

# 基于 Keepalived+Nginx+Tomcat+MySQL 部署双机热备、负载均衡项目

## 一、项目名称

基于 Keepalived+Nginx+Tomcat+MySQL 部署双机热备、负载均衡项目

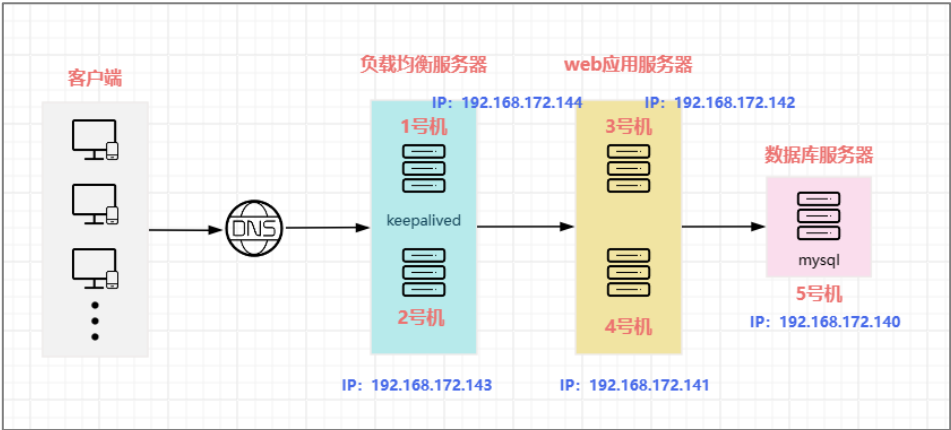
## 二、项目环境

实验环境：VMware+Centos7

表 1 项目机器设置

编号	IP 地址	应用
1 号机	192.168.172.144	负载均衡（主）服务器
2 号机	192.168.172.143	负载均衡（备）服务器
3 号机	192.168.172.142	web 应用服务器 1
4 号机	192.168.172.141	web 应用服务器 2
5 号机	192.168.172.140	数据库服务器

## 三、项目架构



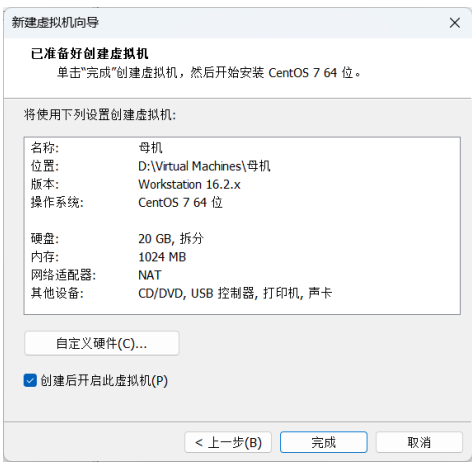
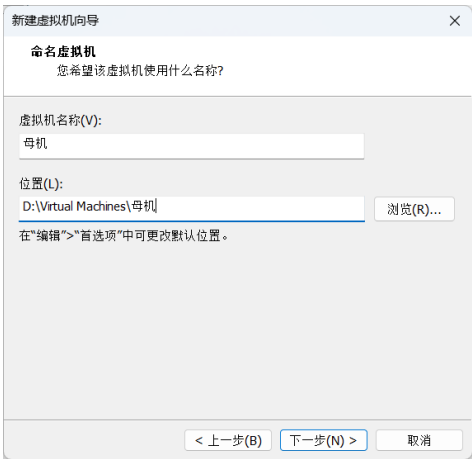
项目架构图

## 四、实现步骤

### （一）创建虚拟机母机

#### 1. 配置虚拟机

在 VMware 中新建一台虚拟机用作项目其他机器的母机：





## 2. 安装 vim

使用 `yum` 命令安装 `vim`，以便后续编辑文件：

```
[root@localhost etc]# yum install -y vim
已加载插件: fastestmirror
Determining fastest mirrors
* base: mirrors.bupt.edu.cn
* extras: mirrors.bupt.edu.cn
* updates: mirrors.bupt.edu.cn
```

## 3. 安装 wget

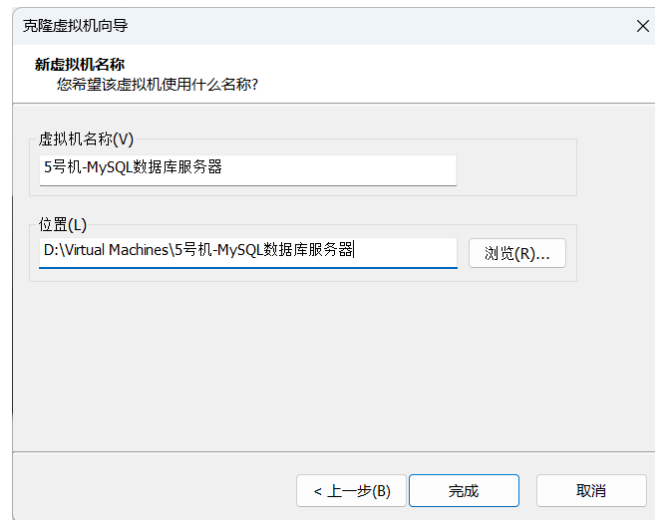
使用 `yum` 命令安装 `wget`，以便后续从互联网下载文件：

```
[root@localhost etc]# yum install -y wget
已加载插件: fastestmirror
Loading mirror speeds from cached hostfile
* base: mirrors.bupt.edu.cn
* extras: mirrors.bupt.edu.cn
* updates: mirrors.bupt.edu.cn
```

## (二) 配置 5 号机-MySQL 数据库服务器

### 1. 克隆

从母机克隆出 5 号机，用作 MySQL 数据库服务器：



### 2. 配置静态 IP 地址

配置 5 号机的 IP 地址，在此路径下更改网卡文件：

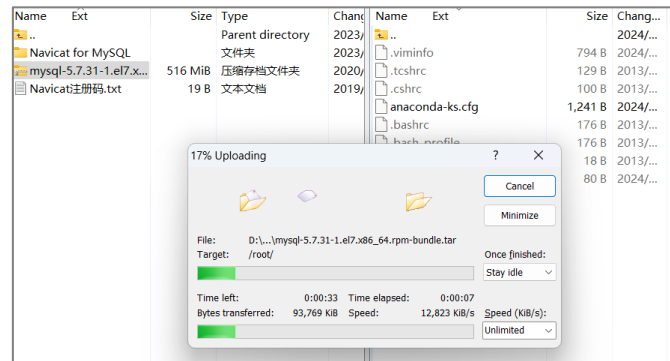
`/etc/sysconfig/network-scripts/ifcfg-ens33`

```
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="static"
DEFROUTE="yes"
IPV4_FAILURE_FATAL="no"
IPV6INIT="yes"
IPV6_AUTOCONF="yes"
IPV6_DEFROUTE="yes"
IPV6_FAILURE_FATAL="no"
IPV6_ADDR_GEN_MODE="stable-privacy"
NAME="ens33"
UUID="36deebff-efc0-4686-b170-1d067898d6ab"
DEVICE="ens33"
ONBOOT="yes"
IPADDR=192.168.172.140
NETMASK=255.255.255.0
GATEWAY=192.168.172.2
DNS1=114.114.114.114
DNS2=8.8.8.8
```

保存并退出后重启网卡：`systemctl restart network`

### 3. 安装 MySQL 数据库

在 5 号机上安装 MySQL 数据库，使用 scp 工具将下载好的 MySQL 数据库的安装包上传到这台机器。



#### (1) 安装依赖

在使用 MySQL 前需要先安装好必须的依赖：

```
yum -y install make gcc-c++ cmake bison-devel ncurses-devel  
libaio libaio-devel nettools
```

#### (2) 卸载 mariadb

在安装 MySQL 之前，由于 Linux 系统（如 CentOS 7）默认已经集成 MariaDB 数据库服务，需要检查并处理 MariaDB 的存在以避免与 MySQL 产生冲突。因为 MariaDB 是从 MySQL 分支出来的一个项目，虽然在大多数功能上与 MySQL 兼容，但二者的文件和配置可能会有重叠部分，直接安装 MySQL 可能会导致文件覆盖、服务冲突等问题。

解决方法是在安装 MySQL 前卸载已有的 MariaDB 及其相关组件。

列出已安装的 MariaDB 相关包：`yum list installed | grep mariadb`

卸载 mariadb：`yum -y remove mariadb-libs`

```
[root@localhost ~]# yum list installed | grep mariadb  
mariadb-libs.x86_64 1:5.5.65-1.el7  
[root@localhost ~]# yum -y remove mariadb-libs  
已加载插件：fastestmirror  
正在解决依赖关系  
--> 正在检查事务
```

