基于 Linux 的 Web 服务器设计与实现*

黄名俊 林宁

南宁学院信息工程学院，南宁，530200

摘 要 通过使用虚拟机技术，在 Linux 系统平台下进行 Web 服务架构的设计，使用负载均衡技术，解决服务器无法承受大量用户频繁的访问，使用 Keepalived 技术实现服务器的高可用，解决服务器单点故障风险，提高服务器架构的稳定性，使用 Zabbix 监控软件对服务器的运行状况监控，保证服务器 24 小时不间断的正常运行。全网数据的备份以及数据库的主从复制，保证数据的安全性。性能测试结果表明，搭建完成的服务器架构具高可用、数据安全性高、架构稳定、支持高并发访问，大幅度提升服务器架构的综合性能。

关键字 服务器，高并发，负载均衡，单点故障

Design and Implementation of Web Server Based on Linux*

Huang Mingjun Lin Ning

School of Information Engineering Nanning University
Nanning 530200, China;
961852656@qq.com

Abstract—Through the use of virtual machine technology, the web service architecture is designed under the Linux system platform, the load balancing technology is used to solve the problem that the server can not bear the frequent access of a large number of users, the keepalived technology is used to realize the high availability of the server, solve the single point of failure risk of the server, and improve the stability of the server architecture, ZABBIX monitoring software is used to monitor the operation status of the server to ensure the normal operation of the server 24 hours a day. The backup of the whole network data and the master-slave replication of the database ensure the security of the data. The performance test results show that the built server architecture has high availability, high data security, stable architecture, supports high concurrent access, and greatly improves the comprehensive performance of the server architecture.

Keyword—Server, High Concurrency, Load balancing, Single point of failure

1 引 言

Web 服务器的主要任务是响应来自浏览器的请求，将用户请求的页面通过浏览器的方式呈现给用户。随着近年互联网技术的迅猛发展，人们通过互联网可以进行购物、社交等活动，享受了互联网带来的巨大便利。根据相关机构发布的“数值 2020”报告中显示，PC 网上用户约为 43.9 亿人，占全球的 57% ,互联网的使用用户大幅度的增长[1]。对于用户请求数量日益增长的今天，一个安全的、稳定的、支持高并发访问的 Web 服务器架构系统尤为重要。

为了解决用户访问量大，提高服务器系统的性能，降低服务器的负载。人们也开始在不断的去寻求更好的解决方法。传统的做法是采用提升服务器的硬件(如使用计算效率更高的 CUP 和更大的内存来作为服务器的硬件)来提升服务器的响应效率和并发量。这种通过升级服务器的软硬件配置，虽然说在一定程度上确实能够提升服务器的性能，但是该方法不光是花费高昂的金钱代价，同时也无法避免服务器架构的单点故障，架构的稳定性不高。

本文针对传统方式通过升级服务器软硬件配置来提升服务器的性能，探索了基于 Linux 系统平台上进行服务器架构搭建的相关技术，使用至少 2 台以上 Web 的服务器，采用负载均衡机制、Keepalived 技术、

*基金资助：本文得到广西高校中青年教师科研基础能力提升项目(2021KY1805)资助.

通讯作者：林宁，副教授，992959839@qq.com

Zabbix 监控软件、全网数据备份以及数据库的主从复制技术来均衡各个服务器的负载量，保障各服务器日常产生数据的安全性，避免服务器出现单点故障以及对服务器的运行状况进行实时的监控，从多方面提升服务器架构的综合性能使其达到最优的效果。该方案不仅实现了服务器架构具高可用、数据安全性高架构稳定及支持提供高并发访问等优点，同时对中小型企业服务器架构部署也有一定的参考价值。

2 相关技术研究

2.1 Web 服务器

Web 服务器主要的功能是可以处理用户的请求，并把处理的结果返回给用户，也可以存放数据供用户下载。

本文采用 Nginx 作为 Web 服务器，Nginx 具有稳定性好，功能丰富，配置简单等特点。在高并发请求的情况下，Nginx 可以同时支持多达 5 万个并发连接的响应，是 Apache 服务的良好替代品。同时，Nginx 上手简单，即使运行几个月也不需要重启，必要时软件版本可以持续升级,由于 Nginx 的诸多优点，得到了广泛的应用。

