**跨境支付系统环境部署文档**

目录

[第一章新建用户 1](#_Toc451440103)

[1.1 新建用户组 1](#_Toc451440104)

[1.2 新建跨境支付项目的目录 1](#_Toc451440105)

[1.3 新建cbpay用户 1](#_Toc451440106)

[1.4 新建discenter用户 2](#_Toc451440107)

[第二章程序上传 3](#_Toc451440108)

[2.1 上传cbpay用户下程序 3](#_Toc451440109)

[2.2 上传discenter用户下的程序 4](#_Toc451440110)

[第三章部署程序 4](#_Toc451440111)

[3.1 部署jdk 4](#_Toc451440112)

[3.2 部署主应用程序 7](#_Toc451440113)

[3.3 部署收银台 31](#_Toc451440114)

[3.4 部署console服务 32](#_Toc451440115)

[3.5 部署定时任务 38](#_Toc451440116)

[第三章部署数据库 40](#_Toc451440117)

[4.1 安装数据库 40](#_Toc451440118)

[4.2 新建数据库 40](#_Toc451440119)

[4.3 导入数据 41](#_Toc451440120)

[第四章启动服务 42](#_Toc451440121)

[4.1 启动activemq 42](#_Toc451440122)

[4.2 启动b2bic 42](#_Toc451440123)

[4.3 启动主服务 43](#_Toc451440124)

[4.4 启动console服务 43](#_Toc451440125)

[4.5 启动timedtask定时任务 44](#_Toc451440126)

[4.6 启动收银台服务 44](#_Toc451440127)

[第五章注意事项 44](#_Toc451440128)

[5.1 超时问题 44](#_Toc451440129)

[5.2 Jdk问题 47](#_Toc451440130)

## 第一章新建用户

### 新建用户组

新建用户组pay。命令如下：

|  |
| --- |
| groupadd pay |

### 新建跨境支付项目的目录

新建跨境支付项目的目录/ztkx，用于存放跨境支付系统所有的程序和配置等文件。命令如下：

|  |
| --- |
| mkdir /ztkx |

### 新建cbpay用户

1. 以root用户登录到新环境，新建cbpay用户，所属组为pay，

主目录为/ztkx/cbpay，命令如下：

|  |
| --- |
| useradd -d /ztkx/cbpay -g pay cbpay |

1. 为cbpay用户设定密码 (示例中设定的密码为cbpay),命令如下：:

|  |
| --- |
| passwd cbpay |

注：然后输入用户密码cbpay回车，确认用户密码，回车。即可完成设定密码操作。

1. 修改配置文件.bash\_profile，修改后内容如下：

|  |
| --- |
| # .bash\_profile  # Get the aliases and functions  if [ -f ~/.bashrc ]; then  . ~/.bashrc  fi  # User specific environment and startup programs  export PS1=`hostname`':$PWD>'  export JAVA\_HOME=/opt/jdk1.7.0\_55/  PATH=$JAVA\_HOME/bin:$PATH:$HOME/bin:.  export PATH  LANG=en\_US.UTF-8  #LANG=zh\_CN.gbk  #LC\_ALL="C"  set -o vi  ulimit -c unlimited  alias l="ls -lrt"  alias ..="cd .."  alias cdb="cd /ztkx/cbpay/CrossBorderPay/bin"  alias cdcfg="cd /ztkx/cbpay/CrossBorderPay/config"  alias cdmsg="cd /ztkx/cbpay/CrossBorderPay/msg"  alias p="ps -ef|grep cbpay"  alias cdlog="cd /ztkx/cbpay/logs/"  alias cdlib="cd /ztkx/cbpay/CrossBorderPay/lib" |

1. 重新加载.bash\_profile文件,命令如下：

|  |
| --- |
| sh /ztkx/cbpay/.bash\_profile |

### 新建discenter用户

1. 以root用户登录到新环境，新建discenter用户，所属组为pay，主目录为/ztkx/discenter，命令如下：

|  |
| --- |
| useradd -d /ztkx/discenter -g pay discenter |

1. 为discenter用户设定密码 (示例中设定的密码为discenter),命令如下：

|  |
| --- |
| passwd discenter |

注：然后输入用户密码（密码为：discenter）回车，确认用户密码，回车。即可完成设定密码操作。

1. 修改配置文件.bash\_profile，修改后内容如下：

|  |
| --- |
| # .bash\_profile  # Get the aliases and functions  if [ -f ~/.bashrc ]; then  . ~/.bashrc  fi  # User specific environment and startup programs  export PS1=`hostname`':$PWD>'  #PATH=$PATH:$HOME/bin:.  export JAVA\_HOME=/opt/jdk1.7.0\_55/  PATH=$JAVA\_HOME/bin:$PATH:$HOME/bin:.  export PATH  set -o vi  alias l="ls -lrt"  alias ..="cd .."  alias cdlog="cd /ztkx/discenter/logs/"  alias cdb="cd /ztkx/discenter/discenter/bin/"  alias cdcfg="cd /ztkx/discenter/discenter/config/"  alias cdlib="cd /ztkx/discenter/discenter/lib/"  alias p="ps -ef|grep discenter" |

1. 重新加载.bash\_profile文件,命令如下：

|  |
| --- |
| sh /ztkx/cbpay/.bash\_profile |

## 第二章程序上传

### 上传cbpay用户下程序

1. 首先以cbpay用户登录到老环境（登陆后自动进入/ztkx/cbpay），将所需程序打包为cbpay.tar。命令如下：

|  |
| --- |
| cd /ztkx/cbpay  tar -cvf cbpay.taractivemq b2bic console CrossBorderPay timedtask |

注：cbpay.tar为压缩包名称，activemq，b2bic，console， CrossBorderPay， timedtask为所需程序，activemq为activmq服务的相关程序，b2bic为中信银行交互时所需程序，console为跨境支付console服务，CrossBorderPay为跨境支付主程序，timedtask为跨境支付的定时任务程序。

1. 使用stfp命令将压缩包cbpay.tar上传到新环境。

|  |
| --- |
| sftp 172.30.12.35  bin  put cbpay.tar  bye |

注:172.30.12.35为新环境IP地址。

1. 使用cbpay用户登录到新环境，解压文件cbpay.tar到/ztkx/cbpay目录下，命令如下：

|  |
| --- |
| cd /ztkx/cbpay  tar -xvf cbpay.tar -C ./ |

1. 新建日志目录(/ztkx/cbpay/logs)，用于存放程序产生的日志。命令如下：

|  |
| --- |
| mkdir /ztkx/cbpay/logs |

1. 新建文件目录(/ztkx/cbpay/file)，用于存放程序产生的购汇文件，付汇文件等。

|  |
| --- |
| mkdir /ztkx/cbpay/file |

### 上传discenter用户下的程序

1. 首先以discenter用户登录到老环境，将discenter目录下所需程序打包为discenter.tar。命令如下：

|  |
| --- |
| tar -cvf discenter.tar cashier discenter |

注：cashier为收银台相关程序，discenter为discenter分发容器的相关程序！

1. 使用stfp命令将压缩包上传到新环境

|  |
| --- |
| sftp 172.30.12.35  bin  put discenter.tar  bye |

注：172.30.12.35为新环境的ip地址！

1. 使用discenter用户登录到新环境，解压文件discenter.tar到，命令如下：

|  |
| --- |
| cd /ztkx/discenter  tar -xvf discenter.tar -C ./ |

1. 新建日志目录(/ztkx/discenter/logs)，用于存放程序产生的日志。命令如下：

|  |
| --- |
| mkdir /ztkx/discenter/logs |

## 第三章部署程序

### 部署jdk

#### 检查jdk

* 首先以root用户登录到新环境，判断环境上是否已经安装jdk，使用如下命令：

|  |
| --- |
| echo $JAVA\_HOME |

注：如果未返回信息则表示该服务器未安装jdk，如果返回信息，则表示该服务器已经安装jdk：

如果jdk未安装，直接进行[**3.1.2安装jdk**](#_安装jdk)操作！如果已安装，则继续向下进行，查看jdk版本是否一致。

* 查看jdk版本是否正确，使用如下命：

|  |
| --- |
| $JAVA\_HOME/bin/java -version |

注：如果java版本为jdk1.7.0\_55,则无需再次进行安装，则继续进行后续操作。如果版本不一致，进行直接进行[**3.1.2安装jdk**](#_安装jdk)操作！