### (3) 解压并安装

解压 MySQL 的安装包: `tar -xvf mysql-5.7.31-1.el7.x86_64.rpm-bundle.tar`

```
[root@localhost ~]# tar -xvf mysql-5.7.31-1.el7.x86_64.rpm-bundle.tar
mysql-community-embedded-devel-5.7.31-1.el7.x86_64.rpm
mysql-community-libs-5.7.31-1.el7.x86_64.rpm
mysql-community-client-5.7.31-1.el7.x86_64.rpm
mysql-community-server-5.7.31-1.el7.x86_64.rpm
mysql-community-embedded-5.7.31-1.el7.x86_64.rpm
mysql-community-embedded-compat-5.7.31-1.el7.x86_64.rpm
mysql-community-common-5.7.31-1.el7.x86_64.rpm
mysql-community-libs-compat-5.7.31-1.el7.x86_64.rpm
mysql-community-devel-5.7.31-1.el7.x86_64.rpm
mysql-community-test-5.7.31-1.el7.x86_64.rpm
[root@localhost ~]# ls
anaconda-ks.cfg
mysql-5.7.31-1.el7.x86_64.rpm-bundle.tar
mysql-community-client-5.7.31-1.el7.x86_64.rpm
mysql-community-common-5.7.31-1.el7.x86_64.rpm
mysql-community-devel-5.7.31-1.el7.x86_64.rpm
mysql-community-embedded-5.7.31-1.el7.x86_64.rpm
mysql-community-embedded-compat-5.7.31-1.el7.x86_64.rpm
mysql-community-embedded-devel-5.7.31-1.el7.x86_64.rpm
mysql-community-libs-5.7.31-1.el7.x86_64.rpm
mysql-community-libs-compat-5.7.31-1.el7.x86_64.rpm
mysql-community-server-5.7.31-1.el7.x86_64.rpm
mysql-community-test-5.7.31-1.el7.x86_64.rpm
```

安装 5 个 rpm 包:

```
rpm -ivh mysql-community-common-5.7.31-1.el7.x86_64.rpm
rpm -ivh mysql-community-libs-5.7.31-1.el7.x86_64.rpm
rpm -ivh mysql-community-libs-compat-5.7.31-1.el7.x86_64.rpm
rpm -ivh mysql-community-client-5.7.31-1.el7.x86_64.rpm
rpm -ivh mysql-community-server-5.7.31-1.el7.x86_64.rpm --
nodeps --force
```

```
[root@localhost ~]# rpm -ivh mysql-community-common-5.7.31-1.el7.x86_64.rpm
警告: mysql-community-common-5.7.31-1.el7.x86_64.rpm: 头V3 DSA/SHA1 Signature, 密钥 ID 5072e1f5:
KEY
准备中... [100%]
正在升级/安装...
1:mysql-community-common-5.7.31-1.el7.x86_64.rpm [100%]
[root@localhost ~]# rpm -ivh mysql-community-libs-5.7.31-1.el7.x86_64.rpm
警告: mysql-community-libs-5.7.31-1.el7.x86_64.rpm: 头V3 DSA/SHA1 Signature, 密钥 ID 5072e1f5:
y
准备中... [100%]
正在升级/安装...
1:mysql-community-libs-5.7.31-1.el7.x86_64.rpm [100%]
[root@localhost ~]# rpm -ivh mysql-community-libs-compat-5.7.31-1.el7.x86_64.rpm
警告: mysql-community-libs-compat-5.7.31-1.el7.x86_64.rpm: 头V3 DSA/SHA1 Signature, 密钥 ID 5072e1f5:
5: NOKEY
准备中... [100%]
正在升级/安装...
1:mysql-community-libs-compat-5.7.31-1.el7.x86_64.rpm [100%]
[root@localhost ~]# rpm -ivh mysql-community-client-5.7.31-1.el7.x86_64.rpm
警告: mysql-community-client-5.7.31-1.el7.x86_64.rpm: 头V3 DSA/SHA1 Signature, 密钥 ID 5072e1f5:
KEY
准备中... [100%]
正在升级/安装...
1:mysql-community-client-5.7.31-1.el7.x86_64.rpm [100%]
[root@localhost ~]# rpm -ivh mysql-community-server-5.7.31-1.el7.x86_64.rpm --nodeps --force
警告: mysql-community-server-5.7.31-1.el7.x86_64.rpm: 头V3 DSA/SHA1 Signature, 密钥 ID 5072e1f5:
KEY
准备中... [100%]
正在升级/安装...
1:mysql-community-server-5.7.31-1.el7.x86_64.rpm [100%]
```

删除 MySQL 安装包 (一个 tar 包和解压得到的多个 rpm 包): `rm -f mysql*`

```
[root@localhost ~]# ls
anaconda-ks.cfg
mysql-5.7.31-1.el7.x86_64.rpm-bundle.tar
mysql-community-client-5.7.31-1.el7.x86_64.rpm
mysql-community-common-5.7.31-1.el7.x86_64.rpm
mysql-community-devel-5.7.31-1.el7.x86_64.rpm
mysql-community-embedded-5.7.31-1.el7.x86_64.rpm
mysql-community-embedded-compat-5.7.31-1.el7.x86_64.rpm
mysql-community-embedded-devel-5.7.31-1.el7.x86_64.rpm
mysql-community-libs-5.7.31-1.el7.x86_64.rpm
mysql-community-libs-compat-5.7.31-1.el7.x86_64.rpm
mysql-community-server-5.7.31-1.el7.x86_64.rpm
mysql-community-test-5.7.31-1.el7.x86_64.rpm
[root@localhost ~]# rm -f mysql*
[root@localhost ~]# ls
anaconda-ks.cfg
```

启动 MySQL 数据库: `systemctl start mysqld`

检查运行状态: `systemctl status mysqld`

```
[root@localhost ~]# systemctl start mysqld
[root@localhost ~]# systemctl status mysqld
● mysqld.service - MySQL Server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
   Active: active (running) since 二 2024-01-02 15:39:40 CST; 7s ago
     Docs: man:mysqld(8)
           http://dev.mysql.com/doc/refman/en/using-systemd.html
   Process: 14204 ExecStart=/usr/sbin/mysqld --daemonize --pid-file=/var/run/mysqld/mysqld.pi
LD_OPTS (code=exited, status=0/SUCCESS)
   Process: 14155 ExecStartPre=/usr/bin/mysqld_pre_systemd (code=exited, status=0/SUCCESS)
   Main PID: 14207 (mysqld)
    CGroup: /system.slice/mysqld.service
            └─14207 /usr/sbin/mysqld --daemonize --pid-file=/var/run/mysqld/mysqld.pid
```

## 4. 设置数据库密码

使用 `grep` 命令在日志文件中搜索数据库的临时密码: `grep "password"`

`/var/log/mysqld.log`

```
[root@localhost ~]# grep "password" /var/log/mysqld.log
2024-01-02T07:39:38.330449Z 1 [Note] A temporary password is generated for root@localhost: aVXx*=7niUrr
```

使用临时密码登录数据库：`mysql -u root -p`

```
[root@localhost ~]# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
Server version: 5.7.31

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

在项目学习环境中，为了便于记忆，设置一个简单的密码。然而数据库默认要求密码必须满足足够复杂。因此，通过配置 `validate_password` 插件的相关系统变量来降低复杂度要求。

`SET GLOBAL validate_password_policy=0;` -- 改为 LOW 级别，允许较弱的密码；

`SET GLOBAL validate_password_length = 1;` -- 设置一个较低的最小长度

```
mysql> set global validate_password_policy=0;
Query OK, 0 rows affected (0.00 sec)

mysql> set global validate_password_length=1;
Query OK, 0 rows affected (0.00 sec)
```

然后，按照下面格式的命令修改密码：

`ALTER USER '用户名'@'主机名' IDENTIFIED BY '新密码';`

```
mysql> ALTER USER 'root'@'localhost' IDENTIFIED BY 'root';
Query OK, 0 rows affected (0.00 sec)
```

退出，使用新密码重新登录。

```
mysql> exit
Bye
[root@localhost ~]# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
```



## 5. 设置远程登录数据库

### (1) 赋予远程登录权限

修改用户表以允许远程访问：在 MySQL 命令行客户端中，执行以下 SQL 命令，为特定用户授予从任意主机（% 表示所有 IP 地址）进行连接的权限。

```
GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' WITH GRANT OPTION;
```

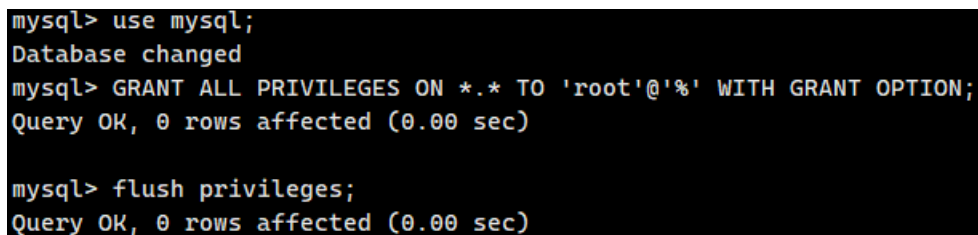
这里 ALL PRIVILEGES 指的是赋予所有权限，可以根据需要指定具体的权限，如 SELECT, INSERT, UPDATE, DELETE, CREATE, 等等。

然后通过命令 `FLUSH PRIVILEGES;` 刷新权限

```
mysql> use mysql;
```

```
mysql> GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY  
'root' WITH GRANT OPTION;
```