2.2 负载均衡技术

负载均衡技术目的是为了降低服务器的负载，扩展带宽。其工作流程主要是：用户发送请求，负载服务器将接收到的请求，转发给 Web 服务器进行处理，在转发过程中，负载均衡服务器会检测后端 Web 服务器的负载情况，将请求分配给负载较低的 Web 服务器进行处理，从而提高整个架构的性能[2]。

2.3 Keepalived 高可用服务

Keepalived 高可用服务可以实现服务器架构高可用，解决服务器单点故障风险。Keepalived 服务采用 Vrrp 协议，对外提供一个虚拟的 IP 地址,谁拿到这个 IP 地址为主服务器，其他的为备份服务器[3]。为了让其他服务器知道主服务器的运行状态正常，在正常工作过程中，会向备份服务器不断的发送心跳消息，表明自己处于正常状态。当主服务器出现故障时，无法发送向其他服务器发消息时,这时认为主服务器出现故障。备份服务器会将对外的虚拟 IP 地址抢占过来，继续工作，保证服务能够正常的运行。当主服务器故障解决了，运行状态正常时，备份服务器将会释放手中的虚拟 IP 地址，恢复到其原始备用角色。

2.4 数据库主从复制

Mysql 数据库来提供动态数据存储访问，为了保证数据的安全性。主从复制，使用另一个数据库将主数据的的所有数据进行备份，称为从数据库，主数据库一般是准实时的业务数据库[4]。

2.5 NFS 网络文件系统

网络文件系统是一种基于 TCP 传输协议的文件共享系统，实现在同一局域网不同主机之间进行文件共享[5]。NFS 服务器提供远程挂载目录， NFS 客户器端可以进行远程的挂载。从本地客户端来看，挂载 NFS 服务器端远端的共享目录，如同 NFS 客户端挂载本地磁盘分区一样。

2.6 Zabbix 监控软件

为了监控服务器的运行状态，一般在网站架构建成后，会进行安装服务器的监控软件，以便管理员可以使用监控工具对服务器的运行状态进行 7*24 小时监控，信息由监控软件自动收集。

Zabbix 监控服务主要是通过 zabbix-agent 软件收集本地的资源后，将本地的资源发送给服务端，服务端可以通过 Web 的显示页面展示给运维人员，以便运维人员能够实时的了解服务的运行状况[6]。

2.7 LNMP 架构

本文采用 LNMP 作为服务器架构，其中 Linux 提供架构平台，Nginx 提供静态文件处理服务，PHP 提供动态文件处理服务,Mysql 提供数据库存储服务。

Nginx 只支持处理用户的静态请求，不支持解析动态请求，当用户发送请求时,如果该请求为动态请求，Nginx 无法处理该请求，于是 Nginx 将用户的请求通过 FastCGI（公共网关接口）接口发送给 PHP 解析器来处理这个请求，处理完成后，数据结果会按原路线返回给客户端[8]。LNMP 原理具体如图 1 所示。

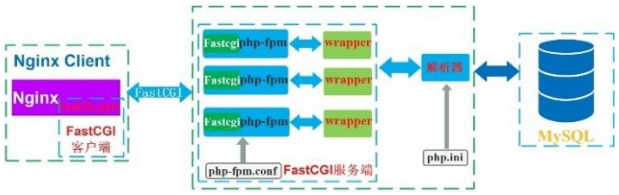

图 1 LNMP 原理具体图

3 服务器架构设计

Web 服务器的总体架构设计如图 2 所示。

服务器架构部署图说明:

(1) 负载均衡服务器可以根据后端的 Web 服务器的负载情况，将用户的请求进转发，减少后端服务器

的压力，缩短响应时间，提高性能。

(2) Web 服务器主要提供 Web 服务，响应用户的请求以及 PHP 解析。

(3) NFS 主要提供文件存储以及文件共享功能，如用户上传的图片；

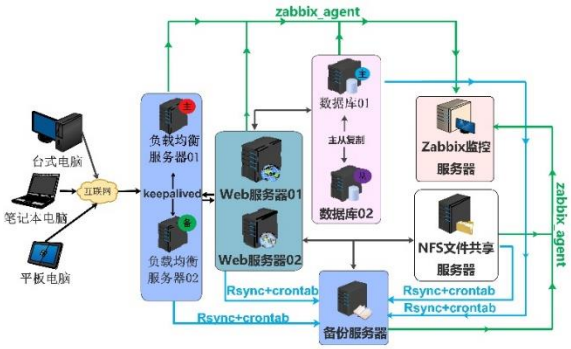

图 2 服务器总体架构设计图

(4) MySQL 数据库用于存储结构化的数据。

(5) 备份服务器主要用于备份各个服务器重要的数据，进行数据冗余，保证数据的安全性，如 Web 服务器的站点目录、开机自启文件、脚本文件、配置文件等。

(6) Zabbix 监控服务器主要通过 zabbix_agent 收集本地资源和应用程序，并将收集到的数据发送给 Zabbix Server，通过 Web 的显示页面展示给运维人员，从而启到对各个服务器进行实时监控。

4 服务器架构实现

按照架构的设计图，先部署内网的服务器，然后再部署外网连接的负载均衡服务器，这样在搭建服务器的初期，对服务器的安全启到一定的保护作用。然后搭建 Rsync 服务器，因为所有服务器的数据都要汇总到备份服务器中，所以最先搭建备份服务器。在搭建 Web 服务器之前，需要挂载 NFS 文件共享服务器和将数据存储到数据库中，所以 NFS 文件共享服务器和数据库服务器在 Web 服务器之前搭建。最后在搭建 Zabbix 监控服务。其安装顺序如下表 1 所示。

4.1 Rsync 备份服务器部署

当 Rsync 客户端想要把本地的文件推送到 Rsync 服务端的 backup 模块对应目录下的时候，首先客户端执行推送命令，明确推送的目录，然后服务器会进行虚拟用户的认证和密码认证，用户和密码认证通过后将数据推送至 backup 模块下，最后完成数据的写入[7]。Rsync 工作原理如图 3 所示。

表 1 安装服务器顺序

安装顺序	服务器名称	服务器说明
1	Rsync 备份服务器	保证数据的安全性
2	NFS 服务器	提供文件共享服务
3	MySQL 服务器	提供数据库的存储
4	Web 服务器	供动网页服务
5	Zabbix 监控服务	监控服务器状态
6	负载均衡服务器	提供负载均衡

Rsync 服务端安装步骤如下：

(1) 下载和编写 Rsync 配置文件。配置文件的执行代码如下:

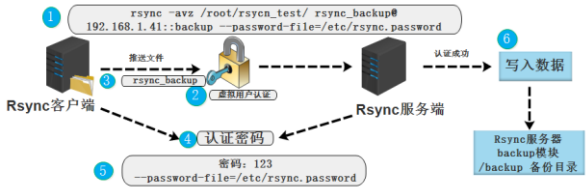

图 3 Rsync 工作原理

```
[root@backup ~]# yum -y install rsync.x86_64
[root@backup ~]# vim /etc/rsyncd.conf
uid = rsync    #指定管理备份目录的用户
gid = rsync    #指定管理备份目录的用户组
port = 873     #定义 Rsync 备份服务的网络端口号
 fake super = yes    #将 Rsync 虚拟用户伪装成为一个超级管理员用户
use chroot = no     #安全相关的配置
max connections = 200    #最大连接数
timeout = 300     #超时时间(单位秒)
pid file = /var/run/rsyncd.pid    #记录进程号码信息
lock file = /var/run/rsync.lock    #锁文件
log file = /var/log/rsyncd.log    #rsync 服务的日志文件，用于排错分析问题
ignore errors              #忽略传输中的简单错误
read only = false          #指定备份目录是可读可写
list = false #使客户端可以查看服务端的模块信息
hosts allow = 172.20.1.0/24    #允许传输备份数据的主机
hosts deny = 0.0.0.0/24 #禁止传输备份数据的主机
auth users = rsync_backup          #指定认证用户
secrets file = /etc/rsync.password    #指定认证用户密码文件
[backup]    #模块信息
comment = "backup dir "    #备份目录信息说明
path = /backup    #模块中配置参数 指定备份目录
```

(2) 创建虚拟用户。具体的命令语句如下:

```
[root@backup ~]# useradd rsync -M -s /sbin/nolog
```

(3) 创建认证用户和密码文件。具体的命令语句如下:

[root@backup01~]# echo “rsync_backup:123456 ” >> /etc/rsync.password

(4) 修改文件权限。具体的命令语句如下：

[root@backup01~]# chmod 600 /etc/rsync.password

(5) 启动服务。具体的命令语句如下：

[root@backup ~]# systemctl start rsyncd.service

4.2 NFS 网络文件共享系统

NFS 文件存储系统主要提供静态文件存储服务器，同时也提供了文件共享服务功能。保证用户上传的文件在 Web 服务器上保持数据同步。

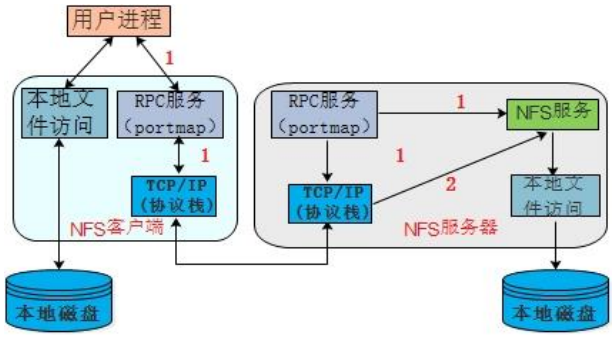

图 4 NFS 的工作原理图

NFS 服务端部署过程如下：

(1) 创建/data/web/目录。创建命令语句如下：

[root@nfs ~]# mkdir -p /data/web/

(2) 修改目录属主。修改命令语句如下：

[root@nfs ~]# chown -R nfsnobody.nfsnobody /data

(3) 下载安装软件。下载命令语句如下：

[root@nfs ~]# yum install -y nfs-utils rpcbind

(4) 配置 NFS 文件。配置命令语句如下：

[root@nfs~]# vim /etc/exports

/data/webs 172.20.1.0/24(insecure,rw,async,no_root_squash)

(5) 启动服务程序。启动服务命令语句如下：

[root@nfs~]# systemctl start rpcbind.service

[root@nfs~]# systemctl start nfs

4.3 SeRsync 文件实时同步备份

SeRsync 实时同步服务首先需要部署好 Rsync 守护进程服务，实现数据传输。为了保证防止 NFS 文件存储服务器宕机导致用户数据丢失等情况，使用 Sersync 文件实时同步服务来保证 NFS 文件存储服务器数据的安全性，其工作原理如图 5 所示。

SeRsync 安装和配置步骤如下：

(1) 解压压缩包到/usr/local 目录下

具体的操作命令如下:

[root@nfs tools]# tar -zxvf sersync -C /usr/local/

(2) 修改文件目录名称

具体的操作命令如下:

[root@nfs tools]# mv GNU-Linux-x86/ sersync

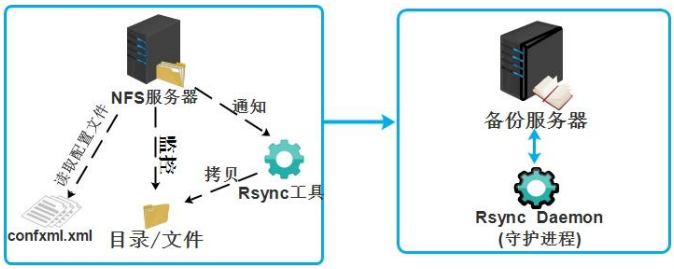

图 5 实时文件同步工作原理

(3) 修改 confxml.xml 配置文件

具体的执行代码如下:

```
[root@nfs sersync]# vim confxml.xml
 <localpath watch="/data/web/">  #本地监听目录
  <remote ip="172.20.1.41" name="web"/> #远端共 享目录 IP 和模块
 </localpath>
 <rsync>  #调用 rsync 参数设置
  <commonParams params="-artuz"/>  #rsync 同步是使用的参数，增量传输
  <auth start="true" users="rsync_backup" passwordfile="/etc/rsync.password"/> #开启 rsyncd 自动认证方式
```

(4) 密码文件。具体的操作命令语句如下:

[root@nfs ~]# echo "123" >> /etc/rsync.password

(5) 修改文件权限。修改文件权限的命令语句如下:

[root@nfs ~]# chmod 600 /etc/rsync.password

(6) 启动 sersync 守护进程。守护程序启动命令如下:

[root@ nfs sersync]# sersync2 -dro /usr/local/sersync/confxml.xml

4.4 数据库主从复制搭建

主从数据复制，可以执行 change master to 命令来完成。命令调用时，需要输入主数据库的 IP 地址,端口号、授权的用户和密码、binlog 的信息等。从库会将这些数据记录到 master.info 文件中。从库执行 start slave 命令来完成相应的工作任务，过程如下：首先开启 SLAVE_SQL_THREAD 和 SLAVE_IO_THREAD 线程；接着从库的 SLAVE_IO_THREAD 线程读取 master.info 文件信息，获取 IP 地址，端口等信息；最后请求连接主数据库，这时主数据会提供一个 Binlog_Dump Thread 线程来和从库的 SLAVE_SQL_THREAD 线程进行交互。从库的 SLAVE_IO_THREAD 线程会根据 binlog 信息请求数

据库提供最新的 binlog 信息，而主数据库通过 Binlog_Dump_Thread 线程将最新的 binlog 信息发送给从库的 SLAVE_IO_THREAD 线程。当从数据库收到最新的 binlog 信息后，会向主数据库发送 ACK 确认控制字段，表示收到最新的 binlog 信息。从数据库读取 relay.info 中的信息,获取上次已经同步的的位置信息，更新从数据库数据。

主从数据库的搭建过程大致如下：

(1) 从库进行数据库恢复，保障数据与主数据库一致。具体的操作命令语句如下:

```
[root@ db01 ~]# mysqldump -uroot -p123456
-x -A -B –events --master-data=2>data.sql
```

(2) 修改置 slave 配置文件。具体的操作命令语句如下：

```
[root@db02 ~]# vim /etc/my.cnf
server_id = 2
```

(3) 恢复数据库数据。具体操作命令语句如下：

```
[root@db02 ~]# mysql -uroot -p123456 < data.sql
```

(4) 查看主库的 log-bin 文件名和 Position 位置点。操作命令语句如下：

```
mysql> show master status;
```

(5) 配置主从复制信息。执行代码如下：

```
mysql> CHANGE MASTER TO
-> MASTER_HOST='172.20.1.51',  #主数据库 IP
-> MASTER_PORT=3306,           #端口号
-> MASTER_USER='root',     #远程登陆用户名
-> MASTER_PASSWORD='123456',  #登陆密码
 -> MASTER_LOG_FILE=' mysql-bin.000011',  #主库文件
 -> MASTER_LOG_POS=1000444;  #主库 bin-log 的位置
```

(6) 在从数据库服务器上执行 start slave 命令。

(7) 执行 show slave status \G 命令，查看 slave 状态。

4.5 Web 服务器搭建

yum 源没有 Nginx 软件，因此需要使用源码编译安装的方法来搭建 Nginx 服务。编译安装步骤如下：

(1)安装 Nginx 主程序。具体的执行代码如下：

```
[root@web01 nginx]# ./configure
--prefix=/application/nginx #程序安装家目录
--user=nginx                #程序属主
--group=nginx               #程序属组
--with-http_ssl_module      #ssl 加密模块
--with-http_stub_status_module #连接状态模块
--with-pcre #正则表达式库
```

(2)编译安装。操作命令语句如下：

```
[root@web01 nginx]# make && make install
```

(3)启动 Nginx。操作命令语句如下：

```
[root@web01 ~]# /application/nginx/sbin/nginx
```

4.6 PHP 服务搭建

完整的 LNMP 架构 PHP 是必不可少的，因为 PHP 提供了动态文件解析功能，目前人们常使用 PHP 来编写动态网站。本架构将 Nginx 和 PHP 分布式的安装在不同的服务器上，采用 FastCGI 方式通信，这样 Nginx 服务器就只负责处理静态服务， PHP 服务器主要任务是处理动态文件，这样性能将会大大提高。

