基于 Zabbix 分布式监控系统的设计与实现*

高鹏　林宁**　　左悦

南宁学院信息工程学院，南宁，530200　　南宁学院土木与建筑工程学院，南宁，530200

摘　要　针对单实例监控平台存在单点故障、跨区域数据采集能力不足及告警响应不够及时等问题，设计并实现了一套基于 Zabbix 的分布式监控系统。系统以 Zabbix 为核心，采用 Keepalived 构建主备 Zabbix Server 高可用架构，结合 MariaDB 主从复制实现监控数据的同步与冗余，并通过部署 Zabbix Proxy 完成跨区域监控数据的分布式采集与缓存上报。同时，集成钉钉告警通知和 Grafana 可视化展示模块，提升了监控事件的响应效率与运行状态的展示效果。通过主备切换、数据库同步、告警触发及可视化展示等测试表明，该系统能够在主节点故障场景下保持监控服务连续运行，并实现监控数据的稳定采集、集中管理与直观展示。研究结果表明，该方案能够有效提升监控平台的可靠性、可扩展性和运维效率。

关键字　Zabbix，分布式监控，高可用，Grafana，告警通知

Research and Implementation of a Zabbix-based Distributed Monitoring System*

GaoPeng　LinNing**　　ZuoYue

College of Information Engineering
Nanning University
Nanning 530200 China

College of Architecture and Civil Engineering
Nanning University
Nanning 530200 China
zuoyue@unn.edu.cn

Abstract—To address the problems of single-point failure, limited cross-region data collection, and delayed alert response in traditional monitoring platforms, a distributed monitoring system based on Zabbix is designed and implemented. The system uses Keepalived to build a high-availability architecture for primary and backup Zabbix Servers, adopts MariaDB master-slave replication for data synchronization and redundancy, and deploys Zabbix Proxy nodes for distributed data collection and cached reporting. In addition, DingTalk alert notification and Grafana visualization are integrated to improve monitoring efficiency. Test results show that the system can maintain continuous monitoring service during primary node failure and achieve stable data collection, centralized management, and intuitive display. The proposed scheme effectively improves the reliability, scalability, and operation efficiency of the monitoring platform.

Key words—Zabbix, Distributed Monitoring, High Availability, Grafana, Alerting

1　引　言

随着企业 IT 基础设施规模不断扩大以及业务连续性要求不断提高，运维则需要对服务器、网络设备和关键应用服务进行统一、规范的监控体系[1]，以便及时发现故障隐患并保障系统稳定运行。Zabbix作为成熟的开源监控平台，具有功能完善、扩展性强、支持分布式部署等特点，已广泛应用于各类IT环境的监控管理中[2]。

然而，传统单实例监控架构在可用性和扩展性方面仍存在不足。当监控中心发生故障时，数据采集与告警链路将受到影响；随着监控对象和监控项数量增加，后端数据库压力也会显著上升；在跨区域网络环境下，监控数据上报还容易受到网络波动影响，从而降低系统的稳定性与可靠性。因此，构建兼顾高可用、分布式采集与集中管理能力的监控系统具有较强的实际意义。

针对上述问题，基于Zabbix分布式监控系统应运而生。该系统采用双机热备的Zabbix Server架构，结合MariaDB主从复制实现数据同步与冗余，通过部署Zabbix Proxy节点完成跨区域数据采集，并集成钉钉告

*基金资助：南宁学院横向课题（2025HX100,2026HX008）资助。
**通讯作者：林宁，教授，linning@unn.edu.cn

警和Grafana可视化功能。

2 相关技术研究

2.1 Zabbix 监控体系

Zabbix是一种开源企业级监控平台，具备主机监控、网络监控、告警管理和数据可视化等功能。其核心组件主要包括Zabbix Server、Zabbix Agent、Web前端以及数据库。Server 负责接收与处理监控数据，Agent 部署在被监控主机上用于采集运行指标，Web 前端用于配置管理和结果展示，数据库用于存储配置数据和历史监控数据。