```
mysql> flush privileges;
```



```
mysql> use mysql;  
Database changed  
mysql> GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' WITH GRANT OPTION;  
Query OK, 0 rows affected (0.00 sec)  
  
mysql> flush privileges;  
Query OK, 0 rows affected (0.00 sec)
```

### (2) 开放 3306 端口

除此之外，还要确保防火墙和 MySQL 配置允许 3306 端口上的入站流量，选择开放 3306 端口

```
firewall-cmd --permanent --add-port=3306/tcp
```

```
firewall-cmd --reload
```

或者关闭防火墙

```
systemctl stop firewalld
```

```
systemctl disable firewalld
```

```
[root@localhost ~]# firewall-cmd --permanent --add-port=3306/tcp
success
[root@localhost ~]# firewall-cmd --list-ports

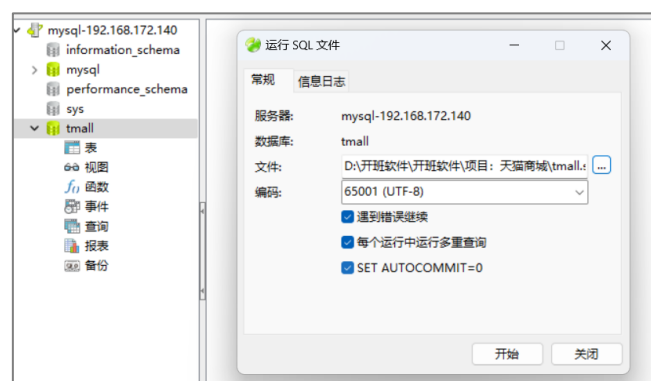
[root@localhost ~]# firewall-cmd --reload
success
[root@localhost ~]# firewall-cmd --list-ports
3306/tcp
```

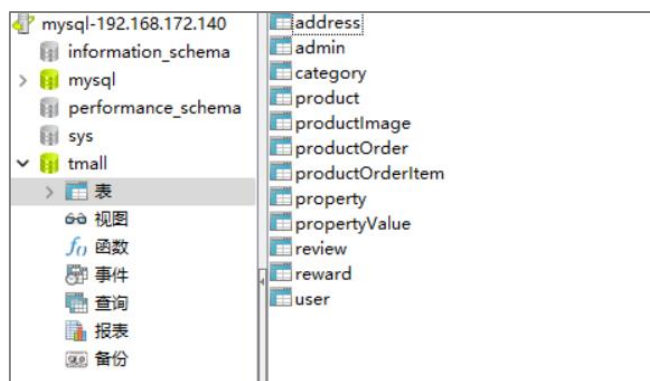
## 6. 导入数据库

使用 navicat 远程登录数据库：



新建一个 tmall 数据库，运行 sql 文件后刷新：

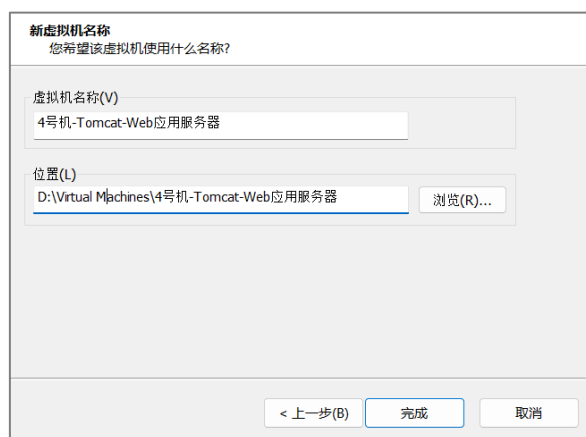




### （三）配置 4 号机-Tomcat-Web 应用服务器

#### 1. 克隆

从母机克隆出 4 号机，用作 Tomcat-Web 应用服务器：



#### 2. 配置静态 IP 地址

在此路径下更改网卡文件：

[/etc/sysconfig/network-scripts/ifcfg-ens33](#)

```
1 TYPE="Ethernet"
2 PROXY_METHOD="none"
3 BROWSER_ONLY="no"
4 BOOTPROTO="static"
5 DEFROUTE="yes"
6 IPV4_FAILURE_FATAL="no"
7 IPV6INIT="yes"
8 IPV6_AUTOCONF="yes"
9 IPV6_DEFROUTE="yes"
10 IPV6_FAILURE_FATAL="no"
11 IPV6_ADDR_GEN_MODE="stable-privacy"
12 NAME="ens33"
13 UUID="36deebff-efc0-4686-b170-1d067898d6ab"
14 DEVICE="ens33"
15 ONBOOT="yes"
16 IPADDR=192.168.172.141
17 NETMASK=255.255.255.0
18 GATEWAY=192.168.172.2
19 DNS1=114.114.114.114
20 DNS2=8.8.8.8
```

保存并退出后重启网卡: `systemctl restart network`

### 3. 安装 tomcat

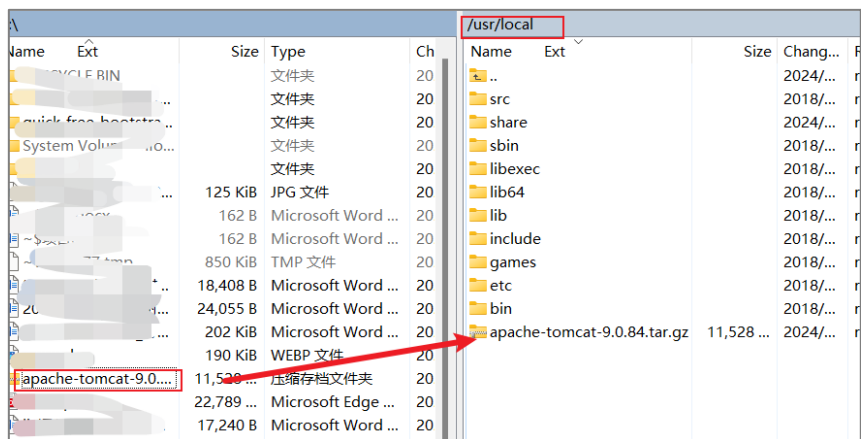
#### (1) 安装 Java 环境

Tomcat 是一个基于 Java 的 Web 应用服务器，用于部署和运行 Java Web 应用程序，这些程序是以 Java Servlet、JSP 等技术编写的。为了能够运行 Tomcat 及其托管的应用程序，必须先系统在系统上安装 Java Development Kit (JDK)。

```
[root@localhost ~]# yum install -y java-1.8.0-openjdk
已加载插件: fastestmirror
Loading mirror speeds from cached hostfile
* base: mirrors.bupt.edu.cn
* extras: mirrors.bupt.edu.cn
* updates: mirrors.bupt.edu.cn
```

#### (2) 安装并解压

将提前下载好的 tomcat 安装包使用 scp 上传到 4 号服务器的 `/usr/local` 目录。



解压缩后放入新建的/usr/local/tomcat 目录。

```
[root@localhost ~]# cd /usr/local
[root@localhost local]# ls
apache-tomcat-9.0.84.tar.gz  bin  etc  games  include  lib  lib64  libexec  sbin  share  src
[root@localhost local]# tar -xvf apache-tomcat-9.0.84.tar.gz
apache-tomcat-9.0.84/conf/
apache-tomcat-9.0.84/conf/catalina.policy
apache-tomcat-9.0.84/conf/catalina.properties
apache-tomcat-9.0.84/conf/context.xml
apache-tomcat-9.0.84/bin/version.sh
[root@localhost local]# ls
apache-tomcat-9.0.84  bin  games  lib  libexec  share
apache-tomcat-9.0.84.tar.gz  etc  include  lib64  sbin  src
[root@localhost local]# rm -f apache-tomcat-9.0.84.tar.gz
[root@localhost local]# ls
apache-tomcat-9.0.84  bin  etc  games  include  lib  lib64  libexec  sbin  share  src
[root@localhost local]# mkdir tomcat
[root@localhost local]# ls
apache-tomcat-9.0.84  bin  etc  games  include  lib  lib64  libexec  sbin  share  src  tomcat
[root@localhost local]# mv apache-tomcat-9.0.84/ tomcat
[root@localhost local]# ls
bin  etc  games  include  lib  lib64  libexec  sbin  share  src  tomcat
[root@localhost local]# cd tomcat
[root@localhost tomcat]# ls
apache-tomcat-9.0.84
```

打开解压后的目录：

```
[root@localhost tomcat]# cd apache-tomcat-9.0.84/
[root@localhost apache-tomcat-9.0.84]# ls
bin  conf  lib  logs  README.md  RUNNING.txt  webapps
BUILDING.txt  CONTRIBUTING.md  LICENSE  NOTICE  RELEASE-NOTES  temp  work
```

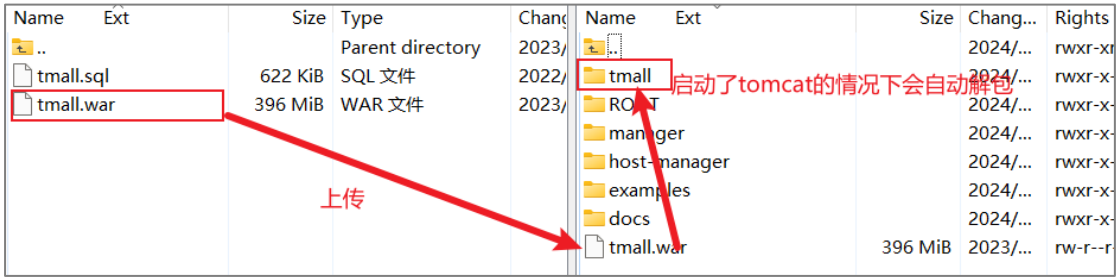
执行 bin 目录中的 startup.sh 启动 tomcat：