* 以cbpay用户登录新环境，使用vi命令打开.bash\_profile文件。具体如下：

|  |
| --- |
| vi /ztkx/cbpay/.bash\_profile |

打开文件后将看到如下配置信息：

|  |
| --- |
| # .bash\_profile  # Get the aliases and functions  if [ -f ~/.bashrc ]; then  . ~/.bashrc  fi  # User specific environment and startup programs  export PS1=`hostname`':$PWD>'  export JAVA\_HOME=/opt/jdk1.7.0\_55/  PATH=$JAVA\_HOME/bin:$PATH:$HOME/bin:.  export PATH  LANG=en\_US.UTF-8  #LANG=zh\_CN.gbk  #LC\_ALL="C"  set -o vi  ulimit -c unlimited  alias l="ls -lrt"  alias ..="cd .."  alias cdb="cd /ztkx/cbpay/CrossBorderPay/bin"  alias cdcfg="cd /ztkx/cbpay/CrossBorderPay/config"  alias cdmsg="cd /ztkx/cbpay/CrossBorderPay/msg"  alias p="ps -ef|grep cbpay"  alias cdlog="cd /ztkx/cbpay/logs/"  alias cdlib="cd /ztkx/cbpay/CrossBorderPay/lib" |

注：红色标注部分即为待修改内容，将JAVA\_HOME设置jdk实际安装目录，例如jdk的实际安装目录为/opt/tool/ jdk1.7.0\_55则将配置修改为；

|  |
| --- |
| export JAVA\_HOME=/opt/tool/jdk1.7.0\_55 |

* 保存后退出cbpay用户。
* 以discenter用户登录到新环境，使用vi命令打开.bash\_profile文件。具体如下：

|  |
| --- |
| vi /ztkx/discenter/.bash\_profile |

打开文件后将看到如下配置信息：

|  |
| --- |
| # .bash\_profile  # Get the aliases and functions  if [ -f ~/.bashrc ]; then  . ~/.bashrc  fi  # User specific environment and startup programs  export PS1=`hostname`':$PWD>'  #PATH=$PATH:$HOME/bin:.  export JAVA\_HOME=/opt/jdk1.7.0\_55/  PATH=$JAVA\_HOME/bin:$PATH:$HOME/bin:.  export PATH  set -o vi  alias l="ls -lrt"  alias ..="cd .."  alias cdlog="cd /ztkx/discenter/logs/"  alias cdb="cd /ztkx/discenter/discenter/bin/"  alias cdcfg="cd /ztkx/discenter/discenter/config/"  alias cdlib="cd /ztkx/discenter/discenter/lib/"  alias p="ps -ef|grep discenter" |

注：红色标注部分即为待修改内容，将JAVA\_HOME设置jdk实际安装目录，例如jdk的实际安装目录为/opt/tool/ jdk1.7.0\_55则将配置修改为；

|  |
| --- |
| export JAVA\_HOME=/opt/tool/jdk1.7.0\_55 |

* 保存后退出discenter用户。

#### 安装jdk

* 首先将jdk1.7.0\_55.tar.gz文件上传到新环境的/opt目录下。
* 以root用户登录到新环境，然后切换到/opt目录下：

|  |
| --- |
| cd /opt |

* 使用tar命令解压文件到当前目录下，具体命令如下：

|  |
| --- |
| tar -zxvf jdk1.7.0\_55.tar.gz |

#### 验证jdk

* 分别以cbpay和discenter用户登录到新环境。
* 验证jdk版本，命令如下：

|  |
| --- |
| $JAVA\_HOME/bin/java -version |

注：如果出现以下信息，则表示jdk已经正确安装。

|  |
| --- |
| java version "1.7.0\_55"  Java(TM) SE Runtime Environment (build 1.7.0\_55-b13)  Java HotSpot(TM) 64-Bit Server VM (build 24.55-b03, mixed mode) |

* 验证完成后退出当前用户。

#### 删除安装包。

* 以root用户登录到新环境。
* 使用rm命令删除压缩包jdk1.7.0\_55.tar.gz,具体命令如下：

|  |
| --- |
| rm /opt/jdk1.7.0\_55.tar.gz |

### 部署主应用程序

##### 部署in容器

1. 修改配置文件：

in容器需要修改配置文件如下：

1. dbpool.xml文件。

该文件用于保存in容器的连接池信息。

* 首先使用cbpay用户登录到新环境，然后使用vi命令打开该文件，具体命令如下：

|  |
| --- |
| vi /ztkx/cbpay/CrossBorderPay/config/in\_conf/dbpool/dbpool.xml |

进入文件后可看到如下配置信息：

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <c3p0-config>  <named-config name="myApp">  <property name="user">cbpay</property>  <property name="password">cbpay</property>  <property name="driverClass">oracle.jdbc.driver.OracleDriver</property>  <property name="jdbcUrl">jdbc:oracle:thin:@172.30.12.22:1521:fund</property>  <property name="initialPoolSize">10</property>  <property name="maxIdleTime">0</property>  <property name="maxIdleTimeExcessConnections">60</property>  <property name="maxPoolSize">30</property>  <property name="minPoolSize">5</property>  <property name="acquireIncrement">1</property>  <property name="idleConnectionTestPeriod">60</property>  <property name="acquireRetryAttempts">30</property>  <property name="acquireRetryDelay">1000</property>  <property name="testConnectionOnCheckout">false</property>  <property name="preferredTestQuery">select \* from dual</property>  </named-config>  </c3p0-config> |

注：红色标注的三项配置即是需要修改的内容，含义如下：

property name="user"标签中配置数据库用户名。

property name="password"配置用户密码。

property name="jdbcUrl"标签中配置数据库的url。

* 修改配置

根据实际数据库的信息进行修改即可。修改完成后保存退出！

1. baseConf.properties文件。

该文件用于存放in容器运行时所需的基本配置。

* 使用vi命令打开该文件，具体命令如下：

|  |
| --- |
| /ztkx/cbpay/CrossBorderPay/config/in\_conf/baseConf.properties |

进入文件后可看到如下配置：

|  |
| --- |
| #主配置文件  #初始化上下文池大小  contextPoolSize=50  #容器名称  CONTAINER\_NAME=IN  #集群节点  cluster=cluster0  #in容器消息监听listener  messagelistener=com.ztkx.cbpay.container.msglistener.TempletMsgListener  #mq重连时间间隔  mq.reconnection.interval=5000  #容器总线程池大小  threadPoolSize=50  #jms消息超时时间  JMS\_MESSAGE\_OVER\_TIME=60000  #是否需要将原始报文传入数据库的标识  TRANS\_LOG\_SWITCH=on  #是否需要更新交易日志  TRANS\_TIMESTAMP\_SWITCH=on |

