# 一 **概述**

本文档是真趣工业互联网安全生产数智化平台（下面简称3.X平台）的运维手册；

ps：本手册适用于V3.15.0及其以上版本，低于V3.15.0的版本会有些许差异；

# **二 面向读者**

本文档面向3.X平台研发人员、运维人员、测试人员、交付经理、运营专员及3.X平台上游产品线的研发及测试人员。

本文档假定：

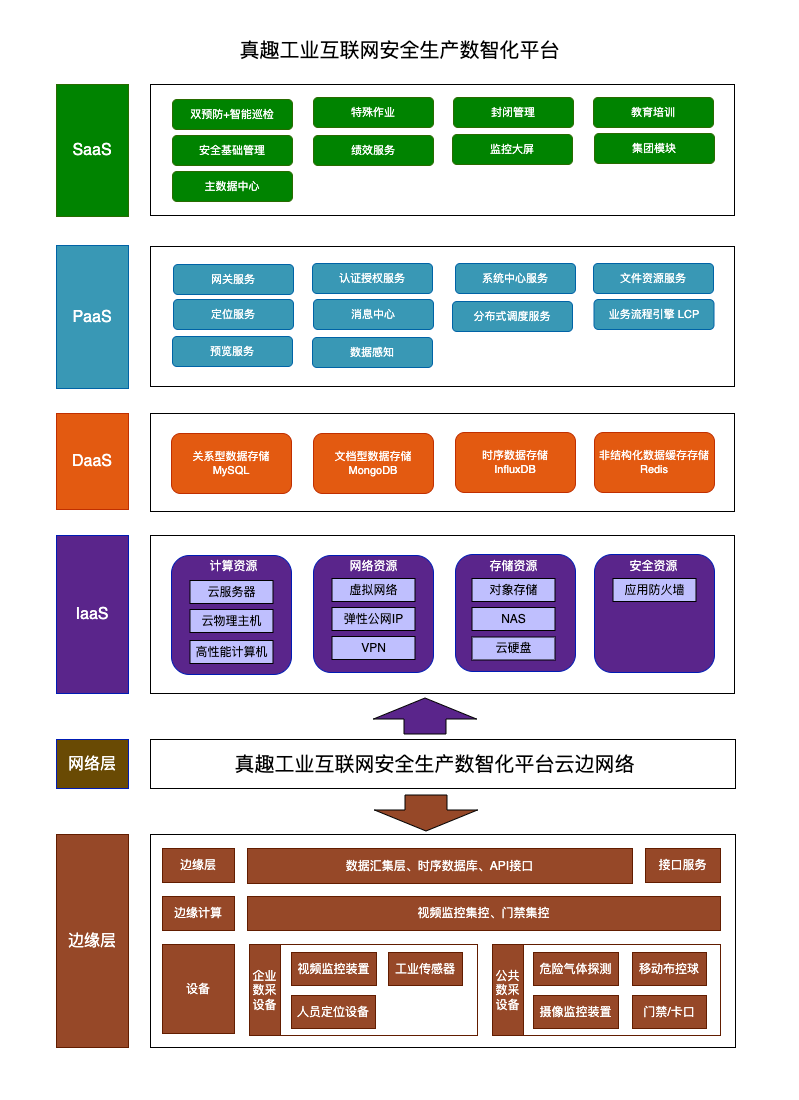
* + 你对Linux有一定了解
  + 你对Docker有一定了解
  + 你能基于安装文档熟练安装3.X平台

# **三 文档说明**

* 本文档**不是**3.X平台的操作手册，**也不是**3.X平台的安装手册。
* 本文档内容适用于通过3.X平台**标准版安装包安装的环境，暂不适用于分布式/集群化环境**。
* 本文档内容适用于V3.15.0版本。（低于V3.15.0的版本会有些许差异，部分适用）
* 如有错误或建议要反馈，请发送至 luweiwei@joysuch.com，内容会持续优化。