```
[root@localhost apache-tomcat-9.0.84]# cd bin
[root@localhost bin]# ls
bootstrap.jar      commons-daemon-native.tar.gz  makebase.sh      tomcat-juli.jar
catalina.bat      configtest.bat               setclasspath.bat  tomcat-native.tar.gz
catalina.sh        configtest.sh                setclasspath.sh   tool-wrapper.bat
catalina-tasks.xml daemon.sh                     shutdown.bat       tool-wrapper.sh
ciphers.bat        digest.bat                   shutdown.sh        version.bat
ciphers.sh         digest.sh                     startup.bat        version.sh
commons-daemon.jar makebase.bat                  startup.sh
[root@localhost bin]# ./startup.sh
Using CATALINA_BASE:   /usr/local/tomcat/apache-tomcat-9.0.84
Using CATALINA_HOME:   /usr/local/tomcat/apache-tomcat-9.0.84
Using CATALINA_TMPDIR: /usr/local/tomcat/apache-tomcat-9.0.84/temp
Using JRE_HOME:        /usr
Using CLASSPATH:        /usr/local/tomcat/apache-tomcat-9.0.84/bin/bootstrap.jar:/usr/local/tomcat/a
                           pache-tomcat-9.0.84/bin/tomcat-juli.jar
Using CATALINA_OPTS:
Tomcat started.
```

4. 部署项目

(1) 上传项目 war 包

使用 scp 部署项目的 war 包到 tomcat 的 webapps 目录：



(2) 修改配置文件

修改项目的配置文件，连接 mysql 数据库服务器。

/usr/local/tomcat/apache-tomcat-9.0.84/webapps/tmall/WEB-INF/classes						
Name	Ext	Size	Chang...	Rights	Owner	
			2024/...	rw-r--r--	root	
public			2024/...	rw-r--r--	root	
mybatis			2024/...	rw-r--r--	root	
META-INF			2024/...	rw-r--r--	root	
com			2024/...	rw-r--r--	root	
log4j2.xml		2,370 B	2022/...	rw-r--r--	root	
application.properties		836 B	2023/...	rw-r--r--	root	

修改配置文件中的数据库服务器的 IP 地址。

```
1 #配置服务端口
2 server.port=8080
3
4 #配置数据源
5 spring.datasource.driver-class-name=com.mysql.jdbc.Driver
6 spring.datasource.url=jdbc:mysql://192.168.172.140:3306/tmall?characterEncoding=utf-8&useSSL
  lse
7 spring.datasource.username=root
8 spring.datasource.password=root
9
```

保存并退出后，重启 tomcat 使配置生效。

```
[root@localhost classes]# pwd
/usr/local/tomcat/apache-tomcat-9.0.84/webapps/tmall/WEB-INF/classes
[root@localhost classes]# /usr/local/tomcat/apache-tomcat-9.0.84/bin/shutdown.sh
Using CATALINA_BASE:   /usr/local/tomcat/apache-tomcat-9.0.84
Using CATALINA_HOME:   /usr/local/tomcat/apache-tomcat-9.0.84
Using CATALINA_TMPDIR: /usr/local/tomcat/apache-tomcat-9.0.84/temp
Using JRE_HOME:        /usr
Using CLASSPATH:        /usr/local/tomcat/apache-tomcat-9.0.84/bin/bootstrap.jar:/usr/local/tomcat/a
pache-tomcat-9.0.84/bin/tomcat-juli.jar
Using CATALINA_OPTS:
[root@localhost classes]# /usr/local/tomcat/apache-tomcat-9.0.84/bin/startup.sh
Using CATALINA_BASE:   /usr/local/tomcat/apache-tomcat-9.0.84
Using CATALINA_HOME:   /usr/local/tomcat/apache-tomcat-9.0.84
Using CATALINA_TMPDIR: /usr/local/tomcat/apache-tomcat-9.0.84/temp
Using JRE_HOME:        /usr
Using CLASSPATH:        /usr/local/tomcat/apache-tomcat-9.0.84/bin/bootstrap.jar:/usr/local/tomcat/a
pache-tomcat-9.0.84/bin/tomcat-juli.jar
Using CATALINA_OPTS:
Tomcat started.
```

### (3) 开放 8080 端口

开放 tomcat 的默认端口 8080:

```
[root@localhost classes]# firewall-cmd --permanent --add-port=8080/tcp
success
[root@localhost classes]# firewall-cmd --list-ports

[root@localhost classes]# firewall-cmd --reload
success
[root@localhost classes]# firewall-cmd --list-ports
8080/tcp
```

5. 测试

使用浏览器测试访问项目：**192.168.172.141:8080/tmall**



注册用户并登录：

192.168.172.141:8080/tmall/register

填写账号信息

设置会员名

用户名：

设置登录密码 登录时验证，保护账号信息

登录密码：

确认密码：

填写基本信息

昵称：

性别：☒男 ☐女

出生日期：

居住地址：

在数据库的 **user** 表中查看新增的记录：

user @tmall (mysql-192.168.172.140) - 表						
文件 编辑 查看 窗口 帮助						
导入向导 导出向导 筛选向导 网格查看 表单查看 备注 十六进制 图像 升序排序						
user_id	user_name	user_nickname	user_password	user_realname	user_gender	user_birthday
1	a120	MRJIANG	123456	aaaa		0 2013-12-01
3	MRJIANG	如有巧合	123456	流年		0 2018-05-11
5	测试	测试	abcd1234	(Null)		0 2024-01-02



## （四）配置 3 号机-Tomcat-Web 应用服务器

### 1. 克隆

用已经配置好的 4 号机克隆出 3 号机，用作另一台 Tomcat-Web 应用服务器：

### 2. 配置静态 IP 地址

更改它的 IP 地址：

```
1 TYPE="Ethernet"
2 PROXY_METHOD="none"
3 BROWSER_ONLY="no"
4 BOOTPROTO="static"
5 DEFROUTE="yes"
6 IPV4_FAILURE_FATAL="no"
7 IPV6INIT="yes"
8 IPV6_AUTOCONF="yes"
9 IPV6_DEFROUTE="yes"
10 IPV6_FAILURE_FATAL="no"
11 IPV6_ADDR_GEN_MODE="stable-privacy"
12 NAME="ens33"
13 UUID="36deebff-efc0-4686-b170-1d067898d6ab"
14 DEVICE="ens33"
15 ONBOOT="yes"
16 IPADDR=192.168.172.142
17 NETMASK=255.255.255.0
18 GATEWAY=192.168.172.2
19 DNS1=114.114.114.114
20 DNS2=8.8.8.8
```

保存并退出后重启网卡：`systemctl restart network`

### 3. 测试

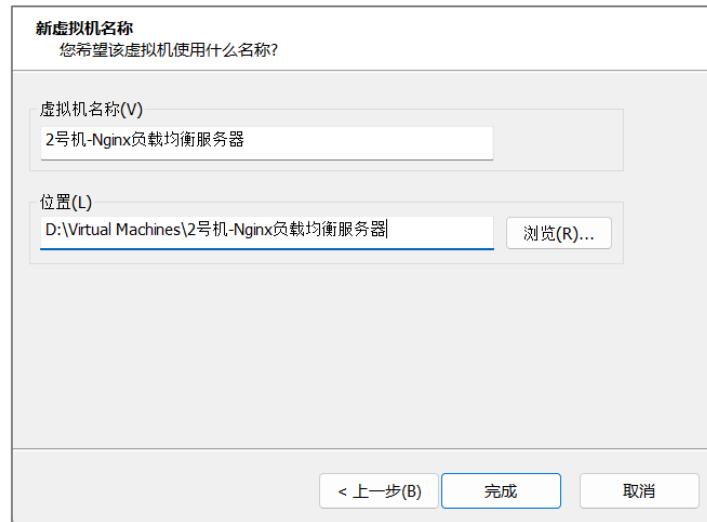
启动 tomcat 后，使用浏览器测试访问项目：**192.168.172.142:8080/tmall**



## （五）配置 2 号机-Nginx 负载均衡服务器

### 1. 克隆

使用母机克隆出 2 号机，用作 Nginx 负载均衡服务器：



### 2. 配置静态 IP 地址

在此路径下更改网卡文件：[/etc/sysconfig/network-scripts/ifcfg-ens33](#)

