

服务器实时监控系統用户使用手册

版本: v3.0.0

适用场景: 中小型服务器集群监控、云服务器运维、监控大屏展示

更新日期: 2025 年 11 月 25 日

目录

- [系统简介](#)
- [前置条件](#)
- [快速部署流程](#)
- [系统功能使用](#)
- [常见问题排查](#)
- [运维管理命令](#)
- [注意事项](#)

一、系统简介

1.1 核心功能

- 多节点监控:** 同时监控多台 Linux 服务器, 采集 CPU、内存、磁盘、进程数、系统负载等核心指标。
- 实时推送:** 基于 WebSocket 技术, 30 秒自动采集数据, 前端无刷新实时更新。
- 智能告警:** 支持自定义告警阈值, 超阈值时触发分级告警 (warning/critical), 支持告警标记与批量解决。
- 无侵入部署:** 通过 SSH 协议采集数据, 被监控节点无需安装额外 Agent。
- 数据持久化:** SQLite 数据库存储监控指标、告警记录, 支持历史数据查询与清理。
- 容器化部署:** Docker 打包应用及依赖, Docker Compose 一键编排, 支持故障自动重启。

1.2 技术架构

- 监控中心: Ubuntu 22.04 LTS (容器化部署后端 + 前端 Nginx)
- 被监控节点: Ubuntu 20.04+/CentOS 7+ (需开启 SSH 服务)
- 核心组件: FastAPI (后端)、Nginx (前端代理)、WebSocket (实时推送)、SQLite (存储)

二、前置条件

2.1 硬件要求

设备类型	配置要求	用途
监控中心服务器	2 核 CPU、4G 内存、50G 磁盘空间以上	运行容器化服务、存储数据
被监控节点	1 核 CPU、1G 内存、20G 磁盘空间以上	提供 SSH 访问，被采集指标

2.2 软件要求

软件名称	版本要求	说明
Docker	20.10+	容器化运行环境
Docker Compose	2.0+	多服务编排工具
Git	2.0+	拉取项目代码（可选，手动上传代码可跳过）
SSH 服务	OpenSSH 7.0+	被监控节点需开启 SSH（默认 22 端口）
浏览器	Chrome 90+/Edge 90+	访问前端监控页面

2.3 网络要求

- 监控中心与被监控节点网络互通，无防火墙拦截以下端口：
- 22 端口：SSH 协议（被监控节点，用于数据采集）
- 8000 端口：后端服务（监控中心，API+WebSocket）
- 8080 端口：前端访问（监控中心，Nginx 代理）
- 被监控节点需开启 SSH 服务，且监控中心能通过 IP + 端口访问。

三、快速部署流程

3.1 步骤 1：登录监控中心服务器

使用 SSH 工具（如 Xshell、Putty）登录监控中心服务器，切换至 root 用户（或具备 sudo 权限的用户）：

```
sudo -i # 切换root用户（可选，避免后续频繁输入sudo）
```

3.2 步骤 2：安装必备工具

2.2.1 安装 Docker 与 Docker Compose

```
\# 更新系统软件包

apt-get update && apt-get upgrade -y
```

```
\# 安装Docker

apt-get install -y docker.io docker-compose-plugin

\# 启动Docker并设置开机自启

systemctl start docker && systemctl enable docker

\# 给当前用户添加Docker权限（避免执行docker命令需sudo）

usermod -aG docker $USER && newgrp docker
```

2.2.2 安装 Git（可选）

若通过 Git 拉取代码，需安装 Git：

```
apt-get install -y git
```

3.3 步骤 3：获取项目代码

方式 1：Git 拉取（推荐）

```
\# 创建项目目录

mkdir -p /opt/server-monitor && cd /opt/server-monitor

\# 拉取项目代码（替换为实际Git仓库地址）

git clone https://github.com/xxx/server-monitor.git .
```

方式 2：手动上传代码

1. 从本地将项目压缩包（`server-monitor.zip`）上传至监控中心服务器 `/opt` 目录。
2. 解压代码：

```
mkdir -p /opt/server-monitor && cd /opt/server-monitor

unzip /opt/server-monitor.zip # 若为tar包，使用tar -zxvf 文件名.tar.gz
```

3.4 步骤 4：配置权限（关键）

确保项目目录及数据目录具备读写权限：

```
\# 进入项目根目录

cd /opt/server-monitor

\# 设置目录权限（Docker容器需读写权限）

chmod -R 755 ./backend ./frontend ./nginx ./data ./logs

chmod +x ./deploy.sh ./backend/startup.sh
```

3.5 步骤 5：配置 Docker 加速器（可选，国内环境推荐）

国内环境拉取 Docker 镜像较慢，配置加速器提升速度：

```
\# 创建Docker配置文件

mkdir -p /etc/docker && tee /etc/docker/daemon.json <<-'EOF'

{

  "registry-mirrors": \["https://docker.1panel.live", "https://hub-
mirror.c.163.com"]

}

EOF

\# 重启Docker使配置生效

systemctl daemon-reload && systemctl restart docker

\# 验证配置（出现镜像源地址即成功）

docker info | grep -A 3 "Registry Mirrors"
```

3.6 步骤 6：一键部署服务

```
\# 进入项目根目录

cd /opt/server-monitor

\# 执行一键部署脚本

./deploy.sh
```

部署脚本执行说明

脚本会自动完成以下操作：

1. 检查 Docker、Docker Compose 是否安装。
2. 创建数据目录（`./data`）、日志目录（`./logs`）。
3. 复制前端文件到 Nginx 发布目录。
4. 构建 Docker 镜像（后端 + Nginx）。
5. 启动容器服务，等待后端初始化（约 30 秒）。

