# MySQL 限制访问的IP范围配置指南

本文档详细介绍如何在 MySQL 数据库中配置 IP 访问限制，实现网络层面的访问控制，符合等保三级对网络访问控制的安全要求。

## 目录

[1. 概述](#Xdf370bae5c8c51c1cf31c7ca53fd3c82d4d170d)

[2. MySQL 用户主机限制原理](#Xef55f76d43d21f9a12a9fc08976b05aada383cb)

[3. 环境准备](#X84df04c7ecc087d85a9f172704441b39251d345)

[4. 基础 IP 限制配置](#X849b8ff2d4cf43e6061780bed07ea0d21689ca0)

[5. 高级访问控制配置](#X99306743791d89149c5e23bba75dc68ac3b1d5b)

[6. 防火墙级别的访问控制](#Xd764219e52b2d36c3892c33228a428c50532ffa)

[7. MySQL 配置文件访问控制](#X7eb7f3739402245973767e23578d06da0648deb)

[8. 访问控制验证与测试](#Xb48c29d0d68889d0560e435c1aa8905d09a4c29)

## 1. 概述

### 1.1 访问控制的重要性

根据等保三级要求，数据库系统必须实现严格的网络访问控制，包括：

**网络层访问控制**：通过 IP 地址限制数据库连接

**用户级访问控制**：为不同用户设置不同的主机访问范围

**端口级访问控制**：限制数据库服务端口的访问

**地理位置控制**：基于网络位置的访问策略

### 1.2 多层防护体系

外部网络 → 防火墙/网关 → 操作系统防火墙 → MySQL bind-address → MySQL 用户主机限制 → 数据库连接

## 2. MySQL 用户主机限制原理

### 2.1 MySQL 用户账户格式

MySQL 中的用户账户由用户名和主机名两部分组成：'username'@'hostname'

-- 用户账户格式示例  
'root'@'localhost' -- 只允许本地连接  
'user'@'192.168.1.100' -- 只允许特定IP连接  
'app'@'192.168.1.%' -- 允许网段连接  
'admin'@'%.example.com' -- 允许特定域名连接  
'guest'@'%' -- 允许任何主机连接（不推荐）

### 2.2 主机名通配符

| 通配符 | 含义 | 示例 | 说明 |
| --- | --- | --- | --- |
| % | 匹配任意字符序列 | 192.168.1.% | 匹配 192.168.1.0/24 网段 |
| \_ | 匹配单个字符 | 192.168.1.1\_ | 匹配如 192.168.1.10-19 |
| localhost | 本地连接 | localhost | 仅本机连接 |
| 127.0.0.1 | 本地IP | 127.0.0.1 | 本机回环地址 |

## 3. 环境准备

### 3.1 前提条件

MySQL 5.7 或更高版本

具有 root 权限的 MySQL 管理员账户

网络规划文档（IP 地址段分配）

防火墙管理权限（如果需要）

### 3.2 网络环境规划示例

# 网络环境示例  
# 管理网段：192.168.10.0/24  
# 应用服务器网段：192.168.20.0/24  
# 办公网段：192.168.30.0/24  
# 数据库服务器：192.168.10.100

### 3.3 连接到 MySQL

# 使用 root 账户连接  
mysql -u root -p -h localhost

## 4. 基础 IP 限制配置

### 4.1 查看现有用户和主机限制

-- 查看所有用户及其主机限制  
SELECT user, host, authentication\_string FROM mysql.user ORDER BY user, host;  
  
-- 查看特定用户的主机限制  
SELECT user, host FROM mysql.user WHERE user = 'root';  
  
-- 查看当前连接信息  
SELECT USER(), CURRENT\_USER(), CONNECTION\_ID();

### 4.2 创建限制特定IP的用户

-- 创建只能从特定IP连接的用户  
CREATE USER 'app\_user'@'192.168.20.10' IDENTIFIED BY 'AppUser@2024!';  
GRANT SELECT, INSERT, UPDATE, DELETE ON business\_db.\* TO 'app\_user'@'192.168.20.10';  
  
-- 创建只能从特定网段连接的用户  
CREATE USER 'web\_user'@'192.168.20.%' IDENTIFIED BY 'WebUser@2024!';  
GRANT SELECT, INSERT, UPDATE, DELETE ON web\_db.\* TO 'web\_user'@'192.168.20.%';  
  
-- 创建管理员用户（限制管理网段）  
CREATE USER 'db\_admin'@'192.168.10.%' IDENTIFIED BY 'DbAdmin@2024!';  
GRANT ALL PRIVILEGES ON \*.\* TO 'db\_admin'@'192.168.10.%' WITH GRANT OPTION;