```
1 TYPE="Ethernet"
2 PROXY_METHOD="none"
3 BROWSER_ONLY="no"
4 BOOTPROTO="static"
5 DEFROUTE="yes"
6 IPV4_FAILURE_FATAL="no"
7 IPV6INIT="yes"
8 IPV6_AUTOCONF="yes"
9 IPV6_DEFROUTE="yes"
10 IPV6_FAILURE_FATAL="no"
11 IPV6_ADDR_GEN_MODE="stable-privacy"
12 NAME="ens33"
13 UUID="36deebff-efc0-4686-b170-1d067898d6ab"
14 DEVICE="ens33"
15 ONBOOT="yes"
16 IPADDR=192.168.172.143
17 NETMASK=255.255.255.0
18 GATEWAY=192.168.172.2
19 DNS1=114.114.114.114
20 DNS2=8.8.8.8
```

保存并退出后重启网卡：`systemctl restart network`

### 3. 安装 nginx

#### (1) 下载

使用 `wget` 命令从官网下载稳定版 nginx:

`wget https://nginx.p2hp.com/download/nginx-1.24.0.tar.gz`

```
[root@localhost ~]# wget https://nginx.p2hp.com/download/nginx-1.24.0.tar.gz
--2024-01-02 19:47:28-- https://nginx.p2hp.com/download/nginx-1.24.0.tar.gz
正在解析主机 nginx.p2hp.com (nginx.p2hp.com) ... 39.104.110.223
正在连接 nginx.p2hp.com (nginx.p2hp.com)|39.104.110.223|:443 ... 已连接。
错误：无法验证 nginx.p2hp.com 的由 “/C=US/O=Let's Encrypt/CN=R3” 颁发的证书：
    颁发的证书已经过期。
要以不安全的方式连接至 nginx.p2hp.com，使用“--no-check-certificate”。
[root@localhost ~]# wget https://nginx.p2hp.com/download/nginx-1.24.0.tar.gz --no-check-certificate
--2024-01-02 19:47:44-- https://nginx.p2hp.com/download/nginx-1.24.0.tar.gz
正在解析主机 nginx.p2hp.com (nginx.p2hp.com) ... 39.104.110.223
正在连接 nginx.p2hp.com (nginx.p2hp.com)|39.104.110.223|:443 ... 已连接。
警告：无法验证 nginx.p2hp.com 的由 “/C=US/O=Let's Encrypt/CN=R3” 颁发的证书：
    颁发的证书已经过期。
已发出 HTTP 请求，正在等待回应... 200 OK
长度：1112471 (1.1M) [application/octet-stream]
正在保存至：“nginx-1.24.0.tar.gz”

100%[====>] 1,112,471  1.67MB/s 用时 0.6s

2024-01-02 19:47:46 (1.67 MB/s) - 已保存 “nginx-1.24.0.tar.gz” [1112471/1112471])
[root@localhost ~]# ls
anaconda-ks.cfg nginx-1.24.0.tar.gz
```

#### (2) 安装依赖

安装编译所需依赖:

`yum install -y gcc make pcre-devel openssl-devel zlib-devel`

这里安装了 GCC（编译器）、PCRE 库的开发文件、OpenSSL 库的开发文件和 `zlib` 压缩库的开发文件，这些都是编译 Nginx 时需要的依赖。

```
已安装:
  openssl-devel.x86_64 1:1.0.2k-26.el7_9

作为依赖被安装:
  keyutils-libs-devel.x86_64 0:1.5.8-3.el7          krb5-devel.x86_64 0:1.15.1-55.el7_9
  libcom_err-devel.x86_64 0:1.42.9-19.el7          libkadm5.x86_64 0:1.15.1-55.el7_9
  libselinux-devel.x86_64 0:2.5-15.el7             libsepol-devel.x86_64 0:2.5-10.el7
  libverto-devel.x86_64 0:0.2.5-4.el7

更新完毕:
  openssl.x86_64 1:1.0.2k-26.el7_9

作为依赖被升级:
  e2fsprogs.x86_64 0:1.42.9-19.el7          e2fsprogs-libs.x86_64 0:1.42.9-19.el7
  krb5-libs.x86_64 0:1.15.1-55.el7_9        libcom_err.x86_64 0:1.42.9-19.el7
  libss.x86_64 0:1.42.9-19.el7             openssl-libs.x86_64 1:1.0.2k-26.el7_9

完毕！
```

### (3) 解压缩

解压缩: `tar -xvf nginx-1.24.0.tar.gz`

进入解压后的目录, 依次执行以下命令。

- 检查环境: `./configure`
- 编译 Nginx: `make`
- 安装: `make install`

检查 nginx 的安装位置, 并将压缩包和安装包删除。

```
[root@localhost ~]# ls
anaconda-ks.cfg  nginx-1.24.0  nginx-1.24.0.tar.gz
[root@localhost ~]# whereis nginx
nginx: /usr/local/nginx
[root@localhost ~]# rm -f nginx*
rm: 无法删除"nginx-1.24.0": 是一个目录
[root@localhost ~]# rm -rf nginx*
[root@localhost ~]# ls
anaconda-ks.cfg
```

## 4. 配置反向代理

进入 nginx 的 conf 目录: `/usr/local/nginx/conf`

使用 `vim nginx.conf` 编辑 nginx 配置文件。

定义一个集群, 分别是 3 号和 4 号应用服务器。设置反向代理模式。

```
34
35     upstream cluster {
36         server 192.168.172.141:8080;
37         server 192.168.172.142:8080;
38     }
39
40     server {
41         listen      80;
42         server_name localhost;
43
44         #charset koi8-r;
45
46         #access_log logs/host.access.log main;
47
48         location / {
49             proxy_pass http://cluster;
50         }
51
```

运行 nginx 目录下的 sbin 目录中的 nginx，启动 nginx。

```
[root@localhost sbin]# ./nginx
[root@localhost sbin]# pidof nginx
14535 14534
```

## 5. 开放 80 端口

开放 nginx 监听的 80 端口：

```
[root@localhost sbin]# firewall-cmd --permanent --add-port=80/tcp
success
[root@localhost sbin]# firewall-cmd --reload
success
[root@localhost sbin]# firewall-cmd --list-ports
80/tcp
```

## 6. 测试

### (1) 故障转移功能

在配置了 Nginx 作为反向代理服务器集群的情况下，可以实现后端服务器的健康检查和负载均衡。当集群中的一台或几台机器发生故障时，Nginx 能够通过健康检查、负载均衡策略和故障转移机制确保服务的连续性。

因此，即使部分后端服务器宕机，由于 Nginx 能动态调整转发策略，客户端仍可以不间断地访问目标服务，从而提高了整个系统的可用性和可靠性。

在浏览器中使用 2 号机的 IP 地址访问 web 项目：

**192.168.172.143/tmall**



此时，关闭 3 号机（或 4 号机）的 tomcat 服务模拟应用服务器故障。

```
[root@localhost ~]# /usr/local/tomcat/apache-tomcat-9.0.84/bin/shutdown.sh
Using CATALINA_BASE:   /usr/local/tomcat/apache-tomcat-9.0.84
Using CATALINA_HOME:   /usr/local/tomcat/apache-tomcat-9.0.84
Using CATALINA_TMPDIR: /usr/local/tomcat/apache-tomcat-9.0.84/temp
Using JRE_HOME:        /usr
Using CLASSPATH:        /usr/local/tomcat/apache-tomcat-9.0.84/bin/bootstrap.jar:/usr/local/tomcat
```

而后重新访问：**192.168.172.143/tmall**

发现仍然能够正常访问。



## (2) 负载均衡算法

服务器负载均衡算法是用于在多台服务器之间分配网络流量和请求的技术，以实现系统资源的高效利用、提升服务可用性和响应速度。下面测试轮询、加权轮询和 IP 哈希三种负载均衡算法。

为了便于区分和观察服务器的请求，即 2 号机请求的是 3 号机还是 4 号机。分别在 3 号机和 4 号机 tomcat 的 webapps/ROOT 下新建 test.txt 文件，内容分别是 ‘This is page 1.’ 和 ‘This is page 2.’。

- 轮询算法：轮询是最简单的负载均衡策略，Nginx 默认采用的就是这一策略。每个请求按顺序逐一分配到后台服务器列表中的下一个服务器。实现效果如下：

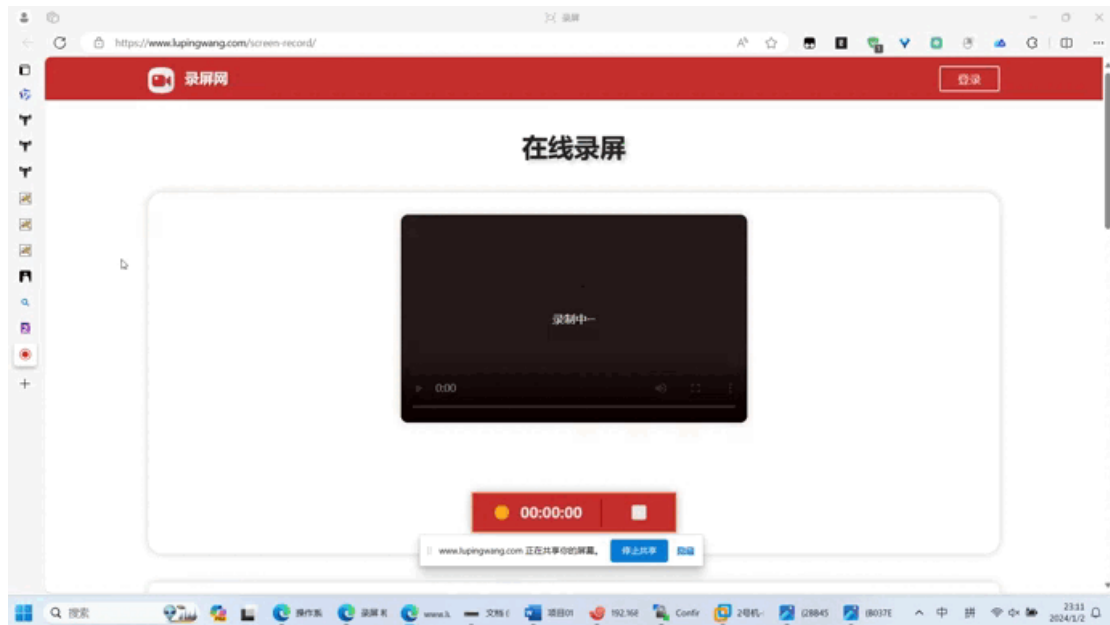


- 加权轮询算法：在轮询的基础上，给每个服务器分配一个权重值，根据权重比例分配请求。权重越高的服务器接收到的请求比例越高。修改 nginx 的配置文件，设置权重。