# **四 架构简述**

3.X平台架构示意图如下图所示：



# **五 端口一览**

下面表格只列了部分重要且常用端口

|  |  |  |  |
| --- | --- | --- | --- |
| 端口类型 | 所属中间件/应用 | 端口 | 端口说明 |
| HTTP | Nginx | 80/9550 | 3.X访问端口 |
| TCP | 30000 | 安帕尔检测仪对接端口 |
| HTTP | tiji-xxl-job | 8351 | 分布式调度任务的管理页面端口 |
| TCP | MySQL | 3306 | MySQL数据库连接端口 |
| TCP | MongoDB | 27017 | MongoDB数据库连接端口 |
| TCP | InfluxDB | 8086 | InfluxDB数据库连接端口 |
| TCP | Redis | 6379 | Redis连接端口 |
| HTTP | Nacos | 8848 | Nacos管理页面的端口 |
| TCP | RocketMQ | 9876 | RocketMQ nameserver 端口 |
| TCP | 10911 | RocketMQ broker端口 |
| HTTP | 30809 | RocketMQ 控制台端口（使用时） |

# **六 文件目录解读**

整个系统的所有文件、数据等，基本都存放在**/opt/application/**文件夹下，这个文件夹下的操作需要慎重。

## **6.1 平台的前端文件**

/opt/application/hse/nginx/web/tiji-front

/opt/application/hse/nginx/web/tiji-front-workflow

/opt/application/hse/nginx/web/h5

/opt/application/hse/nginx/web/map-h5

这4个目录都是系统的前端文件

## **6.2 地图文件**

/opt/application/hse/nginx/web/map-data

该文件夹下，每个租户都有一个随机字符串的文件夹与之对应，里面存放租户地图文件。

如果5G大小以下的2D、3D地图，可以直接在平台侧租户管理菜单里面上传，大于5G可以联系产研侧处理，不可以随意往该文件夹内手动上传地图；

## **6.3 系统附件**

附件分两部分，一部分为信息化平台附件，一部分为流程引擎附件；

信息化平台附件：/opt/application/hse/nginx/attachment 该文件夹下，每个租户会有一个独立的文件夹存放附件

流程引擎附件：/opt/application/hse/nginx/workflow/file 该文件夹下，-prod结尾的文件夹为每个租户流程引擎内上传的附件；pdf下是流程表单数据导出的pdf文件；sign下是所有的签名图片；

## **6.4 安装/升级日志文件**

安装日志存储位置：/tmp/install/

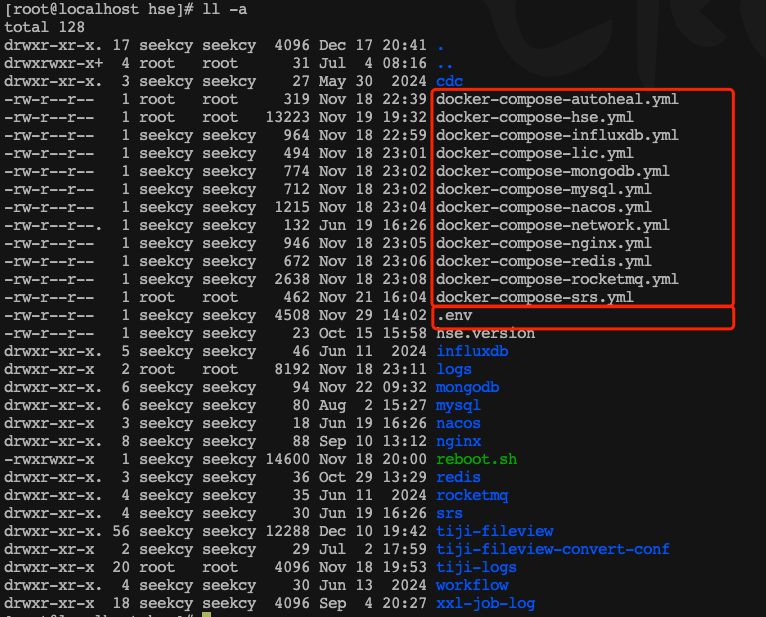
升级日志存储位置：/tmp/upgrade/

## **6.5 升级备份文件**

执行一键升级时，会整个备份上个版本的数据库数据、各项中间件配置、应用的配置，存放在/opt/application/backup/V3.14.0 最后面的版本号为上个版本值

