基于 Linux 的高可用服务器集群架构设计与实现*

何榆锋　　林宁**

南宁学院信息工程学院，　南宁 530200

摘　要　为了给企业提供稳定可靠的基础网站部署平台，需要搭建一个高可用网站集群架构，该架构必须要支持高并发、高可用、低延迟，才能保证企业网站能对外提供稳定服务。架构整体采用免费开源软件搭建，首先采用 LNMP 来做基础框架。此基础上增加代理，把用户流量分散提交给服务器，这样能有效地缓解和分流用户访问压力，大幅度提高服务器在响应数据上的效率，达到负载均衡。在使用率高的代理和数据库中配置 keepalived 高可用，可以保证服务不间断。增加监控，监控集群状态，发现异常可及时处理，保证了集群架构 7*24 小时运行。数据库配置互为主从，可以相互备份数据，避免数据丢失或损坏。搭建堡垒机，通过它对集群内服务器进行管理。配置防火墙防止网络攻击，保护架构的安全。

关键字　LNMP，高可用，监控，堡垒机，集群

Design and Implementation of High-availability Server Cluster Architecture Based on Linux

He Yufeng　　Lin Ning

School of Information Engineering
Nanning University
Nanning 530200,China

Abstract—In order to provide a stable and reliable basic website deployment platform for enterprises, it is necessary to build a high availability website cluster architecture, the architecture must support high concurrency, high availability, low latency, to ensure that the enterprise website can provide stable services. The overall architecture is built by free and open source software. Firstly, LNMP is used as the basic framework. On this basis, adding a proxy to distribute the user traffic to the server, which can effectively alleviate and shunt the pressure of user access, greatly improve the efficiency of the server in responding to data, and achieve load balancing. keepalived High availability is configured in high usage agents and databases to ensure uninterrupted service. Monitoring is added to monitor the status of the cluster, and anomalies can be handled in time to ensure that the cluster architecture runs 7*24 hours. The database configuration is the master and slave of each other, which can backup data with each other to avoid data loss or damage. The fortress machine is built to manage the servers in the cluster. Configure firewall to prevent network attacks and protect the security of the architecture.

Key words—LNMP, High-availability, Monitoring, Bastion Host, Cluster

1　引　言

随着互联网的蓬勃发展，企业网站变得非常重要，可以通过它来建立推广企业形象和产品、发布重要信息或招聘信息、与客户互动联系等，这样能为企业带来更大的收益[1][2]。在《中国互联网络发展状况统计报告》中提到二零二一年六月止，我国的上网人数已经达到了10.11亿，互联网的普及程度达到了71.6%[3]。网站在面对庞大访问量传统的单服务器网站，会导致系统不能稳定运行，提供不了稳定的服务。

针对以上问题，本文基于LNMP架构进行扩展，结合zabbix和jumpserver以及keepalived软件，实现一个能为中小企业提供满足高可用、高并发、可扩展稳定安全基础的网站部署平台。

2　相关技术研究

2．1　LNMP 架构

LNMP为当前流行的一种网站服务器体系结构，该体系结构是指以Linux系统为平台，以Nginx服务器为核心，采用MySQL数据库与编译工具PHP相结合的网站服务器体系结构。是对LAMP改进后的一种新的网站服务器架构。其性能比LAMP优越、效率高、体积小等特点，许多网管人员对其倍加青睐[4]。

*基金资助：本文得到南宁学院 2019 年度教授培育工程项目（2019JSGC12）资助。
**通讯作者：林宁，副教授，bgy_2009@163.com

2．2 高可用软件

Keepalived作为一款集群管理的服务软件，最初专为Linux虚拟服务器进行负载均衡，同时还能够对Linux虚拟服务器进行集群系统内各服务节点进行状态管理。直至增加高可用VRRP（虚拟路由冗余协议）功能为止，确保了单个节点宕机后整个网络仍然能够正常工作。所以其他服务（例如Nginx）的高可用解决方案可以采用Keepalived软件，Keepalived可根据检测服务器的状态来剔除无法使用的机器，这样我们就可以实现Nginx集群和高可用技术[5]。

2．3 监控技术

Zabbix是企业中常用的监控系统，它可以支持对网络设备、服务器等实时监测，监测的数量能达到上万个，可以把数百万级别的监控指标收集起来，并且完全开源的，免费的。Zabbix具有指标采集、问题检测、告警和修复、安全和认证、轻松搭建部署、强大的接口支持等实用的运维管理功能[6]。

2．4 堡垒机

JumpServer是FIT2CLOUD旗下免费开源的软件产品，被许多企业应用于内部资产（物理机，云主机等）管理。通过Jumpserver堡垒机来确保IT系统内的网络、数据不会受到内、外用户的侵入与损坏，实现了用户到网络设备的访问、对服务器和业务系统运行维护过程进行全程监控，实时记录已输命令。同时，通过设置权限，实现资产隔离管理，这样就能保证非专业人员对其它的服务器造成破坏和入侵[7]。

