

**JumpServer运维手册**

2022.06修订

目录

[1. 系统架构说明 4](#_Toc3708)

[1.1. 前提 4](#_Toc66)

[1.2. 系统说明 4](#_Toc14656)

[1.3. 依赖组件 4](#_Toc21764)

[1.4. 组件功能介绍 5](#_Toc6256)

[1.4.1. Core 组件 5](#_Toc2361)

[1.4.2. Celery 组件 5](#_Toc18468)

[1.4.3. Koko 组件 6](#_Toc20542)

[1.4.4. Lion 组件 6](#_Toc16203)

[1.4.5. Omnidb 组件 6](#_Toc6727)

[1.4.6. Xrdp 组件 6](#_Toc5724)

[1.4.7. Magnus 组件 6](#_Toc10930)

[1.5. 文件夹用途解析 6](#_Toc1242)

[1.6. 服务运行状态 8](#_Toc24085)

[2. 服务端口说明 8](#_Toc21435)

[2.1. 服务器端口说明 8](#_Toc1399)

[2.2. 防火墙配置说明 10](#_Toc13352)

[3. 日常基本操作 11](#_Toc10390)

[3.1. 服务运维基本命令 11](#_Toc9531)

[3.2. 修改配置文件 12](#_Toc20271)

[3.2.1. 配置文件详解 12](#_Toc10399)

[3.2.2. 修改config.txt 14](#_Toc12697)

[3.2.3. 修改其他配置文件 14](#_Toc17728)

[3.3. 数据库备份 14](#_Toc29497)

[3.4. 数据库恢复 15](#_Toc4207)

[4. 日志查看 15](#_Toc23962)

[4.1. 系统组件日志 15](#_Toc31009)

[4.2. Docker 查看 16](#_Toc2344)

[5. 升级步骤 17](#_Toc11828)

[6. 故障恢复 17](#_Toc2270)

[6.1. JumpServer程序故障 18](#_Toc32740)

[6.2. 数据库故障 18](#_Toc6259)

[6.2.1. 单机数据库 18](#_Toc24019)

[6.2.2. 主主数据库 18](#_Toc2239)

[6.3. 服务器宕机 20](#_Toc24137)

[7. 安全建议 20](#_Toc17517)

# 系统架构说明

## 系统说明

JumpServer 堡垒机采用微服务架构设计，通过 docker 承载服务，不同的功能模块在不同的容器上运行。通过 docker-compose 编排整个JumpServer服务的运行与维持不同功能模块之间的依赖关系。

## 依赖组件

JumpServer 堡垒机依赖于如下组件：

|  |  |
| --- | --- |
| 名称 | 描述 |
| jms\_web | JumpServer 的高性能的HTTP和反向代理web服务器 |
| jms\_lb | JumpServer 的高性能的HTTPS和反向代理web服务器,如果未开启LB配置，则不会运行此容器；运行此容器后,jms\_web容器将不再对宿主机映射80端口 |
| jms\_core | JumpServer的核心组件，主要功能是鉴权及授权中心 |
| jms\_celery | 任务队列，提供任务实时处理及任务调度 |
| Jms\_koko | JumpServer提供的Linux资产终端字符流连接及操作的工具，是1.5.2版本后用来代替jms\_coco，用来弥补coco处理高并发能力不足的缺点 |
| jms\_lion | JumpServer提供的，通过webgui方式访问Windows资产图形界面连接及操作的工具，是2.10.0版本后用来替代jms\_guacamole,用来弥补guacamole报错日志不全面、不准确的缺点 |
| jms\_omnidb | 主要功能是通过可视化界面纳管数据 |
| jms\_razor | JumpServer提供的，通过拉起原生mstsc访问Windows资产图形界面连接及操作的工具 |
| jms\_magnus | 提供代理直连Redis、MySQL、MariaDB、Redis以及PostgreSQL，也可以使用客户端图形界面化工具进行连接。 |
| jms\_redis | 提供数据缓存服务，如果使用外置 Redis，则不会运行此容器 |
| jms\_mysql | 提供数据库服务，如果使用外置 MySQL，则不会运行此容器 |

