**品优购电商系统部署**

**第2章**

**MyCat & Nginx**

传智播客.黑马程序员

# 课程目标

目标1：理解MyCat分片，能够配置MyCat分片

目标2：掌握Nginx的安装与静态网站部署

目标3：掌握Nginx的静态网站部署

目标4：理解Nginx的反向代理与负载均衡，能够配置反向代理与负载均衡

目标5：了解品优购的整体部署方案

# 1.开源数据库中间件-MyCat

如今随着互联网的发展，数据的量级也是撑指数的增长，从GB到TB到PB。对数据的各种操作也是愈加的困难，传统的关系性数据库已经无法满足快速查询与插入数据的需求。这个时候NoSQL的出现暂时解决了这一危机。它通过降低数据的安全性，减少对事务的支持，减少对复杂查询的支持，来获取性能上的提升。

但是，在有些场合NoSQL一些折衷是无法满足使用场景的，就比如有些使用场景是绝对要有事务与安全指标的。这个时候NoSQL肯定是无法满足的，所以还是需要使用关系性数据库。如果使用关系型数据库解决海量存储的问题呢？此时就需要做数据库集群，为了提高查询性能将一个数据库的数据分散到不同的数据库中存储。

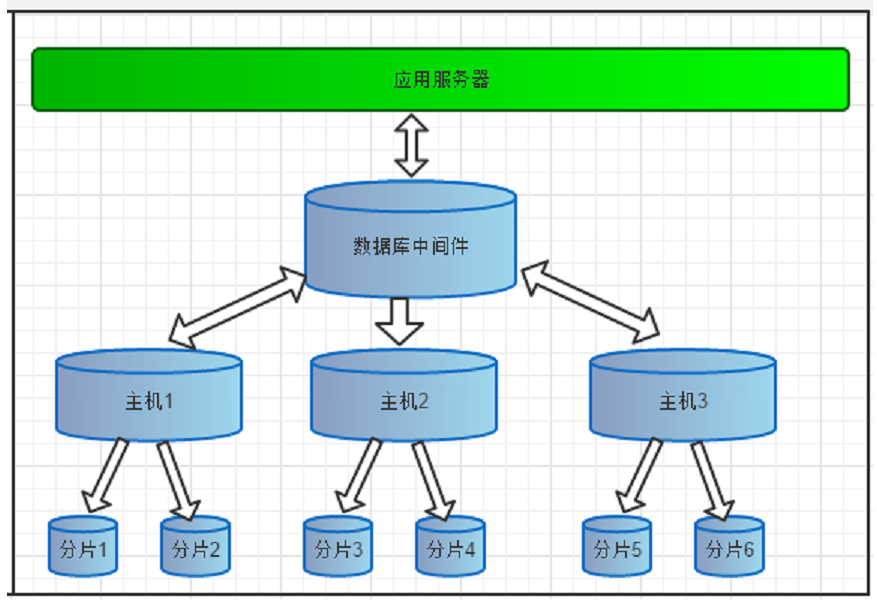
## 1.1 MyCat简介

Mycat 背后是阿里曾经开源的知名产品——Cobar。Cobar 的核心功能和优势是 MySQL 数据库分片，此产品曾经广为流传，据说最早的发起者对 Mysql 很精通，后来从阿里跳槽了，阿里随后开源的 Cobar，并维持到 2013 年年初，然后，就没有然后了。

Cobar 的思路和实现路径的确不错。基于 Java 开发的，实现了 MySQL 公开的二进制传输协议，巧妙地将自己伪装成一个 MySQL Server，目前市面上绝大多数 MySQL 客户端工具和应用都能兼容。比自己实现一个新的数据库协议要明智的多，因为生态环境在哪里摆着。

Mycat 是基于 cobar 演变而来，对 cobar 的代码进行了彻底的重构，使用 NIO 重构了网络模块，并且优化了 Buffer 内核，增强了聚合，Join 等基本特性，同时兼容绝大多数数据库成为通用的数据库中间件。

简单的说，MyCAT就是：一个新颖的数据库中间件产品支持mysql集群，或者mariadb cluster，提供高可用性数据分片集群。你可以像使用mysql一样使用mycat。对于开发人员来说根本感觉不到mycat的存在。



**MyCat支持的数据库：**



## 1.2MyCat下载及安装

### 1.2.1 MySQL安装与启动

JDK：要求jdk必须是1.7及以上版本

MySQL：推荐mysql是5.5以上版本

**MySQL安装与启动步骤如下：( 步骤1-5省略 )**

（1）将MySQL的服务端和客户端安装包（RPM）上传到服务器



（2）查询之前是否安装过MySQL

|  |
| --- |
| rpm -qa|grep -i mysql |

（3）卸载旧版本MySQL

|  |
| --- |
| rpm -e --nodeps  软件名称 |

1. 安装服务端

|  |
| --- |
| rpm -ivh MySQL-server-5.5.49-1.linux2.6.i386.rpm |

1. 安装客户端

|  |
| --- |
| rpm -ivh MySQL-client-5.5.49-1.linux2.6.i386.rpm |

（6）启动MySQL服务

|  |
| --- |
| service mysql start |

（7）登录MySQL

|  |
| --- |
| mysql -u root |

1. 设置远程登录权限

|  |
| --- |
| GRANT ALL PRIVILEGES ON \*.\* TO 'root'@'%'IDENTIFIED BY '123456' WITH GRANT OPTION; |

在本地SQLyog 连接远程MySQL进行测试

### 1.2.2 MyCat安装及启动

MyCat：

MyCat的官方网站：

<http://www.mycat.org.cn/>

下载地址：

<https://github.com/MyCATApache/Mycat-download>

第一步：将Mycat-server-1.4-release-20151019230038-linux.tar.gz上传至服务器

第二步：将压缩包解压缩。建议将mycat放到/usr/local/mycat目录下。

|  |
| --- |
| tar -xzvf Mycat-server-1.4-release-20151019230038-linux.tar.gz  mv mycat /usr/local |