注：红色标注参数cluster为该in容器在分布集群上的编号值。

1. ActiveMQconfig.xml文件

该文件存放使用activemq发送消息和接受消息消息时的相关配置。

* 首先使用vi命令打开该文件，具体命令如下：

|  |
| --- |
| vi /ztkx/cbpay/CrossBorderPay/config/in\_conf/protocol/ActiveMQconfig.xml |

进入文件后，可看到如下配置信息：

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <recipe>  <inboxs>  <inbox>  <service name="discenter\_in">  <INITIAL\_CONTEXT\_FACTORY>org.apache.activemq.jndi.ActiveMQInitialContextFactory</INITIAL\_CONTEXT\_FACTORY>  <PROVIDER\_URL>tcp://127.0.0.1:61616</PROVIDER\_URL>  <queueName>dynamicQueues/discenter\_in.queue</queueName>  <overTime>60000</overTime>  <isTransaction>false</isTransaction>  <autoAcknowledge>false</autoAcknowledge>  <sessionCounts>5</sessionCounts>  <messageListener></messageListener>  </service>  </inbox>  <inbox>  <service name="out\_in">  <INITIAL\_CONTEXT\_FACTORY>org.apache.activemq.jndi.ActiveMQInitialContextFactory</INITIAL\_CONTEXT\_FACTORY>  <PROVIDER\_URL>tcp://127.0.0.1:61616</PROVIDER\_URL>  <queueName>dynamicQueues/out\_in.queue</queueName>  <overTime>0</overTime>  <isTransaction>false</isTransaction>  <autoAcknowledge>true</autoAcknowledge>  <sessionCounts>5</sessionCounts>  <messageListener></messageListener>  </service>  </inbox>  <inbox>  <service name="console\_in">  <INITIAL\_CONTEXT\_FACTORY>org.apache.activemq.jndi.ActiveMQInitialContextFactory</INITIAL\_CONTEXT\_FACTORY>  <PROVIDER\_URL>tcp://127.0.0.1:61616</PROVIDER\_URL>  <queueName>dynamicQueues/console\_in.queue</queueName>  <overTime>0</overTime>  <isTransaction>false</isTransaction>  <autoAcknowledge>false</autoAcknowledge>  <sessionCounts>5</sessionCounts>  <messageListener></messageListener>  </service>  </inbox>  </inboxs>  <outboxs>  <outbox>  <service name="in\_out">  <INITIAL\_CONTEXT\_FACTORY>org.apache.activemq.jndi.ActiveMQInitialContextFactory</INITIAL\_CONTEXT\_FACTORY>  <PROVIDER\_URL>tcp://127.0.0.1:61616</PROVIDER\_URL>  <queueName>dynamicQueues/in\_out.queue</queueName>  <overTime>0</overTime>  <isTransaction>false</isTransaction>  <autoAcknowledge>true</autoAcknowledge>  <sessionCounts>5</sessionCounts>  <messageListener></messageListener>  </service>  </outbox>  <outbox>  <service name="in\_discenter">  <INITIAL\_CONTEXT\_FACTORY>org.apache.activemq.jndi.ActiveMQInitialContextFactory</INITIAL\_CONTEXT\_FACTORY>  <PROVIDER\_URL>tcp://127.0.0.1:61616</PROVIDER\_URL>  <queueName>dynamicQueues/in\_discenter.queue</queueName>  <overTime>0</overTime>  <isTransaction>false</isTransaction>  <autoAcknowledge>true</autoAcknowledge>  <sessionCounts>5</sessionCounts>  <messageListener></messageListener>  </service>  </outbox>  </outboxs>  </recipe> |

注：红色标注部分为待修改部分，即所有的PROVIDER\_URL标签的内容；

PROVIDER\_URL：该标签表示activemq的服务器ip地址和端口号，这两项配置需要和in容器连接的activemq服务的实际信息保持一致。例如：某个activemq服务的ip地址为172.30.12.35，

开放的服务端口为61617，而此in容器需要连接到该activemq上，则将所有的PROVIDER\_URL标签的值改为tcp://172.30.12.35:61617

另：如果activemq服务和in容器在同一台服务器上，ip地址使用127.0.0.1即可，如果不在同一台服务器上，则使用activemq所在服务器的真实ip地址。

1. 修改启停脚本

in容器的启动脚本为startIn.sh，停止脚本为stopIn.sh。

1. 修改启动脚本。

* 使用grep命令查看待修改参数的配置信息，具体命令如下：

|  |
| --- |
| grep -E 'CLOSE\_PORT=|DEBUG\_PORT=' /ztkx/cbpay/CrossBorderPay/bin/startIn.sh |

注：待修改内容含义如下：

CLOSE\_PORT：此配置为服务关闭端口。

DEBUG\_PORT：此配置为远程debug端口，用于远程调试程序。

* 验证端口是否被占用。

以root用户登录到新环境，然后使用netstat命令，确认远程debug端口和关闭端口（这两个端口的配置在startIn.sh和stopIn.sh文件中）是否被占用，命令如下：

netstat -anp | grep 7687//7687为关闭端口号

netstat -anp | grep 9999//9999为远程debug端口号

注：如果端口被占用则修改端口号，重新验证端口是否被占用，直到验证通过，记录下新端口号，并退出root用户。

* 切换回cbpay用户，使用vi命令进入文件startIn.sh文件，具体命令如下：

|  |
| --- |
| vi /ztkx/cbpay/CrossBorderPay/bin/startIn.sh |

* 进入文件后可看到如下配置信息

|  |
| --- |
| #!/bin/sh  # function: 此脚本功能是启动in模块  #a author: tianguangzhao  # version:1.0 2015/12/24  #设置编码格式，保证中文正确显示  export LANG=en\_US.UTF-8  #所需参数  #jar包路径  FILE\_PATH=run.jar  #模块名称  MODULAR\_NAME=IN  #关闭端口  CLOSE\_PORT=7687  #debug端口  DEBUG\_PORT=9999  #日志目录  NOHUP\_PATH="/ztkx/cbpay/logs/in\_nohup.out "  #监测进程是否已经启动  function check {  flag=`ps -ef|grep ${FILE\_PATH}|grep ${MODULAR\_NAME}|grep -v grep|grep ${LOGNAME}|wc -l`  if [ $? -ne 0 ]  then  echo "模块检查失败，模块名为 ${MODULAR\_NAME} !"  exit 1  fi  #根据查询进程判断，如果查询到相应进程，则模块已经启动  if [ $flag -ne 0 ]  then  echo "模块${MODULAR\_NAME}正在运行，无需重新启动 ! "  exit 1  fi  }  #程序入口  function start {  check  sh ./controlService.sh ${FILE\_PATH} ${MODULAR\_NAME} startup ${CLOSE\_PORT} ${NOHUP\_PATH} ${DEBUG\_PORT}  if [ $? -ne 0 ]  then  echo "模块启动失败，模块名为 $2，请检查 !"  exit 1  fi  }  start  exit 0 |