部署成功标识

脚本输出以下内容即表示部署成功：

```
\=====

部署成功！

访问地址：

监控大屏：http://192.168.211.149:8080/monitoring.html

数据管理：http://192.168.211.149:8080/data-management.html

API文档：http://192.168.211.149:8080/api/docs

\=====
```

3.7 步骤 7：验证服务状态

```
\# 查看容器运行状态（需显示2个容器"up"状态）

docker compose ps

\# 查看后端服务日志（验证是否启动成功）

docker compose logs -f server-monitor

\# 查看Nginx日志（验证前端是否正常）

docker compose logs -f nginx
```

四、系统功能使用

4.1 访问系统页面

打开浏览器，输入以下地址（替换为监控中心服务器 IP）：

功能页面	访问地址	说明
监控大屏	http:// 监控中心 IP:8080/monitoring.html	实时展示服务器指标与告警
数据管理	http:// 监控中心 IP:8080/data-management.html	管理服务器、清理数据（前端完善后可用）
API 文档	http:// 监控中心 IP:8080/api/docs	测试 API 接口、添加服务器

五、常见问题排查

5.1 部署失败

问题 1：Docker 权限不足

- 现象：执行 `./`deploy.sh` 时提示「Permission denied」。
- 解决：执行 `usermod -aG docker $USER && newgrp docker`，重新登录服务器后重试。

问题 2：端口被占用

- 现象：启动容器时提示「port is already allocated」。
- 解决：
 1. 查看占用端口的进程：`netstat -tulpn | grep 8080`（或 8000）。
 2. 停止占用进程：`kill -9 进程ID`，或修改 `docker-compose.yaml` 中的端口映射（如将 8080 改为 8081）。

问题 3：镜像拉取缓慢 / 失败

- 现象：部署脚本卡在「Building server-monitor」。
- 解决：配置 Docker 加速器（参考 3.5 节），或手动拉取基础镜像：

```
docker pull python:3.9-slim
```

```
docker pull nginx:alpine
```

5.2 服务器添加失败

问题 1: SSH 连接超时

- 现象：添加服务器时返回 `connection_status: offline`，日志提示「connect timeout」。
- 解决：
 1. 验证监控中心能否 ping 通被监控节点： `ping 被监控节点IP`。
 2. 检查被监控节点防火墙是否开放 22 端口： `ufw allow 22` (Ubuntu) 或 `firewall-cmd --permanent --add-port=22/tcp` (CentOS)。
 3. 确认被监控节点 SSH 服务已启动： `systemctl status sshd`。

问题 2: SSH 认证失败

- 现象：日志提示「PermissionDenied」。
- 解决：
 1. 验证 SSH 用户名 / 密码是否正确。
 2. 被监控节点 SSH 配置允许密码登录：编辑 `/etc/ssh/sshd_config`，确保 `PasswordAuthentication yes`，重启 SSH 服务： `systemctl restart sshd`。

5.3 前端页面无法显示数据

问题 1: WebSocket 连接失败

- 现象：页面显示「连接断开」，日志提示「WebSocket Disconnect」。
- 解决：检查 Nginx 配置文件（`./nginx/nginx.conf`）中的 WebSocket 代理配置是否正确：

```
location /ws {  
  
    proxy_pass http://localhost:8000/ws;  
  
    proxy_http_version 1.1;  
  
    proxy_set_header Upgrade $http_upgrade;  
  
    proxy_set_header Connection "upgrade";  
  
    proxy_set_header Host $host;  
  
}
```

重启 Nginx 容器： `docker compose restart nginx`。

问题 2：后端服务未启动

- 现象：访问 API 文档提示「502 Bad Gateway」。
- 解决：查看后端日志：`docker compose logs server-monitor`，根据错误信息排查（如数据库初始化失败、端口占用）。

六、运维管理命令

6.1 服务管理

命令	功能说明
<code>docker compose up -d</code>	启动所有服务（后台运行）
<code>docker compose down</code>	停止并删除所有服务
<code>docker compose restart</code>	重启所有服务
<code>docker compose restart server-monitor</code>	单独重启后端服务
<code>docker compose restart nginx</code>	单独重启 Nginx 服务

6.2 日志查看

命令	功能说明
<code>docker compose logs -f</code>	查看所有服务实时日志
<code>docker compose logs -f server-monitor</code>	查看后端服务实时日志
<code>docker compose logs -f nginx</code>	查看 Nginx 服务实时日志
<code>tail -f ./logs/app.log</code>	查看后端应用日志（宿主机目录）

6.3 数据管理

命令	功能说明
<code>sqlite3 ./data/monitor.db</code>	进入数据库交互模式（需安装 sqlite3）
<code>docker exec -it monitor-backend sqlite3 /app/data/monitor.db</code>	进入容器内数据库
<code>cp ./data/monitor.db /backup/</code>	备份数据库文件

七、注意事项

1. **数据安全**：数据库文件（`./data/monitor.db`）建议定期备份，避免数据丢失。
2. **权限控制**：被监控节点的 SSH 用户建议授予最小权限（需执行 `top`、`free`、`df` 等命令）。
3. **性能优化**：若监控节点超过 15 台，建议提升监控中心服务器配置（4 核 8G 内存），或调整采集间隔（修改 `main.py` 中 `asyncio.sleep(30)` 为更大值）。
4. **防火墙配置**：生产环境建议限制 8080 端口访问权限，仅允许指定 IP 访问监控页面。
5. **日志清理**：定期清理日志文件（`./logs/app.log`、`./logs/nginx/*.log`），避免占用过多磁盘空间。