# **七 应用与中间件介绍**

3.x整个平台一共有 20个应用服务，9个中间件服务；系统是运行在docker容器中的，每个应用和中间件都是以docker-compose来编排启动的，所以核心的启动文件是下面docker-compose-\*.yml文件，以及.env环境变量配置文件（/opt/application/hse/下）；



系统是运行在docker容器中的，每个应用和中间件都是以docker-compose来编排启动的，所以核心的启动文件是上面docker-compose-\*.yml文件，以及.env环境变量配置文件；

* docker-compose-autoheal.yml 监控所有应用、中间件的健康状态，以及不健康状态下的重启；
* docker-compose-hse.yml 管理所有的平台应用服务（tiji-开头的微服务）
* docker-compose-influxdb.yml 管理influxdb中间件
* docker-compose-lic.yml 授权服务
* docker-compose-mongodb.yml 管理mongodb中间件
* docker-compose-mysql.yml 管理mysql中间件
* docker-compose-nacos.yml 管理注册/配置中心中间件
* docker-compose-nginx.yml 管理nginx中间件
* docker-compose-redis.yml 管理redis中间件
* docker-compose-rocketmq.yml 管理rocketmq中间件
* docker-compose-srs.yml 管理视频流srs中间件
* docker-compose-network.yml 管理系统所有容器的桥接网络
* .env 文件中有每个应用的版本号，以及JVM设置