## 组件功能介绍

### Core 组件

core 组件是其他组件依赖的核心组件，该组件主要功能：

1. 负责与组件数据库(MySQL、Redis)进行通讯，进行核心数据的持久化。

* 例如：将用户信息、资产信息、应用信息、权限管理、资产会话管理等信息保存至默认的 MySQL 数据库中。
* 将用户登录后保持会话的凭证 Session 或者 Token 保存至 Redis 中。
* 将一些不易变动的数据缓存进 Redis 中。例如：组织下的人员数，用户组数，资产数等。

1. 负责用户身份认证和其他组件注册，及保持用户登录会话状态。
2. 负责 RESTful API，及时响应接口的请求。
3. 异步任务的下发。

* 例如：测试资产可连接性、更新资产硬件信息、执行改密计划、推送系统用户进资产、发送消息给用户(邮件、第三方平台)

### Celery 组件

celery 是处理 JumpServer 异步任务的组件。

***异步任务指的是当前下发任务的线程无需等待任务结果，直接拿到任务id即可。后续持续根据任务id，去获取任务执行结果即可。***

***celery 是异步任务的主体框架，Ansible 异步任务只是其中的一个命名空间，是 celery 异步任务的子集。***

该组件主要功能：

1. 负责将 core 组件下发在 Redis 中的任务取出并执行，将结果保存在 Mysql 中。

### Koko 组件

koko 是服务于类 Unix 系统平台的组件，该组件主要功能：

1. 命令行方式纳管Linux、Windows、数据库、k8s 等资产及应用
2. 作为 sshd 服务，可通过 2222 端口进行 ssh 或者 ftp 资产进行操作