2.2 Zabbix Proxy 分布式采集机制

Zabbix Proxy 是 Zabbix 分布式架构中的关键组件，可部署在不同业务区域，负责本地采集并缓存监控数据，再统一上报至中心 Server。在大规模或跨区域网络环境中，若所有被监控主机均直接连接中心 Server，容易导致中心节点负载过高，并受到网络质量波动的影响。而该机制能够降低中心节点压力，提高复杂网络环境下的数据采集可靠性，并增强系统的可扩展性。

2.3 Keepalived 与 MariaDB 高可用技术

Keepalived 是 Linux 环境下常用的高可用软件，主要通过虚拟路由冗余协议实现虚拟 IP 漂移和主备节点切换[7]。当主节点发生故障时，备用节点可自动接管服务地址，从而提高系统连续性。数据库主从复制技术通过日志同步机制实现数据复制与冗余备份[8]，能够在主库故障时为数据恢复提供支撑。将 Keepalived 与 MariaDB 主从复制结合使用，分别解决 Zabbix Server 单点故障和监控数据可靠存储问题，为监控平台的高可用运行提供保障。

2.4 告警通知与可视化技术

告警通知由钉钉机器人通过 Webhook 接口实现监控事件的即时推送，可在故障发生后快速通知运维人员，提高监控系统响应效率。Grafana 则是一种常用的数据可视化平台，支持将监控数据库中的关键指标以仪表盘形式进行展示，便于从整体上掌握系统运行状态[10]。

3 系统总体设计

3.1 系统总体架构设计

本系统总体架构由 Zabbix Server、Zabbix Proxy、Zabbix Agent、MariaDB 数据库、告警模块和可视化模块组成。其中，Zabbix Agent 部署在被监控主机上，负责采集 CPU、内存、磁盘、网络以及应用服务等运行指标；Zabbix Proxy 部署在不同业务区域或网络环境中，负责本地监控数据的汇聚、缓存与转发；中心 Zabbix Server 负责统一接收监控数据、完成触发器判断、事件处理与告警调度；MariaDB 数据库用于存储监控配置数据和历史监控数据；Zabbix Web 与 Grafana 则提供监控查看、状态分析和图形化展示能力。系统架构如图 1 所示。

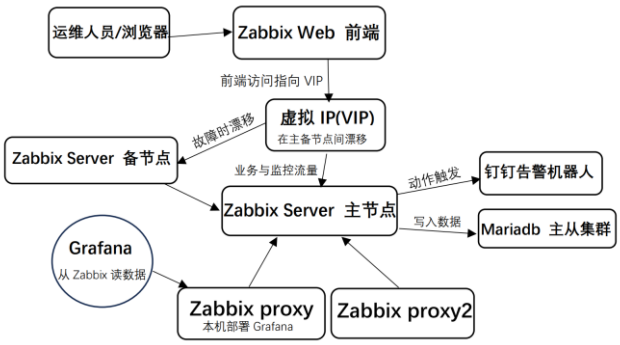

图 1 系统总体架构图

3.2 Zabbix Server 高可用设计

Zabbix Server 是监控平台的核心组件，承担监控数据接收、事件计算和告警触发等关键任务。采用双机热备方式部署两台 Zabbix Server，并借助 Keepalived 提供的虚拟 IP 作为系统对外访问入口，可避免单节点故障导致整个监控系统中断。运维人员访问 Web 前端以及 Agent、Proxy 上报监控数据时，均通过该虚拟 IP 与监控中心通信。

3.3 数据库主从架构设计

在数据层，系统采用 MariaDB 主从复制架构来确保数据库的可靠性和容灾能力，其中主库负责接收 Zabbix Server 写入的配置数据与历史监控数据，从库通过复制机制实时同步主库变更内容，实现数据冗余备份。