PHP 服务的搭建与配置流程：

(1) 更新 yum 源卸载系统自带的 PHP 软件。具体的执行代码如下：

```
[root@php ~]# yum remove php-mysql php php-fpm
php-common
[root@php~]#rpm -Uvh
https://dl.fedoraproject.org/pub/epel/epel-release-latest-
7.noarch.rpm
[root@web01 ~]#rpm -Uvh
https://mirror.webtatic.com/yum/el7/webtatic-release.rpm
```

(2) 安装 PHP 软件。具体的执行代码如下:

```
[root@php ~]# yum install -y php71w php71w-cli
php71w-common php71w-devel php71w-embedded  php71w-gd
php71w-mcrypt php71w-mbstring php71w-pdo php71w-xml
php71w-fpm php71w-mysqlnd php71w-opcache
php71w-pecl-memcached php71w-pecl-redis
php71w-pecl-mongodb
```

(3) 编写配置文件。具体的操作命令语句如下:

```
[root@php ~]# vim /etc/php-fpm.d/www.conf
user = www
group = www   #保证 Nginx 进程的管理用户和 PHP 服务进
程的管理用户保持一致
```

(4) 启动 php 服务。具体的操作命令语句如下：

```
[root@PHP ~]# systemctl enbale php-fpm
[root@PHP ~]# systemctl start php-fpm
```

4.7 实现 Nginx + PHP 建立关系

PHP 和 Nginx 部署在不同的服务器上，若想让 PHP 和 Nginx 相互配合工作，需要建立关系，监听端口为 9000。其建立过程如下：

(1) 编辑 Nginx 配置文件在 server 区内添加 location 块。具体的操作命令语句如下：

```
[root@web01 ~]# vim /etc/nginx/conf.d/default.conf
location ~ \.php$ {
root                /html/web/;
fastcgi_pass 127.0.0.1:9000;  #指定连接 php-fpm 监听端口
fastcgi_index   index.php;
fastcgi_param SCRIPT_FILENAME
$document_root$fastcgi_script_name;
```

```
include        fastcgi_params;
}
```

(2) 编写 phpinfo()函数，测试配置是否正常。

4.8 验证 PHP 与 Mysql 的连通性

由于 Mysql 数据库是独立的，所以要测试 PHP 服务是否可以连接上 Mysql 数据库，如果正常则返回 SUCCESS。其测试代码如下：

```
[root@web01 ~]# vim /html/web/conn-mysql.php
<?php
$servername = "172.20.1.100";   #数据库 IP 地址
$username = "root";             #远程登陆用户
$password = "123456";           #登陆密码
$conn = mysqli_connect($servername, $username, $password);
if ($conn) {
        echo "mysql successful by root !\n";   #连接成功输出内容
        }
else{
        die("Connection failed:“.mysqli_connect_error());   #连接失败输出内容
                }
?>
```

4.9 负载均衡服务器

使用 Nginx 的 upstream 模块来实现将用户的请求均衡的分配给后端 Web 服务器，从而降低 Web 服务器的压力，提高处理数据的效率，本文采用的指定权重的分配方式，其配置如下：

```
[root@lb01 ~]# vim /etc/nginx/conf.d/default.conf
upstream conf{
  server 172.20.1.7 weight=2;
  server 172.20.1.8 weight=1;
  server 172.20.1.9 backup;   #热备份
}
```

4.10 实现 Keepalived 高可用服务

Keepalived 通过虚拟 IP 的漂移实现高可用,可以有效的避免服务器架构出现单点故障，提高架构的稳定性。

Keepalived 高可用服务配置过程如下:

(1) 安装 keepalived 软件后，编写 keepalived.conf 文件。相应的执行代码如下：

```
global_defs {
vrrp_script implement {
script   "/check_web.sh"   #指定脚本文件
}
 vrrp_instance df {     #VRRP 协议配置
 state MASTER          #服务器状态
 interface ens34       #网卡选择
 virtual_router_id 51   #定义虚拟路由 id
 mcast_src_ip 172.20.1.5   #本机 IP 地址
  priority 150         #定义优先级
 authentication {       #认证方式
    auth_type PASS
    auth_pass 1111
}
   virtual_ipaddress {
      172.20.29.200/24  #指定虚拟 IP 地址
}
   track_script {
      implement   #执行监控脚本
      }
}
```