### Lion 组件

lion 是服务于 Windows 系统平台的组件，该组件主要功能：

1. 通过 Web 端访问 Windows 资产

### Omnidb 组件

omnidb 是服务于数据库的组件，该组件主要功能：

1. 通过Web 端访问数据库

### Razor 组件

Razor 是服务于 Windows 系统平台的组件，该组件主要功能：

1. 通过 JumpServer Client 方式拉起原生mstsc，访问 Windows 资产

### Magnus 组件

magnus 是服务数据库通过代理连接的组件。该组件主要功能：

1. 以数据库代理的方式支持用户通过原生数据库客户端（例如 Navicat、SQLyog 等）对数据库进行直连操作。

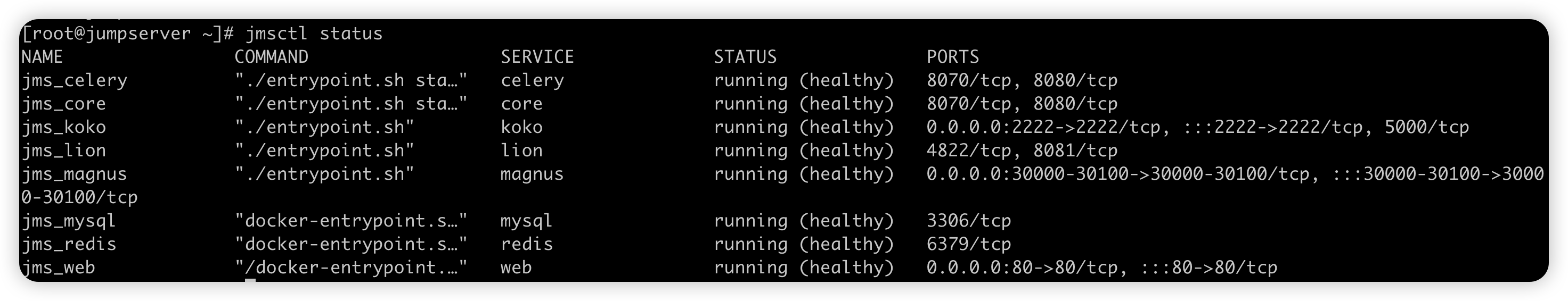
## 文件夹用途解析

JumpServer的默认安装路径：/opt/jumpserver，视安装路径而定。

|  |  |
| --- | --- |
| 文件路径 | 作用 |
| config/ | 保存了各个组件的配置信息，最主要的是 /opt/jumpserver/config/config.txt 文件，保存的是 JumpServer 的核心配置 |
| core/data/ansible/ | 此目录下有 3 层子目录，日志文件最终会以任务 id 的前 3 位字母，选择相应的文件目录下存放  日志文件以任务 id + .log 命令。  例如：111ecf75-d4b1-4a54-b2b5-e6b111be7545.log  存放路径:/opt/jumpserver/core/data/ansible/1/1/1/111ecf75-d4b1-4a54-b2b5-e6b111be7545.log |
| core/data/celery/ | 此目录下有 3 层子目录，日志文件最终会以任务 id 的前 3 位字母，选择相应的文件目录下存放此目录下最终保存的是使用 Celery 执行任务的日志，日志文件以任务 id + .log 命令。 |
| core/data/media/replay/ | 此目录下存放的是资产录像文件 |
| core/data/static/ | 此目录下存放的是静态文件，例如：img、css、js文件 |
| core/logs/ | 此文件保存的是使用 core 组件时产生的日志信息 |
| koko/data/keys/ | 此目录下存放的是和 core 组件注册使用的秘钥 |
| koko/data/logs/ | 此目录下存放的是使用 koko 组件时产生的日志信息 |
| koko/data/replays/ | 此目录下存放的是使用 koko 组件时的录像文件，在资产会话结束后，此录像文件会发送给 core 组件 |
| lion/data/keys/ | 此目录下存放的是和 core 组件注册使用的秘钥 |
| lion/data/logs/ | 此目录下存放的是使用 lion 组件时产生的日志信息 |
| lion/data/replays/ | 此目录下存放的是使用 lion 组件时的录像文件，在资产会话结束后，此录像文件会发送给 core 组件 |
| nginx/log/ | 此目录下存放的是 nginx 产生的日志信息 |
| omnidb/data/omnidb.db | 此文件是 omnidb 组件提供服务时所需的本地数据库文件 |
| omnidb/data/omnidb.log | 此文件保存的是使用 omnidb 组件时产生的日志信息 |
| razor/data/keys/ | 此目录下存放的是和 core 组件注册使用的秘钥 |
| razor/data/logs/ | 此目录下存放的是使用 xrdp 组件时产生的日志信息 |
| razor/data/replays | 此目录下存放的是使用 xrdp 组件时的录像文件，在资产会话结束后，此录像文件会发送给 core 组件 |
| redis/data/ | 此目录下存放的是 Redis 的配置文件及持久化文件(内置 redis ) |
| mysql | 此目录下存放的是 mysql 的持久化文件(内置 MySQL ) |

## 服务运行状态

JumpServer堡垒机的模块都是独立打包和运行的，基于微服务架构和容器化部署。下图是平台各模块的正常运行时状态截图：



# 服务端口说明

## 服务器端口说明

从客户端到 JumpServer 服务的访问，需要开通的端口如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 端口 | 作用 | 说明 | 需开通与否 |
| 80、443 | web端，http、https服务端口 | http、https 访问端口，二者需开通一个 | 需要 |
| 2222 | SSH（堡垒机用户） | koko服务组件默认端口，如果启用 SSH Client方式访问堡垒机及登录资产，则需要开启 | 按需 |
| 3389 | docker 内部 jms\_xrdp 组件服务接口 | xrdp 服务组件默认端口，如果启用RDP 组件(XRDP)方式访问Windows资产，则需要开启 | 按需 |
| 8080 | celery, core服务接口 | celery，core服务接口 | 不需要 |
| 8070 | celery, core服务接口 | celery，core服务接口 | 不需要 |
| 30000-30100 | Mangnus 服务端口 | Magnus 服务组件默认端口，如果启用DB 组件(Magnus)方式访问数据库，则需要开启 | 按需 |