第三步：进入mycat目录的bin目录，启动mycat

|  |
| --- |
| ./mycat start |

停止：

|  |
| --- |
| ./mycat stop |

mycat 支持的命令{ console | start | stop | restart | status | dump }

Mycat的默认端口号为：8066

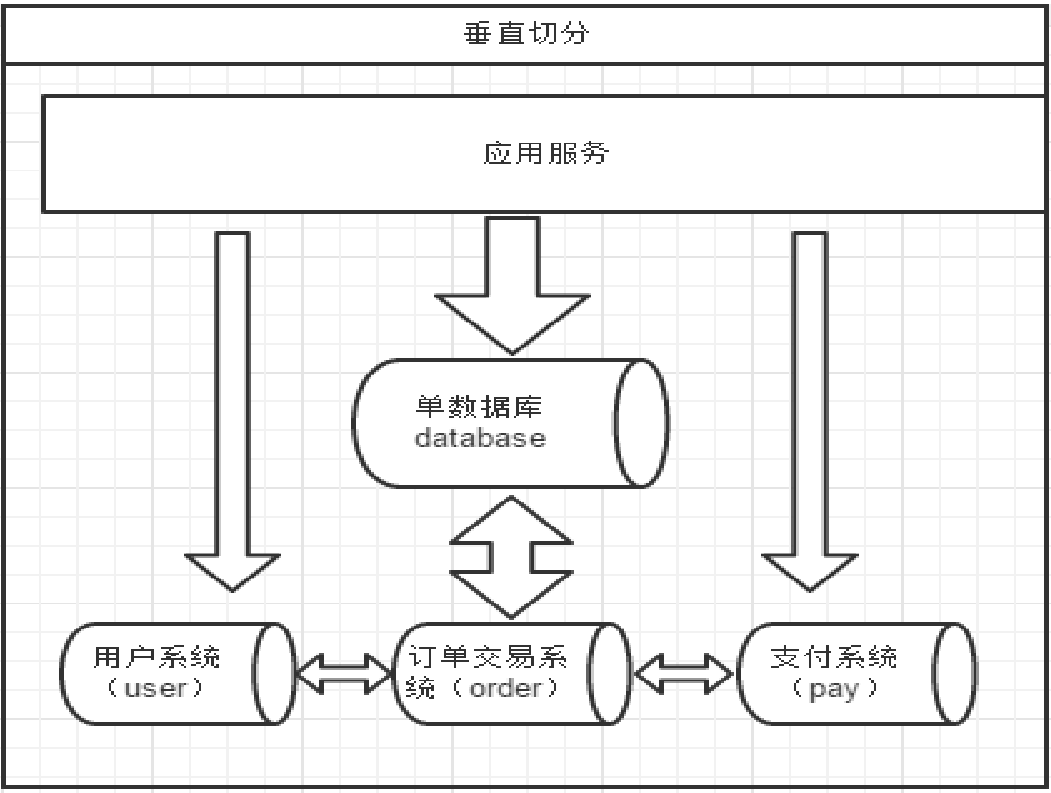
## 1.3MyCat分片-海量数据存储解决方案

### 1.3.1 什么是分片

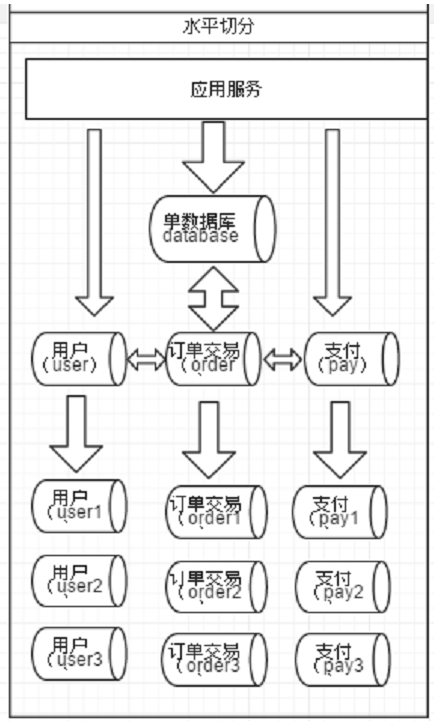
简单来说，就是指通过某种特定的条件，将我们存放在同一个数据库中的数据分散存放到多个数据库（主机）上面，以达到分散单台设备负载的效果。

数据的切分（Sharding）根据其切分规则的类型，可以分为两种切分模式。

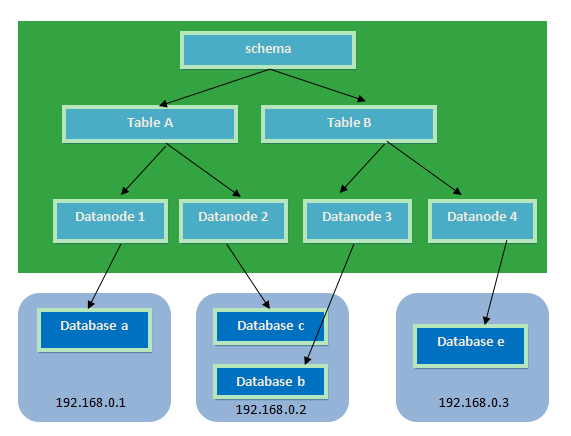
（1）一种是按照不同的表（或者Schema）来切分到不同的数据库（主机）之上，这种切分可以称之为数据的垂直（纵向）切分



（2）另外一种则是根据表中的数据的逻辑关系，将同一个表中的数据按照某种条件拆分到多台数据库（主机）上面，这种切分称之为数据的水平（横向）切分。



**MyCat分片策略：**



### 1.3.2 分片相关的概念

**逻辑库(schema) ：**

前面一节讲了数据库中间件，通常对实际应用来说，并不需要知道中间件的存在，业务开发人员只需要知道数据库的概念，所以数据库中间件可以被看做是一个或多个数据库集群构成的逻辑库。

**逻辑表（table）：**

既然有逻辑库，那么就会有逻辑表，分布式数据库中，对应用来说，读写数据的表就是逻辑表。逻辑表，可以是数据切分后，分布在一个或多个分片库中，也可以不做数据切分，不分片，只有一个表构成。

分片表：是指那些原有的很大数据的表，需要切分到多个数据库的表，这样，每个分片都有一部分数据，所有分片构成了完整的数据。 总而言之就是需要进行分片的表。

非分片表：一个数据库中并不是所有的表都很大，某些表是可以不用进行切分的，非分片是相对分片表来说的，就是那些不需要进行数据切分的表。

**分片节点(dataNode)**

数据切分后，一个大表被分到不同的分片数据库上面，每个表分片所在的数据库就是分片节点（dataNode）。

**节点主机(dataHost)**

数据切分后，每个分片节点（dataNode）不一定都会独占一台机器，同一机器上面可以有多个分片数据库，这样一个或多个分片节点（dataNode）所在的机器就是节点主机（dataHost）,为了规避单节点主机并发数限制，尽量将读写压力高的分片节点（dataNode）均衡的放在不同的节点主机（dataHost）。

**分片规则(rule)**

前面讲了数据切分，一个大表被分成若干个分片表，就需要一定的规则，这样按照某种业务规则把数据分到某个分片的规则就是分片规则，数据切分选择合适的分片规则非常重要，将极大的避免后续数据处理的难度。