注：其中红色标注部分为待修改内容。将验证端口时记录的新端口号替换原端口号保存退出即可。

1. 修改停止脚本。

* 使用vi命令进入文件stopIn.sh文件，具体命令如下：

|  |
| --- |
| vi /ztkx/cbpay/CrossBorderPay/bin/stopIn.sh |

* 进入文件后可看到如下配置信息

|  |
| --- |
| #!/bin/sh  # function: 此脚本功能是关闭in模块  # author: tianguangzhao  # version:1.0 2015/12/24  #设置编码格式，保证中文正确显示  export LANG=en\_US.UTF-8  #所需参数  #jar包路径  FILE\_PATH=run.jar  #模块名称  MODULAR\_NAME=IN  #关闭端口  CLOSE\_PORT=7687  #日志目录和打印方式  NOHUP\_PATH=" /ztkx/cbpay/logs/in\_nohup.out "  #监测进程是否已经启动  function check {  flag=`ps -ef|grep ${FILE\_PATH}|grep ${MODULAR\_NAME}|grep -v grep|grep ${LOGNAME}|wc -l`  if [ $? -ne 0 ]  then  echo -e "模块检查失败，模块名为 ${MODULAR\_NAME} ! \n"  exit 1  fi  #根据查询进程判断，如果查询到相应进程，则模块已经启动  if [ $flag -eq 0 ]  then  echo -e "模块${MODULAR\_NAME}并未运行，无需停止 ! \n"  exit 1  fi  }  #程序入口  function start {  check  #参数说明：$1 是jar包路径,$2 模块名称,$3是相应操作,startup或者stop,$4是关闭端口,$5是报文位置  sh ./controlService.sh ${FILE\_PATH} ${MODULAR\_NAME} stop ${CLOSE\_PORT} ${NOHUP\_PATH}  if [ $? -ne 0 ]  then  echo -e "模块停止失败，模块名为 $2，请检查 ! \n"  exit 1  fi  }  start  exit 0 |

注：停止服务时，不需要设定远程调试端口，所以只需修改关闭端口（即CLOSE\_PORT参数）即可。

注意：停止脚本中的关闭端口必须和启动脚本中保持一致。

##### 部署out容器

1. 修改配置文件

out容器待修改配置文件如下：

1. dbpool.xml文件。

该文件用于保存out容器的连接池信息。

* 首先使用cbpay用户登录到新环境，然后使用vi命令打开该文件，具体命令如下：

|  |
| --- |
| vi /ztkx/cbpay/CrossBorderPay/config/out\_conf/dbpool/dbpool.xml |

进入文件后可看到如下配置信息：

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <c3p0-config>  <named-config name="myApp">  <property name="user">cbpay</property>  <property name="password">cbpay</property>  <property name="driverClass">oracle.jdbc.driver.OracleDriver</property>  <property name="jdbcUrl">jdbc:oracle:thin:@172.30.12.22:1521:fund</property>  <property name="initialPoolSize">10</property>  <property name="maxIdleTime">0</property>  <property name="maxIdleTimeExcessConnections">60</property>  <property name="maxPoolSize">30</property>  <property name="minPoolSize">5</property>  <property name="acquireIncrement">1</property>  <property name="idleConnectionTestPeriod">60</property>  <property name="acquireRetryAttempts">30</property>  <property name="acquireRetryDelay">1000</property>  <property name="testConnectionOnCheckout">false</property>  <property name="preferredTestQuery">select \* from dual</property>  </named-config>  </c3p0-config> |

注：红色标注的三项配置即是需要修改的内容，含义如下：

property name="user"标签中配置数据库用户名。

property name="password"配置用户密码。

property name="jdbcUrl"标签中配置数据库的url。

* 修改配置

根据实际数据库的信息进行修改即可。修改完成后保存退出！

1. flowno\_conf.properties文件。

该文件用于存放生成序列号时所需信息。

注：如果discenter分发容器下只有一路in容器，则该文件无需修改。

* 首先使用vi命令，打开该文件，具体命令如下：

|  |
| --- |
| vi /ztkx/cbpay/CrossBorderPay/config/out\_conf/flowno\_conf.properties |

进入文件后可看到如下配置信息:

|  |
| --- |
| #auto save  #Tue May 17 09:53:01 CST 2016  initsequence=1  steplength=2  currentsequence=170601  maxsequence=999999 |

注：红色标注部分即为待修改参数。含义如下：

Initsequence：该参数表示序列初始值。

Steplength：该参数表示序列增长的步长。

1. ActiveMQconfig.xml文件

该文件存放使用activemq发送消息和接受消息消息时的相关配置。

* 首先使用vi命令打开该文件，具体命令如下：

|  |
| --- |
| vi /ztkx/cbpay/CrossBorderPay/config/out\_conf/protocol/ActiveMQconfig.xml |

进入文件后，可看到如下配置信息：

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <recipe>  <inboxs>  <inbox>  <service name="in\_out">  <INITIAL\_CONTEXT\_FACTORY>org.apache.activemq.jndi.ActiveMQInitialContextFactory</INITIAL\_CONTEXT\_FACTORY>  <PROVIDER\_URL>tcp://127.0.0.1:61616</PROVIDER\_URL>  <queueName>dynamicQueues/in\_out.queue</queueName>  <overTime>0</overTime>  <isTransaction>false</isTransaction>  <autoAcknowledge>true</autoAcknowledge>  <sessionCounts>5</sessionCounts>  <messageListener></messageListener>  </service>  </inbox>  <!--  <inbox>  <service name="out\_in">  <INITIAL\_CONTEXT\_FACTORY>org.apache.activemq.jndi.ActiveMQInitialContextFactory</INITIAL\_CONTEXT\_FACTORY>  <PROVIDER\_URL>tcp://127.0.0.1:61616</PROVIDER\_URL>  <queueName>dynamicQueues/out\_in.queue</queueName>  <sessionCounts>5</sessionCounts>  <messageListener></messageListener>  </service>  </inbox>  <inbox>  <service name="console\_in">  <INITIAL\_CONTEXT\_FACTORY>org.apache.activemq.jndi.ActiveMQInitialContextFactory</INITIAL\_CONTEXT\_FACTORY>  <PROVIDER\_URL>tcp://127.0.0.1:61616</PROVIDER\_URL>  <queueName>dynamicQueues/console\_in.queue</queueName>  <sessionCounts>5</sessionCounts>  </service>  </inbox>  -->  </inboxs>  <outboxs>  <outbox>  <service name="out\_in">  <INITIAL\_CONTEXT\_FACTORY>org.apache.activemq.jndi.ActiveMQInitialContextFactory</INITIAL\_CONTEXT\_FACTORY>  <PROVIDER\_URL>tcp://127.0.0.1:61616</PROVIDER\_URL>  <queueName>dynamicQueues/out\_in.queue</queueName>  <overTime>0</overTime>  <isTransaction>false</isTransaction>  <autoAcknowledge>true</autoAcknowledge>  <sessionCounts>5</sessionCounts>  <messageListener></messageListener>  </service>  </outbox>  <!--  <outbox>  <service name="in\_discenter">  <INITIAL\_CONTEXT\_FACTORY>org.apache.activemq.jndi.ActiveMQInitialContextFactory</INITIAL\_CONTEXT\_FACTORY>  <PROVIDER\_URL>tcp://127.0.0.1:61616</PROVIDER\_URL>  <queueName>dynamicQueues/in\_discenter.queue</queueName>  <sessionCounts>5</sessionCounts>  </service>  </outbox>  -->  </outboxs>  </recipe> |