数据库节点服务器需要开通的访问端口如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 端口 | 作用 | 说明 | 需开通与否 |
| 3306 | 数据库服务接口 | MySQL 服务使用，存放 JumpServer 服务数据 | 如使用内置 mysql，不需要，如使用外置mysql需开通JumpServer 服务器到mysql 的3306端口 |
| 6379 | Redis 服务接口 | 缓存使用，存放 JumpServer 缓存 | 如使用内置 redis，不需要，如使用外置redis需开通 JumpServer 服务器到redis 的6379端口 |

JumpServer服务器到资产或应用需要开通的访问端口如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 端口 | 作用 | 说明 | 需开通与否 |
| ssh 协议端口，默认为22 | 通过 ssh 协议访问资产 | 根据客户实际访问所用端口号做规则放通 | 按需开启 |
| rdp 协议端口，默认为3389 | 通过 rdp 协议访问资产 | 根据客户实际访问所用端口号做规则放通 | 按需开启 |
| vnc 协议端口，默认为5901 | 通过 vnc 协议访问资产 | 根据客户实际访问所用端口号做规则放通 | 按需开启 |
| telnet 协议端口，默认为23 | 通过 telnet 协议访问资产 | 根据客户实际访问所用端口号做规则放通 | 按需开启 |
| 其他协议端口，如mysql，默认为3306 | 通过其他协议访问应用 | 根据客户实际访问所用端口号做规则放通 | 按需开启 |

## 防火墙配置说明

在生产环境中，已经在实施过程中放开 JumpServer 相应访问端口。如有防火墙以及网络限制等需提前放行规则。

如在 JumpServer 已经运行的情况下需要添加修改 JumpServer 节点服务器防火墙策略以及修改机器网络配置等，需要重启 JumpServer服务与 docker 服务：

1. 进入 JumpServer 安装包目录
2. 停止 JumpServer 服务
3. 重启 docker
4. 启动 JumpServer 服务

以下为以上操作的命令示例:

|  |
| --- |
| $ cd /opt/jumpserver-offline-release-v2.22.0-amd64-203  $ ./jmsctl down  $ systemctl restart docker  $ ./jmsctl start |

# 日常基本操作

## 服务运维基本命令

JumpServer 的安装脚本的默认存放位置于/opt/jumpserver-offline-release-版本号-系统架构-补丁号/jmsctl.sh

使用方法为：

$ cd /opt/jumpserver-offline-release-v2.22.0-amd64-203

命令格式：./jmsctl.sh [COMMAND]

参数详解如下：

$ ./jmsctl.sh -h 获取帮助

$ ./jmsctl.sh start 启动JumpServer服务所有容器

$ ./jmsctl.sh stop 停止JumpServer服务所有容器

$ ./jmsctl.sh down 关闭JumpServer服务所有容器

$ ./jmsctl.sh restart 重启JumpServer服务所有容器 （会删除容器缓存）

$ ./jmsctl.sh status 查看JumpServer服务容器运行状态

$ ./jmsctl.sh backup\_db 备份JumpServer数据库文件

$ ./jmsctl.sh uninstall 卸载JumpServer服务（操作此项需谨慎）

备注：使用docker restart jms\_容器名的操作不会删除容器缓存，故如需保留容器中变更的配置且重启容器使配置生效，可使用此命令（如要保证容器缓存被清除后依旧保留此变更配置，请 docker commit 重新打包镜像）；如需重新加载变更在宿主机上的配置，请使用./jmsctl.sh restart重启整个服务，或者单独重启组件，这里以koko为例，使用./jmsctl.sh restart koko。