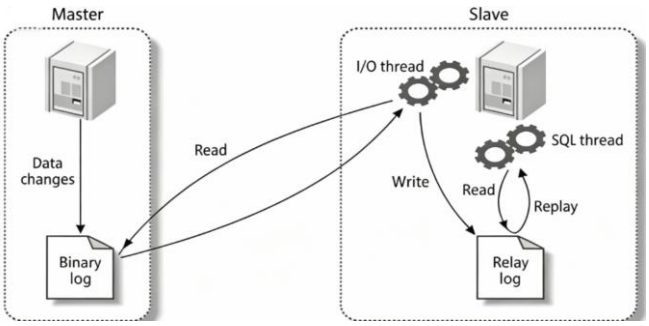

图 2 数据库主从架构原理图

3.4 Proxy 分布式采集设计

通过在不同业务区域部署 Zabbix Proxy 节点，构建“Zabbix Server + 多 Proxy + Agent”的分布式监控架构，解决中心节点高负载和数据采集稳定性低等问题。Proxy 节点负责就近接入所属区域内的被监控

主机，执行监控数据采集，并将采集结果先缓存在本地，再按批次上传至中心 Zabbix Server。该机制能够有效减轻中心节点的连接压力与数据处理压力，同时在链路短时异常时保留本地缓存数据，待网络恢复后继续补传，从而提高复杂网络环境下监控数据上报的完整性和可靠性。Proxy 分布式采集结构图见图 3。

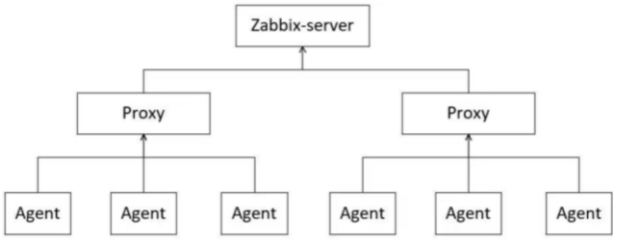

图 3 分布式采集结构图

3.5 告警与可视化设计

告警采用 Zabbix 自带告警机制的基础上集成钉钉告警通知功能，根据预设触发器和动作规则生成告警事件，并通过告警脚本调用钉钉机器人接口，将故障主机、异常内容和告警时间等信息及时推送给运维人员，实现异常状态的快速通知。在可视化方面，系统采用 Zabbix Web 前端与 Grafana 相结合的方式展示监控结果。Zabbix Web 用于完成主机管理、模板配置和基础图形查看，Grafana 则用于对关键性能指标、主机状态、数据库复制情况及 Proxy 运行情况进行统一可视化展示。通过图形化仪表盘更加直观地掌握系统整体运行状态，并结合告警信息对故障进行快速定位与分析。

4 系统实现

在完成系统总体设计之后，本文进一步对基于 Zabbix 的分布式监控系统进行实现。系统实现主要包括部署环境搭建、Zabbix Server 高可用部署、MariaDB 主从配置、Proxy 与 Agent 部署、钉钉告警接入以及 Grafana 可视化集成等内容。

4.1 系统概况与部署环境

根据系统总体设计，本文设计了一个基于 Zabbix 的分布式监控系统架构，采用“Zabbix Server + MariaDB + Zabbix Proxy + Zabbix Agent”的分布式部署方式，并在服务端引入 Keepalived 构建主备高可用架构。

系统整体运行于虚拟化环境之上，统一部署在 CentOS 7 x86_64 平台中，平台共部署 6 台核心节点，并统一采用静态 IP 进行网络规划。192.168.164.233 作为主 Zabbix Server 承载 Web 前端服务；备 Zabbix Server 用 192.168.164.236；两台服务器通过 Keepalived 共享虚拟 IP 192.168.164.110，对外提供统一访问入口。

数据库层采用 MariaDB 主从结构，主库地址为 192.168.164.234，从库地址为 192.168.164.210，用于监控配置数据与历史数据的集中存储及冗余备份；192.168.164.235 和 192.168.164.237 则分别作为两个 Zabbix Proxy 节点，负责各区域监控数据的本地采集、缓存与转发。

