# 二级标识数据托管客户端使用文档

## 简介

本文档为二级标识数据托管客户端使用帮助文档。用于二级标识数据托管。应用名称为deposit-client。

本应用只提供托管客户端，不涉及状态监控与维护方面。也不涉及定时任务维护。

主要功能是从数据库中查询标识，生成指定格式的xml文件，然后zip压缩，sm2加密，如果大于1G则按照1G分卷，为一个加密文件签名，将加密文件与签名文件上传到配置好的sftp服务器。

应用包见附件deposit-client-mysql.jar。

## 配置详解

配置采用yaml格式配置，如下表：

注：下边配置为全部的配置信息，具体使用时只需要修改个别部分配置即可。使用时需修改的配置项见使用方式部分章节。

|  |
| --- |
| deposit:  client:  #客户端工作目录默认目录为${user.home}/deposit-data/data  #workDir: /home/gtld/apps/deposit-client/data/  #客户端产生的托管文件备份目录默认目录为${user.home}/deposit-data/backup  #backupDir: /home/gtld/apps/deposit-client/backup/  #单文件大小限制，超出这个值，需要文件分卷,请勿修改  fileLimitSize: 1G  #分组加密，不要改动  encryptGroupSize: 256M  #对接的托管平台sm2公钥  publicKey: MFkwEwYHKoZIzj0CAQYIKoEcz1UBgi0DQgAEupeawqs29qgtV8DhzACaP8/FaxdoDHsLSEYvZQb5YDhdx+hrRIjQVOtxpIPK8BCeJgfbOhDdgonmYWUXl/Y7KA==  #客户端使用的sm2私钥  privateKey: MIGTAgEAMBMGByqGSM49AgEGCCqBHM9VAYItBHkwdwIBAQQg+6m/7aH2DalpRT9rW6sNoYP40ILaqBh4UlqzLo1ntVygCgYIKoEcz1UBgi2hRANCAATOW4ppWhvE1gJuwqzn2yk9U1tVXlS2KBG1SIN/LqNrF9H+UANbIlsIUZfSzrBvzPEQKbLnCVyvCF9QHmtQIwdE  #客户端支持的标签前缀配置  handleLists:  - prefix: "86.100"  type: FULL  host: 202.173.9.16  username: ylnstn1  password: yln@Zdns1208  port: 22  path: upload  - prefix: "88.100"  type: FULL  host: 202.173.9.16  username: sftp88100  password: sFtp88100  port: 22  path: upload  spring:  #缓存  cache:  type: ehcache  ehcache:  config: classpath:/ehcache.xml  #配置数据源  datasource:  druid:  db-type: com.alibaba.druid.pool.DruidDataSource  driverClassName: net.sf.log4jdbc.DriverSpy  #数据库连接地址  url: jdbc:log4jdbc:mysql://202.173.9.71:3306/stndata?useUnicode=true&characterEncoding=utf8&serverTimezone=Asia/Shanghai  #数据库用户名  username: stndata  #数据库连接密码  password: caiCt@0104  # 初始连接数  initial-size: 5  # 最小连接数  min-idle: 10  # 最大连接数  max-active: 20  # 获取连接超时时间  max-wait: 5000  # 连接有效性检测时间  time-between-eviction-runs-millis: 60000  # 连接在池中最小生存的时间  min-evictable-idle-time-millis: 300000  # 连接在池中最大生存的时间  max-evictable-idle-time-millis: 900000  test-while-idle: true  test-on-borrow: true  test-on-return: false  # 检测连接是否有效  validation-query: select 1 from dual  # 配置监控统计  webStatFilter:  enabled: true  stat-view-servlet:  enabled: true  url-pattern: /druid/\*  reset-enable: false  filter:  stat:  enabled: true  # 记录慢SQL  log-slow-sql: true  slow-sql-millis: 1000  merge-sql: true  wall:  config:  multi-statement-allow: true  jta:  enabled: false  main:  #禁用web端口  web-application-type: none  profiles:  active: production  jackson:  date-format: yyyy-MM-dd HH:mm:ss  time-zone: GMT+8  mybatis-plus:  global-config:  db-config:  select-strategy: not\_empty  configuration:  log-impl: org.apache.ibatis.logging.stdout.StdOutImpl |

具体配置项参考注释部分，标识托管配置如下：

|  |
| --- |
| #客户端支持的标签前缀配置,为一个set配置，至少需要配置一个，可多个  handleLists:  - prefix: "86.100" #支持的托管前缀  type: FULL #托管类型，取值FULL/DIFF  host: 202.173.9.16 #SFTP服务器地址  username: ylnstn1 #SFTP用户名  password: yln@Zdns1208 #SFTP用户密码  port: 22 #SFTP服务端口号  path: upload #SFTP上传目录  - prefix: "88.100"  type: FULL  host: 202.173.9.16  username: sftp88100  password: sFtp88100  port: 22  path: upload |