## 修改配置文件

### 配置文件详解

该配置文件的位置在于：/opt/jumpserver/config/config.txt

|  |
| --- |
| # 以下设置如果为空系统会自动生成随机字符串填入  ## 迁移请修改 SECRET\_KEY 和 BOOTSTRAP\_TOKEN 为原来的设置  ## 完整参数文档 https://docs.jumpserver.org/zh/master/admin-guide/env/  ## 安装配置, 可以使用华为云加速下载, arm64 用户需要注释掉 DOCKER\_IMAGE\_PREFIX  # DOCKER\_IMAGE\_PREFIX=swr.cn-south-1.myhuaweicloud.com  VOLUME\_DIR=/opt/jumpserver  DOCKER\_DIR=/var/lib/docker  SECRET\_KEY=OTJjZTNiNDItMjYxZC05ODMzLTE2NTctOWU0MzhhOTc0YWQy #用来加密解密的key  BOOTSTRAP\_TOKEN=OTJjZTNiNDItMjYxZC05ODMz #koko/lion用来向jms注册使用的token  LOG\_LEVEL=ERROR #日志级别  ## MySQL 配置, USE\_EXTERNAL\_MYSQL=1 表示使用外置 MySQL, 请输入正确的 MySQL 信息  USE\_EXTERNAL\_MYSQL=0 #是否使用外部的mysql  DB\_HOST=mysql #数据库的连接地址  DB\_PORT=3306 #数据库的连接端口  DB\_USER=root #数据库的连接用户  DB\_PASSWORD=OTJjZTNiNDItMjYxZC05ODMzLT #数据库的连接用户的密码  DB\_NAME=jumpserver #数据库的连接数据库，即写入数据的数据库  ## Redis 配置, USE\_EXTERNAL\_REDIS=1 表示使用外置 Redis, 请输入正确的 Redis 信息  USE\_EXTERNAL\_REDIS=0 #是否使用外部Redis  REDIS\_HOST=redis #外部redis的连接地址  REDIS\_PORT=6379 #外部redis的连接端口  REDIS\_PASSWORD=OTJjZTNiNDItMjYxZC05ODMzLT #外部redis的连接密码  ## Compose 项目设置, 如果 192.168.250.0/24 网段与你现有网段冲突, 请修改然后重启 JumpServer  COMPOSE\_PROJECT\_NAME=jms  COMPOSE\_HTTP\_TIMEOUT=3600  DOCKER\_CLIENT\_TIMEOUT=3600  DOCKER\_SUBNET=192.168.250.0/24 #docker有关于JumpServer的网段  ## IPV6 设置, 容器是否开启 ipv6 nat, USE\_IPV6=1 表示开启, 为 0 的情况下 DOCKER\_SUBNET\_IPV6 定义不生效  USE\_IPV6=0  DOCKER\_SUBNET\_IPV6=fc00:1010:1111:200::/64  ## 访问配置  HTTP\_PORT=80 #JumpServer的web界面访问的端口  SSH\_PORT=2222 #JumpServer的ssh方式访问的端口  MAGNUS\_MYSQL\_PORT=33060 #JumpServer使用magnus组件连接数据库时使用的端口  MAGNUS\_MARIADB\_PORT=33061  ## HTTPS 配置, 参考 https://docs.jumpserver.org/zh/master/admin-guide/proxy/ 配置  # USE\_LB=1  # HTTPS\_PORT=443  # SERVER\_NAME=your\_domain\_name  # SSL\_CERTIFICATE=your\_cert  # SSL\_CERTIFICATE\_KEY=your\_cert\_key  ## Nginx 文件上传大小  CLIENT\_MAX\_BODY\_SIZE=4096m  ## Task 配置, 是否启动 jms\_celery 容器, 单节点必须开启  USE\_TASK=1  ## XPack, USE\_XPACK=1  USE\_XPACK=1 #是否启动企业版功能（注意：社区版不能开启）  RDP\_PORT=3389 # #RDP协议连接资产使用的端口  MAGNUS\_POSTGRE\_PORT=54320 #JumpServer使用Magnus组件连接PostgerSql数据库使用的端口。  ## Core 配置, Session 定义, SESSION\_COOKIE\_AGE 表示闲置多少秒后 session 过期, SESSION\_EXPIRE\_AT\_BROWSER\_CLOSE=true 表示关闭浏览器即 session 过期  # SESSION\_COOKIE\_AGE=86400 #SESSION有效期，默认界面用户不需要密码自动化登录时间  SESSION\_EXPIRE\_AT\_BROWSER\_CLOSE=true #关闭浏览器失效登录的session  其他CORE组件的参数请参考：https://docs.jumpserver.org/zh/v2.20.1/admin-guide/env/#core  ## Koko Lion XRDP 组件配置  CORE\_HOST=http://core:8080 #JumpServer项目的url，api请求注册使用  JUMPSERVER\_ENABLE\_FONT\_SMOOTHING=true #启用平滑字体true。  TCP\_SEND\_BUFFER\_BYTES=4194304  TCP\_RECV\_BUFFER\_BYTES=6291456  其他 KOKO参数请参考：https://docs.jumpserver.org/zh/v2.20.1/admin-guide/env/  ## 终端使用宿主 HOSTNAME 标识  SERVER\_HOSTNAME=${HOSTNAME}  ## 额外的配置  CURRENT\_VERSION=v2.22.2  #JumpServer连接VScode的参数  ENABLE\_LOCAL\_PORT\_FORWARD=true  ENABLE\_VSCODE\_SUPPORT=true  MAGNUS\_REDIS\_PORT=63790 #JumpServer使用Magnus组件连接Redis数据库的端口  其他参数请参考：https://docs.jumpserver.org/zh/v2.20.1/admin-guide/env/ |