3 中小型企业网站集群架构设计

中小型企业网站总体集群架构设计如图 1 所示。

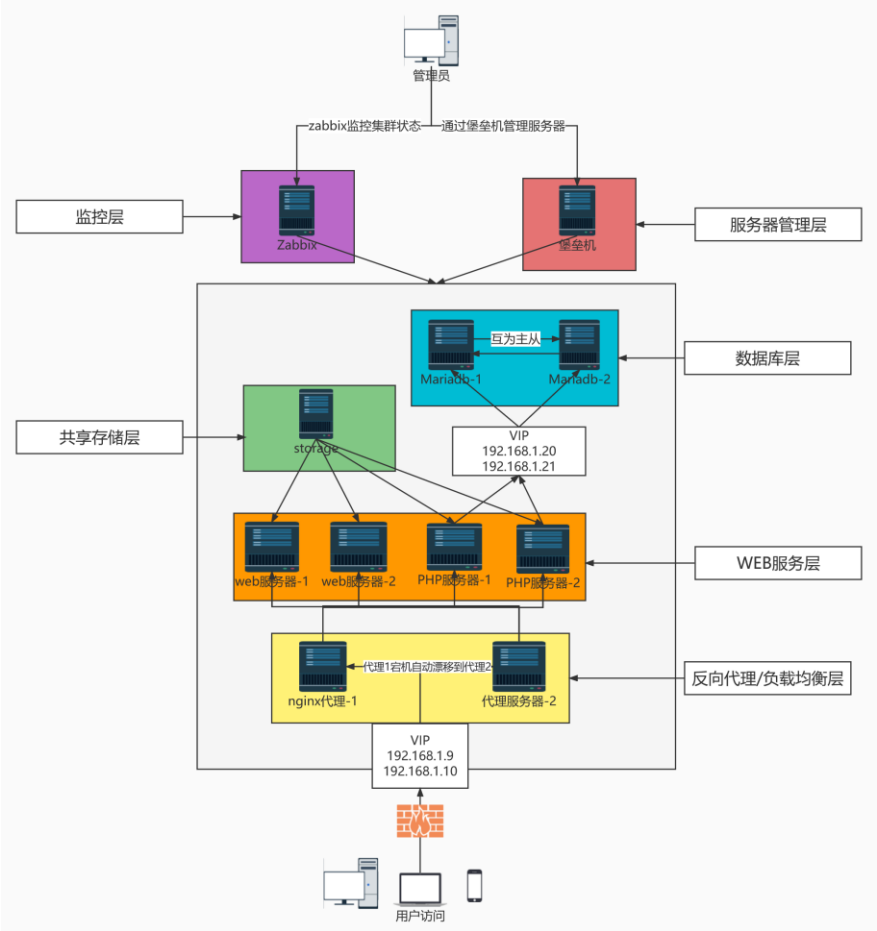

图 1 中小型企业网站集群架构设计图

（1） 反向代理/负载均衡层

代理负责转发用户请求到后端 WEB 服务层中处理。负载均衡主要解决单机的处理性能达不到需求的问题。由于互联网的飞速发展，用户量增多，当访问量达到峰值时需利用 Proxy 将请求数据交给负载均衡器，根据算法（权重、轮询等）将请求转发到后端 WEB 中处理，WEB 处理完请求数据后，会把响应数据返回给 Proxy[8]。算法合理的负载均衡和分配恰当的

请求可以让后端 WEB 服务层共同处理请求，这样就能减少单机压力，降低负载，保证服务的稳定性[9]。对该层配置 Keepalived，如某台服务器宕机了，另一台可以接着使用，保障服务正常运行。提供单独 VIP 供用户访问。

（2） WEB服务层

WEB服务是处理用户请求的。WEB根据用户的访问需求，查询相关的数据库、web和php服务器代码等。本架构用到2台web来增加架构可用性和稳定性。如某台服务器宕机时，另一台仍然可以接着使用。

（3） 共享存储层

共享存储是给web和php服务器进行共享，在部署网站时可以直接在共享存储层进行代码上传，这样web和php服务器也能共享到代码，可以减少部署网站的工作量和时间。

（4） 数据库层

数据库负责存储数据。网站的数据库的性能好坏，很大程度会影响用户体验，本架构对该层，配置为互为主从，保证不同数据库的数据都能同步。对该层配置Keepalived，当一台宕机了，另一台可以接着使用，保障架构正常运行。提供单独VIP供用户访问。

（5） 监控层

监控层主要负责监控集群架构，如果服务器出现故障会第一时间发送信息通知运维进行维护。还能实时监控各服务器的性能，如果性能不足可以第一时间对设备进行扩展，增加性能，保障架构7*24小时运行。