```
35     upstream cluster {
36         server 192.168.172.141:8080 weight=1;
37         server 192.168.172.142:8080 weight=2;
38     }
```

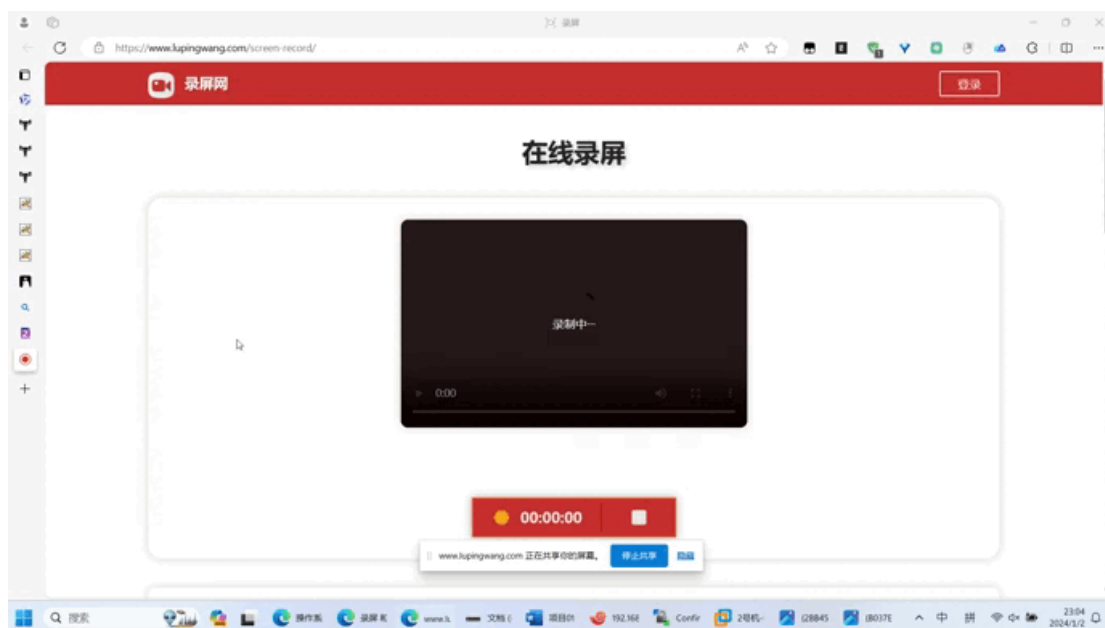
保存并退出后重启 nginx。实现效果如下：



- IP 哈希算法：使用 `ip_hash` 算法，会根据客户端 IP 地址计算哈希值，然后将来自同一客户端 IP 地址的请求定向到同一台后端服务器上，这样可以实现会话持久化。修改 nginx 的配置文件如下。

```
35     upstream cluster {
36         ip_hash;
37         server 192.168.172.141:8080;
38         server 192.168.172.142:8080;
39     }
```

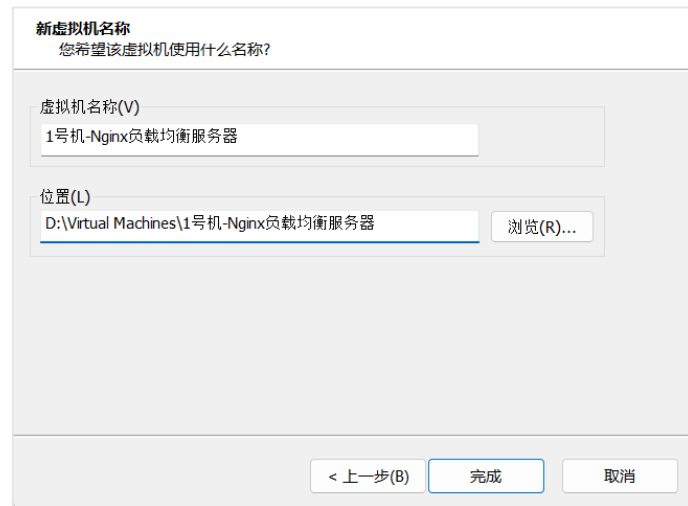
保存并退出后重启 nginx。实现效果如下：



## （六）配置 1 号机-Nginx 负载均衡服务器

### 1. 克隆

用当前配置好的 2 号机克隆出一台 1 号机，同样用作 nginx 负载均衡服务器：



新虚拟机名称  
您希望该虚拟机使用什么名称？

虚拟机名称(V)  
1号机-Nginx负载均衡服务器

位置(L)  
D:\Virtual Machines\1号机-Nginx负载均衡服务器 浏览(R)...

< 上一步(B) 完成 取消

### 2. 配置静态 IP 地址

在此路径下更改网卡文件：

[/etc/sysconfig/network-scripts/ifcfg-ens33](#)

```
1 TYPE="Ethernet"
2 PROXY_METHOD="none"
3 BROWSER_ONLY="no"
4 BOOTPROTO="static"
5 DEFROUTE="yes"
6 IPV4_FAILURE_FATAL="no"
7 IPV6INIT="yes"
8 IPV6_AUTOCONF="yes"
9 IPV6_DEFROUTE="yes"
10 IPV6_FAILURE_FATAL="no"
11 IPV6_ADDR_GEN_MODE="stable-privacy"
12 NAME="ens33"
13 UUID="36deebff-efc0-4686-b170-1d067898d6ab"
14 DEVICE="ens33"
15 ONBOOT="yes"
16 IPADDR=192.168.172.144
17 NETMASK=255.255.255.0
18 GATEWAY=192.168.172.2
19 DNS1=114.114.114.114
20 DNS2=8.8.8.8
```

保存并退出后重启网卡：`systemctl restart network`

启动 nginx 后，同样进行测试，效果与 2 号机相同。

## (七) keepalived 双机热备

为了避免单点故障对服务的影响，确保服务的高可用性。分别在 1 号机和 2 号机两台负载均衡服务器上部署 Keepalived 实现双机热备。

### 1. 安装 keepalived

分别在 1 号机和 2 号机上安装 keepalived。

```
[root@localhost ~]# yum install -y keepalived
已加载插件: fastestmirror
Loading mirror speeds from cached hostfile
* base: mirrors.bupt.edu.cn
* extras: mirrors.bupt.edu.cn
* updates: mirrors.bupt.edu.cn
```

### 2. 配置主从关系和虚拟 IP

分别进入 keepalived 的配置文件，配置主从关系和虚拟 IP。

设置 1 号机为主路由如下：

```
14 #vrrp_strict
15 vrrp_garp_interval 0
16 vrrp_gna_interval 0
17 }
18
19 vrrp_script check_nginx {
20     script "/etc/keepalived/check_nginx.sh"
21     interval 1
22     weight -25
23 }
24
25 vrrp_instance VI_1 {
26     state MASTER ← 主路由
27     interface ens33 ← 网卡
28     virtual_router_id 51
29     priority 100 ← 权限
30     advert_int 1
31     authentication {
32         auth_type PASS
33         auth_pass 1111
34     }
35     virtual_ipaddress {
36         192.168.172.88 ← 虚拟IP
37     }
38     track_script {
39         check_nginx
40     }
41 }
```

设置 2 号机为备用路由如下：

```
14 #vrrp_strict
15 vrrp_garp_interval 0
16 vrrp_gna_interval 0
17 }
18
19 vrrp_script check_nginx {
20     script "/etc/keepalived/check_nginx.sh"
21     interval 1
22     weight -25
23 }
24
25 vrrp_instance VI_1 {
26     state BACKUP ← 备用路由
27     interface ens33 ← 网卡
28     virtual_router_id 51
29     priority 80 ← 优先级
30     advert_int 1
31     authentication {
32         auth_type PASS
33         auth_pass 1111
34     }
35     virtual_ipaddress {
36         192.168.172.88 ← 虚拟IP
37     }
38     track_script {
39         check_nginx
40     }
41 }
```

### 3. 编写 nginx 心跳检测脚本

在 2 台机器上分别创建编写 nginx 心跳检测脚本 **check\_nginx.sh**, 用于检查 Nginx 服务是否正在运行。