### 修改config.txt

JumpServer 配置文件默认位置：/opt/jumpserver/config/config.txt

如 JumpServer 在运行过程中，修改了 config.txt 配置文件中的参数，需要运行：

|  |
| --- |
| $ cd /opt/jumpserver-offline-release-v2.22.0-amd64-203  $ ./jmsctl.sh restart |

### 修改其他配置文件

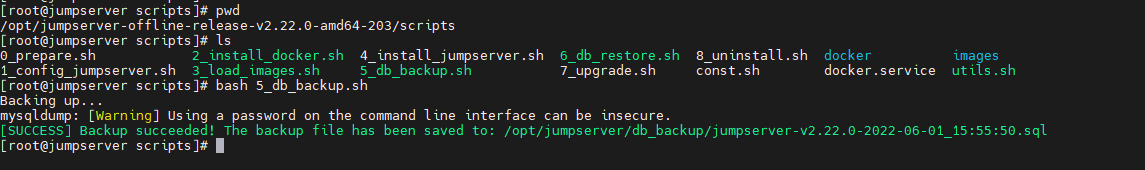
如 JumpServer 在运行，修改了其他配置文件中的参数，如 nginx 配置文件（http\_server.conf），mysql的配置文件（my.cnf），redis配置文件（redis.conf）此变更需要启动服务才能生效。需要运行：

|  |
| --- |
| $ cd /opt/jumpserver-offline-release-v2.22.0-amd64-203  $ ./jmsctl.sh restart |

## 数据库备份

如 JumpServer 在运行时，需要对 JumpServer 数据库进行备份，可以执行安装包目录下的数据库备份脚本：

|  |
| --- |
| $ cd /opt/jumpserver-offline-release-v2.22.0-amd64-203/scripts  $ bash 5\_db\_backup.sh  或者：  $ cd /opt/jumpserver-offline-release-v2.22.0-amd64-203  $ ./jmsctl.sh backup\_db |



## 数据库恢复

在升级失败或者其他需要回滚数据库时，可进行数据库回滚操作：

数据库回滚之前需要进行操作备份，具体操作请参考3.3数据库备份。

|  |
| --- |
| 停止JumpServer服务。  $ jmsctl stop  进入数据库容器  $ docker exec -it jms\_mysql /bin/bash #JumpServer使用本地数据库容器  登录数据库  $ mysql -uroot -p$MYSQL\_ROOT\_PASSWORD #amd架构  删除旧的JumpServer数据库  $ drop database jumpserver;  创建新的数据库并退出  $create database jumpserver default charset 'utf-8';  恢复数据库  $ jmsctl restore\_db 备份数据库脚本名称  重新启动JumpServer  $ jmsctl restart |