## **7.1 应用篇**

### **tiji-auth 授权服务**

容器名：tiji-auth

端口：8100（内部端口，不开放）

日志文件：/opt/application/hse/tiji-logs/tiji-auth/

配置文件：nacos中tiji-auth.yaml

### **tiji-gateway 网关服务**

容器名：tiji-gateway

端口：8888（内部端口，不开放）

日志文件：/opt/application/hse/tiji-logs/tiji-gateway/

配置文件：nacos中tiji-gateway.yaml

### **tiji-system 系统中心**

容器名：tiji-system

端口：8106（内部端口，不开放）

日志文件：/opt/application/hse/tiji-logs/tiji-system/

配置文件：nacos中tiji-system.yaml

### **tiji-master-data 主数据服务**

容器名：tiji-master-data

端口：8211、8212（内部端口，不开放）

日志文件：/opt/application/hse/tiji-logs/tiji-master-data/

配置文件：nacos中tiji-master-data.yaml

### **tiji-message-center 消息中心**

容器名：tiji-message-center

端口：8271（内部端口，不开放）

日志文件：/opt/application/hse/tiji-logs/tiji-message-center/

配置文件：nacos中tiji-message-center.yaml

### **tiji-resource 文件服务**

容器名：tiji-resource

端口：8251（内部端口，不开放）

日志文件：/opt/application/hse/tiji-logs/tiji-resource/

配置文件：nacos中tiji-resource.yaml

### **tiji-monitor-screen 大屏服务**

容器名：tiji-monitor-screen

端口：8261（内部端口，不开放）

日志文件：/opt/application/hse/tiji-logs/tiji-monitor-screen/

配置文件：nacos中tiji-monitor-screen.yaml

### **tiji-risk-prevention 双预防服务**

容器名：tiji-risk-prevention

端口：8291（内部端口，不开放）

日志文件：/opt/application/hse/tiji-logs/tiji-risk-prevention/

配置文件：nacos中tiji-risk-prevention.yaml

### **tiji-business-process 全流程服务**

容器名：tiji-business-process

端口：8221（内部端口，不开放）

日志文件：/opt/application/hse/tiji-logs/tiji-business-process/

配置文件：nacos中tiji-business-process.yaml

### **tiji-education 教育培训服务**

容器名：tiji-education

端口：8301（内部端口，不开放）

日志文件：/opt/application/hse/tiji-logs/tiji-education/

配置文件：nacos中tiji-education.yaml

### **tiji-location-middleware 定位服务**

容器名：tiji-location-middleware

端口：8311（内部端口，不开放）

日志文件：/opt/application/hse/tiji-logs/tiji-location-middleware/

配置文件：nacos中tiji-location-middleware.yaml

### **tiji-immersion-management 封闭管理服务**

容器名：tiji-immersion-management

端口：8321（内部端口，不开放）

日志文件：/opt/application/hse/tiji-logs/tiji-immersion-management/

配置文件：nacos中tiji-immersion-management.yaml

### **tiji-performance 绩效服务**

容器名：tiji-performance

端口：8311（内部端口，不开放）

日志文件：/opt/application/hse/tiji-logs/tiji-performance/

配置文件：nacos中tiji-performance.yaml

### **tiji-group 集团服务**

容器名：tiji-group

端口：8341（内部端口，不开放）

日志文件：/opt/application/hse/tiji-logs/tiji-group/

配置文件：nacos中tiji-group.yaml

### **tiji-data-integrating 数据对接服务**

容器名：tiji-data-integrating

端口：8281（内部端口，不开放）

日志文件：/opt/application/hse/tiji-logs/tiji-data-integrating/

配置文件：nacos中tiji-data-integrating.yaml

### **tiji-workflow 流程引擎**

容器名：tiji-workflow

端口：8202（内部端口，不开放）

日志文件：/opt/application/hse/workflow/logs/

配置文件：/opt/application/hse/workflow/conf/

### **tiji-cdc 数据感知服务**

容器名：tiji-cdc

端口：8111（内部端口，不开放）

日志文件：/opt/application/hse/tiji-logs/tiji-cdc/

配置文件：/opt/application/hse/cdc/tiji-cdc-conf/

### **tiji-fileview 预览服务**

容器名：tiji-fileview

端口：8012（内部端口，不开放）

日志文件：未挂载日志文件，可直接使用docker logs -fn 200 tiji-fileview 命令来看日志

配置文件：/opt/application/hse/tiji-fileview-convert-conf/

已转码附件存储位置：/opt/application/hse/tiji-fileview

### **tiji-xxl-job 调度服务**

容器名：tiji-xxl-job

端口：8351（默认内网该端口防火墙是放开的）

日志：tiji-xxl-job本身没有日志文件，应用执行定时任务的日志文件存储于 /opt/application/hse/xxl-job-log/，如果想要看tiji-xxl-job的日志可直接使用docker logs -fn 200 tiji-xxl-job命令来看日志

配置文件：无

tiji-xxl-job访问地址：http://ip:8351/xxl-job/ 用户名密码：admin/Joysuch@Locate2022

该地址可以用来排查定时任务问题，手动执行定时任务（不可随意执行，因为有些定时任务不能重复执行），查看定时任务执行情况等；

### **tiji-license-agent license授权服务**

容器名：tiji-lic

端口：30001

日志：docker logs -fn 200 tiji-lic命令来看日志

配置文件：无

## **7.2 中间件篇**

### **MySQL**

**容器名**：mysql

**端口**：3306（防火墙已添加）

**常用的用户名密码**：

root/\_fs&dX@8OOmm

tiji/C05d3~!FPLzu

**日志**：未单独挂载，可使用docker logs命令来查看

**配置**：/opt/application/hse/mysql/conf.d和/opt/application/hse/mysql/mysql.conf.d

**数据**：/opt/application/hse/mysql/data

**服务器上直连**：mysql -h127.0.0.1 -uroot -p"\_fs&dX@8OOmm"

**进容器再连接**：

1、docker exec -it mysql sh

2、mysql -h127.0.0.1 -uroot -p"\_fs&dX@8OOmm"

整合成一行即：docker exec -it mysql mysql -h127.0.0.1 -uroot -p"\_fs&dX@8OOmm"

### **MongoDB**

**容器名**：mongodb

**端口**：27017（防火墙已添加）

**常用的用户名密码**：

库：admin 用户名密码：admin/Seekcy@2020

库：tiji\_location\_middleware 用户名密码：tiji\_location\_middleware/seekcyLocation\_2022

库：tiji\_message\_center 用户名密码：tiji\_message\_center/seekcyLocation\_2022

**日志**：/opt/application/hse/mongodb/log （默认保留15天）

**配置**：/opt/application/hse/mongodb/conf/mongod.conf

**数据**：/opt/application/hse/mongodb/db/

**连接进mongodb：**

1、docker exec -it mongodb sh

2、mongo

3、use tiji\_location\_middleware;

4、db.auth("tiji\_location\_middleware", "seekcyLocation\_2022");