4.2 Zabbix 服务器与数据库部署

为实现监控平台核心服务的稳定运行，本模块完成了 Zabbix Server 与 MariaDB 数据库的基础部署。Zabbix Server 负责监控数据的接收、处理与管理，MariaDB 负责监控配置数据和历史监控数据的存储。通过完成 Zabbix 软件源配置、数据库安装初始化以及 Server 端数据库连接配置，为后续高可用切换和数据库主从复制提供基础运行环境。

（1） Zabbix Server 和数据库部署步骤如下

①安装 Zabbix 官方 Yum 源和相关组件

yum install -y https://repo.zabbix.com/zabbix/5.0/rhel/7/x86_64/zabbix-release-5.0-1.el7.noarch.rpm

yum -y install zabbix-server-mysql zabbix-agent zabbix-get zabbix-sender

② 数据库安装与初始化

#编辑源文件

vim /etc/yum.repos.d/mariadb.repo

[mariadb]

name = MariaDB

baseurl = https://mirrors.ustc.edu.cn/mariadb/yum/10.6/centos7-amd64

gpgkey=https://mirrors.ustc.edu.cn/mariadb/yum/RPM-GPG-KEY-MariaDB

gpgcheck=1

安装 mariadb

yum install MariaDB-server MariaDB-client

修改配置文件

vim /etc/my.cnf.d/server.cnf

[mysqld]

skip_name_resolve = ON

innodb_file_per_table = ON

innodb_buffer_pool_size = 256M

max_connections = 2000

log-bin = master-log

启动数据库

systemctl start mariadb

进入数据库进行数据初始化，创建 Zabbix 数据库与用户

mysql

MariaDB [(none)]> create database zabbix character set utf8 collate utf8_bin;

MariaDB [(none)]> grant all on zabbix.* to 'zabbix'@'%' identified by 'keer';

MariaDB [(none)]> flush privileges;

#数据表导入
zcat /usr/share/doc/zabbix-server-mysql-5.0.47/create.sql.gz | mysql -uzabbix -pkeer -h 192.168.164.234 zabbix

③在两台 Zabbix Server 节点配置服务端参数，使其统一连接至 MariaDB 主库。

#备节点仅将 SourceIP 改为 192.168.164.236，其余保持一致
vim /etc/zabbix/zabbix_server.conf
ListenPort=10051
SourceIP=192.168.164.233
DBHost=192.168.164.234
DBName=zabbix
DBUser=zabbix
DBPassword=keer
DBPort=3306
#开启服务
systemctl start zabbix-server.service zabbix-agent

④在 zabbix-server 端配置 web GUI
vim /etc/yum.repos.d/zabbix.repo
[zabbix-frontend] # 将这个软件源设置为启动
enabled=1
yum install centos-release-scl
yum install zabbix-web-mysql-scl zabbix-nginx-conf-scl
vim /etc/opt/rh/rh-nginx116/nginx/nginx.conf
#将配置文件中的 server 模块删除掉包括 server{}
server {
...
} # 删除
修改用户和时区
vim /etc/opt/rh/rh-php72/php-fpm.d/zabbix.conf
listen.acl_users = apache,nginx
php_value[date.timezone] = Asia/Shanghai

⑤ 在初始部署阶段，可通过主节点地址 http://192.168.164.233/进入安装向导，按照提示填入数据库连接信息和管理员账户（默认 Admin/zabbix）。安装完成后登录 Zabbix 前端，进入“配置→主机”，按需添加监控主机和对应模板。

4.3 高可用与分布式配置

本模块采用两台 Zabbix Server 通过 Keepalived 共享虚拟 IP 192.168.164.110，对外提供统一访问入口；数据库层采用 MariaDB 主从结构，其中 192.168.164.234 为主库，192.168.164.210 为从库，用于实现监控数据的同步与冗余备份；同时部署两台 Proxy 节点 192.168.164.235 和 192.168.164.237，负责不同区域监控数据的本地采集、缓存与转发。实现系统能够在主节点异常时完成服务切换，在数据库层实现数据同步，提升复杂网络环境下监控数据上报的稳定性。