```
#!/bin/bash
if /usr/sbin/pidof nginx &>/dev/null ;then
    exit 0
else
    exit 1
fi
```

开启该脚本的权限，让 keepalived 能够执行。

```
[root@localhost keepalived]# chmod +x check_nginx.sh
[root@localhost keepalived]# ll
总用量 8
-rwxr-xr-x. 1 root root 89 1月 3 22:04 check_nginx.sh
-rw-r--r--. 1 root root 3700 1月 3 01:29 keepalived.conf
[root@localhost keepalived]#
```

## 4. 开启 keealived

为了避免出现 keepalived 脑裂问题，分别关闭两台机器的防火墙。使用 `systemctl start keepalived` 命令开启 keepalived 服务。

```
[root@localhost ~]# systemctl stop firewalld
[root@localhost ~]# systemctl status firewalld      关闭防火墙
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset: enabled)
   Active: inactive (dead) since 三 2024-01-03 22:27:58 CST; 2s ago
     Docs: man:firewalld(1)
   Process: 685 ExecStart=/usr/sbin/firewalld --nofork --nopid $FIREWALLD_ARGS (code=exited, status=0/SUCCESS)
   Main PID: 685 (code=exited, status=0/SUCCESS)

1月 03 19:01:49 localhost.localdomain systemd[1]: Starting firewalld - dynamic firewall daemon ...
1月 03 19:01:50 localhost.localdomain systemd[1]: Started firewalld - dynamic firewall daemon.
1月 03 22:27:57 localhost.localdomain systemd[1]: Stopping firewalld - dynamic firewall daemon ...
1月 03 22:27:58 localhost.localdomain systemd[1]: Stopped firewalld - dynamic firewall daemon.
[root@localhost ~]# systemctl start keepalived      开启keepalived
[root@localhost ~]#
```

## 5. 测试

在分别开启 keepalived 后使用 `ip addr` 命令查看两台机器网络接口的 IP 地址。

1 号机（主路由）如下：

```
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
   link/ether 00:0c:29:af:d6:f6 brd ff:ff:ff:ff:ff:ff
   inet 192.168.172.144/24 brd 192.168.172.255 scope global noprefixroute ens33
      valid_lft forever preferred_lft forever
   inet 192.168.172.88/32 scope global ens33
      valid_lft forever preferred_lft forever
   inet6 fe80::f854:36d9:b0b8:f0a3/64 scope link tentative noprefixroute dadfailed
      valid_lft forever preferred_lft forever
   inet6 fe80::cece:2bb1:3fd6:2ffd/64 scope link tentative noprefixroute dadfailed
      valid_lft forever preferred_lft forever
   inet6 fe80::562f:dd75:d79e:4ef3/64 scope link tentative noprefixroute dadfailed
      valid_lft forever preferred_lft forever
```

2 号机（备用路由）如下：

```
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
   link/ether 00:0c:29:7e:a0:2e brd ff:ff:ff:ff:ff:ff
   inet 192.168.172.143/24 brd 192.168.172.255 scope global noprefixroute ens33
      valid_lft forever preferred_lft forever
   inet6 fe80::f854:36d9:b0b8:f0a3/64 scope link tentative noprefixroute dadfailed
      valid_lft forever preferred_lft forever
   inet6 fe80::cece:2bb1:3fd6:2ffd/64 scope link tentative noprefixroute dadfailed
      valid_lft forever preferred_lft forever
   inet6 fe80::562f:dd75:d79e:4ef3/64 scope link tentative noprefixroute dadfailed
      valid_lft forever preferred_lft forever
```

此时，可以通过使用虚拟 ip 访问 web 应用。



### (1) 工作原理

keepalive 实现双机热备的工作原理如下：

- 在两台服务器上分别部署 **Keepalived**，并配置为同一个 **VRRP** 实例的 **Master** 和 **Backup** 角色。**Master** 节点正常运行时，会对外提供服务，例如作为 **Web** 服务器或数据库服务器。
- **Master** 节点会定期发送心跳信息给 **Backup** 节点以及网络中的其他设备，表明自己状态正常。
- 若 **Master** 节点出现故障（如进程异常退出、网络中断等），**Backup** 节点在一定时间内收不到心跳信息，就会认为 **Master** 节点失效，此时 **Backup** 节点将自动转换为 **Master** 角色，接管对外的服务。
- 当原 **Master** 节点恢复正常后，它会再次尝试成为 **Master**，但在此过程中，由于 **Backup** 节点当前是活动的 **Master**，因此原 **Master** 会自动转为 **Backup** 角色，等待下一次切换机会。

### (2) 模拟主路由故障

关闭 1 号机主路由的 **nginx** 服务，模拟主路由故障。

```
[root@localhost keepalived]# pidof nginx
1336 1335
[root@localhost keepalived]# kill -9 1335 1336
[root@localhost keepalived]#
```

发现此时仍然能够使用虚拟 ip 访问 web 项目。



查看 1 号机的 IP 地址：

```
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:af:d6:f6 brd ff:ff:ff:ff:ff:ff
    inet 192.168.172.144/24 brd 192.168.172.255 scope global noprefixroute ens33
        valid_lft forever preferred_lft forever
    inet6 fe80::f854:36d9:b0b8:f0a3/64 scope link tentative noprefixroute dadfailed
        valid_lft forever preferred_lft forever
    inet6 fe80::cece:2bb1:3fd6:2ffd/64 scope link tentative noprefixroute dadfailed
        valid_lft forever preferred_lft forever
    inet6 fe80::562f:dd75:d79e:4ef3/64 scope link tentative noprefixroute dadfailed
        valid_lft forever preferred_lft forever
```

查看 2 号机的 IP 地址：

```
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:7e:a0:2e brd ff:ff:ff:ff:ff:ff
    inet 192.168.172.143/24 brd 192.168.172.255 scope global noprefixroute ens33
        valid_lft forever preferred_lft forever
    inet 192.168.172.88/32 scope global ens33 虚拟IP
        valid_lft forever preferred_lft forever
    inet6 fe80::f854:36d9:b0b8:f0a3/64 scope link tentative noprefixroute dadfailed
        valid_lft forever preferred_lft forever
    inet6 fe80::cece:2bb1:3fd6:2ffd/64 scope link tentative noprefixroute dadfailed
        valid_lft forever preferred_lft forever
    inet6 fe80::562f:dd75:d79e:4ef3/64 scope link tentative noprefixroute dadfailed
        valid_lft forever preferred_lft forever
```

通过虚拟 IP 的归属发现进行了主备转换。

开启 1 号机主路由的 nginx 服务。查看 1 号机的 IP 地址：

```
[root@localhost keepalived]# /usr/local/nginx/sbin/nginx
[root@localhost keepalived]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:af:d6:f6 brd ff:ff:ff:ff:ff:ff
    inet 192.168.172.144/24 brd 192.168.172.255 scope global noprefixroute ens33
        valid_lft forever preferred_lft forever
    inet 192.168.172.88/32 scope global ens33
        valid_lft forever preferred_lft forever
    inet6 fe80::f854:36d9:b0b8:f0a3/64 scope link tentative noprefixroute dadfailed
        valid_lft forever preferred_lft forever
    inet6 fe80::cece:2bb1:3fd6:2ffd/64 scope link tentative noprefixroute dadfailed
        valid_lft forever preferred_lft forever
    inet6 fe80::562f:dd75:d79e:4ef3/64 scope link tentative noprefixroute dadfailed
        valid_lft forever preferred_lft forever
```



查看 2 号机的 IP 地址：

```
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:7e:a0:2e brd ff:ff:ff:ff:ff:ff
    inet 192.168.172.143/24 brd 192.168.172.255 scope global noprefixroute ens33
        valid_lft forever preferred_lft forever
    inet6 fe80::f854:36d9:b0b8:f0a3/64 scope link tentative noprefixroute dadfailed
        valid_lft forever preferred_lft forever
    inet6 fe80::cece:2bb1:3fd6:2ffd/64 scope link tentative noprefixroute dadfailed
        valid_lft forever preferred_lft forever
    inet6 fe80::562f:dd75:d79e:4ef3/64 scope link tentative noprefixroute dadfailed
        valid_lft forever preferred_lft forever
```

发现此时 1 号机恢复主路由身份。

再通过模拟主路由机器故障和 `keepalived` 故障同样可以出现主备转换的情况。

## 五、问题与解决

### 1. 网络设置问题

在使用 yum 下载安装 mysql 的依赖时，出现下载失败的问题：