### **InfluxDB**

**容器名**：influxdb

**端口**：8086（防火墙未开放）

**常用的用户名密码**：

admin密码：SeekcyWWYTInflux

库：major\_hazard\_monitor\_point\_data 用户名密码：tiji/seekcyLocation\_2022

保留策略：两个月

**日志**：未挂载，基本上用不到该中间件的日志，想看可以用docker logs 来看

**配置**：/opt/application/hse/influxdb/config/influxdb.conf

**数据**：/opt/application/hse/influxdb/data

### **Nacos**

3x的配置中心兼注册中心；

**容器名**：nacos

**端口**：8848（防火墙已添加）

**用户名密码**：nacos/DnZj0qj0Kp4cbAVK

**日志**：未挂载，基本上用不到该中间件的日志，想看可以用docker logs 来看

**配置**：/opt/application/hse/nacos/conf/application.properties

**数据**：存在MySQL数据库中，库名：nacos

### **Redis**

**容器名**：redis

**端口**：6379（防火墙已添加）

**授权密码**：DrdGFT0Ib8duVKe

**日志**：未挂载，基本上用不到该中间件的日志，想看可以用docker logs 来看

**配置**：/opt/application/hse/redis/redis.conf

**数据**：/opt/application/hse/redis/data

### **RocketMQ**

**容器名**：rktmqnamesrv、rktmqbroker

**端口**：nameserver-9876，broker-10911（仅限内部使用，端口未开放）

**授权密码**：DrdGFT0Ib8duVKe

**日志**：/opt/application/hse/rocketmq/namesrv/logs；/opt/application/hse/rocketmq/broker/logs；

**配置**：/opt/application/hse/rocketmq/broker/broker.conf

**数据**：/opt/application/hse/rocketmq/namesrv/store；/opt/application/hse/rocketmq/broker/store；

默认不运行rocketmq dashboard，如果想要排查，请先确保服务器剩余内存至少大于2G，然后编辑/opt/application/hse/docker-compose-rocketmq.yml文件，将最后一段注释放开，切记第一行中文注释继续保留，下面每一行只要删除第一个#字符即可，然后保存退出；再执行hsectl start命令，静等1分钟，即可使用**http://ip:30809/** 来访问mq dashboard；排查问题结束后，请将注释还原回去，再执行docker stop rktmqdashboard && docker rm rktmqdashboard 这一行命令，将dashboard停掉。

### **SRS**

**容器名**：srs

**端口**：19350、8101、1985（防火墙已添加）

**日志**：未挂载，基本上用不到该中间件的日志，想看可以用docker logs 来看

**配置**：/opt/application/hse/srs/conf

### **Nginx**

**容器名**：nginx

**端口**：80，9550，443，8443，18443，18444，8886，20001，20002，30000（防火墙已添加），一般只使用其中的**80或者9550**即可，申请外网映射也只需要映射其中一个端口就行；

**日志**：access\_log未挂载，可直接使用docker logs查看日志

**配置**：/opt/application/nginx/conf/；/opt/application/nginx/conf.d/

**SSL证书存放位置**：/opt/application/hse/nginx/ssl/

### **autoheal**

一个比较小的中间件，用于监控容器的健康状态，并在应用不健康时，能做到按照策略拉起应用

**容器名**：autoheal

# **八 Docker 容器运维知识**

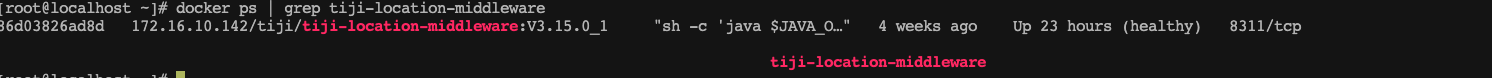
## **8.1 查看进程**

docker ps 显示所有启动中的容器进程

docker ps -a 显示所有容器进程（包括启动正常的、异常的、退出的都会出来）

根据过滤条件查看某一个应用的容器状态

docker ps | grep tiji-location-middleware 如下图



一共会返回7列，分别如下

CONTAINER ID：86d03826ad8d 容器ID

IMAGE：172.16.10.142/tiji/tiji-location-middleware:V3.15.0\_1 镜像地址

COMMAND："sh -c 'java $JAVA\_O…" 容器启动时执行的命令

CREATED：4 weeks ago 容器被创建的时间

STATUS：Up 23 hours (healthy) 容器启动的时间

PORTS：8311/tcp 容器端口

最后一列为容器名

其中CREATED表示容器是4周前创建的，中间可以restart容器很多次，每次restart都会刷新STATUS，Up表示容器是正常状态，healthy表示容器是健康的，23 hours表示23小时前重启的；