### 1.3.3 MyCat分片配置

**（1）配置schema.xml**

schema.xml作为MyCat中重要的配置文件之一，管理着MyCat的逻辑库、逻辑表以及对应的分片规则、DataNode以及DataSource。弄懂这些配置，是正确使用MyCat的前提。这里就一层层对该文件进行解析。

schema 标签用于定义MyCat实例中的逻辑库

Table 标签定义了MyCat中的逻辑表 rule用于指定分片规则，auto-sharding-long的分片规则是按ID值的范围进行分片 1-5000000 为第1片 5000001-10000000 为第2片.... 具体设置我们会在第5小节中讲解。

dataNode 标签定义了MyCat中的数据节点，也就是我们通常说所的数据分片。

dataHost标签在mycat逻辑库中也是作为最底层的标签存在，直接定义了具体的数据库实例、读写分离配置和心跳语句。

在服务器上创建3个数据库，分别是db1 db2 db3

修改schema.xml如下：

|  |
| --- |
| <?xml version="1.0"?>  <!DOCTYPE mycat:schema SYSTEM "schema.dtd">  <mycat:schema xmlns:mycat="http://org.opencloudb/">  <schema name="PINYOUGOUDB" checkSQLschema="false" sqlMaxLimit="100">  <table name="tb\_test" dataNode="dn1,dn2,dn3" rule="auto-sharding-long" />  </schema>  <dataNode name="dn1" dataHost="localhost1" database="db1" />  <dataNode name="dn2" dataHost="localhost1" database="db2" />  <dataNode name="dn3" dataHost="localhost1" database="db3" />  <dataHost name="localhost1" maxCon="1000" minCon="10" balance="0"  writeType="0" dbType="mysql" dbDriver="native" switchType="1" slaveThreshold="100">  <heartbeat>select user()</heartbeat>  <writeHost host="hostM1" url="192.168.25.142:3306" user="root"  password="123456">  </writeHost>  </dataHost>  </mycat:schema> |

**（2）配置 server.xml**

server.xml几乎保存了所有mycat需要的系统配置信息。最常用的是在此配置用户名、密码及权限。在system中添加UTF-8字符集设置，否则存储中文会出现问号

|  |
| --- |
| <property name="charset">utf8</property> |

修改user的设置 , 我们这里为 PINYOUGOUDB设置了两个用户

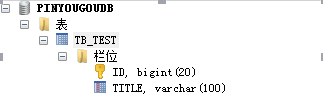
|  |
| --- |
| <user name="test">  <property name="password">test</property>  <property name="schemas">PINYOUGOUDB</property>  </user>  <user name="root">  <property name="password">123456</property>  <property name="schemas">PINYOUGOUDB</property>  </user> |

### 1.3.4 MyCat分片测试

进入mycat ，执行下列语句创建一个表:

|  |
| --- |
| CREATE TABLE tb\_test (  id BIGINT(20) NOT NULL,  title VARCHAR(100) NOT NULL ,  PRIMARY KEY (id)  ) ENGINE=INNODB DEFAULT CHARSET=utf8 |

创建后你会发现，MyCat会自动将你的表转换为大写，这一点与Oracle有些类似。



我们再查看MySQL的3个库，发现表都自动创建好啦。好神奇。

接下来是插入表数据，注意，在写INSERT语句时一定要写把字段列表写出来，否则会出现下列错误提示：

错误代码： 1064

partition table, insert must provide ColumnList

我们试着插入一些数据：

|  |
| --- |
| INSERT INTO TB\_TEST(ID,TITLE) VALUES(1,'goods1');  INSERT INTO TB\_TEST(ID,TITLE) VALUES(2,'goods2');  INSERT INTO TB\_TEST(ID,TITLE) VALUES(3,'goods3'); |

我们会发现这些数据被写入到第一个节点中了，那什么时候数据会写到第二个节点中呢？

我们插入下面的数据就可以插入第二个节点了

|  |
| --- |
| INSERT INTO TB\_TEST(ID,TITLE) VALUES(5000001,'goods5000001'); |

因为我们采用的分片规则是每节点存储500万条数据，所以当ID大于5000000则会存储到第二个节点上。

目前只设置了两个节点，如果数据大于1000万条，会怎么样呢？执行下列语句测试一下

|  |
| --- |
| INSERT INTO TB\_TEST(ID,TITLE) VALUES(10000001,'goods10000001'); |

### 1.3.5 MyCat分片规则

rule.xml用于定义分片规则 ，我们这里讲解两种最常见的分片规则

（1）按主键范围分片rang-long

在配置文件中我们找到

|  |
| --- |
| <tableRule name="auto-sharding-long">  <rule>  <columns>id</columns>  <algorithm>rang-long</algorithm>  </rule>  </tableRule> |

tableRule 是定义具体某个表或某一类表的分片规则名称 columns用于定义分片的列 algorithm代表算法名称 我们接着找rang-long的定义

|  |
| --- |
| <function name="rang-long"  class="org.opencloudb.route.function.AutoPartitionByLong">  <property name="mapFile">autopartition-long.txt</property>  </function> |

Function用于定义算法 mapFile 用于定义算法需要的数据，我们打开autopartition-long.txt

|  |
| --- |
| # range start-end ,data node index  # K=1000,M=10000.  0-500M=0  500M-1000M=1  1000M-1500M=2 |

1. 一致性哈希murmur

当我们需要将数据平均分在几个分区中，需要使用一致性hash规则

我们找到function的name为murmur 的定义，将count属性改为3，因为我要将数据分成3片

|  |
| --- |
| <function name="murmur"  class="org.opencloudb.route.function.PartitionByMurmurHash">  <property name="seed">0</property><!-- 默认是0 -->  <property name="count">3</property><!-- 要分片的数据库节点数量，必须指定，否则没法分片 -->  <property name="virtualBucketTimes">160</property><!-- 一个实际的数据库节点被映射为这么多虚拟节点，默认是160倍，也就是虚拟节点数是物理节点数的160倍 -->  <!-- <property name="weightMapFile">weightMapFile</property> 节点的权重，没有指定权重的节点默认是1。以properties文件的格式填写，以从0开始到count-1的整数值也就是节点索引为key，以节点权重值为值。所有权重值必须是正整数，否则以1代替 -->  <!-- <property name="bucketMapPath">/etc/mycat/bucketMapPath</property>  用于测试时观察各物理节点与虚拟节点的分布情况，如果指定了这个属性，会把虚拟节点的murmur hash值与物理节点的映射按行输出到这个文件，没有默认值，如果不指定，就不会输出任何东西 -->  </function> |

我们再配置文件中可以找到表规则定义

|  |
| --- |
| <tableRule name="sharding-by-murmur">  <rule>  <columns>id</columns>  <algorithm>murmur</algorithm>  </rule>  </tableRule> |

但是这个规则指定的列是id ,如果我们的表主键不是id ,而是order\_id ,那么我们应该重新定义一个tableRule：

|  |
| --- |
| <tableRule name="sharding-by-murmur-order">  <rule>  <columns>order\_id</columns>  <algorithm>murmur</algorithm>  </rule>  </tableRule> |