（1） Keepalived 高可用配置实现步骤如下

① 在两台 Zabbix Server 节点部署 Keepalived。
#在两台 zabbix-server 上
yum -y install ipvsadm keepalived

② 通过不同优先级实现主备切换；共享虚拟 IP 地址为 192.168.164.110/24。
#编辑配置文件
vim /etc/keepalived/keepalived.conf
vrrp_instance VI_1 {
    state MASTER   #备用机上改为 BACKUP
    interface ens33
    virtual_router_id 51
    priority 200 # 备机优先级设置为 100
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        192.168.164.110/24
    }
}

③ 完成后启动 Keepalived 服务：
systemctl enable keepalived
systemctl start keepalived

（2） 数据库主从复制实现步骤如下

① 主库开启二进制日志并设置唯一 server-id，从库配置独立的 server-id 与中继日志参数。
主库关键参数示例
vim /etc/my.cnf.d/server.cnf
[mysqld]
server-id=1
log-bin=master-log
从库关键参数示例
vim /etc/my.cnf.d/server.cnf
[mysqld]
server-id=2
relay-log=relay-log

② 在主库创建复制账户，在从库指定主库地址、复制用户和日志位点，建立主从同步关系。
主库创建复制用户
mysql
grant replication slave on *.* to 'repl'@'%' identified by 'repl123';
flush privileges;
show master status;
从库指定主库并启动复制
change master to
master_host='192.168.164.234',
master_user='repl',
master_password='repl123',
master_log_file='master-log.xxx',
master_log_pos=xxx;
start slave;
show slave status\G

（3） 分布式监控实现步骤如下

① 在两台 Proxy 节点上安装 Zabbix Proxy 与本地 MariaDB。

```
yum install -y
https://repo.zabbix.com/zabbix/5.0/rhel/7/x86_64/zabbix-release-5.0-1.el7.noarch.rpm
yum install -y zabbix-proxy-mysql MariaDB-server MariaDB-client
# 启动数据库
systemctl restart mariadb
```

② 分别创建代理数据库。

```
# 第二台 Proxy 节点采用相同方法完成安装与初始化，需将数据库名和节点地址调整为对应值。
mysql
create database zabbix_proxy character set utf8 collate utf8_bin;
grant all on zabbix_proxy.* to zabbix@'%' identified by 'zabbixproxy';
flush privileges;
# 导入代理数据库表结构
zcat /usr/share/doc/zabbix-proxy-mysql-5.0.47/schema.sql.gz | mysql -uzabbix -pzabbixproxy -h 192.168.164.235 zabbix_proxy
```

③ 编辑 Proxy 配置文件设置为主动模式，使通过虚拟 IP 与中心 Zabbix Server 通信。

```
vim /etc/zabbix/zabbix_proxy.conf
ProxyMode=0
Server=192.168.164.110
#与 Zabbix 前端中创建的 Proxy 名称对应
Hostname=proxy
#对应本地代理数据库信息
DBHost=192.168.164.235
DBName=zabbix_proxy
DBUser=zabbix
DBPassword=zabbixproxy
#启动 proxy 代理
systemctl start zabbix-proxy
systemctl enable zabbix-proxy
```

4.4 数据采集与处理模块实现

本模块完成了 Zabbix Agent 部署、主机接入、模板关联以及应用监控项配置。以 Zabbix Agent 作为主要采集组件，在被监控主机上采集 CPU、内存、磁盘、网络等基础资源指标，并通过模板和自定义监控项扩展 Nginx、PHP-FPM 等应用层数据采集能力。各主机将监控数据上报至中心 Zabbix Server 或对应 Proxy 节点，由 Server 统一完成数据处理、状态判断和历史数据入库，形成完整的监控数据采集与处理链路。