# 日志查看

## 系统组件日志

以默认路径：/opt/jumpserver 为例：

JumpServer的日志存放在/opt/jumpserver/core/logs目录下：



* ansible.log：该日志是ansible执行任务产生的日志（linux测试资产可连接性、更新硬件信息、推送系统用户、linux执行改密计划等）
* beat.log: 该日志是定时任务的日志
* celery\_ansible.log：该日志是异步任务ansible队列下的任务日志。
* celery\_default.log: 该日志是异步任务默认队列下的任务日志。
* daphne.log：该日志是Django的一部服务，主要用来支持websocket。
* drf\_exception.log：该日志是使用DRF框架抛出的异常信息。
* flower.log：该日志是作业中心的任务监控组件日志。
* gunicorn.log：该日志是用来记录请求的日志。
* jumpserver.log：JumpServer的总日志。

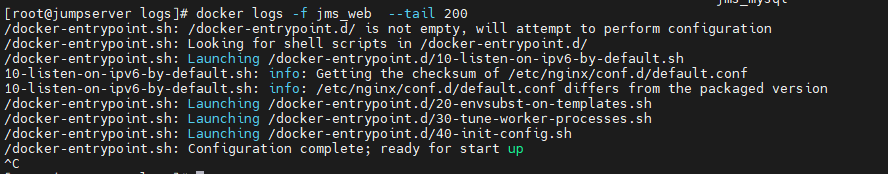
其他组件日志文件位置：

* Core：/opt/jumpserver/core/logs
* Celery：/opt/jumpserver/core/logs
* Lion：/opt/jumpserver/lion/data/logs
* Koko：/opt/jumpserver/koko/data/logs
* Xrdp：/opt/jumpserver/xrdp/data/logs
* Omnidb：/opt/jumpserver/omnidb/data
* Magnus：/opt/jumpserver/omnidb/data/logs
* Web：/opt/jumpserver/nginx/log/

## Docker 查看

例：查看 web 容器后200行日志

$ docker logs -f jms\_core --tail 200



如需查看其他组件日志：

|  |
| --- |
| $ docker logs -f [Container ID]  $ docker logs -f [Container] |

# 升级步骤

注意：

如需升级，建议首先查看官方的release note，了解近期版本变化，选择适合当前环境的版本。同时建议联系售后技术支持，进一步确认。

如自己升级，需注意：JumpServer升级会停止服务一段时间，根据环境情况，时间为10-30分钟，建议至少申请一个小时的变更窗口，留足验证以及回退时间。

JumpServer支持采用一键快速升级方式（会重启平台全部服务），步骤如下：

1. 门户下载最新安装包，上传最新安装包至 JumpServer服务器
2. 备份 JumpServer 服务数据

|  |
| --- |
| $ cd /opt/jumpserver-offline-release-v2.22.0-amd64-203/scripts  $ bash 5\_db\_backup.sh  或者  $ cd /opt/jumpserver-offline-release-v2.22.0-amd64-203  $ ./jmsctl.sh backup\_db |

1. 解压安装包，进入安装包目录
2. 运行jmsctl.sh脚本升级

|  |
| --- |
| $ ./jmsctl.sh upgrade |

1. 升级完毕，启动JumpServer服务，具体步骤如下：

|  |
| --- |
| $ ./jmsctl.sh start |

注意：如特殊部署的情况，升级步骤需要根据部署情况执行，请联系售后技术工程师。

# 故障恢复

故障的情况一般分为三种：JumpServer程序宕机、MySQL宕机、服务器宕机。

## JumpServer程序故障

1. 重启故障容器

|  |
| --- |
| $ docker restart 容器名 |

1. 全部重启

|  |
| --- |
| $ cd /opt/jumpserver-offline-release-v2.22.0-amd64-203  $ ./jmsctl.sh down # 停止JumpServer  $ docker ps -a # 检查是否还有没停的容器，有则kill并删除  $ docker kill ID  $ docker rm ID  $ ./jmsctl.sh start # 启动JumpServer,若有个别启动不成功的可以稍等再启一次  $ ./jmsctl.sh status # 检查JumpServer是否启动成功 |