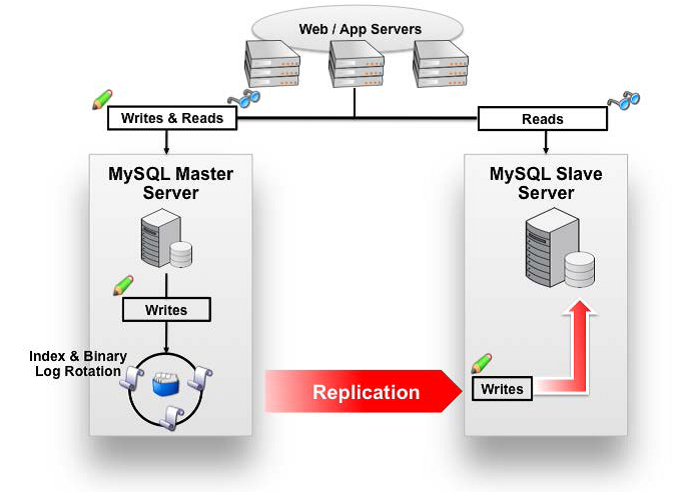
在schema.xml中配置逻辑表时，指定规则为sharding-by-murmur-order

|  |
| --- |
| <table name="tb\_order" dataNode="dn1,dn2,dn3" rule="sharding-by-murmur-order" /> |

我们测试一下，创建品优购的订单表 ,并插入数据，测试分片效果。

## 1.4了解数据库读写分离

数据库读写分离对于大型系统或者访问量很高的互联网应用来说，是必不可少的一个重要功能。对于MySQL来说，标准的读写分离是主从模式，一个写节点Master后面跟着多个读节点，读节点的数量取决于系统的压力，通常是1-3个读节点的配置





Mycat读写分离和自动切换机制，需要mysql的主从复制机制配合。

具体配置步骤参见配套的扩展文档。

# 2.Nginx的安装与启动

## 2.1什么是Nginx

Nginx 是一款高性能的 http 服务器/反向代理服务器及电子邮件（IMAP/POP3）代理服务器。由俄罗斯的程序设计师伊戈尔·西索夫（Igor Sysoev）所开发，官方测试 nginx 能够支支撑 5 万并发链接，并且 cpu、内存等资源消耗却非常低，运行非常稳定。



**Nginx 应用场景：**

1、http 服务器。Nginx 是一个 http 服务可以独立提供 http 服务。可以做网页静态服务器。

2、虚拟主机。可以实现在一台服务器虚拟出多个网站。例如个人网站使用的虚拟主机。

3、反向代理，负载均衡。当网站的访问量达到一定程度后，单台服务器不能满足用户的请求时，需要用多台服务器集群可以使用 nginx 做反向代理。并且多台服务器可以平均分担负载，不会因为某台服务器负载高宕机而某台服务器闲置的情况。

## 2.2 Nginx在Linux下的安装

重新准备一台虚拟机作为服务器。比如IP地址为192.168.25.141

### 2.2.1环境准备

（1）需要安装 gcc 的环境【此步省略】

|  |
| --- |
| yum install gcc-c++ |

（2）第三方的开发包。【此步省略】

 **PCRE**

PCRE(Perl Compatible Regular Expressions)是一个 Perl 库，包括 perl 兼容的正则表达式库。nginx 的 http 模块使用 pcre 来解析正则表达式，所以需要在 linux 上安装 pcre 库。

|  |
| --- |
| yum install -y pcre pcre-devel |

注：pcre-devel 是使用 pcre 开发的一个二次开发库。nginx 也需要此库。

 **zlib**

zlib 库提供了很多种压缩和解压缩的方式，nginx 使用 zlib 对 http 包的内容进行 gzip，所以需要在 linux 上安装 zlib 库。

|  |
| --- |
| yum install -y zlib zlib-devel |

 **OpenSSL**

OpenSSL 是一个强大的安全套接字层密码库，囊括主要的密码算法、常用的密钥和证书封装管理功能及 SSL 协议，并提供丰富的应用程序供测试或其它目的使用。nginx 不仅支持 http 协议，还支持 https（即在 ssl 协议上传输 http），所以需要在 linux安装 openssl 库。

|  |
| --- |
| yum install -y openssl openssl-devel |

### 2.2.2 Nginx下载

官方网站下载 nginx：http://nginx.org/

我们课程中使用的版本是 1.8.0 版本。

### 2.2.3 Nginx安装

第一步：把 nginx 的源码包nginx-1.8.0.tar.gz上传到 linux 系统

Alt+p 启动sftp ,将nginx-1.8.0.tar.gz上传

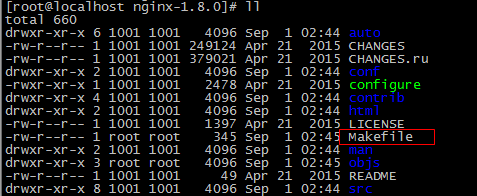
第二步：解压缩

|  |
| --- |
| tar zxvf nginx-1.8.0.tar.gz |

第三步：进入nginx-1.8.0目录 使用 configure 命令创建一 makeFile 文件。

|  |
| --- |
| ./configure \  --prefix=/usr/local/nginx \  --pid-path=/var/run/nginx/nginx.pid \  --lock-path=/var/lock/nginx.lock \  --error-log-path=/var/log/nginx/error.log \  --http-log-path=/var/log/nginx/access.log \  --with-http\_gzip\_static\_module \  --http-client-body-temp-path=/var/temp/nginx/client \  --http-proxy-temp-path=/var/temp/nginx/proxy \  --http-fastcgi-temp-path=/var/temp/nginx/fastcgi \  --http-uwsgi-temp-path=/var/temp/nginx/uwsgi \  --http-scgi-temp-path=/var/temp/nginx/scgi |

执行后可以看到Makefile文件



|  |
| --- |
| **---- 知识点小贴士 ----**  Makefile是一种配置文件， Makefile 一个工程中的源文件不计数，其按类型、功能、模块分别放在若干个目录中，makefile定义了一系列的规则来指定，哪些文件需要先编译，哪些文件需要后编译，哪些文件需要重新编译，甚至于进行更复杂的功能操作，因为 makefile就像一个Shell脚本一样，其中也可以执行操作系统的命令。 |
| **---- 知识点小贴士 ----**  **configure参数**  **./configure \**  **--prefix=/usr \                                                        指向安装目录**  **--sbin-path=/usr/sbin/nginx \                                 指向（执行）程序文件（nginx）**  **--conf-path=/etc/nginx/nginx.conf \                      指向配置文件**  **--error-log-path=/var/log/nginx/error.log \              指向log**  **--http-log-path=/var/log/nginx/access.log \            指向http-log**  **--pid-path=/var/run/nginx/nginx.pid \                      指向pid**  **--lock-path=/var/lock/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=nginx \**  **--group=nginx \**  **--with-http\_ssl\_module \                      启用ngx\_http\_ssl\_module支持（使支持https请求，需已安装openssl）**  **--with-http\_flv\_module \                       启用ngx\_http\_flv\_module支持（提供寻求内存使用基于时间的偏移量文件）**  **--with-http\_stub\_status\_module \     启用ngx\_http\_stub\_status\_module支持（获取nginx自上次启动以来的工作状态）**  **--with-http\_gzip\_static\_module \   启用ngx\_http\_gzip\_static\_module支持（在线实时压缩输出数据流）**  **--http-client-body-temp-path=/var/tmp/nginx/client/ \ 设定http客户端请求临时文件路径**  **--http-proxy-temp-path=/var/tmp/nginx/proxy/ \ 设定http代理临时文件路径**  **--http-fastcgi-temp-path=/var/tmp/nginx/fcgi/ \ 设定http fastcgi临时文件路径**  **--http-uwsgi-temp-path=/var/tmp/nginx/uwsgi \ 设定http uwsgi临时文件路径**  **--http-scgi-temp-path=/var/tmp/nginx/scgi \ 设定http scgi临时文件路径**  **--with-pcre 启用pcre库** |