（6） 管理层

服务器管理层主要是堡垒机，可以在这层对整个架构的服务器进行管理操作。

4 中小型企业网站集群架构部署实现

本网站集群架构都是基于开源软件实现的，集群服务中的数据都做了高可用备份，数据安全性高以及支持高并发访问，能够网站平台来解决搭建自己家网站而产生不必要的麻烦。

4．1 部署 Web 服务器

web 结合 php 服务器一起使用。Web Server 和用户间的通讯过程原理：开始，用户请求发送到 Web Server 上，Web Server 收到请求后，对请求进行响应，Web Server 需监听用户请求发送的端口，若 Server 端口为 80，需先将请求整理归类，继续监听服务器的端口，监听结束后，需要对用户进行应答，我们要协调 php 服务器，或者协调 mariadb 数据库，去访问硬盘，读取对应内容在硬盘上，待有关内容组装完成后，再转换为标准 HTML 源文件页面，然后发送给用户，这里面有两个过程，一个为请求过程，一个为响应过程，从而实现 Web 服务工作流程，用户以 URL 地址的方式向服务器发送申请，服务器对用户的申请进行处理，完成了整个前后端以及数据库端的网站工作原理。

Web服务器实现步骤如下：

① 安装 Nginx 服务。

```
# yum -y install nginx
```

② 打开编辑 Nginx 站点配置文件并将站点文件 server.conf 放入 conf.d 目录中方便网站的站点管理，设置服务端口为 8080，并指定新的站点目录 /var/www/web。

```
server {
listen 8080; #指定服务的监听端口，端口为 1--65535 的整数值
server_name www.heyufeng.com;
charset utf-8;
access_log /usr/local/nginx/logs/heyufeng.access.log main;
location / {
root /var/www/web; #指定站点空间根目录
index index.html index.htm;}
error_page 500 502 503 504 /50x.html;
location = /50x.html {
root /usr/local/nginx/html;}}
```

③ 创建站点目录/var/www/web 并启动 Nginx。

```
#创建目录  mkdir -p /var/www/web
#启动  nginx /usr/local/nginx/sbin/nginx
```

4．2 部署 PHP 服务器

php 服务器，用来处理动态数据。我们访问网站时，会发现后缀名未必就是 html，有的 php、asp、jsp、ruby、python 等或者其他开发的网页，这类网页我们需要借助处理动态数据的服务器来处理这类动态资源，web 服务器仅承担静态页面资源的管理，要求二者相结合，共同完成任务。比如：当我们要访问数据库时我们是无法让 HTML 或者图片等这种静态文件去访问，必须得采用动态服务器才能实现对数据库访问。

PHP服务器实现步骤如下

① 网上找 PHP 软件包并安装到 Linux 系统。

② 创建/var/www/web 站点目录，启动 PHP。

```
#创建目录  mkdir -p /var/www/web
#启动 PHP  /usr/local/php5.6/sbin/php-fpm
```

4．3 代理服务器配置部署

Web 和 PHP 服务器是通过 Nginx_Proxy 代理来实现负载均衡的，部署代理服务器主要通过修改 Nginx 配置内容，设置 upstream 的集群节点来实现。Nginx_Proxy 架构如图 2 所示，配置双机热备 Keepalived 达到一定的负载均衡，用户通过 VIP 来访

问两台代理，如果代理 1 宕机，代理 2 还能继续工作，达到高可用。

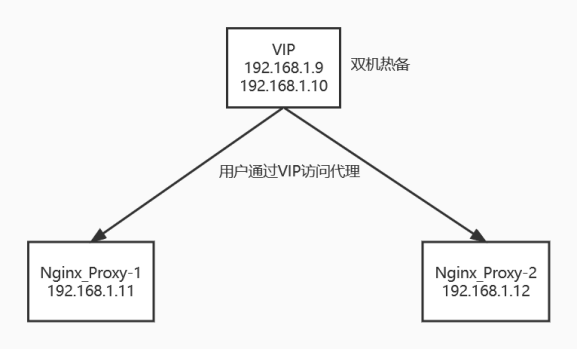

图 2 Nginx_Proxy 架构图

（1） 代理服务器实现步骤如下。

① 编辑修改代理的主配置文件 vim nginx.conf。

② http 节点下面，加入 upstream 节点，将节点命名 webServer 和 phpServer，配置参与负载均衡的服务器 IP 信息，权重比，最大失败次数和超时时间。

#默认算法为轮询，设置了 weight 权重比，可以根据权重分配任务

```
upstream webServer {
server 192.168.1.13:8080 weight=1 max_fails=3 fail_timeout=10s;
server 192.168.1.14:8080 weight=1 max_fails=3 fail_timeout=10s;}
upstream phpServer {
ip_hash;
server 192.168.1.15:9000 max_fails=3 fail_timeout=10s;
server 192.168.1.16:9000 max_fails=3 fail_timeout=10s;}
```

配置 2 个节点，分别是 webServer 集群和 phpServer 集群。1.13、1.14 是两台 web 服务器 IP，1.15、1.16 是两台 php 服务器 IP。Nginx 的 upstream 节点默认是通过轮询来向后端分配任务，这样一来，请求将按特定时间进行分配，最后把所有的请求分配到集群架构的后端服务器上。这里把 webServer 按照权重比的方式实现负载，如此一来请求就会根据权重比的高低分配。将 phpServer 按 ip_hash 方式达到负载均衡，这样，请求将按照访问 IP 时的 hash 结果被逐个分发，各用户接入的后端服务器是保持不变的，解决了 session 不能分享的问题，若任一服务器宕机，就自动在负载均衡里剔除掉，并且不会影响到业务的正常运行。