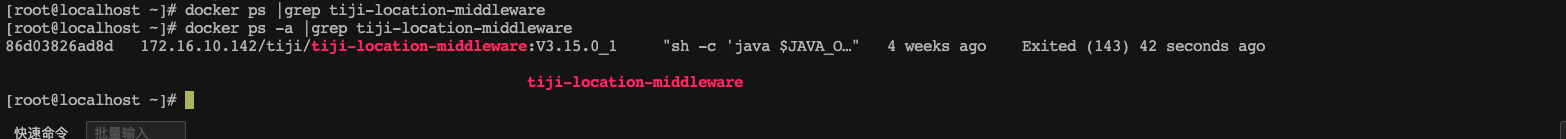
## **8.2 启停容器**

启动：通常使用docker start {容器ID或容器名} 来启动一个容器，例如docker start tiji-location-middleware

停止：通常使用docker stop {容器ID或容器名} 来启动一个容器，例如docker stop tiji-location-middleware

重启：通常使用docker restart {容器ID或容器名} 来启动一个容器，例如docker restart tiji-location-middleware

执行docker stop 停止容器后，再次使用docker ps |grep tiji-location-middleware 将不会返回任何结果，需要使用docker ps -a |grep tiji-location-middleware



此时可以看到tiji-location-middleware容器的状态变成了Exited (143) 42 seconds ago ，表示退出，42秒前退出；

## **8.3 查看容器日志**

此处只介绍查看容器日志的方法，日志挂载到宿主机磁盘的，可直接进指定文件夹下看日志文件；此方式在某些场景下不一定方便，不方便请直接找日志文件的存放处查看；

* 从结尾处查看某一容器日志：

例：docker logs -fn 200 tiji-location-middleware

docker logs 查看日志的固定语法

-f 表示从结尾看日志

-n 表示结束多少行，需要在-n 后面加上行数，比如此处的200，最后面再跟上容器日志

* 查看几分钟前容器日志：

例：docker logs --since 5m tiji-location-middleware

* 查看指定时间内的容器日志

例：docker logs --since "2024-12-18T00:00:00Z" --until "2024-12-18T00:01:00Z" tiji-location-middleware

这里会查询2024-12-18 08:00:00 ~ 2024-12-18 08:01:00 之间的日志，需要特别注意下上面命令行中是UTC时区，服务器一般都是CST，所以会有8小时的偏差。