第四步：编译

|  |
| --- |
| make |

第五步：安装

|  |
| --- |
| make install |

## 2.3 Nginx启动与访问

注意：启动nginx 之前，上边将临时文件目录指定为/var/temp/nginx/client， 需要在/var 下创建此 目录

|  |
| --- |
| mkdir /var/temp/nginx/client -p |

进入到Nginx目录下的sbin目录

|  |
| --- |
| cd /usr/local/ngiux/sbin |

输入命令启动Nginx

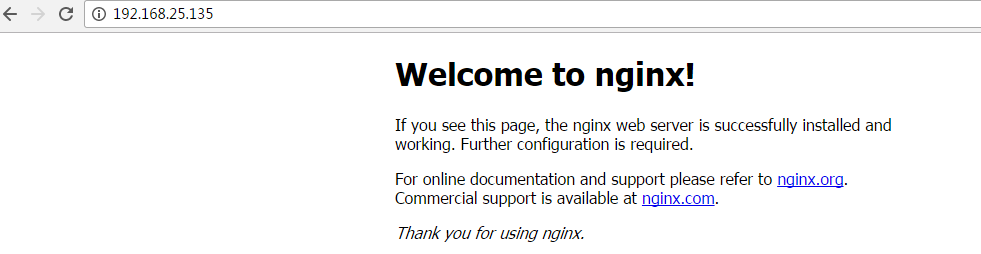
|  |
| --- |
| ./nginx |

启动后查看进程

|  |
| --- |
| ps aux|grep nginx |



地址栏输入虚拟机的IP即可访问（默认为80端口）



关闭 nginx：

|  |
| --- |
| ./nginx -s stop |

或者

|  |
| --- |
| ./nginx -s quit |

重启 nginx：

1、先关闭后启动。

2、刷新配置文件：

|  |
| --- |
| ./nginx -s reload |

# 3.Nginx静态网站部署

## 3.1 静态网站的部署

将我们之前生成的静态页（d:\item）上传到服务器的/usr/local/nginx/html下即可访问



## 3.2 配置虚拟主机

虚拟主机，也叫“网站空间”，就是把一台运行在互联网上的物理服务器划分成多个“虚拟”服务器。虚拟主机技术极大的促进了网络技术的应用和普及。同时虚拟主机的租用服务也成了网络时代的一种新型经济形式。

### 3.2.1 端口绑定

1. 上传静态网站：

将前端静态页cart.html 以及图片样式等资源 上传至 /usr/local/nginx/cart 下

将前端静态页search.html 以及图片样式等资源 上传至 /usr/local/nginx/search 下

（2）修改Nginx 的配置文件：/usr/local/nginx/conf/nginx.conf

|  |
| --- |
| server {  listen 81;  server\_name localhost;  location / {  root cart;  index cart.html;  }  }  server {  listen 82;  server\_name localhost;  location / {  root search;  index search.html;  }  } |

1. 访问测试：

地址栏输入http://192.168.25.141:81 可以看到购物车页面

地址栏输入http://192.168.25.141:82 可以看到搜索页面

### 3.2.2 域名绑定

**什么是域名：**

[域名](https://baike.baidu.com/item/%E5%9F%9F%E5%90%8D" \t "https://baike.baidu.com/item/%E5%9F%9F%E5%90%8D/_blank)（[Domain Name](https://baike.baidu.com/item/Domain Name" \t "https://baike.baidu.com/item/%E5%9F%9F%E5%90%8D/_blank)），是由一串用“点”分隔的[字符](https://baike.baidu.com/item/%E5%AD%97%E7%AC%A6" \t "https://baike.baidu.com/item/%E5%9F%9F%E5%90%8D/_blank)组成的Internet上某一台计算机或计算机组的名称，用于在数据传输时标识计算机的电子方位（有时也指地理位置，地理上的域名，指代有行政自主权的一个地方区域）。域名是一个[IP地址](https://baike.baidu.com/item/IP%E5%9C%B0%E5%9D%80" \t "https://baike.baidu.com/item/%E5%9F%9F%E5%90%8D/_blank)上有“面具” 。域名的目的是便于记忆和沟通的一组服务器的地址（[网站](https://baike.baidu.com/item/%E7%BD%91%E7%AB%99/155722" \t "https://baike.baidu.com/item/%E5%9F%9F%E5%90%8D/_blank)，电子邮件，[FTP](https://baike.baidu.com/item/FTP/13839" \t "https://baike.baidu.com/item/%E5%9F%9F%E5%90%8D/_blank)等）。域名作为力所能及难忘的互联网参与者的名称。域名按[域名系统](https://baike.baidu.com/item/%E5%9F%9F%E5%90%8D%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/%E5%9F%9F%E5%90%8D/_blank)（DNS）的规则流程组成。在[DNS](https://baike.baidu.com/item/DNS/427444" \t "https://baike.baidu.com/item/%E5%9F%9F%E5%90%8D/_blank)中注册的任何名称都是域名。域名用于各种网络环境和应用程序特定的命名和寻址目的。通常，域名表示[互联网协议](https://baike.baidu.com/item/%E4%BA%92%E8%81%94%E7%BD%91%E5%8D%8F%E8%AE%AE" \t "https://baike.baidu.com/item/%E5%9F%9F%E5%90%8D/_blank)（IP）资源，例如用于访问因特网的个人计算机，托管网站的服务器计算机，或网站本身或通过因特网传送的任何其他服务。世界上第一个注册的域名是在1985年1月注册的。

**域名级别：**

**（1）顶级域名**

顶级域名又分为两类：

一是国家顶级域名（national top-level domainnames，简称nTLDs），200多个国家都按照ISO3166国家代码分配了顶级域名，例如中国是cn，美国是us，日本是jp等；

二是国际顶级域名（international top-level domain names，简称iTDs），例如表示工商企业的 .Com .Top，表示网络提供商的.net，表示非盈利组织的.org，表示教育的.edu，以及没有限制的中性域名如.xyz等。大多数域名争议都发生在com的顶级域名下，因为多数公司上网的目的都是为了赢利。但因为自2014年以来新顶级域名的发展，域名争议案件数量增长幅度越来越大[5] 。为加强域名管理，解决域名资源的紧张，Internet协会、Internet分址机构及世界知识产权组织（WIPO）等国际组织经过广泛协商， 在原来三个国际通用顶级域名：（com）的基础上，新增加了7个国际通用顶级域名：firm（公司企业）、store（销售公司或企业）、Web（突出WWW活动的单位）、arts（突出文化、娱乐活动的单位）、rec (突出消遣、娱乐活动的单位）、info(提供信息服务的单位）、nom(个人），并在世界范围内选择新的注册机构来受理域名注册申请。