③ 单独配置 server 站点便于独立管理，存放在/usr/local/nginx/conf.d/目录下后启动 nginx。该目录只能存在一个.conf 的代理配置文件，配置如下：server {

```
listen 80；
server_name www.heyufeng.com;
charset utf-8;
access_log /usr/local/nginx/logs/heyufeng.access.log main;
location / {
root /var/www/web/;
index index.html index.htm;
proxy_pass http://webServer;
proxy_set_header X_Real_IP $remote_addr;
proxy_set_header X_Forwarded-For $proxy_add_x_
forwarded_for;
proxy_redirect off;
proxy_connect_timeout 30;
proxy_send_timeout 15;
proxy_read_timeout 15;
}
fastcgi_connect_timeout 30;
fastcgi_send_timeout 30;
fastcgi_read_timeout 30;
fastcgi_buffer_size 64K;
fastcgi_buffers 4 64K;
fastcgi_busy_buffers_size 128K;
fastcgi_temp_file_write_size 128K;
location ~ \.(php|php5)$ {
root /var/www/web;
fastcgi_pass phpSerer;
fastcgi_index_index.php;
fastcgi_param SCRIPT_FILENAME $document_
root$fastcgi_script_name;
include fastcgi_params;
}}
```

（2） Keepalived 服务安装部署实现步骤

为了服务器能达到高可用则需要安装部署 Keepalived，实现方式主要是通过修改它的主配置文件，配置 vrrp_instanse 节点来实现，本系统中要实现 Keepalived 双机热备。

① 打开并编辑 Keepalived.conf 配置文件。

#编辑配置文件 vim keepalived.conf

② 设置 router_id，添加 VI-1 和 VI-2 两个节点，配置主备，组 id，优先级，主备检查间隔时间，密码，虚拟 VIP。 keepalived 节点配置如下：

```
global_defs { router_id keep_11 } #主机 ID
vrrp_instance_VI_1 {
state MASTER #主机
interface eno16777726 #绑定网卡
virtual_router_id 50 #表示哪些服务器一个组
priority 100 #优先级/权重，谁的优先级高会成为主机
advert_int 1 #同步检测时间（1 秒）
authentication { #密码，阻挡非法的节点加入
auth_type PASS
auth_pass 1111
}
virtual_ipaddress { #虚拟 VIP
192.168.1.10
}
}
vrrp_instance_VI_2 {
state BACHUP
interface eno16777736 #绑定网卡
virtual_router_id 55
priority 100
advert_int 1
authentication {
auth_type PASS
auth_pass 2222
}
```

```
    virtual_ipaddress {
        192.168.1.9
    }  }
```

本系统要实现双机热备主要是在两台不同服务器（S_1 和 S_2）上分别配置 VI_1 和 VI_2 节点，这两个节点一个为主 MASTER，一个为备 BACKUP，这样 2 台服务器都有主备节点。S_1 的主节点配置虚拟 VIP 为 192.168.1.10，S_2 的主节点配置虚拟 VIP 为 192.168.1.9。由于 Keepalived 选出 MASTER 后所有用户的请求都是通过 MASTER 访问的，那么说只要配置了双机热备就相当于 2 台服务器都是 MASTER，一台为 1.10，一台为 1.9，这两个 VIP 都提供给用户进行访问，这样也算是为代理配置实现了一定的负载均衡。

当 MASTER 宕掉了，就需从备用的服务器中选出一个来接替它的工作。MASTER 选举方式，如果虚拟的 VIP 设置为服务器本身的 IP 那么这台服务器的优先级为最大，只要它存在那么它就一直都是 MASTER。如果不设置虚拟 VIP 那么在同一个节点中的服务器都会对外宣告自己是 MASTER，发送 VRRP 宣告信息。当收到其他服务器发来的宣告，就会比较它们的优先级，谁的优先级高就为 MASTER。如果优先级都是相等的值时就看 IP 值，哪个 IP 值最大就为 MASTER。

③ 立即启动 keepalived 服务，并设置自启。

```
# systemctl enable --now keepalived.service
```

4．4　共享存储服务器部署

Storage 共享存储，主要是将几块新的硬盘配置成 RAID10 阵列（镜像阵列条带），然后把 RAID10 阵列制作成 LVM 分区，便于扩展管理，最后通过 NFS 共享给其他服务器使用。

共享存储服务器实现步骤如下:

① 为服务器添加 B、C、D、E 四块硬盘。

② 安装 mdadm 软件（做 RAID10 需要使用）。

```
# yum install -y mdadm.x86_64
```

③ 创建 RAID10 并创建挂载目录/webData。

```
# mdadm -Cv /dev/md10 -a yes -n 4 -l 10 /dev/sdb /dev/sdc /dev/sdd /dev/sde
```

④ 将 RAID10 做成 LVM 分区，便于扩展管理。

```
#添加物理卷   pvcreate /dev/md10
#创建卷组   vgcreate webData /dev/md10
#创建逻辑卷   lvcreate -n lv_web -L20G webData
#格式化逻辑卷   mkfs.xfs /dev/webData/lv_web
```