注：红色标注部分为待修改部分，即所有的PROVIDER\_URL标签的内容(注释掉的配置除外)；

* 修改配置信息

PROVIDER\_URL：该标签表示activemq的服务器ip地址和端口号，这两项配置需要和out容器连接的activemq服务的实际信息保持一致。例如：某个activemq服务的ip地址为172.30.12.35，

开放的服务端口为61617，而此out容器需要连接到该activemq上，则将所有的PROVIDER\_URL标签的值改为tcp://172.30.12.35:61617

另：如果activemq服务和out容器在同一台服务器上，ip地址使用127.0.0.1即可，如果不在同一台服务器上，则使用activemq所在服务器的真实ip地址。

1. 修改启停脚本

out容器的启动脚本为startOut.sh，停止脚本为stopOut.sh。

1. 修改启动脚本。

* 使用grep命令查看待修改参数的配置信息，具体命令如下：

|  |
| --- |
| grep -E 'CLOSE\_PORT=|DEBUG\_PORT=' /ztkx/cbpay/CrossBorderPay/bin/startOut.sh |

注：待修改内容含义如下：

CLOSE\_PORT：此配置为服务关闭端口。

DEBUG\_PORT：此配置为远程debug端口，用于远程调试程序。

* 验证端口是否被占用。

以root用户登录到新环境，然后使用netstat命令，确认远程debug端口和关闭端口（这两个端口的配置在startOut.sh和stopOut.sh文件中）是否被占用，命令如下：

netstat -anp | grep 7686//7686为关闭端口号

netstat -anp | grep 9998//9998为远程debug端口号

如果端口被占用则修改端口号，重新验证端口是否被占用，直至验证通过，记录下新端口号，并退出root用户。

* 使用vi命令进入文件startOut.sh文件，具体命令如下：

|  |
| --- |
| vi /ztkx/cbpay/CrossBorderPay/bin/startOut.sh |

* 进入文件后可看到如下配置信息

|  |
| --- |
| #!/bin/sh  # function: 此脚本功能是启动out模块  # author: tianguangzhao  # version:1.0 2015/12/24  #设置编码格式，保证中文正确显示  export LANG=en\_US.UTF-8  #所需参数  #jar包路径  FILE\_PATH=run.jar  #模块名称  MODULAR\_NAME=OUT  #关闭端口  CLOSE\_PORT=7686  #debug端口  DEBUG\_PORT=9998  #日志目录和打印方式  NOHUP\_PATH="/ztkx/cbpay/logs/out\_nohup.out"  #监测进程是否已经启动  function check {  flag=`ps -ef|grep ${FILE\_PATH}|grep ${MODULAR\_NAME}|grep -v grep|grep ${LOGNAME}|wc -l`  if [ $? -ne 0 ]  then  echo "模块检查失败，模块名为 ${MODULAR\_NAME} !"  exit 1  fi  #根据查询进程判断，如果查询到相应进程，则模块已经启动  if [ $flag -ne 0 ]  then  echo "模块${MODULAR\_NAME}正在运行，无需重新启动 ! "  exit 1  fi  }  #程序入口  function start {  check  sh ./controlService.sh ${FILE\_PATH} ${MODULAR\_NAME} startup ${CLOSE\_PORT} ${NOHUP\_PATH} ${DEBUG\_PORT}  if [ $? -ne 0 ]  then  echo "模块启动失败，模块名为 $2，请检查 !"  exit 1  fi  }  start  exit 0 |

注：其中红色标注部分为待修改内容。将验证端口时记录的新端口号替换原端口号保存退出即可。

1. 修改停止脚本。

* 使用vi命令进入文件stopOut.sh文件，具体命令如下：

|  |
| --- |
| vi /ztkx/cbpay/CrossBorderPay/bin/stopOut.sh |

* 进入文件后可看到如下配置信息

|  |
| --- |
| #!/bin/sh  # function: 此脚本功能是关闭out模块  # author: tianguangzhao  # version:1.0 2015/12/24  #设置编码格式，保证中文正确显示  export LANG=en\_US.UTF-8  #所需参数  #jar包路径  FILE\_PATH=run.jar  #模块名称  MODULAR\_NAME=OUT  #关闭端口  CLOSE\_PORT=7686  #debug端口  DEBUG\_PORT=  #日志目录和打印方式  NOHUP\_PATH=" /ztkx/cbpay/logs/out\_nohup.out "  #监测进程是否已经启动  function check {  flag=`ps -ef|grep ${FILE\_PATH}|grep ${MODULAR\_NAME}|grep -v grep|grep ${LOGNAME}|wc -l`  if [ $? -ne 0 ]  then  echo -e "模块检查失败，模块名为 ${MODULAR\_NAME} ! \n"  exit 1  fi  #根据查询进程判断，如果查询到相应进程，则模块已经启动  if [ $flag -eq 0 ]  then  echo -e "模块${MODULAR\_NAME}并未运行，无需停止 ! \n"  exit 1  fi  }  #程序入口  function start {  check  #参数说明：$1 是jar包路径,$2 模块名称,$3是相应操作,startup或者stop,$4是关闭端口,$5是报文位置  sh ./controlService.sh ${FILE\_PATH} ${MODULAR\_NAME} stop ${CLOSE\_PORT} ${NOHUP\_PATH} ${DEBUG\_PORT}  if [ $? -ne 0 ]  then  echo -e "模块停止失败，模块名为 $2，请检查 ! \n"  exit 1  fi  }  start  exit 0 |

注：停止服务时，不需要设定远程调试端口，所以只需修改关闭端口（即CLOSE\_PORT参数）即可。

注意：停止脚本中的关闭端口必须和启动脚本中保持一致。

##### 部署discenter容器

1. 修改配置文件

discenter容器待修改配置文件如下：

1. dbpool.xml文件。

该文件用于保存in容器的连接池信息。

* 首先使用cbpay用户登录到新环境，然后使用vi命令打开该文件，具体命令如下：

|  |
| --- |
| vi /ztkx/cbpay/CrossBorderPay/config/in\_conf/dbpool/dbpool.xml |

进入文件后可看到如下配置信息：

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <c3p0-config>  <named-config name="myApp">  <property name="user">cbpay</property>  <property name="password">cbpay</property>  <property name="driverClass">oracle.jdbc.driver.OracleDriver</property>  <property name="jdbcUrl">jdbc:oracle:thin:@172.30.12.22:1521:fund</property>  <property name="initialPoolSize">10</property>  <property name="maxIdleTime">0</property>  <property name="maxIdleTimeExcessConnections">60</property>  <property name="maxPoolSize">30</property>  <property name="minPoolSize">5</property>  <property name="acquireIncrement">1</property>  <property name="idleConnectionTestPeriod">60</property>  <property name="acquireRetryAttempts">30</property>  <property name="acquireRetryDelay">1000</property>  <property name="testConnectionOnCheckout">false</property>  <property name="preferredTestQuery">select \* from dual</property>  </named-config>  </c3p0-config> |