## 使用方式

### 环境要求

centos服务器

jdk1.8

### 表结构要求

需在二级标识业务库创建如下表：

表结构如下：

|  |  |  |  |
| --- | --- | --- | --- |
| ESCROW\_RECORD\_LOG | | | |
| 字段 | 类型 | 描述 | 备注 |
| ID | int | 主键，自增 |  |
| PREFIX | VARCHAR2(255) | 前缀 |  |
| WATERMARK | DATETIME | 截止时间线 |  |
| REV | int | 重新发送号 |  |
| DATA\_TYPE | VARCHAR2(10) | 托管类型 | FULL/DIFF |
| START\_TIME | DATETIME | 开始时间 |  |
| END\_TIME | DATETIME | 结束时间 |  |
| STATUS | VARCHAR2(32) | 状态 | SUCCESS/FAIL |
| REMARK | VARCHAR2(255) | 备注 |  |
| DEPOSIT\_ID | VARCHAR2(50) | 托管ID |  |

使用前需初始化客户端使用的表结构

|  |
| --- |
| **CREATE** **TABLE** `ESCROW\_RECORD\_LOG` **(**  `ID` **int(**11**)** **NOT** **NULL** AUTO\_INCREMENT**,**  `**PREFIX**` **varchar(**255**)** **CHARACTER** **SET** utf8 **COLLATE** utf8\_general\_ci **NULL** **DEFAULT** **NULL,**  `WATERMARK` **datetime(**0**)** **NULL** **DEFAULT** **NULL,**  `REV` **int(**255**)** **NULL** **DEFAULT** **NULL,**  `DATA\_TYPE` **varchar(**255**)** **CHARACTER** **SET** utf8 **COLLATE** utf8\_general\_ci **NULL** **DEFAULT** **NULL,**  `START\_TIME` **datetime(**0**)** **NULL** **DEFAULT** **NULL,**  `END\_TIME` **datetime(**0**)** **NULL** **DEFAULT** **NULL,**  `STATUS` **varchar(**20**)** **CHARACTER** **SET** utf8 **COLLATE** utf8\_general\_ci **NULL** **DEFAULT** **NULL,**  `REMARK` **varchar(**255**)** **CHARACTER** **SET** utf8 **COLLATE** utf8\_general\_ci **NULL** **DEFAULT** **NULL,**  `DEPOSIT\_ID` **varchar(**50**)** **CHARACTER** **SET** utf8 **COLLATE** utf8\_general\_ci **NULL** **DEFAULT** **NULL,**  **PRIMARY** **KEY** **(**`ID`**)** **USING** BTREE**,**  **INDEX** `pre\_wat\_sta`**(**`**PREFIX**`**,** `WATERMARK`**,** `STATUS`**)** **USING** BTREE  **)** ENGINE **=** InnoDB**;** |

### 使用方式

以centos系统用户escrow为例，建议用户目录如下

mkdir /home/escrow/apps

mkdir /home/escrow/apps/deposit-client/ #应用目录

mkdir /home/escrow/apps/deposit-client/data #工作目录，临时文件

mkdir /home/escrow/apps/deposit-client/backup #数据备份目录

其中,backup目录备份时子目录创建规则如下：

/home/escrow/apps/deposit-client/backup/${prefix}/${yyyy-MM-dd}

${prefix}为前缀

${yyyy-MM-dd}为托管日期

/home/escrow/apps/deposit-client/backup/${prefix}/${yyyy-MM-dd}目录下为当前日期的托管文件备份。

/home/escrow/apps/deposit-client/backup/${prefix}/${yyyy-MM-dd}/slices为子文件备份目录

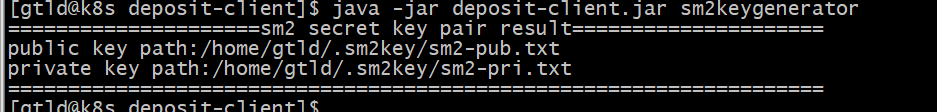
如下使用示例中涉及到的配置均以上述目录为例

#### 生成SM2密钥对

|  |
| --- |
| java **-**jar deposit-client-mysql.jar sm2keygenerator |

执行之后，会在console控制台输出sm2密钥对结果文件地址

示例如下:



具体密钥值查看console输出的路径文件内容即可。

#### 3.3.2 默认配置方式执行

默认下，应用配置文件已集成在deposit-client-mysql.jar中，即解压后目录BOOT-INF/classes/config中。默认生效的配置文件为其中的application.yaml与application-production.yaml。由于其中有数据库信息和SFTP服务器和密钥信息配置，这些配置都是需要修改的。这些需要自定义修改的配置都在application-production.yaml中。