例如：baidu.com

**（2）二级域名**

二级域名是指顶级域名之下的域名，在国际顶级域名下，它是指域名注册人的网上名称，例如 ibm，yahoo，microsoft等；在国家顶级域名下，它是表示注册企业类别的符号，例如.top，com，edu，gov，net等。

中国在国际互联网络信息中心（Inter NIC） 正式注册并运行的顶级域名是CN，这也是中国的一级域名。在顶级域名之下，中国的二级域名又分为类别域名和行政区域名两类。类别域名共7个， 包括用于科研机构的ac；用于工商金融企业的com、top；用于教育机构的edu；用于政府部门的 gov；用于互联网络信息中心和运行中心的net；用于非盈利组织的org。而行政区域名有34个，分别对应于中国各省、自治区和直辖市。

例如：map.baidu.com

**（3）三级域名**

三级域名用字母（ A～Z，a～z，大小写等）、数字（0～9）和连接符（－）组成， 各级域名之间用实点（.）连接，三级域名的长度不能超过20个字符。如无特殊原因，建议采用申请人的英文名（或者缩写）或者汉语拼音名 （或者缩写） 作为三级域名，以保持域名的清晰性和简洁性。

例如：

item.map.baidu.com

**域名与IP绑定：**

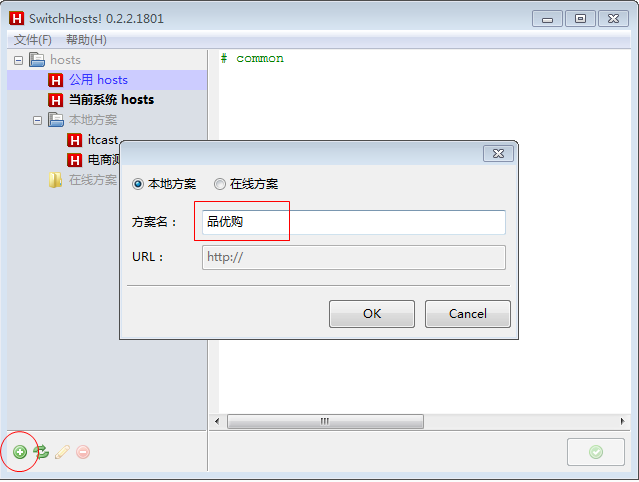
一个域名对应一个 ip 地址，一个 ip 地址可以被多个域名绑定。

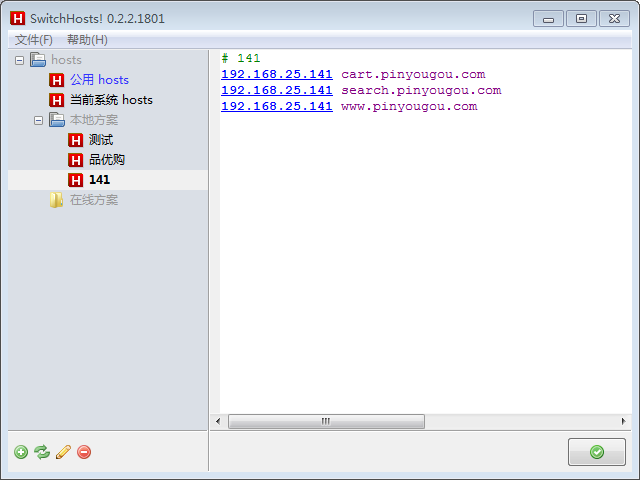
本地测试可以修改 hosts 文件（C:\Windows\System32\drivers\etc）

可以配置域名和 ip 的映射关系，如果 hosts 文件中配置了域名和 ip 的对应关系，不需要走dns 服务器。

我们可以通过一个叫SwitchHosts的软件来修改域名指向

新增方案：





指定绑定规则 IP 域名 然后确定。

做好域名指向后，修改nginx配置文件

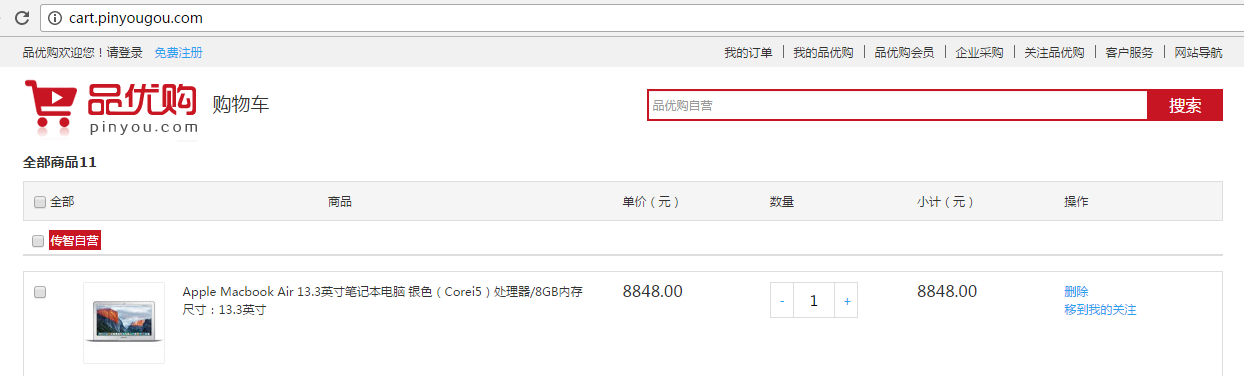
|  |
| --- |
| server {  listen 80;  server\_name cart.pinyougou.com;  location / {  root cart;  index cart.html;  }  }  server {  listen 80;  server\_name search.pinyougou.com;  location / {  root search;  index search.html;  }  } |