注：红色标注的三项配置即是需要修改的内容，含义如下：

property name="user"标签中配置数据库用户名。

property name="password"配置用户密码。

property name="jdbcUrl"标签中配置数据库的url。

* 修改配置

根据实际数据库的信息进行修改即可。修改完成后保存退出！

1. ActiveMQconfig.xml文件

该文件存放使用activemq发送消息和接受消息消息时的相关配置。

* 首先使用vi命令打开该文件，具体命令如下：

|  |
| --- |
| vi /ztkx/discenter/discenter/config/protocol/ActiveMQconfig.xml |

进入文件后，可看到如下配置信息：

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <recipe>  <inboxs>  <!--  <inbox>  <service name="cashier\_discenter">  <INITIAL\_CONTEXT\_FACTORY>org.apache.activemq.jndi.ActiveMQInitialContextFactory</INITIAL\_CONTEXT\_FACTORY>  <PROVIDER\_URL>tcp://127.0.0.1:61616</PROVIDER\_URL>  <queueName>dynamicQueues/cashier\_discenter.queue</queueName>  <overTime>0</overTime>  <isTransaction>false</isTransaction>  <autoAcknowledge>true</autoAcknowledge>  <sessionCounts>5</sessionCounts>  <messageListener></messageListener>  </service>  </inbox>  -->  <inbox>  <service name="console\_other">  <INITIAL\_CONTEXT\_FACTORY>org.apache.activemq.jndi.ActiveMQInitialContextFactory</INITIAL\_CONTEXT\_FACTORY>  <PROVIDER\_URL>tcp://172.0.0.1:61616</PROVIDER\_URL>  <queueName>dynamicQueues/console\_other.queue</queueName>  <overTime>0</overTime>  <isTransaction>false</isTransaction>  <autoAcknowledge>true</autoAcknowledge>  <sessionCounts>3</sessionCounts>  <messageListener>com.ztkx.cbpay.container.msglistener.CommanMsgListener</messageListener>  <selector>containid='{CONTAINER\_NAME}'</selector>  </service>  </inbox>  </inboxs>  <outboxs>  <!--  <outbox>  <service name="in\_out">  <INITIAL\_CONTEXT\_FACTORY>org.apache.activemq.jndi.ActiveMQInitialContextFactory</INITIAL\_CONTEXT\_FACTORY>  <PROVIDER\_URL>tcp://127.0.0.1:61616</PROVIDER\_URL>  <queueName>dynamicQueues/in\_out.queue</queueName>  <overTime>0</overTime>  <isTransaction>false</isTransaction>  <autoAcknowledge>false</autoAcknowledge>  <sessionCounts>5</sessionCounts>  <messageListener></messageListener>  </service>  </outbox>  -->  <outbox>  <service name="console\_other">  <INITIAL\_CONTEXT\_FACTORY>org.apache.activemq.jndi.ActiveMQInitialContextFactory</INITIAL\_CONTEXT\_FACTORY>  <PROVIDER\_URL>tcp://172.0.0.1:61616</PROVIDER\_URL>  <queueName>dynamicQueues/console\_other.queue</queueName>  <overTime>0</overTime>  <isTransaction>false</isTransaction>  <autoAcknowledge>true</autoAcknowledge>  <sessionCounts>3</sessionCounts>  <messageListener>com.ztkx.cbpay.container.msglistener.CommanMsgListener</messageListener>  <selector>containid='{CONTAINER\_NAME}'</selector>  </service>  </outbox>  </outboxs>  </recipe> |

注：红色标注部分为待修改部分，即所有的PROVIDER\_URL标签的内容(注释掉的配置除外)；

* 修改配置信息

PROVIDER\_URL：该标签表示activemq的服务器ip地址和端口号，这两项配置需要和discenter容器连接的activemq服务的实际信息保持一致。例如：某个activemq服务的ip地址为172.30.12.35，

开放的服务端口为61617，而此discenter容器需要连接到该activemq上，则将所有的PROVIDER\_URL标签的值改为tcp://172.30.12.35:61617

另：如果activemq服务和discenter容器在同一台服务器上，ip地址使用127.0.0.1即可，如果不在同一台服务器上，则使用activemq所在服务器的真实ip地址。

1. 修改启停脚本

discenter容器启动脚本为startDis.sh，停止脚本为stopDis.sh。

1. 修改启动脚本。

* 使用grep命令查看待修改参数的配置信息，具体命令如下：

|  |
| --- |
| grep -E 'CLOSE\_PORT=|DEBUG\_PORT=' /ztkx/discenter/discenter/bin/startDis.sh |

注：待修改内容含义如下：

CLOSE\_PORT：此配置为服务关闭端口。

DEBUG\_PORT：此配置为远程debug端口，用于远程调试程序。

* 验证端口是否被占用。

以root用户登录到新环境，然后使用netstat命令，确认远程debug端口和关闭端口（这两个端口的配置在startDis.sh和stopDis.sh文件中）是否被占用，命令如下：

netstat -anp | grep 7689//7689为关闭端口号

netstat -anp | grep 9997//9997为远程debug端口号

如果端口被占用则修改端口号，重新验证端口是否被占用，直至验证通过，记录下新端口号，并退出root用户。

* 使用vi命令进入文件startDis.sh文件，具体命令如下：

|  |
| --- |
| vi /ztkx/discenter/discenter/bin/startDis.sh |

* 进入文件后可看到如下配置信息

|  |
| --- |
| #!/bin/sh  # function: 此脚本功能是启动discenter模块  # author: tianguangzhao  # version:1.0 2015/12/24  #设置编码格式，保证中文正确显示  export LANG=en\_US.UTF-8  #所需参数  #jar包路径  FILE\_PATH=run.jar  #模块名称  MODULAR\_NAME=DISCENTER  #关闭端口  CLOSE\_PORT=7690  #debug端口  DEBUG\_PORT=9890  #日志目录  NOHUP\_PATH="/ztkx/discenter/logs/dis\_nohup.out "  #监测进程是否已经启动  function check {  flag=`ps -ef|grep ${FILE\_PATH}|grep ${MODULAR\_NAME}|grep -v grep|wc -l`  if [ $? -ne 0 ]  then  echo "模块检查失败，模块名为 ${MODULAR\_NAME} !"  exit 1  fi  #根据查询进程判断，如果查询到相应进程，则模块已经启动  if [ $flag -ne 0 ]  then  echo "模块${MODULAR\_NAME}正在运行，无需重新启动 ! "  exit 1  fi  }  #程序入口  function start {  check  sh ./controlService.sh ${FILE\_PATH} ${MODULAR\_NAME} startup ${CLOSE\_PORT} ${NOHUP\_PATH} ${DEBUG\_PORT}  if [ $? -ne 0 ]  then  echo "模块启动失败，模块名为 $2，请检查 !"  exit 1  fi  }  start  exit 0 |