修改示例，假设应用包在~/apps/deposit-client目录下

|  |
| --- |
| **cd** **~/**apps**/**deposit-client**/**  **mkdir** d **&&** **cd** d  jar **-**xvf **../**deposit-client.jar  **vi** BOOT-INF**/**classes**/**config**/**application-production.yaml |

需要修改的配置项列表如下：

可选修改

#托管客户端工作目录，托管过程中产生的中间文件都在此目录中默认目录为${user.home}/deposit-data/data。

deposit.client.workdir= /home/escrow/apps/deposit-client/data/

#备份目录，历史托管文件会备份到此目录，默认目录为${user.home}/deposit-data/backup/

deposit.client.backupdir=/home/escrow/apps/deposit-client/backup/

#必要修改

deposit.client.publicKey #托管平台sm2公钥

deposit.client.privateKey #客户端使用的sm2私钥

deposit.client.handleLists #支持的标识前缀，配置方式见章节配置详解

#数据库配置

spring.datasource.druid.url #数据库连接字符串

spring.datasource.druid.username #数据库用户名

spring.datasource.druid.password #数据库用户密码

上述配置项修改之后，将配置项修改更新到原始jar包中

|  |
| --- |
| jar **-**uvf **../**deposit-client.jar BOOT-INF**/**classes**/**config**/**application-production.yaml |

执行托管，配置文件中配置的前缀全部执行

|  |
| --- |
| java **-**jar deposit-client-mysql.jar **--**escrow.client**=**true |

执行托管，执行配置文件中指定前缀的托管

|  |
| --- |
| java **-**jar deposit-client-mysql.jar **--**escrow.client**=**true **--**escrow.prefix**=**88**.**100 |

#### 3.3.3 外部配置文件方式执行

参考配置详解中的配置内容，也可以参照jar包内BOOT-INF/classes/config目录下的application-demo.yaml。创建自己使用的配置文件applicatin-demo.yaml，修改好之后

调用指令如下：

|  |
| --- |
| java **-**jar deposit-client.jar **--**spring.config.location**=**application-demo.yaml **--**escrow.client**=**true  #指定前缀  java **-**jar deposit-client.jar **--**spring.config.location**=**application-demo.yaml **--**escrow.client**=**true **--**escrow.prefix**=**88**.**100 |

--spring.config.location指定加载的配置文件，绝对路径或当前路径下的相对路径均可。

#### 3.3.4 支持的命令行参数

见下表

|  |  |  |
| --- | --- | --- |
| 序号 | 参数key值 | 参数说明 |
| 1 | --spring.config.location | 外部配置文件，当指定外部配置文件时，应用执行的配置会使用指定的这个配置文件 |
| 2 | --escrow.client | 客户端方式执行，默认为true,当为false时不会执行托管 |
| 3 | --escrow.prefix | 执行托管的前缀,如不指定，则配置文件中的前缀都会执行一次托管 |
| 4 | --escrow.type | 托管类型,FULL|DIFF，当指定时，会覆盖配置文件中的托管类型。如不指定，则按照配置文件的配置执行 |
| 5 | sm2keygenerator | 当指定参数为此值时  表示生成sm2密钥对 |
| 6 | --deposit.client.publicKey | 托管平台使用的sm2公钥 |
| 7 | --deposit.client.privateKey | 客户端使用的sm2私钥 |
| 8 | --deposit.client.workDir | 工作目录，托管业务产生的临时文件在此目录中 |
| 9 | --deposit.client.backupDir | 备份目录，托管结束后将本次托管文件备份到此目录 |
| 10 | --spring.datasource.druid.url | mysql数据库连接字符串 |
| 11 | --spring.datasource.druid.username | Mysql数据库用户名 |
| 12 | --spring.datasource.druid.password | Mysql数据库密码 |

参数优先级，命令行参数的优先级大于配置文件

#### 3.3.5 状态维护

每一次托管执行，都会在escrow\_record\_log表生成一条记录。

指令执行结束后，如果escrow\_record\_log表中增加了一条记录并且status字段=SUCCESS，则代表托管提交成功。不过，这个状态并不代表托管平台的处理结果，最终结果应以托管平台的通知为准。如托管平台处理失败，则需要根据托管平台的提示信息针对处理后重新提交。

根据托管规范，需要在每天北京时间8点之后提交托管文件。定时任务维护需客户端使用方自行解决。

#### 3.3.5 结束状态码

客户端结束时，会返回调用终端一个状态码，状态码说明见下表

|  |  |
| --- | --- |
| 状态码 | 状态说明 |
| 0 | 本次客户端提交托管成功 |
| -1 | 应用内部发生未知错误，需查看日志排查 |
| 1 | 配置文件中缺失托管的前缀相关的托管配置 |
| >1 | 托管失败,需查看日志排查错误原因 |