⑤ 将 lv_web 写入文件，开机自动加载。

```
# echo "/dev/webData/lv_web /webData xfs defaults 0 0" >> /etc/fstab
```

⑥ 安装 NFS 服务，启动并设置开机自启。

```
#安装 NFS 服务   yum install rpcbind.x86_64 -y
#启动并设自启   systemctl enable --now rpcbind
```

⑦ 编辑修改 NFS 的配置文件，启动 NFS 服务。

```
#编辑文件   vim /etc/exports
/webData *(rw,sync,all_squash,anonuid=99.anongid=99)
# systemctl start nfs          #启动 nfs
```

⑧ Web 及 PHP 服务器将共享挂载到本地使用。

```
# yum install -y rpcbind.x86_64   安装 NFS，才可以支持 NFS 类型文件系统挂载
# 挂载 NFS 光盘  mount -tnfs 192.168.1.17:/webData /var/www/web/
```

4．5　数据库服务器部署

数据库部署，主要实现对数据进行组织、存储、管理；主要表现在数据维护性、数据安全性、数据一致性、数据恢复等方面。数据库架构如图 3 所示，mariadb 实现互为主从，其他服务通过 VIP 连接数据库，当 mariadb-1 宕机，mariadb-2 可以接着工作，达到高可用。为数据库配置 Keepalived 双机热备能达到高可用以及一定的负载均衡，具体配置参考 4.3 代理服务器的 keepalived 服务安装部署。

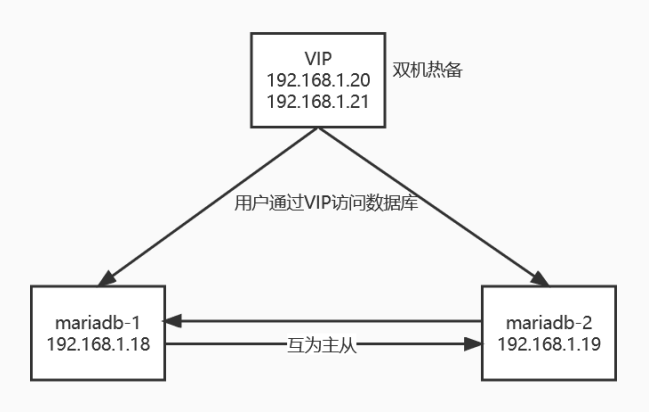

图 3　mariadb 架构图

（1）数据库服务器实现步骤

① 利用 yum 安装 mariadb 的服务端（MariaDB-server）和客户端（MariaDB-client）。

② 启动并初始化 mariaDB 数据库，设开机自启。

```
# mysql_secure_installation     #初始化数据库
```

互为主从也叫做主主同步，在主从同步原有的基础上添加了主也同步从的 bin-log（日志），使用 Keepalived 的虚拟 IP 又称 VIP 监视两个数据库，实现 ip 漂移可以去除了之前一主一从以及一主多从的主单点问题，这样能达到数据库的高可用。

互为主从的原理如图 4 所示，从图中可以看出是 master_1 和 master_2 都是通过 io 线程来读取 bin-log 日志文件内的数据，之后再从 io 线程将在 bin-log 日志中读出的全部数据写入 relay-log 日志中，最后在交给 sql 线程从本地把数据写入到数据库中。配置 mariaDB 互为主从可以使数据实时同步到另一个服务器中，能实现容灾效果。

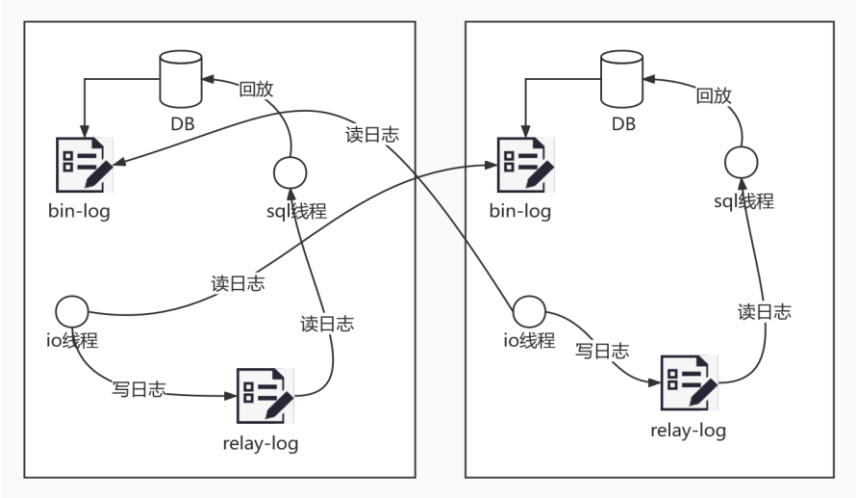

图 4 数据库互为主从原理

（2） 互为主从实现步骤如下

① 分别在 mariadb-1 和 2 主服务器中授权账号，用于给从服务器登录主服务器复制同步日志使用。

grant replication slave on *.* to 'slave'@'192.168.1.19' identified by "heyufeng";

grant replication slave on *.* to 'slave'@'192.168.1.18' identified by "heyufeng";