注：其中红色标注部分为待修改内容。将验证端口时记录的新端口号替换原端口号保存退出即可。

1. 修改停止脚本。

* 使用vi命令进入文件stopDis.sh文件，具体命令如下：

|  |
| --- |
| vi /ztkx/discenter/discenter/bin/stopDis.sh |

* 进入文件后可看到如下配置信息

|  |
| --- |
| #!/bin/sh  # function: 此脚本功能是关闭discenter模块  # author: tianguangzhao  # version:1.0 2015/12/24  #设置编码格式，保证中文正确显示  export LANG=en\_US.UTF-8  #所需参数  #jar包路径  FILE\_PATH=run.jar  #模块名称  MODULAR\_NAME=DISCENTER  #关闭端口  CLOSE\_PORT=7690  #日志目录和打印方式  NOHUP\_PATH=" /ztkx/discenter/logs/dis\_nohup.out "  #监测进程是否已经启动  function check {  flag=`ps -ef|grep ${FILE\_PATH}|grep ${MODULAR\_NAME}|grep -v grep|wc -l`  if [ $? -ne 0 ]  then  echo -e "模块检查失败，模块名为 ${MODULAR\_NAME} ! \n"  exit 1  fi  #根据查询进程判断，如果查询到相应进程，则模块已经启动  if [ $flag -eq 0 ]  then  echo -e "模块${MODULAR\_NAME}并未运行，无需停止 ! \n"  exit 1  fi  }  #程序入口  function start {  check  #参数说明：$1 是jar包路径,$2 模块名称,$3是相应操作,startup或者stop,$4是关闭端口,$5是报文位置  sh ./controlService.sh ${FILE\_PATH} ${MODULAR\_NAME} stop ${CLOSE\_PORT} ${NOHUP\_PATH}  if [ $? -ne 0 ]  then  echo -e "模块停止失败，模块名为 $2，请检查 ! \n"  exit 1  fi  }  start  exit 0 |

注：停止服务时，不需要设定远程调试端口，所以只需修改关闭端口（即CLOSE\_PORT参数）即可。

注意：停止脚本中的关闭端口必须和启动脚本中保持一致。

### 部署收银台

1. 修改配置文件
2. baseparam.properties文件

该文件用于存放收银台服务运行时需要的基本配置。

* 首先使用cbpay用户登录到新环境，然后使用vi命令打开该文件

具体命令如下：

|  |
| --- |
| vi /ztkx/discenter/cashier/apache-tomcat-7.0.64/webapps/cashier/WEB-INF/classes/baseparam.properties |

打开文件后，看到如下配置：

|  |
| --- |
| <!--http写入的超时时间-->  connectTimeout=10000  <!--http读取的超时时间-->  readTimeout=10000  <!--编码格式-->  encode=GBK  <!--文件编码格式-->  file\_encode=utf-8  <!--discenter\_host-->  discenter\_host=172.30.12.35  <!--discenter\_port-->  discenter\_port=60001  <!--url的后缀-->  suffix=.action  <!--传输内容的参数-->  param=CASHIERDATA  <!--报文最大长度-->  maxLength=99999  <!--报文格式的截取-->  msgheadlength=6  <!--是否立即购汇 0立即 1不立即-->  nowbuybat=1 |

注：红色标注部分即为待修改部分；

discenter\_host：该参数表示处理请求的discenter的ip地址，

discenter\_port：该参数表示处理请求的discenter的端口号。

此处配置应该与数据库中的协议配置保持一致。

* 查询数据库中的配置

以cbpay用户登录到（discenter容器连接的）数据库，运行以下sql。

|  |
| --- |
| select xmlconf from protocol t where serverid ='CASHIER\_CLI' |

查询到如下内容：

|  |
| --- |
| <protocol><common id="cashier\_cli" type="tcp" flag="client" inOut="DataIn/DataOut" mode="syn" encoding="UTF-8" sessionCount="5" connecMode="short"/><request host="127.0.0.1"port="60001" policy="dynamic\_len:start:0,end:5" dataParams="" file="" encryption="" connectTimeout="60000" method="POST"/><response policy="dynamic\_len:start:0,end:5" readTimeout="60000" /></protocol> |

* 修改配置内容

修改discenter\_host参数与discenter所在服务器保持一致，如果discenter容器部署在172.30.12.35机器上，则discenter\_host=172.30.12.35。

修改discenter\_port：该参数与查询数据库记录中的port参数的值保持一致（如上所示则为60001）。

### 部署console服务

1. 修改配置文件
2. baseparam.properties文件

该文件用于存放收银台服务运行时需要的基本配置。

* 首先使用discenter用户登录到新环境，然后使用vi命令打开该文件

具体命令如下：

|  |
| --- |
| vi /ztkx/discenter/console/config/baseConf.properties |

打开文件后，看到如下配置：

|  |
| --- |
| #console发送消息到discenter用到  tcpserverhost=172.30.12.35  tcpserverport=8812  tcpconnecttimeout=60000  tcpreadtimeout=60000  #主配置文件  #容器名称  CONTAINER\_NAME=CONSOLE  #mq重连时间间隔  mq.reconnection.interval=5000  #jms消息超时时间  JMS\_MESSAGE\_OVER\_TIME=60000 |

注：红色标注部分即为待修改部分；

tcpserverhost：该参数表示处理请求的discenter的ip地址，

tcpserverport：该参数表示处理请求的discenter的端口号。

此处配置应该与数据库中的协议配置保持一致。

* 查询数据库中的配置

以cbpay用户登录到（discenter容器连接的）数据库，运行以下sql。

|  |
| --- |
| select xmlconf from protocol t where serverid ='CONSOLE\_CLI' |

查询到如下内容：

|  |
| --- |
| <protocol><common id="console\_cli" type="tcp" flag="server" inOut="DataIn/DataOut" mode="syn" encoding="GBK" sessionCount="5" connecMode="short"/><request host="127.0.0.1" port="8812" policy="dynamic\_len:start:0,end:5" dataParams="" file="" encryption="" connectTimeout="60000" method=""/><response policy="dynamic\_len:start:0,end:5" dataParams="" readTimeout="60000" /></protocol> |

* 修改配置内容：
* 修改tcpserverhost参数与discenter所在服务器保持一致，如果discenter容器部署在172.30.12.35机器上，则tcpserverhost=172.30.12.35。
* 修改tcpserverport：该参数与查询数据库记录中的port参数的值保持一致（如上所示则为8812）。

1. ActiveMQconfig.xml文件。

该文件存放使用activemq发送消息和接受消息消息时的相关配置。

* 首先使用vi命令打开该文件，具体命令如下：

|  |
| --- |
| vi /ztkx/discenter/console/config/protocol/ActiveMQconfig.xml |

进入文件后，可看到如下配置信息：

|  |
| --- |
| <recipe>  <inboxs>  <inbox>  <service name="console\_other">  <INITIAL\_CONTEXT\_FACTORY>org.apache.activemq.jndi.ActiveMQInitialContextFactory</INITIAL\_CONTEXT\_FACTORY>  <PROVIDER\_URL>tcp://172.30.12.35:61616</PROVIDER\_URL>  <queueName>dynamicQueues/console\_other.queue</queueName>  <overTime>0</overTime>  <isTransaction>false</isTransaction>  <autoAcknowledge>true</autoAcknowledge>  <sessionCounts>3</sessionCounts>  <messageListener>cn.msec.cbpay.listener.ConsoleComMsgListener</messageListener>  <selector>containid='CONSOLE'</selector>  </service>  </inbox>  </inboxs>  <outboxs>  <outbox>  <service name="console\_other">  <INITIAL\_CONTEXT\_FACTORY>org.apache.activemq.jndi.ActiveMQInitialContextFactory</INITIAL\_CONTEXT\_FACTORY>  <PROVIDER\_URL>tcp://172.30.12.35:61616</PROVIDER\_URL>  <queueName>dynamicQueues/console\_other.queue</queueName>  <overTime>0</overTime>  <isTransaction>false</isTransaction>  <autoAcknowledge>true</autoAcknowledge>  <sessionCounts>3</sessionCounts>  <messageListener>cn.msec.cbpay.listener.ConsoleComMsgListener</messageListener>  <selector>containid='CONSOLE'</selector>  </service>  </outbox>  </outboxs>  </recipe> |