### 4.3 修改现有用户的主机限制

-- 重命名用户以更改主机限制  
-- 注意：这实际上是创建新用户并删除旧用户  
  
-- 1. 先查看现有用户权限  
SHOW GRANTS FOR 'existing\_user'@'%';  
  
-- 2. 创建具有相同权限但限制主机的新用户  
CREATE USER 'existing\_user'@'192.168.20.%' IDENTIFIED BY 'NewPassword@2024!';  
  
-- 3. 复制权限（根据SHOW GRANTS的结果）  
GRANT SELECT, INSERT, UPDATE, DELETE ON business\_db.\* TO 'existing\_user'@'192.168.20.%';  
  
-- 4. 删除原来的用户  
DROP USER 'existing\_user'@'%';  
  
-- 5. 刷新权限  
FLUSH PRIVILEGES;

### 4.4 删除不安全的用户

-- 查找允许任意主机连接的用户  
SELECT user, host FROM mysql.user WHERE host = '%';  
  
-- 删除不必要的通配符用户  
DROP USER 'username'@'%';  
  
-- 删除空用户名的账户  
DELETE FROM mysql.user WHERE user = '';  
  
-- 删除测试数据库相关用户  
DROP USER IF EXISTS ''@'localhost';  
DROP USER IF EXISTS ''@'hostname';  
  
-- 刷新权限  
FLUSH PRIVILEGES;

## 5. 高级访问控制配置

### 5.1 基于角色的IP限制

-- 数据库管理员（仅限管理网段）  
CREATE USER 'dba\_primary'@'192.168.10.101' IDENTIFIED BY 'DBA1@2024!';  
CREATE USER 'dba\_backup'@'192.168.10.102' IDENTIFIED BY 'DBA2@2024!';  
GRANT ALL PRIVILEGES ON \*.\* TO 'dba\_primary'@'192.168.10.101' WITH GRANT OPTION;  
GRANT ALL PRIVILEGES ON \*.\* TO 'dba\_backup'@'192.168.10.102' WITH GRANT OPTION;  
  
-- 应用服务器用户（限制应用网段）  
CREATE USER 'app\_web'@'192.168.20.%' IDENTIFIED BY 'AppWeb@2024!';  
CREATE USER 'app\_api'@'192.168.21.%' IDENTIFIED BY 'AppAPI@2024!';  
GRANT SELECT, INSERT, UPDATE, DELETE ON app\_db.\* TO 'app\_web'@'192.168.20.%';  
GRANT SELECT, INSERT, UPDATE, DELETE ON api\_db.\* TO 'app\_api'@'192.168.21.%';  
  
-- 只读用户（限制办公网段）  
CREATE USER 'readonly\_report'@'192.168.30.%' IDENTIFIED BY 'ReadOnly@2024!';  
GRANT SELECT ON report\_db.\* TO 'readonly\_report'@'192.168.30.%';  
  
-- 备份用户（仅限备份服务器）  
CREATE USER 'backup\_user'@'192.168.10.200' IDENTIFIED BY 'Backup@2024!';  
GRANT SELECT, LOCK TABLES, RELOAD, REPLICATION CLIENT ON \*.\* TO 'backup\_user'@'192.168.10.200';

### 5.2 动态IP白名单管理

-- 创建IP白名单表  
CREATE TABLE ip\_whitelist (  
 id INT AUTO\_INCREMENT PRIMARY KEY,  
 ip\_address VARCHAR(45) NOT NULL,  
 ip\_range VARCHAR(45),  
 description VARCHAR(255),  
 created\_by VARCHAR(32),  
 created\_time TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,  
 is\_active BOOLEAN DEFAULT TRUE,  
 expires\_at TIMESTAMP NULL,  
 INDEX idx\_ip (ip\_address),  
 INDEX idx\_active (is\_active)  
);  
  
-- 插入白名单IP  
INSERT INTO ip\_whitelist (ip\_address, ip\_range, description, created\_by) VALUES  
('192.168.10.0/24', '192.168.10.%', '管理网段', 'admin'),  
('192.168.20.0/24', '192.168.20.%', '应用服务器网段', 'admin'),  
('192.168.30.0/24', '192.168.30.%', '办公网段', 'admin');  
  
-- 查看白名单  
SELECT \* FROM ip\_whitelist WHERE is\_active = TRUE;