(2) 编写 Nginx 状态检测脚本 check_web.sh。相应的执行代码如下：

```
#!/bin/bash
num=`ps -ef|grep -c [n]ginx`
if [ $num -lt 2 ]
then
systemctl stop keepalived
echo "服务异常,请检查"
fi
```

4.11 Zabbix 监控服务器部署

当一个集群部署完成后，在后期的运行过程中难免会产生问题，然而管理员不可能实时的去监控每一台服务器的运行状态，所以要想实现管理员可以对网站架构实时监控，出现故障可以及时处理，必须在服务器架构中部署监控软件，来方便管理员对服务器的运行状态进行监控。Zabbix 监控软件部署分为服务端和客户端。

服务端搭建过程如下：

(1)下载 zabbix 软件

```
[root@zabbix~]#rpm -ivh https://mirrors.tuna.tsinghua.edu.cn/zabbix/zabbix/4.0/rhel/7/x86_64/zabbix-release-4.0-1.el7.noarch.rpm
```

(2) 配置 zabbix_server.conf 文件

```
[root@zabbix~]#  vim  /etc/zabbix_server.conf
126  DBPassword=zabbix
[root@zabbix~]#  vim  /etc/httpd/conf.d/zabbix.conf
21 php_value date.timezone Asia/Shanghai
```

(3) 启动 mariadb 数据库

```
[root@zabbix~]# systemctl start mariadb.service
```

(4) 创建 Zabbix 数据库

```
mysql>create database zabbix character set utf8 collate utf8_bin;
```

(5) 创建数据库管理用户。执行代码如下：

```
mysql> grant all privileges on zabbix.* to zabbix@localhost identified by 'zabbix';
```

(6) 在 zabbix 数据库中导入相应的表信息。执行代码如下：

```
[root@zabbix~]# zcat
/usr/share/doc/zabbix-server-mysql-4.0.27/create.sql.gz|mysq
l -uzabbix -pzabbix zabbix
```

(7) 登录 zabbix 服务 web 页面,测试是否安装成功。

客户端搭建过程如下：

(1) 下载、配置客户端软件。执行的命令语句如下：

```
[root@web01~]#vim/etc/zabbix/zabbix_agentd.conf
98 Server=172.20.1.71
```

(2) 启动服务。执行的命令语句如下：

```
[root@web01 ~]# systemctl start zabbix-agent
```

(3) 查看服务运行状态。执行的命令语句如下：

```
[root@web01 ~]# netstat -lntup|grep 10050
```

5 结束语

本文基于 Linux 平台下，进行 Web 服务器架构的搭建，提供 Web 服务给用户访问，采用负载均衡技术解决了 Web 服务器无法承受大量用户频繁访问的问题支持用户高并发的访问。服务器架构具备高可用，能够有效杜绝服务器架构出现单点故障而引起网站宕机，提高整体服务器架构的稳定性。全网数据的定期备份以及 Mysql 数据库的主从复制，保证了服务器日常运行产生数据的安全。使用 Zabbix 监控软件以及编写的脚本文件实现了管理员对服务器的运行状态进行 7*24 小时不间断的监控，出现故障可以及时处理，对中小型企业服务器架构部署一定的参考价值。

参 考 文 献

[1] 李亚非. 高并发 WEB 服务器集群负载均衡器的设计与实现[D].成都：电子科技大学,2019.

[2] 王永辉. 基于 Nginx 高性能 Web 服务器性能优化与负载均衡的改进与实现[D]. 成都：电子科技大学.

[3] 汪海洋, 凌永兴, 包丽红,等. 基于 keepalived 的高可用性应用研究[J]. 电子技术, 2014(7):21-24.

[4] 何小川. MySQL 数据库主从复制的实现[J]. 广东通信技术, 2017(10):58-60.

[5] E·扎达克. Linux 网络文件系统管理指南[M]. 北京：电子工业出版社, 2001.

[6] 郭晓慧, 李润知, 张茜等. 基于 Zabbix 的分布式服务器监控应用研究[J]. 通信学报, 2013, 34((S2):94-98.

[7] 孙东奇, 王秀梅. 用 Rsync 实现 Linux 文件系统备份[J]. 教育信息化, 2006(3):35-36

[8] 鸟哥.鸟哥的 Linux 私房菜 基础学习篇第四版.北京：人民邮电出版社,2018.10: 289-4