② 分别打开编辑 mariadb-1 和 2 的 server.cnf 配置文件。该配置文件如下：

```
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
server-id=18
log_bin=mysql-bin
relay-log=relay-mysql  #开启二进制日志
relay-log-index=relay-mysql.index #开启中继日志
log_slave_updates=on
auto_increment-increment=2
auto_increment-offset=1
```

③ 查询 MASTER 的二进制状态，File 列是日志名，Position 列是偏移量，这 2 个数据在 slave 配置时需要用到。

④ 配置从服务器（192.168.1.18 和 1.19），输入主服务器的 IP，输入前面创建用于复制的账号密码，输入上面查询出的记录操作日志的文件名和偏移量。

```
MariaDB [(none)]> change master to
master_host='192.168.1.18',master_user='slave',master_password=
'heyufeng',master_log_file='mysql-bin.000005',master_log_pos=5
298;
```

参数说明如下：

```
CHANGE MASTER TO MASTER_HOST = ‘host’, #MASTER地址
MASTER_PORT = 3306,   #MASTER端口
MASTER_USER = ‘slave’,  #MASTER上创建的用户名
MASTER_PASSWORD = ‘heyufeng’,#MASTER上创建的账户密码
MASTER_LOG_FILE = ‘mysql-bin.000005’,  #File列的数据内容
MASTER_LOG_POS = 5298;   #Position列的数据内容
```

⑤ 激活从功能 start slave;查看 slave 状态是否成功，确保 Slave_IO_Running 和 Slave_SQL_Running 两个值为 YES，则表示成功。

⑥ 验证互为主从是否成功，分别在 2 台服务器上创建数据库，mariadb-1 新建 test_1 库，mariadb-2 新建 test_2 库。

```
MariaDB [(none)]> create database test_1;
MariaDB [(none)]> create database test_2;
```

⑦ show databases 在 2 台数据库分别查询库，如果都能看到新建的 2 个库，表示互为主从是能使用的。

4．6 监控服务器部署

我们通过 Zabbix 来实现针对服务器的 cpu，内存，磁盘，网卡，安全进行检测，采用的是 Zabbix 通用监控模板“Template OS Linux by Zabbix agent”。Zabbix 是利用客户机/服务器模式来进行数据采集，采用浏览器/服务器模式在前端可视化页面进行展示以及配置相关参数。被监控的服务器，是采用 agent2 方式来配合服务器端进行数据采集。服务器端收集 agent2 发送的数据，写进 MariaDB 数据库，同时利用 php+apache 显现在 web 前端[9]。

（1） 服务端监控服务器实现步骤如下

① 安装 Zabbix 源，然后修改 Zabbix 源的下载链接为阿里云的，从而提高整个的下载速度。

```
# rpm -ivh zabbix-release-5.0-1.el7.noarch.rpm
# sed -i
's#http://repo.zabbix.com#https://mirrors.aliyun.com/zabbix#'
/etc/yum.repos.d/zabbix.repo
```

② 安装 Zabbix 服务。并安装 scl 工具，使用 scl 可以运行多个不同版本的软件，并且不会影响其他软件的系统依赖环境。（会默认把下载的软件或生成的配置放入/opt/rh/下）

yum -y install zabbix-server-mysql zabbix-agent centos-release-scl

③ 修改 zabbix-front 前端源，将[zabbix-frontend]下的 enabled=0 改为 enabled=1。

④ 安装 Zabbix 前端环境，要安装到 scl 环境下。

yum install zabbix-web-mysql-scl zabbix-apache-conf-scl -y

⑤ 安装 Zabbix 所需的数据库，这里选择 mariadb 数据库。安装后启动并加入开机自启，初始化数据库。

#安装数据库 yum -y install mariadb-server

#启动并自启 systemctl enable --now mariadb

#初始化数据库 mysql_secure_installation

⑥ 添加数据库用户，以及 Zabbix 所需的数据库信息。

#创建库 create database zabbix character set utf8 collate utf8_bin;

#添加用户 create user 11@localhost identified by '123456';

#为用户授权使用 zabbix 库

grant all privileges on zabbix.* to 11@localhost;

#刷新授权表 flush privileges;

⑦ 使用 zabbix-mysql 命令，导入数据库信息。

zcat create.sql.gz | mysql -uzabbix -p zabbix

⑧ 修改 zabbix_server.conf，改为如下字段。

DBPassword=123456

⑨ 修改 Zabbix 的 php 配置文件 zabbix.conf，最后一行改为 Asia/Shanghai。

⑩ 启动 Zabbix 相关服务器，并设置自启动。

systemctl enable --now zabbix-server.service zabbix-agent.service httpd rh-php72-php-fpm.service

访问入口：192.168.1.100/zabbix 账号：Admin 密码：zabbix

（2） 客户端监控服务器实现步骤如下

① 安装客户端软件 Zabbix_agent2，配合服务器进行数据采集。

② 设置正确时间，时区的统一配置。

ntpdate -u ntp.aliyun.com
mv /etc/localtime{,.bak}
ln -s /usr/share/zoneinfo/Asia/Shanghai /etc/localtime