## 6. 防火墙级别的访问控制

### 6.1 Linux iptables 配置

#!/bin/bash  
# MySQL 端口访问控制脚本  
  
# 清除现有规则  
iptables -F  
iptables -X  
iptables -t nat -F  
iptables -t nat -X  
  
# 设置默认策略  
iptables -P INPUT DROP  
iptables -P FORWARD DROP  
iptables -P OUTPUT ACCEPT  
  
# 允许本地回环  
iptables -A INPUT -i lo -j ACCEPT  
iptables -A OUTPUT -o lo -j ACCEPT  
  
# 允许已建立的连接  
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT  
  
# 允许管理网段访问MySQL (3306端口)  
iptables -A INPUT -p tcp -s 192.168.10.0/24 --dport 3306 -j ACCEPT  
  
# 允许应用服务器网段访问MySQL  
iptables -A INPUT -p tcp -s 192.168.20.0/24 --dport 3306 -j ACCEPT  
  
# 允许办公网段的只读访问（可以结合MySQL用户权限）  
iptables -A INPUT -p tcp -s 192.168.30.0/24 --dport 3306 -j ACCEPT  
  
# 允许SSH访问（管理用）  
iptables -A INPUT -p tcp -s 192.168.10.0/24 --dport 22 -j ACCEPT  
  
# 记录被拒绝的连接  
iptables -A INPUT -p tcp --dport 3306 -j LOG --log-prefix "MySQL Access Denied: "  
iptables -A INPUT -p tcp --dport 3306 -j DROP  
  
echo "Firewall rules applied successfully"

### 6.2 Ubuntu UFW 配置

# 启用UFW  
sudo ufw enable  
  
# 设置默认策略  
sudo ufw default deny incoming  
sudo ufw default allow outgoing  
  
# 允许SSH（管理用）  
sudo ufw allow from 192.168.10.0/24 to any port 22  
  
# 允许MySQL访问  
sudo ufw allow from 192.168.10.0/24 to any port 3306 comment "Admin Network"  
sudo ufw allow from 192.168.20.0/24 to any port 3306 comment "App Servers"  
sudo ufw allow from 192.168.30.0/24 to any port 3306 comment "Office Network"  
  
# 查看规则  
sudo ufw status numbered  
  
# 启用日志  
sudo ufw logging on

## 7. MySQL 配置文件访问控制

### 7.1 bind-address 配置

# /etc/mysql/mysql.conf.d/mysqld.cnf (Ubuntu/Debian)  
# /etc/my.cnf (CentOS/RHEL)  
  
[mysqld]  
# 绑定到特定IP地址，而不是所有接口  
# bind-address = 127.0.0.1 # 仅本地访问  
bind-address = 192.168.10.100 # 绑定到特定IP  
  
# 或者绑定到多个IP（MySQL 8.0.13+）  
# bind-address = 192.168.10.100,127.0.0.1  
  
# 端口配置  
port = 3306  
  
# 禁用网络连接（仅socket连接）  
# skip-networking  
  
# 连接限制  
max\_connections = 200  
max\_user\_connections = 50  
  
# 超时设置  
wait\_timeout = 3600  
interactive\_timeout = 3600  
connect\_timeout = 10

### 7.2 SSL/TLS 配置

[mysqld]  
# SSL配置  
ssl-ca=/etc/mysql/ssl/ca-cert.pem  
ssl-cert=/etc/mysql/ssl/server-cert.pem  
ssl-key=/etc/mysql/ssl/server-key.pem  
  
# 要求SSL连接  
require\_secure\_transport = ON  
  
# TLS版本限制  
tls\_version = TLSv1.2,TLSv1.3

-- 为用户要求SSL连接  
CREATE USER 'secure\_user'@'192.168.20.%' IDENTIFIED BY 'SecureUser@2024!' REQUIRE SSL;  
GRANT SELECT, INSERT, UPDATE, DELETE ON secure\_db.\* TO 'secure\_user'@'192.168.20.%';  
  
-- 要求特定的SSL证书  
CREATE USER 'cert\_user'@'192.168.20.%' IDENTIFIED BY 'CertUser@2024!' REQUIRE X509;  
  
-- 要求特定CA签发的证书  
CREATE USER 'ca\_user'@'192.168.20.%' IDENTIFIED BY 'CAUser@2024!' REQUIRE ISSUER '/C=CN/ST=Beijing/L=Beijing/O=Company/CN=MySQL-CA';

## 8. 访问控制验证与测试

### 8.1 本地连接测试

# 测试本地连接  
mysql -u root -p -h localhost  
mysql -u root -p -h 127.0.0.1  
mysql -u root -p --socket=/var/run/mysqld/mysqld.sock

