is compatible with the “%B” parameter of the mod\_log\_config Apache module |
| $http\_referer | 防盗链，记录上一站的地址 |
| $http\_user\_agent |  |
| http\_x\_forwarded\_for |  |

### 2.9Nginx模块讲解

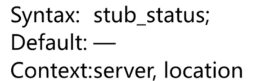


# 查看编译的模块

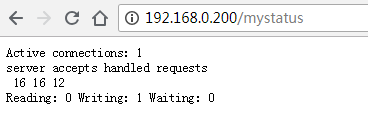
|  |
| --- |
| [root@s200 /usr/share/nginx/html]#nginx -V  nginx version: nginx/1.14.2  built by gcc 4.8.5 20150623 (Red Hat 4.8.5-28) (GCC)  built with OpenSSL 1.0.2k-fips 26 Jan 2017  TLS SNI support enabled  configure arguments:  --prefix=/etc/nginx  --sbin-path=/usr/sbin/nginx  --modules-path=/usr/lib64/nginx/modules  --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-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  --with-cc-opt='-O2 -g -pipe -Wall -Wp,-D\_FORTIFY\_SOURCE=2 -fexceptions -fstack-protector-strong  --param=ssp-buffer-size=4 -grecord-gcc-switches -m64 -mtune=generic -fPIC'  --with-ld-opt='-Wl,-z,relro -Wl,-z,now -pie' |

#### 2.9.1http\_stub\_status\_module配置

--with-http\_stub\_status\_module



|  |
| --- |
| [root@s200 /usr/share/nginx/html]#cat /etc/nginx/conf.d/default.conf | grep -v "^$" | grep -v "#"  server {  listen 80;  server\_name localhost;  location /mystatus {  stub\_status;  }  location / {  root /usr/share/nginx/html;  index index.html index.htm;  }  error\_page 500 502 503 504 404 /50x.html;  location = /50x.html {  root /usr/share/nginx/html;  }  }  [root@s200 /usr/share/nginx/html]#nginx -tc /etc/nginx/nginx.conf  nginx: the configuration file /etc/nginx/nginx.conf syntax is ok  nginx: configuration file /etc/nginx/nginx.conf test is successful  [root@s200 /usr/share/nginx/html]#nginx -s reload -c /etc/nginx/nginx.conf |

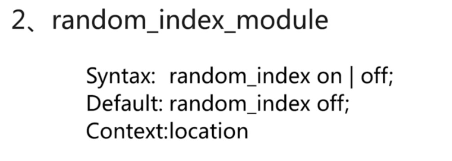


|  |
| --- |
| # Nginx当前活跃的连接数  Active connections: 1  # Nginx接受的握手总次数、Nginx处理的连接数、Nginx总的请求数  # 正常握手和连接数相等，表示没有丢失  server accepts handled requests  16 16 12  # 正在从Nginx读的个数、正在往Nginx写的个数、Nginx开启KeepAlive后客户端既没有读也没有写，但是建立连接的数量  Reading: 0 Writing: 1 Waiting: 0 |

#### 2.9.2http\_random\_index\_module

目录中随机选择一个主页

--with-http\_random\_index\_module



|  |
| --- |
| location / {  root /opt/nginx/app/code;  random\_index on;  } |

/opt/nginx/app/code/1.html

|  |
| --- |
| <html>  <head>  <meta charset="utf-8">  <title>index\_blue</title>  </head>  <body style="background-color:blue;">  </body>  </html> |

/opt/nginx/app/code/2.html

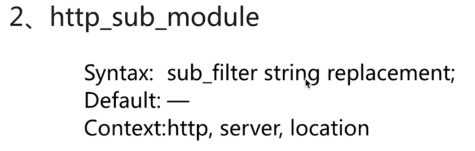
|  |
| --- |
| <html>  <head>  <meta charset="utf-8">  <title>index\_red</title>  </head>  <body style="background-color:red;">  </body>  </html> |

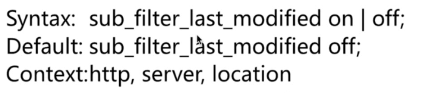
/opt/nginx/app/code/3.html

|  |
| --- |
| <html>  <head>  <meta charset="utf-8">  <title>index\_green</title>  </head>  <body style="background-color:green;">  </body>  </html> |

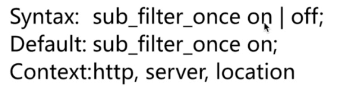
#### 2.9.3http\_sub\_module







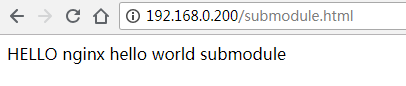
缓存使用较常见



|  |
| --- |
| location / {  root /opt/nginx/app/code;  index index.html index.htm;  sub\_filter '<a>hello' '<a>HELLO';  } |