## 数据库故障

### 单机数据库

1. 备份数据库
2. 登录数据库服务器，停止JumpServer，保证mysql无数据写入后。检查数据库具体问题。

### 主主数据库

1. 关闭服务

登录数据库服务器，停止JumpServer，保证MySQL无数据写入。

|  |
| --- |
| $ cd /opt/jumpserver-offline-release-v2.22.0-amd64-203  $ ./jmsctl.sh down  $ mysql -uroot -p'\*\*\*' -e "stop slave" # 停止同步并检测  $ mysql -uroot -p'\*\*\*' -e "show slave status \G" | grep Running |

1. Mysql数据同步

将正常mysql的jumpserver库备份

|  |
| --- |
| $ grep "DB" /opt/jumpserver/config/config.txt # 查看数据库配置  $ mysqldump -uroot -p'PASSWORD' jumpserver > jms\_bak.sql # 备份相应库 |

拷贝备份文件到故障服务器，导入数据

|  |
| --- |
| $ systemctl start mysqld  $ mysql -uroot -p'PASSWORD'  $ mysql> show create database jumpserver; # 查询创表语句以及字符集编码  $ mysql> drop database jumpserver;  $ mysql> show databases;  $ mysql> commit;  $ mysql> source jms\_bak.sql;  $ mysql> commit; |

检查两个库的数据是否一致

|  |
| --- |
| $ mysql> use jumpserver;  $ mysql> select count(\*) from users\_user;  $ mysql> select count(\*) from assets\_asset;  $ mysql> select count(\*) from assets\_systemuser;  $ mysql> select count(\*) from assets\_commandfilter;  $ mysql> select count(\*) from assets\_commandfilterrule;  $ mysql> select count(\*) from assets\_label;  $ mysql> select count(\*) from applications\_application; |

两个库进行双主同步，以下命令两个库都要执行。

|  |
| --- |
| $ mysql> grant replication slave on \*.\* to 'root'@'%' identified by '\*\*\*'; # 给另一个库授权  $ mysql> show master status; # 列出了日志位点信息,binlog file,binlog position  $ mysql> change master to master\_host='IP',master\_user='root',master\_password='\*\*\*',master\_log\_file='mysql-bin.0000\*\*',master\_log\_pos=\*\*\*; # 需要改为另一个库的IP、mysql-bin、master\_log\_pos  $ mysql> start slave; # 开始同步  $ mysql> show slave status\G; # 查看同步状态，Slave\_IO和Slave\_SQL都为yes说明正常。  $ mysql> flush privileges; |

1. 启动服务

启动前将录像等数据同步到宕机的服务器。

|  |
| --- |
| $ /bin/rsync -a /data/jumpserver/core/data/ \*.\*.\*.\*:/data/jumpserver/core/data |

主机启动Keepalived，先拉起VIP，用于连接mysql。

|  |
| --- |
| $ vim /opt/scripts/chk\_port.sh # 主节点port\_list只保留3306  $ systemctl start keepalived # 主节点执行，先获取VIP后才能连数据库 |

启动JumpServer。

|  |
| --- |
| $ cd /opt/jumpserver-offline-release-v2.13.2-amd64-57  $ ./jmsctl.sh start # 启动JumpServer  $ ./jmsctl.sh status # 检查JumpServer  $ vim /opt/scripts/chk\_port.sh # 主节点port\_list改回原来的配置 |

备机启动 JumpServer 和keepalived

|  |
| --- |
| $ cd /opt/jumpserver-offline-release-v2.13.2-amd64-57  $ ./jmsctl.sh start # 启动JumpServer  $ systemctl start keepalived.service # 启动keepalived，检查VIP |

## 服务器宕机

出现服务器宕机后应恢复服务器，保证服务器运行正常，然后按照 MySQL 故障的处理方式恢复数据库，并做主从，最后启动JumpServer、Keepalived等程序。

# 安全建议

1. 建议针对 JumpServer 数据库进行定期备份
2. 建议针对 JumpServer 配置文件进行定期备份
3. 建议全局开启 MFA 功能，避免因堡垒机用户密码泄露导致的安全问题
4. 建议开启账号备份计划，定期备份系统用户账号到邮箱作为逃生方案