```
Error downloading packages:
net-tools-2.0-0.25.20131004git.el7.x86_64: [Errno 256] No more mirrors to try.
libarchive-3.1.2-14.el7_7.x86_64: [Errno 256] No more mirrors to try.
gcc-c++-4.8.5-44.el7.x86_64: [Errno 256] No more mirrors to try.
gcc-4.8.5-44.el7.x86_64: [Errno 256] No more mirrors to try.
libgomp-4.8.5-44.el7.x86_64: [Errno 256] No more mirrors to try.
glibc-common-2.17-326.el7_9.x86_64: [Errno 256] No more mirrors to try.
libgcc-4.8.5-44.el7.x86_64: [Errno 256] No more mirrors to try.
libaio-devel-0.3.109-13.el7.x86_64: [Errno 256] No more mirrors to try.
cpp-4.8.5-44.el7.x86_64: [Errno 256] No more mirrors to try.
kernel-headers-3.10.0-1160.105.1.el7.x86_64: [Errno 256] No more mirrors to try.
glibc-devel-2.17-326.el7_9.x86_64: [Errno 256] No more mirrors to try.
libmpc-1.0.1-3.el7.x86_64: [Errno 256] No more mirrors to try.
libstdc++-4.8.5-44.el7.x86_64: [Errno 256] No more mirrors to try.
bison-devel-3.0.4-2.el7.x86_64: [Errno 256] No more mirrors to try.
cmake-2.8.12.2-2.el7.x86_64: [Errno 256] No more mirrors to try.
mpfr-3.1.1-4.el7.x86_64: [Errno 256] No more mirrors to try.
glibc-headers-2.17-326.el7_9.x86_64: [Errno 256] No more mirrors to try.
glibc-2.17-326.el7_9.x86_64: [Errno 256] No more mirrors to try.
libstdc++-devel-4.8.5-44.el7.x86_64: [Errno 256] No more mirrors to try.
ncurses-devel-5.9-14.20130511.el7_4.x86_64: [Errno 256] No more mirrors to try.
```

这个错误提示 No more mirrors to try 表示在尝试从可用的镜像站点下载软件包时，所有列出的镜像都无法完成下载任务。因此，首先考虑是否是网络问题，检查网络连接，确保虚拟机可以正常访问互联网：尝试 ping 一个公共的互联网域名或 IP 地址，例如：ping baidu.com

```
[root@localhost ~]# ping baidu.com
ping: baidu.com: 未知的名称或服务
```

发现无法正常访问互联网，所以初步确定存在网络问题。考虑到在此期间更改过机器的网卡文件，因此对网卡文件进行检查。

```
TYPE="Ethernet"
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="static"
DEFROUTE="yes"
IPV4_FAILURE_FATAL="no"
IPV6INIT="yes"
IPV6_AUTOCONF="yes"
IPV6_DEFROUTE="yes"
IPV6_FAILURE_FATAL="no"
IPV6_ADDR_GEN_MODE="stable-privacy"
NAME="ens33"
UUID="36deebff-efc0-4686-b170-1d067898d6ab"
DEVICE="ens33"
ONBOOT="yes"
IPADDR=192.168.172.140
NETMASK=255.255.255.0
GATWAY=192.168.172.2
DNS1=114.114.114.114
DNS2=8.8.8.8
```

检查发现设置网关时关键字设置错误，将 GATWAY 更改为正确的 GATEWAY。

保存并退出，重启网卡。再次 ping baidu.com 测试网络。

```
[root@localhost ~]# ping baidu.com
PING baidu.com (110.242.68.66) 56(84) bytes of data.
64 bytes from 110.242.68.66 (110.242.68.66): icmp_seq=1 ttl=128 time=270 ms
64 bytes from 110.242.68.66 (110.242.68.66): icmp_seq=2 ttl=128 time=75.7 ms
64 bytes from 110.242.68.66 (110.242.68.66): icmp_seq=3 ttl=128 time=50.6 ms
64 bytes from 110.242.68.66 (110.242.68.66): icmp_seq=4 ttl=128 time=181 ms
^C
--- baidu.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3007ms
rtt min/avg/max/mdev = 50.666/144.594/270.967/87.836 ms
```

此时网络连接正常。

再次使用 yum 安装 mysql 的依赖。

```
已安装:
bison-devel.x86_64 0:3.0.4-2.el7          cmake.x86_64 0:2.8.12.2-2.el7
gcc-c++.x86_64 0:4.8.5-44.el7             libaio-devel.x86_64 0:0.3.109
ncurses-devel.x86_64 0:5.9-14.20130511.el7_4 net-tools.x86_64 0:2.0-0.25.1

作为依赖被安装:
cpp.x86_64 0:4.8.5-44.el7                 gcc.x86_64 0:4.8.5-44.el7
glibc-devel.x86_64 0:2.17-326.el7_9       glibc-headers.x86_64 0:2.17-326.el7_9
kernel-headers.x86_64 0:3.10.0-1160.105.1.el7 libarchive.x86_64 0:3.1.1-2.el7
libmpc.x86_64 0:1.0.1-3.el7               libstdc++.devel.x86_64 0:4.8.5-44.el7
mpfr.x86_64 0:3.1.1-4.el7

作为依赖被升级:
glibc.x86_64 0:2.17-326.el7_9  glibc-common.x86_64 0:2.17-326.el7_9  libgcc.x86_64 0:4.8.5-44.el7
libgomp.x86_64 0:4.8.5-44.el7  libstdc++.x86_64 0:4.8.5-44.el7

完毕！
```

成功安装，问题得以解决。

## 2. keepalived 脑裂问题

在测试双机热备的过程中，1 号机和 2 号机两台机器出现都认为自己是主路由，同时绑定 VIP 的情况。

```

link/ether 00:0c:29:7e:a0:2e brd ff:ff:ff:ff:ff:ff
inet 192.168.172.143/24 brd 192.168.172.255 scope global noprefixroute ens33
    valid_lft forever preferred_lft forever
inet 192.168.172.88/32 scope global ens33
    valid_lft forever preferred_lft forever
inet6 fe80::f854:36d9:b0b8:f0a3/64 scope link tentative noprefixroute dadfailed
    valid_lft forever preferred_lft forever
inet6 fe80::cece:2bb1:3fd6:2ffd/64 scope link tentative noprefixroute dadfailed
    valid_lft forever preferred_lft forever
inet6 fe80::562f:dd75:d79e:4ef3/64 scope link tentative noprefixroute dadfailed
    valid_lft forever preferred_lft forever

inet 192.168.172.144/24 brd 192.168.172.255 scope global noprefixroute ens33
    valid_lft forever preferred_lft forever
inet 192.168.172.88/32 scope global ens33
    valid_lft forever preferred_lft forever
inet6 fe80::f854:36d9:b0b8:f0a3/64 scope link tentative noprefixroute dadfailed
    valid_lft forever preferred_lft forever
inet6 fe80::cece:2bb1:3fd6:2ffd/64 scope link tentative noprefixroute dadfailed
    valid_lft forever preferred_lft forever
inet6 fe80::562f:dd75:d79e:4ef3/64 scope link tentative noprefixroute dadfailed
    valid_lft forever preferred_lft forever

```

通过查询资料可知这是脑裂（Split-Brain）问题。脑裂是指集群的一部分节点认为自己是活跃状态并继续提供服务，而另一部分节点也认为自己是活跃状态，从而导致集群内出现两个或多个“主节点”，这可能会造成数据不一致和资源冲突等问题。

在 Keepalived 场景下，脑裂可能发生在以下情况：

1. 网络分区：集群中的网络通信出现问题，使得一部分节点无法与其他节点通信，这些节点可能错误地认为其他节点已经失效，并尝试接管 VIP。
2. 配置不当：如果 Keepalived 的仲裁机制设置不当，比如优先级、超时时间等参数不合理，可能导致在正常情况下发生不必要的主备切换。

更改优先级、超时时间等参数后发现无法解决问题。最后通过关闭防火墙的方式，使两台机器能够正常通信，进行主备切换。

```

inet 192.168.172.143/24 brd 192.168.172.255 scope global noprefixroute ens33
    valid_lft forever preferred_lft forever
inet6 fe80::f854:36d9:b0b8:f0a3/64 scope link tentative noprefixroute dadfailed
    valid_lft forever preferred_lft forever
inet6 fe80::cece:2bb1:3fd6:2ffd/64 scope link tentative noprefixroute dadfailed
    valid_lft forever preferred_lft forever
inet6 fe80::562f:dd75:d79e:4ef3/64 scope link tentative noprefixroute dadfailed
    valid_lft forever preferred_lft forever

inet 192.168.172.144/24 brd 192.168.172.255 scope global noprefixroute ens33
    valid_lft forever preferred_lft forever
inet 192.168.172.88/32 scope global ens33
    valid_lft forever preferred_lft forever
inet6 fe80::f854:36d9:b0b8:f0a3/64 scope link tentative noprefixroute dadfailed
    valid_lft forever preferred_lft forever
inet6 fe80::cece:2bb1:3fd6:2ffd/64 scope link tentative noprefixroute dadfailed
    valid_lft forever preferred_lft forever
inet6 fe80::562f:dd75:d79e:4ef3/64 scope link tentative noprefixroute dadfailed
    valid_lft forever preferred_lft forever

```

## 六、项目收获

1. 熟悉了 Linux 系统的一些常用基本命令，学会了如何配置不同应用场景的虚拟机服务器。
2. 学习了 Tomcat 与 MySQL 结合为基于 Java 的 Web 应用程序实现动态内容生成与持久化数据存储。
3. 理解了单机架构和集群架构的区别。明白了高可用、高并发和高性能对于服务器的重要性。
4. 理解了负载均衡的作用以及应对不同情况的负载均衡算法。
5. 理解了基于 keepalived 的双机热备的原理与作用。