执行以下命令，刷新配置

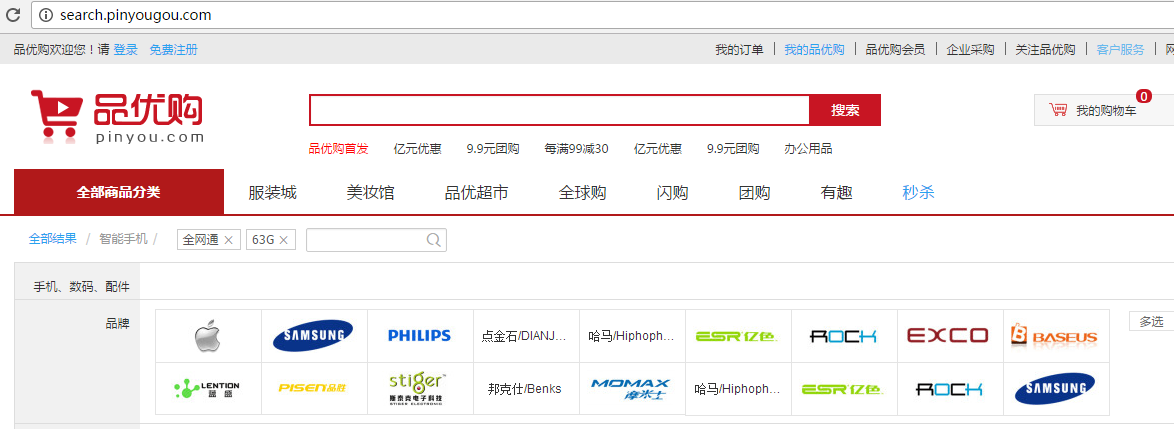
|  |
| --- |
| [root@localhost sbin]# ./nginx -s reload |

测试：

地址栏输入http://cart.pinyougou.com/



地址栏输入http://search.pinyougou.com/



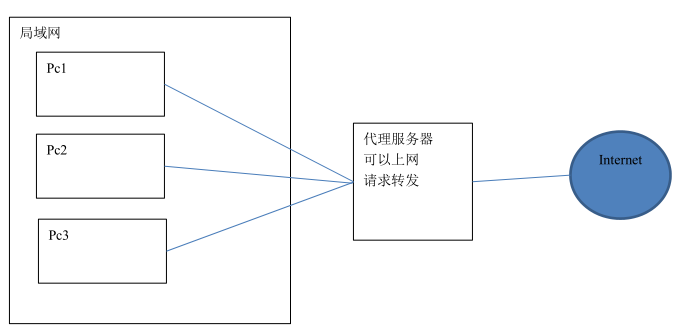
# 4.Nginx反向代理与负载均衡

## 4.1 反向代理

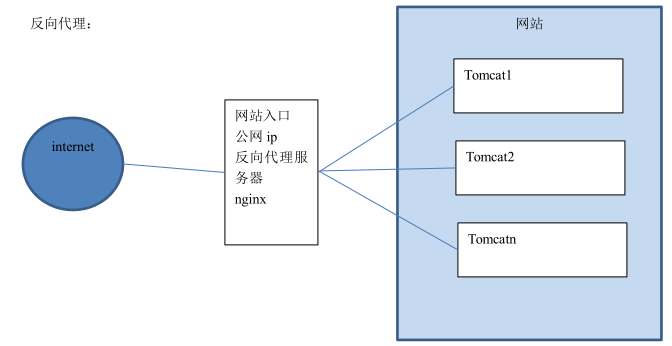
### 4.1.1 什么是反向代理

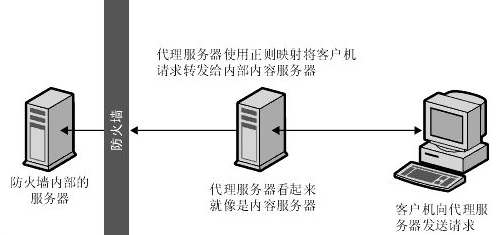
反向代理（Reverse Proxy）方式是指以[代理服务器](http://baike.baidu.com/item/%E4%BB%A3%E7%90%86%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "http://baike.baidu.com/_blank)来接受internet上的连接请求，然后将请求转发给内部网络上的服务器，并将从服务器上得到的结果返回给internet上请求连接的客户端，此时代理服务器对外就表现为一个反向代理服务器。

首先我们先理解正向代理，如下图：



正向代理是针对你的客户端，而反向代理是针对服务器的，如下图





### 4.1.2 配置反向代理-准备工作

（1） 将网站首页页面部署到tomcat中（ROOT目录），上传到服务器。

（2）启动TOMCAT，输入网址http://192.168.25.141:8080可以看到网站首页

### 4.1.3 配置反向代理

（1）在Nginx主机修改 Nginx配置文件

|  |
| --- |
| upstream tomcat-portal {  server 192.168.25.141:8080;  }  server {  listen 80;  server\_name www.pinyougou.com;  location / {  proxy\_pass http://tomcat-portal;  index index.html;  }  } |

1. 重新启动Nginx 然后用浏览器测试： [www.pinyougou.com](http://www.pinyougou.com) （此域名须配置域名指向）

## 4.2 负载均衡

### 4.2.1 什么是负载均衡

负载均衡 建立在现有网络结构之上，它提供了一种廉价有效透明的方法扩展[网络设备](http://baike.baidu.com/item/%E7%BD%91%E7%BB%9C%E8%AE%BE%E5%A4%87" \t "http://baike.baidu.com/_blank)和[服务器](http://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "http://baike.baidu.com/_blank)的带宽、增加[吞吐量](http://baike.baidu.com/item/%E5%90%9E%E5%90%90%E9%87%8F" \t "http://baike.baidu.com/_blank)、加强网络数据处理能力、提高网络的灵活性和可用性。

负载均衡，英文名称为Load Balance，其意思就是分摊到多个操作单元上进行执行，例如Web[服务器](http://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "http://baike.baidu.com/_blank)、[FTP服务器](http://baike.baidu.com/item/FTP%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "http://baike.baidu.com/_blank)、[企业](http://baike.baidu.com/item/%E4%BC%81%E4%B8%9A" \t "http://baike.baidu.com/_blank)关键应用服务器和其它关键任务服务器等，从而共同完成工作任务。

### 4.2.2 配置负载均衡-准备工作

1. 将刚才放有首页工程的tomcat复制两份，端口分别为8180 和8280 。
2. 分别启动这两个tomcat服务器的tomcat服务。
3. 为了能够区分是访问哪个服务器的网站，可以在首页标题加上标记以便区分。

### 4.2.3 配置负载均衡

修改 Nginx配置文件：

|  |
| --- |
| upstream tomcat-portal {  server 192.168.25.141:8080;  server 192.168.25.141:8180;  server 192.168.25.141:8280;  }  server {  listen 80;  server\_name www.pinyougou.com;    location / {  proxy\_pass http://tomcat-portal;  index index.html;  }    } |

地址栏输入http://www.pinyougou.com/ 刷新观察每个网页的标题，看是否不同。

经过测试，三台服务器出现的概率各为33.3333333%，交替显示。

如果其中一台服务器性能比较好，想让其承担更多的压力，可以设置权重。

比如想让NO.1出现次数是其它服务器的2倍，则修改配置如下：

|  |
| --- |
| upstream tomcat-portal {  server 192.168.25.141:8080;  server 192.168.25.141:8180 weight=2;  server 192.168.25.141:8280;  } |

经过测试，每刷新四次，有两次是8180

## 4.3 了解高可用

### 4.3.1什么是高可用

nginx 作为负载均衡器，所有请求都到了 nginx，可见 nginx 处于非常重点的位置，如果nginx 服务器宕机后端 web 服务将无法提供服务，影响严重。