### 8.2 远程连接测试

# 从允许的IP测试连接  
mysql -u app\_user -p -h 192.168.10.100 -P 3306  
  
# 测试SSL连接  
mysql -u secure\_user -p -h 192.168.10.100 --ssl-mode=REQUIRED  
  
# 验证连接来源  
mysql -u test\_user -p -h 192.168.10.100 -e "SELECT USER(), CURRENT\_USER(), CONNECTION\_ID();"

### 8.3 连接验证脚本

#!/bin/bash  
# MySQL连接测试脚本  
  
MYSQL\_HOST="192.168.10.100"  
MYSQL\_PORT="3306"  
  
# 测试用户列表  
declare -A test\_users=(  
 ["app\_user@192.168.20.10"]="AppUser@2024!"  
 ["admin\_user@192.168.10.50"]="AdminUser@2024!"  
 ["readonly@192.168.30.100"]="ReadOnly@2024!"  
)  
  
echo "Testing MySQL connection restrictions..."  
echo "============================================"  
  
for user\_host in "${!test\_users[@]}"; do  
 username=$(echo $user\_host | cut -d'@' -f1)  
 source\_ip=$(echo $user\_host | cut -d'@' -f2)  
 password=${test\_users[$user\_host]}  
   
 echo "Testing: $username from $source\_ip"  
   
 # 注意：实际测试需要从对应的IP地址进行  
 timeout 10 mysql -u "$username" -p"$password" -h "$MYSQL\_HOST" -P "$MYSQL\_PORT" -e "SELECT 'Connection successful' as Result, USER(), CONNECTION\_ID();" 2>/dev/null  
   
 if [ $? -eq 0 ]; then  
 echo "✓ Connection successful for $username"  
 else  
 echo "✗ Connection failed for $username"  
 fi  
 echo "---"  
done  
  
echo "Test completed."

### 8.4 权限验证查询

-- 查看当前连接的详细信息  
SELECT   
 Id,  
 User,  
 Host,  
 db,  
 Command,  
 Time,  
 State,  
 Info  
FROM information\_schema.PROCESSLIST;  
  
-- 查看用户权限矩阵  
SELECT   
 GRANTEE,  
 TABLE\_SCHEMA,  
 PRIVILEGE\_TYPE,  
 IS\_GRANTABLE  
FROM information\_schema.USER\_PRIVILEGES   
UNION ALL  
SELECT   
 GRANTEE,  
 TABLE\_SCHEMA,  
 PRIVILEGE\_TYPE,  
 IS\_GRANTABLE  
FROM information\_schema.SCHEMA\_PRIVILEGES  
ORDER BY GRANTEE, TABLE\_SCHEMA, PRIVILEGE\_TYPE;  
  
-- 检查特定用户的连接权限  
SHOW GRANTS FOR 'app\_user'@'192.168.20.%';  
  
-- 查看失败的连接尝试（需要启用日志）  
SELECT \* FROM mysql.general\_log WHERE command\_type = 'Connect' AND argument LIKE '%Access denied%';

## 常见问题与最佳实践

### 9.1 常见问题排除

**问题1：连接被拒绝**

# 检查MySQL服务状态  
sudo systemctl status mysql  
  
# 检查端口监听  
sudo netstat -tlnp | grep 3306  
  
# 检查防火墙状态  
sudo ufw status  
# 或  
sudo iptables -L

**问题2：权限配置错误**

-- 检查用户存在性  
SELECT user, host FROM mysql.user WHERE user = 'your\_username';  
  
-- 检查权限分配  
SHOW GRANTS FOR 'your\_username'@'your\_host';  
  
-- 检查当前连接  
SELECT USER(), CURRENT\_USER();

### 9.2 最佳实践建议

**权限最小化原则**

仅分配完成工作所需的最小权限

定期审查和清理不必要的权限

避免使用通配符权限

**网络分段**

将不同类型的服务器放在不同的网络段

使用VLAN进行网络隔离

实施网络访问控制列表（ACL）

**监控和审计**

启用详细的审计日志

监控异常连接行为

设置告警机制

**定期维护**

定期更新MySQL版本

定期审查用户权限

定期测试访问控制规则

### 9.3 安全注意事项

**密码安全**：使用强密码策略，定期更换密码

**证书管理**：妥善管理SSL证书，及时更新过期证书

**日志分析**：定期分析访问日志，识别异常行为

**备份策略**：确保配置变更有备份和回滚方案

**应急响应**：制定安全事件应急响应预案

**注意事项：**

本指南中的IP地址和密码仅为示例，实际使用时请使用您环境的真实配置

在生产环境实施前，请在测试环境充分验证所有配置

确保遵循组织的安全策略和合规要求

定期审查和更新访问控制配置

建立完善的变更管理流程，确保配置变更的可追溯性 ```