③ 每一台服务器分别修改 agent2 配置文件，不同是服务器注意 Hostname 修改为自己的主机名。

# vim /etc/zabbix/zabbix_agent2.conf
Server=192.168.1.100
ServerActive=192.168.1.100
Hostname=jumpserver

④ 重启 zabbix-agent2。

systemctl restart zabbix-agent2.service

4．7 堡垒机部署

Jumpserver 是 python 开发的一个开源系统，可以实现前置机所有功能。使用 ssh 来远程登录其他服务器进行管理，客户端不需要安装代理软件就能进行管理。完全开源并获得 GPL 许可。jumpserver 堡垒机实现主要是安装部署一些相应的组件，才能实现相应的功能，运行后台时需要准备好 Python3 解释器、MySQL 数据库、redis 缓存数据库。安装 Nginx 来代理组件的 web 页面。Jumpserver 架构如图 5 所示：

（1） jumpserver 堡垒机实现步骤如下

① 准备 yum 源、安装依赖，修改中文字符集。

② 安装 MySQL 数据库并启动，后初始化密码。

③ 创建 jumpserver 数据库和它的用户并设置密码，给 jumpserver 用户授权访问 jumpserver 库的权限。

mysql> create database jumpserver default charset 'utf8' collate 'utf8_bin';
mysql> create user 'jumpserver'@'%' IDENTIFIED BY 'hyf123456';
mysql> grant all privileges on jumpserver.* to 'jumpserver'@'%' identified by 'hyf123456';
mysql> flush privileges; #刷新授权

④ yum 安装 Python3.6 和 Redis 数据库并启动，Redis 的端口为 6379。

⑤ 部署 Jumpserver（CORE）服务并启动 jms。

tar -xvf jumpserver-v2.1.0.tar.gz
ln -s /teach_jmp/jumpserver-v2.1.0 /teach_jmp/jumpserver
/teach_jmp/jumpserver/jms start -d

⑥ 部署 koko 组件并启动。

Koko 为 ssh 服务和 Web 终端服务，提供 ssh 和 websocket 接口。

vim /teach_jmp/koko/config.yml #编辑配置文件

修改以下内容，没有的参数则在文件后手动添加：

BOOTSTRAP_TOKEN: gLNcW1hflJrc1TuM
LOG_LEVEL: INFO 日志，便于排错
REDIS_HOST: 127.0.0.1
REDIS_PORT: 6379
REDIS_PASSWORD:
REDIS_CLUSTERS:
REDIS_DB_ROOM:
#启动 koko 服务 ./koko -d

⑦ 部署 Guacamole 组件。

Guacamole 是 Apache 跳板机项目，支持 VNC（RFB）、RDP、SSH 等多种标准管理协议。

⑧ 部署 Lina 和 Luna 组件。

Lina 为前端源代码、前端进程，提供美化的前端网页页面。Luna 为 Web Terminal 和 ssh 协议服务。

步骤如下：首先获取 lina 和 luna 压缩包到/teach_jmp 目录

wget https://github.com/jumpserver/lina/releases/download/v2.1.0/lina-v2.1.0.tar.gz

wget https://github.com/jumpserver/luna/releases/download/v2.1.1/luna-v2.1.1.tar.gz

解压 lina 和 luna 的压缩包并修改解压出来目录名

tar -zxf lina-v2.1.0.tar.gz　tar -zxf luna-v2.1.1.tar.gz　#解压缩包
mv lina-v2.1.0 lina　　mv luna-v2.1.1 luna #改名
赋予 lina 和 luna 目录权限并修改用户和组
chown -R nginx:nginx lina
chown -R root:root luna