为了屏蔽负载均衡服务器的宕机，需要建立一个备份机。主服务器和备份机上都运行高可用（High Availability）监控程序，通过传送诸如“I am alive”这样的信息来监控对方的运行状况。当备份机不能在一定的时间内收到这样的信息时，它就接管主服务器的服务 IP 并继续提供负载均衡服务；当备份管理器又从主管理器收到“I am alive”这样的信息时，它就释放服务 IP 地址，这样的主服务器就开始再次提供负载均衡服务。

### 4.3.2 keepalived简介

keepalived 是集群管理中保证集群高可用的一个服务软件，用来防止单点故障。

Keepalived 的作用是检测 web 服务器的状态，如果有一台 web 服务器死机，或工作出现故障，Keepalived 将检测到，并将有故障的 web 服务器从系统中剔除，当 web 服务器工作正常后 Keepalived 自动将 web 服务器加入到服务器群中，这些工作全部自动完成，不需要人工干涉，需要人工做的只是修复故障的 web 服务器。

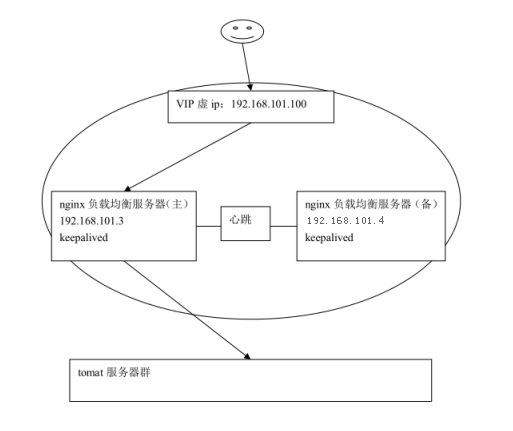
keepalived 是以 VRRP 协议为实现基础的，VRRP 全称 Virtual Router Redundancy Protocol，即虚拟路由冗余协议。

虚拟路由冗余协议，可以认为是实现路由器高可用的协议，即将 N 台提供相同功能的

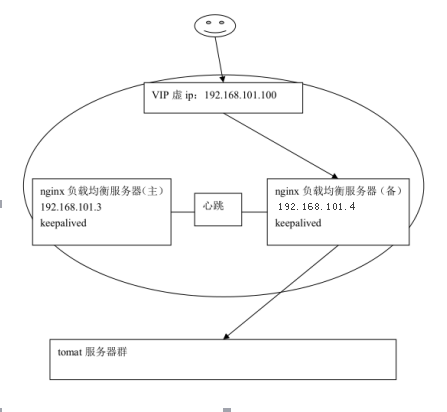
路由器组成一个路由器组，这个组里面有一个 master 和多个 backup，master 上面有一个对外提供服务的 vip（VIP = Virtual IPAddress，虚拟 IP 地址，该路由器所在局域网内其他机器的默认路由为该 vip），master 会发组播，当 backup 收不到 VRRP 包时就认为 master 宕掉了，这时就需要根据 VRRP 的优先级来选举一个 backup 当 master。这样的话就可以保证路由器的高可用了。

keepalived 主要有三个模块，分别是 core、check 和 VRRP。core 模块为 keepalived 的核心，负责主进程的启动、维护以及全局配置文件的加载和解析。check 负责健康检查，包括常见的各种检查方式。VRRP 模块是来实现 VRRP 协议的。

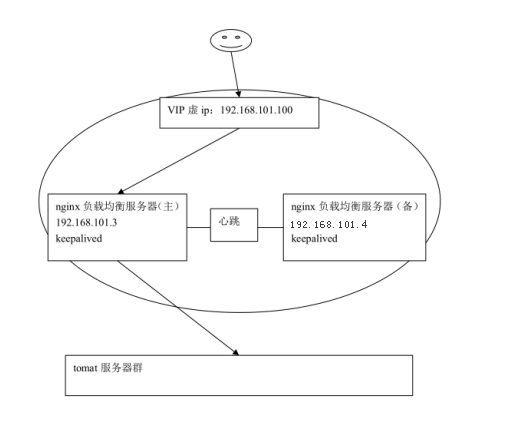
初始状态：



主机宕机：



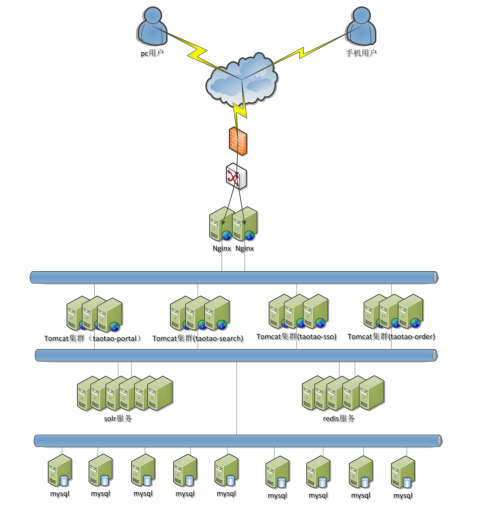
主机恢复：



Keepalived的安装与配置详见配套的扩展文档

# 5.品优购部署方案

## 5.1 品优购网络拓扑图



## 5.2 品优购服务器列表

|  |  |  |
| --- | --- | --- |
| **用途** | **服务器数量** | **安装软件** |
| 反向代理服务器 | 3 | Nginx Keepalived |
| 小计 | 3 |  |
| 网站门户web | 5 | tomcat |
| 搜索web | 3 | tomcat |
| 商品详细页web + 商品详细页生成服务 | 3 | Nginx + tomcat |
| 购物车web | 3 | tomcat |
| 秒杀频道 | 3 | tomcat |
| 用户中心 | 2 | tomcat |
| 评论中心 | 2 | tomcat |
| 单点登录 | 1 | Tomcat |
| 商家后台 | 1 | Tomcat |
| 运营商管理后台 | 1 | Tomcat |
| 小计 | 25 |  |
| 短信发送网关 | 2 | 微服务 |
| 商家商品服务 | 5 | tomcat |
| 广告内容服务 | 3 | tomcat |
| 购物车服务 | 3 | tomcat |
| 订单服务 | 3 | tomcat |
| 支付服务 | 3 | tomcat |
| 秒杀服务 | 3 | tomcat |
| 评论服务 | 3 | tomcat |
| 搜索服务 | 3 | tomcat |
| 用户服务 | 3 | tomcat |
| 小计 | 31 |  |
| dubbox注册中心 | 3 | Zookeeper |
| Solr入口 | 3 | Zookeeper |
| Redis | 6 |  |
| Solr | 4 |  |
| 消息中间件 | 3 | ActiveMQ |
| 小计 | 19 |  |
| MySQL | 6 |  |
| MyCAT | 1 |  |
| 小计 | 7 |  |
| Tracker | 3 |  |
| Storage | 5 |  |
| 小计 | 8 |  |
| 总计 | 93 |  |