（1） 数据采集与处理模块实现步骤如下

① 在被监控主机上安装 Zabbix Agent，并配置监控中心地址。

```
yum install -y zabbix-agent zabbix-sender
# 修改 Agent 配置文件
vim /etc/zabbix/zabbix_agentd.conf
Server=192.168.164.233,192.168.164.236,192.168.164.110
ServerActive=192.168.164.110:10051
Hostname=mariadb
# 不同主机按实际情况设置对应 Hostname
# 启动并设置开机自启
systemctl start zabbix-agent
systemctl enable zabbix-agent
```

② 在 Zabbix Web 前端添加主机并关联监控模板，实现基础监控数据自动采集。在前端“配置→主机”中创建对应主机记录，填写 Agent 接口地址并关联 Template OS Linux 等基础模板后，系统即可自动采集 CPU、内存、磁盘、网络等常用指标。模板化配置方式能够减少逐项定义监控项的工作量，提高监控对象接入效率和配置一致性。

③ 针对应用服务配置监控项，扩展数据采集范围。在基础资源监控之外，系统进一步对 Nginx、MySQL、PHP-FPM 等应用服务进行监控。对于常见服务，可直接关联 Zabbix 内置模板；对于业务相关指标，则可通过用户参数或自定义脚本方式扩展监控项，使 Agent 能够按设定周期采集更细粒度的应用运行数据。

4.5 告警与可视化模块实现

告警与可视化模块在 Zabbix 自带告警机制基础上，引入钉钉群机器人作为消息通知媒介，通过告警脚本将故障主机、告警等级、异常内容和恢复状态等信息实时推送给运维人员；同时，在可视化方面引入 Grafana，并通过 Zabbix 数据源插件实现对监控数据的统一展示。

（1）告警与可视化模块实现步骤如下

① 在钉钉中创建告警群和自定义机器人，获取 Webhook 地址及加签密钥；随后在 Zabbix 服务器 /usr/lib/zabbix/alertscripts/ 目录下编写 dingding.py 告警脚本。通过接收 Zabbix 传入的告警参数，按钉钉机器人要求生成签名，并以 Markdown 消息格式通过 HTTP POST 方式发送到钉钉群，实现故障与恢复信息的自动推送，同时将发送结果写入本地日志文件。

```
import requests,json,sys,time,hmac,hashlib,base64,urllib.parse
def msg(user, content):
    secret = 'SEC20602dd12fb89ec72a548de5ac274e2159051a9f2ea88bccbdd29a5131f9d6e2'
    timestamp = str(round(time.time() * 1000))
    secret_enc = secret.encode('utf-8')
    string_to_sign = '{}\n{}'.format(timestamp, secret)
    string_to_sign_enc = string_to_sign.encode('utf-8')
    hmac_code = hmac.new(secret_enc, string_to_sign_enc, digestmod=hashlib.sha256).digest()
```

```
        sign = urllib.parse.quote_plus(base64.b64encode(hmac_code))
        api_url = dingding_url + "&timestamp={}&sign={}".format(timestamp, sign)
        headers = {'Content-Type': 'application/json;charset=utf-8'}
        # 根据消息关键词设置标题
        if "告警" in content and "恢复" not in content:
            title = "服务告警，请尽快处理"
        elif "恢复" in content:
            title = "故障已恢复"
        elif "更新" in content:
            title = "故障处理有更新"
        else:
            title = "人工测试"
        # 组装 markdown 消息
        json_text = {
            "msgtype": "markdown",
            "markdown": {
                "title": title,
                "text": "@" + str(user) + "\n" + str(content)
            },
            "at": {
                "atMobiles": [str(user)],
                "isAtAll": False
            }
        }
        r = requests.post(api_url, data=json.dumps(json_text), headers=headers, timeout=10)
        print("HTTP:", r.status_code)
        print("RESP:", r.text)
        try:
            j = r.json()
            if j.get("errcode", -1) != 0:
                sys.exit(1)
        except Exception:
            sys.exit(1)
    if __name__ == '__main__':
        sent_to_user = sys.argv[1]
        input_message = sys.argv[2]
        msg(sent_to_user, input_message)
```