* 其他

docker logs 有很多其他命令，比如加上 -t 会在每一行日志前面加上时间戳，更多使用方法，可自行网上查阅；

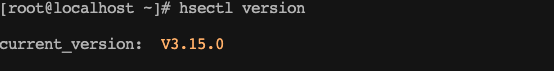
# **九 hsectl 命令使用**

hsectl是3.X安装包环境下特有的命令

有以下常用命令

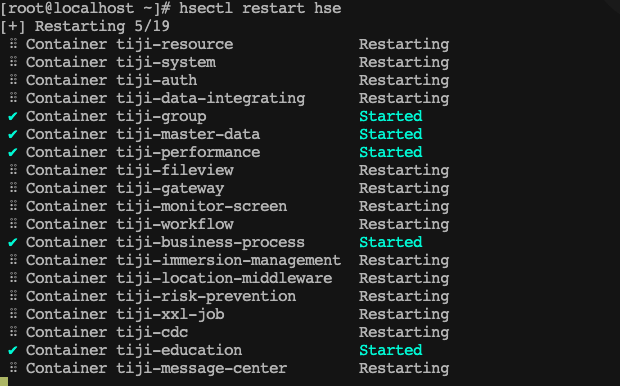
## **hsectl version**

查看当前平台版本，效果如下图所示：



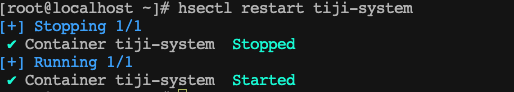
## **hsectl restart hse**

重启所有的应用，不包含中间件，效果如下图所示：



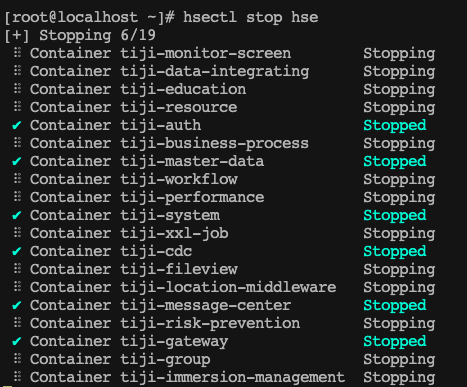
## **hsectl restart 容器名**

重启某一个容器，例如使用hsectl restart tiji-system，效果如下图所示：



## **hsectl stop hse**

停止所有的应用，不包含中间件，效果如下图所示：



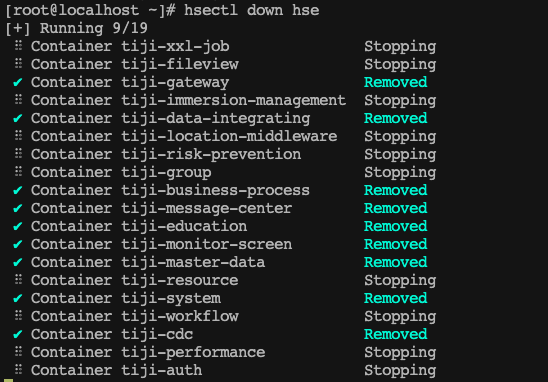
## **hsectl stop 容器名**

停止某一个应用或中间件，例如hsectl stop tiji-system，效果如下图所示：



## **hsectl down hse**

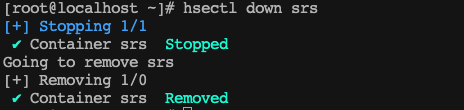
移除所有的应用，应用容器会一并删除，数据不会丢失，应用属于无状态的，效果如下图所示：