⑨ 安装配置 Nginx 并启动。

yum -y install nginx　#安装 nginx

修改 nginx 配置文件，添加新主机
nginx -t 检查配置文件语法，无错后启动 nginx

⑩ 进入 jumpserver 前端页面，进行登录查看。

访问地址：http://192.168.1.101　账号：admin　密码：admin

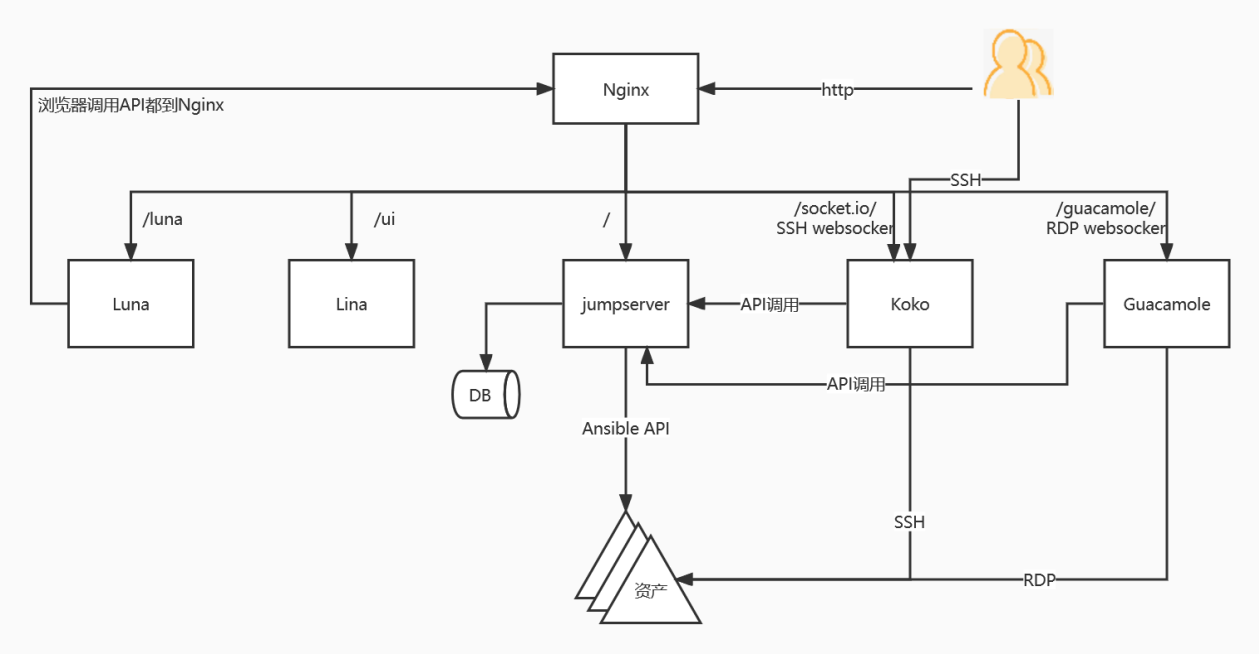

图 5　jumpserver 架构图

4．8　防火墙

利用防火墙设置禁止全部数据进入和出去，然后再将需要放行的数据允许它们所使用的端口通过，将 ssh 的 22 端口设置成只允许堡垒机远程登录，这样能保障了一定的安全性。本系统是使用 Linux 自带 iptables 工具进行完成，Linux 自带的防火墙有两个，一个是 Iptables：netfilter 配置工具，Centos7 版本之前的一个默认的防火墙；还有一个是 firewalld：Centos7 及之后版本的默认防火墙。

防火墙规则设置如下:

① 把默认规则改为禁止，禁止全部数据的进出。

iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

② 设置防火墙状态表，配置了状态表后只需要配置允许数据进来就行，数据进来时就记录再状态表里，数据出去时会自动放行。如果没有状态表，则需要进出都配置允许该数据通过。

iptables -I INPUT 1 -p tcp -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
iptables -I OUTPUT 1 -p tcp -m conntrack --ctstate NEW,RELATED,ESTABLISHED -j ACCEPT

③ 每台服务器上分别添加只允许 jumpserver 堡垒机访问 22 端口，并允许需要对外服务的端口通行。第一条允许堡垒机 22 端口通过，第二条允许源端口号通过，第三条允许目的端口号通过。

iptables -A INPUT -s [源地址] -d [目的地址] -p tcp --dport 22 -i eno16777736 -j ACCEPT　#注意修改目的地址
iptables -A INPUT -p tcp -m multiport --sport [端口号] -j ACCEPT
iptables -A INPUT -p tcp -m multiport --dport [端口号] -j ACCEPT

④ 允许 ping 流量通过。

iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
iptables -A OUTPUT -p icmp --icmp-type echo-reply -j ACCEPT

⑤ 允许 lo 环回接口流量通过。

iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

⑥ 将防火墙规则保存到 iptables 里，让重启服务器不清空规则。

```
# iptables-save > /etc/sysconfig/iptable
```

5 结束语

本架构整体采用免费开源软件，基于 LNMP 基础架构进行扩展，是为了给中小企业提供一个基础的网站部署平台而搭建。

参 考 文 献

[1] .“互联网+”背后的本质到底是什么?[J].信息与电脑(理论版),2017(10):1-2.

[2] 朱启贞.“互联网+”时代我国民营企业战略研究企业网站评价[J].中国管理信息化,2016,19(04):104-105.

[3] 袁瑀. 我国C2C电子商务税收征管问题研究[D].昆明：云南财经大学,2022.

[4] 周毅. 基于LNMP的设备远程管理平台开发[D].成都：西南科技大学,2019.

[5] 胡科,陈勇.基于Nginx+Keepalived实现高可用集群及负载均衡配置与分析[J].微型电脑应用,2022,38(05):200-202.

[6] 陈志伟,李道强,杨江,曾志勇.应用Zabbix构建高校数据中心监控平台[J].福建电脑,2022,38(07):42-44.

[7] 赵伟.基于JumpServer的IT系统安全运维审计的研究与实现[J].电子元器件与信息技术,2021,5(10):1-3.

[8] 李天翼,许鲁,常致全.一种新型的基于网络存储的Web集群解决方案[J].计算机应用研究,2003(10):78-79+112.

[9] 戴春雷. 中小型电子商务网站架构研究与部署实践[D].苏州：苏州大学,2017.