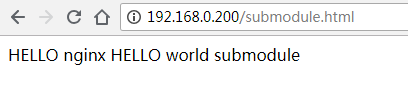
opt/nginx/app/code/submodule.html

|  |
| --- |
| <html>  <head>  <meta charset="utf-8">  <title>index\_blue</title>  </head>  <body>  <a>hello</a>  <a>nginx</a>  <a>hello</a>  <a>world</a>  <a>submodule</a>  </body>  </html> |

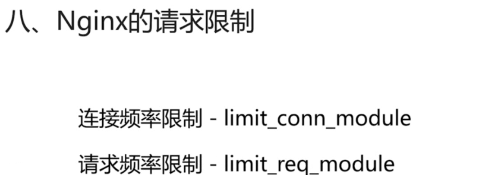


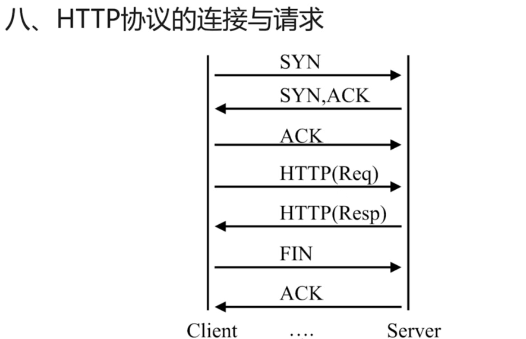
|  |
| --- |
| location / {  root /opt/nginx/app/code;  index index.html index.htm;  sub\_filter '<a>hello' '<a>HELLO';  sub\_filter\_once off;  } |

$ nginx -s reload -c /etc/nginx/nginx.conf



2.9.4Nginx的请求限制





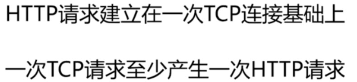
·三次握手

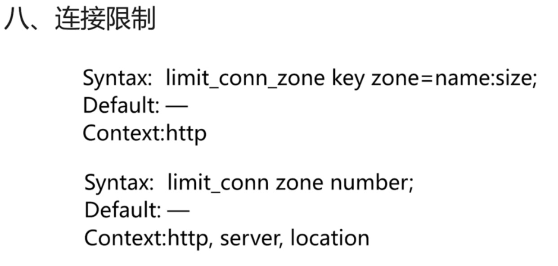
·发包收包

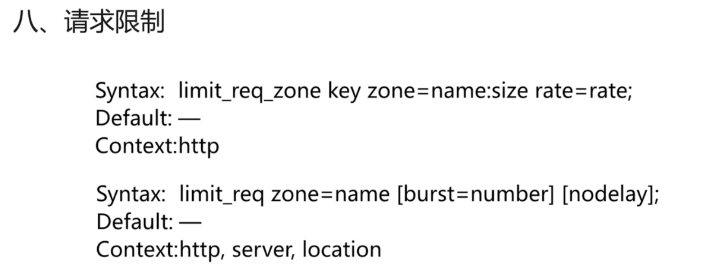
·FIN ACK来保持连接的状态，也就是长连接和KeepAlive

·HTTP建立在TCP连接的基础之上









|  |
| --- |
| [root@s200 /etc/nginx/conf.d]#ab -n 4000 -c 1000 http://192.168.0.200/1.html  This is ApacheBench, Version 2.3 <$Revision: 1430300 $>  Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/  Licensed to The Apache Software Foundation, http://www.apache.org/  Benchmarking 192.168.0.200 (be patient)  Completed 400 requests  Completed 800 requests  Completed 1200 requests  Completed 1600 requests  Completed 2000 requests  Completed 2400 requests  Completed 2800 requests  Completed 3200 requests  Completed 3600 requests  Completed 4000 requests  Finished 4000 requests  Server Software: nginx/1.14.2  Server Hostname: 192.168.0.200  Server Port: 80  Document Path: /1.html  Document Length: 126 bytes  Concurrency Level: 1000  Time taken for tests: 0.605 seconds  Complete requests: 4000  Failed requests: 0  Write errors: 0  Total transferred: 992000 bytes  HTML transferred: 504000 bytes  Requests per second: 6611.80 [#/sec] (mean)  Time per request: 151.245 [ms] (mean)  Time per request: 0.151 [ms] (mean, across all concurrent requests)  Transfer rate: 1601.30 [Kbytes/sec] received  Connection Times (ms)  min mean[+/-sd] median max  Connect: 0 8 15.0 0 57  Processing: 10 20 16.6 12 212  Waiting: 0 20 16.6 12 212  Total: 11 28 26.7 12 217  Percentage of the requests served within a certain time (ms)  50% 12  66% 13  75% 63  80% 64  90% 72  95% 76  98% 82  99% 84  100% 217 (longest request) |