最终状态变成Removed，表示已删除

## **hsectl down 容器名**

移除某一个应用或中间件，例如hsectl down srs，效果如下图所示：



会先stop再移除；

## **hsectl start**

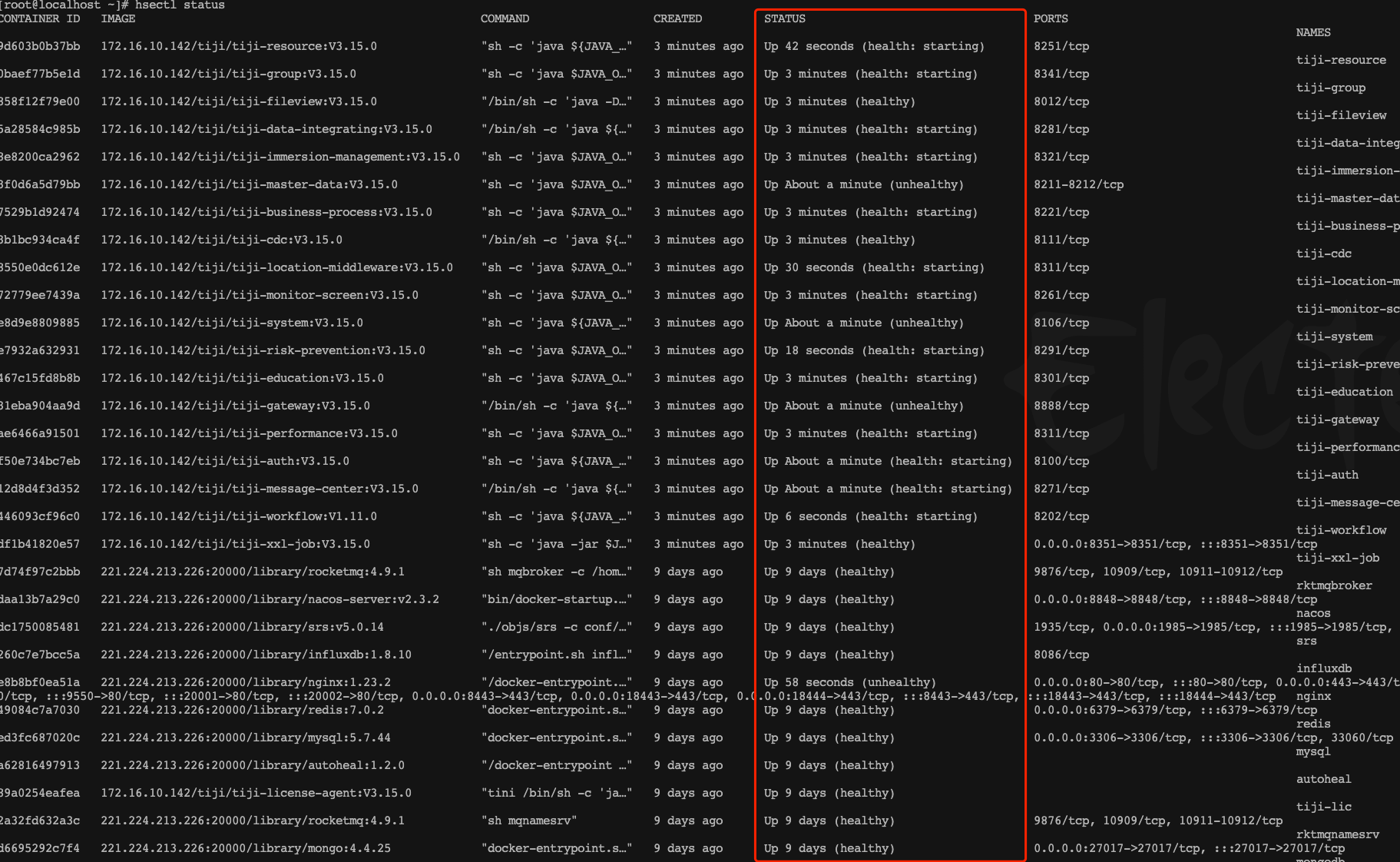
一键启动所有应用和中间件，如果已经启动着的应用和中间，并不会重启，效果如下图所示：



最终状态Started，表示重新创建应用/中间件容器，已经开始启动，但是并不表示启动成功，状态是Running的，表示的是这个应用/中间件是启动着的，并不是因此命令重启，无变化；

## **hsectl status**

查看所有应用和中间件的状态，效果如下图所示：



重点关注括号内的，health：starting表示正在启动中；healthy 表示启动完成并且健康；unhealthy 表示启动失败并且不健康，等待自我重启；

# **十 备份**

seekcy下有一个crontab定时任务，如下：

15 2 \* \* \* sh /opt/application/hse/backup\_data.sh >>/var/log/hse/backup.log 2>&1

每天夜里2点15分开始执行备份脚本；（如果有需要改的，自行修改）

备份范围和目录：

* MySQL：/opt/application/backup/mysql
* MongoDB：/opt/application/backup/mongo
* InfluxDB：/opt/application/backup/influx

保留天数：8天

# **十一 基础软件版本清单**

|  |  |  |
| --- | --- | --- |
| 分类 | 软件名称 | 版本 |
| 应用 | 3.X微服务应用 | V3.15.0 |
| tiji-workflow 流程引擎 | V1.11.0 |
| tiji-cdc 数据感知 | V3.15.0 |
| tiji-fileview 预览服务 | V3.15.0 |
| tiji-xxl-job 分布式调度服务 | V3.15.0 |
| tiji-lic 授权服务 | V3.15.0 |
| 基础框架 | Docker | 24.0.5 |
| JDK | 11.0.24 |
| 中间件 | MySQL | 5.7.44 |
| MongoDB | 4.4.25 |
| InfluxDB | 1.8.10 |
| Nacos | 2.3.2 |
| Redis | 7.0.2 |
| RocketMQ | 4.9.1 |
| EMQX（默认不安装） | 4.3.14 |
| SRS | 5.0.14 |
| Nginx | 1.23.2 |
| autoheal | 1.2.0 |