② 创建告警脚本日志文件，并将脚本赋予执行权限。

```
touch /var/log/zabbix/dingding.log
chown zabbix.zabbix /var/log/zabbix/dingding.log
chmod +x /usr/lib/zabbix/alertscripts/dingding.py
```

③ 在 Zabbix 前端“管理→媒体类型”中新建“钉钉”媒体类型，选择“脚本”作为媒介类型，脚本名称设置为 dingding.py，并配置 {ALERT.SENDTO}、{ALERT.MESSAGE} 等参数，使 Zabbix 在触发告警时能够将接收目标和告警内容传递给脚本。

④ 给对应用户添加钉钉媒体类型，并在“配置→动作”中创建告警动作与恢复动作。当触发器满足设定条件时，Zabbix 会自动调用告警脚本，将主机 IP、告警组、告警时间、告警等级、异常类别和当前状态等内容推送到钉钉群；当故障恢复后，系统还会继续发送恢复通知

⑤ 部署 Grafana 增强图形化展示能力。

```
# 配置 Grafana 官方 Yum 源
cat <<EOF > /etc/yum.repos.d/grafana.repo
[grafana]
name=grafana
baseurl=https://packages.grafana.com/oss/rpm
repo_gpgcheck=1
enabled=1
gpgcheck=1
gpgkey=https://packages.grafana.com/gpg.key
EOF
#部署并启动 Grafana
yum install -y grafana
systemctl enable grafana-server && systemctl start grafana-server
grafana-cli plugins install alexanderzobnin-zabbix-app
systemctl restart grafana-server
```

⑥ 在 Grafana Web 前端“Configuration→Data Sources”中添加 Zabbix 数据源，将接口地址设置为 http://192.168.164.233/api_jsonrpc.php，并填写有效的 Zabbix 账号信息，连接成功后即可创建仪表盘展示基础资源趋势。

4.6 系统性能测试与分析

（1）高可用故障切换测试

主节点正常运行且持有虚拟 IP，通过停止主节点 Keepalived 服务或模拟主节点宕机的方式制造故障，备节点短时间内自动接管虚拟 IP，测试结果见图 4。

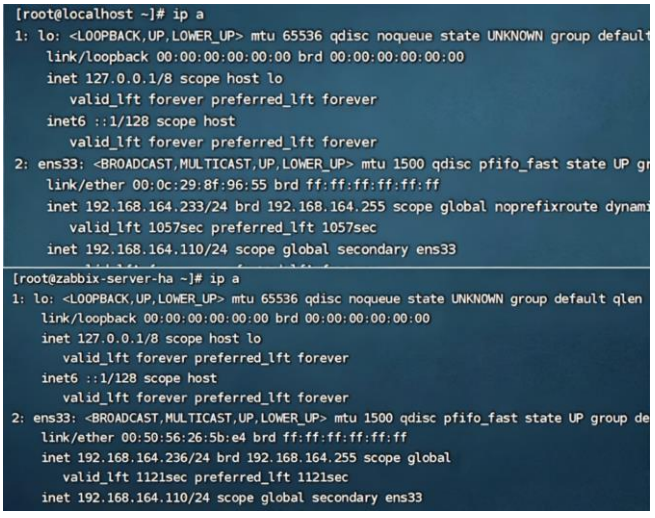

图 4 节点漂移状态图

（2）数据库主从同步验证

从库执行 SHOW SLAVE STATUS\G 后，复制状态见图 5。Slave_IO_Running 与 Slave_SQL_Running 均为 Yes，从库已成功建立主从复制链路，I/O 线程

与 SQL 线程运行正常；同时 Seconds_Behind_Master 为 0，说明当前主从同步延迟较小，主库产生的二进制日志已及时在从库完成重放。

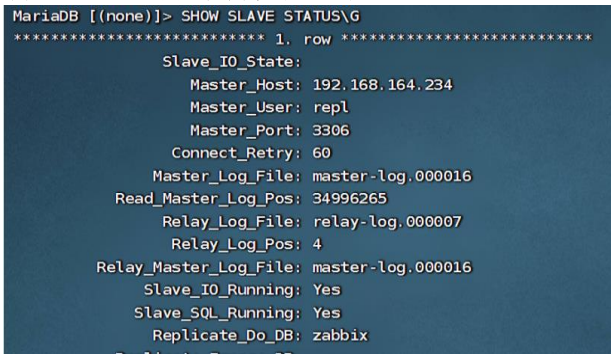

图 5 从库节点状态图

（3）告警触发与通知测试

在数据库主库上配置 ICMP Ping 监控项并执行 iptables -I INPUT -p icmp --icmp-type echo-request -j DROP 阻断 ICMP 回显请求，模拟主机网络异常场景。满足触发条件后，Zabbix 前端生成问题事件并在短时间内向目标用户发送告警通知，如图 6 所示，说明系统告警链路运行正常，能够实现异常状态的及时告警。

图 6 告警通知示意图

（4）Grafana 可视化展示测试

如图 7 所示，Grafana 准确获取并展示与 Zabbix 前端一致的监控数据且数据更新无明显延迟，其多指标展示与模板化筛选功能提升了跨主机、跨指标的对比分析能力。

图 7 Grafana 仪表盘示例图

5 结束语

本方案基于 Zabbix 设计并实现了一套分布式高可用监控系统，通过 Zabbix Server 主备部署、MariaDB 主从复制、Proxy 分布式采集以及钉钉告警和 Grafana 可视化，实现了监控数据的集中管理与稳定运行。实践表明，该系统能够提高监控平台的可靠性和运维效率，可为企业监控系统建设提供参考。

参 考 文 献

[1] 陈豪. 基于Zabbix构建企业监控系统的设计与优化[J]. 信息与电脑, 2025, 37(08):114-117.

[2] 冯永照, 邓伟杰, 宋志君, 等. 基于Zabbix和SNMP的网络磁盘监控系统的研究与应用[J]. 内蒙古石油化工, 2025, 51(09):21-25.

[3] 朱建彬. 基于Keepalived的MySQL双主复制高可用集群架构研究与实现[J]. 电脑知识与技术, 2025, 21(25):82-84+87. DOI:10.14004/j.cnki.ckt.2025.1263.

[4] 陈玲, 基于Zabbix+Grafana的信息监测告警系统开发及应用. 上海市, 上海市地震局, 2024-04-19.

[5] 邢峥嵘, 薛晶, 李存. Grafana在智能运维管理中的应用[J]. 包钢科技, 2025, 51(02):85-88. DOI:10.13647/j.cnki.btgkj.2025.02.017.

[6] 孔令, 袁黎晖, 刘毅. 基于Zabbix平台的高校数据中心告警推送方式实现和应用[J]. 通讯世界, 2024, 31(05):91-93.

[7] 谷文华, 王帅, 钱飞. 基于Keepalived的高可用研究与设计[J]. 网络安全技术与应用, 2025, (01):26-29.

[8] 许彪, 王湘渝, 朱爱梅. 基于Mariadb Galera的高可用数据库集群技术[J]. 信息技术与信息化, 2021, (10):25-27.

[9] 杨澎涛, 范永合, 孙友凯, 等. 基于Zabbix的告警推送技术[J]. 电子技术与软件工程, 2019, (18):59-61. DOI:10.20109/j.cnki.etse.2019.18.035.

[10] 王路. 基于Zabbix与Grafana的制播一体化运维监控系统构建[J]. 西部广播电视, 2025, 46(03):178-182.