注：红色标注部分为待修改部分，即所有的PROVIDER\_URL标签的内容(注释掉的配置除外)；

* 修改配置信息

PROVIDER\_URL：该标签表示activemq的服务器ip地址和端口号，这两项配置需要和console服务连接的activemq服务的实际信息保持一致。例如：某个activemq服务的ip地址为172.30.12.35，

开放的服务端口为61617，而此console服务需要连接到该activemq上，则将所有的PROVIDER\_URL标签的值改为tcp://172.30.12.35:61617

另：如果activemq服务和console服务在同一台服务器上，ip地址使用127.0.0.1即可，如果不在同一台服务器上，则使用activemq所在服务器的真实ip地址。

1. SpringContext-oracle.xml文件

该文件用于存放console服务连接数据库时的配置信息。

* 首先使用vi命令打开该文件，具体命令如下：

|  |
| --- |
| vi /ztkx/discenter/console/config/SpringContext-oracle.xml |

打开此文件后，可看到如下配置信息：

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <beans xmlns="http://www.springframework.org/schema/beans"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:jdbc="http://www.springframework.org/schema/jdbc"  xmlns:context="http://www.springframework.org/schema/context" xmlns:tx="http://www.springframework.org/schema/tx"  xmlns:aop="http://www.springframework.org/schema/aop"  xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd  http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx.xsd  http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd  http://www.springframework.org/schema/jdbc http://www.springframework.org/schema/jdbc/spring-jdbc.xsd  http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop.xsd">  <bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource" destroy-method="close">  <property name="driverClass" value="oracle.jdbc.driver.OracleDriver" />  <property name="jdbcUrl" value="jdbc:oracle:thin:@172.30.12.22:1521:fund" />  <property name="user" value="cbpay" />  <property name="password" value="cbpay" />  <property name="initialPoolSize" value="2"/>  <property name="maxPoolSize" value="2"/>  <property name="minPoolSize" value="2"/>  </bean>  <bean id="gsettleJdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">  <property name="dataSource" ref="dataSource" />  </bean>  <bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">  <property name="dataSource" ref="dataSource" />  <property name="configLocation" value="classpath:/sql-map-config.xml"></property>  </bean>  <bean id="sqlSession" class="org.mybatis.spring.SqlSessionTemplate">  <constructor-arg index="0" ref="sqlSessionFactory" />  </bean>  <bean id="transactionManager" class="org.springframework.jdbc.datasource.DataSourceTransactionManager">  <property name="dataSource" ref="dataSource" />  </bean>  <tx:annotation-driven transaction-manager="transactionManager" />  <tx:advice id="OracleAdvice" transaction-manager="transactionManager">  <tx:attributes>  <tx:method name="\*elete\*" propagation="REQUIRED" />  <tx:method name="\*pdate\*" propagation="REQUIRED" />  <tx:method name="\*nsert\*" propagation="REQUIRED" />  <tx:method name="\*add\*" propagation="REQUIRED" />  </tx:attributes>  </tx:advice>  <bean id="commonSqlMapper" class="org.mybatis.spring.mapper.MapperFactoryBean">  <property name="mapperInterface" value="cn.msec.rest.db.dao.CommonSqlMapper" />  <property name="sqlSessionFactory" ref="sqlSessionFactory" />  </bean>  <!-- 配置参与事务的类 -->  <aop:config>  <aop:pointcut id="allTestServiceMethod"  expression="execution(\* cn.msec.rest.db.\*.dao.\*.\*(..)) || execution(\* cn.msec.rest.db.service.impl.\*.\*(..))||execution(\* cn.msec.cbpay.action.\*.\*(..))" />  <aop:advisor pointcut-ref="allTestServiceMethod" advice-ref="OracleAdvice" />  </aop:config>  </beans> |

注：红色标注部分即为待修改参数，含义如下：

property name="jdbcUrl"标签中配置数据库的url地址；

property name="user"标签中存放数据库用户名

property name="password"标签中存放用户密码

* 修改配置文件

根据console服务所连数据库的实际信息修改即可！

<!—下面4,5步为测试的环境配置，不用测试模块的不需要配置-->

4)修改测试配置文件vi /ztkx/discenter/console/config/test.properties

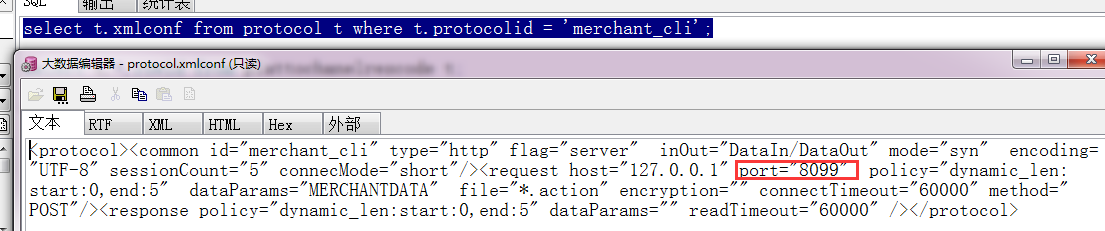
|  |
| --- |
| #测试加密解密类  keyStoreFile=/ztkx/discenter/console/cers/zlex.keystore  keyStoreAlias=www.zlex.org  keyStorePassword=123456  publicCertificateFile=/ztkx/discenter/console/cers/zlex.cer  #测试HTTP请求参数  url=http://172.30.12.35:8099  connectTimeout=30000  readTimeout=30000 |

上表格红色部分需要根据具体环境位置。

url：为discenter容器所在地址。

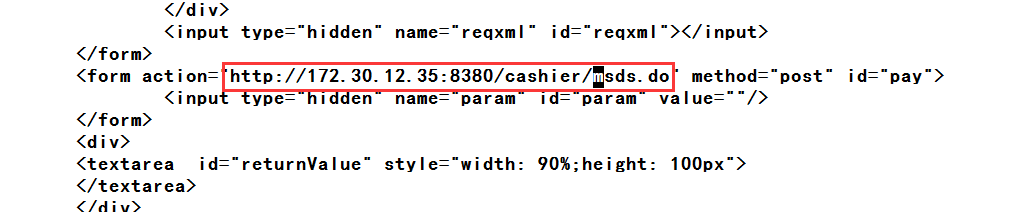
Port：根据数据库中

select t.xmlconf from protocol t where t.protocolid = 'merchant\_cli';



5）修改支付报文发送位置vi /ztkx/discenter/console/apache-tomcat-7.0.64/webapps/console/www/cbpay/cashier/testpay.html

查找msds.do发现



Ip：为收银台地址。

Port:为收银台配置的端口。

剩余的不做改变。

### 部署定时任务

1. 修改配置文件
2. baseConf.properties文件

该文件用于配置定时任务运行时所需的基本参数。

* 首先使用vi命令打开该文件，具体命令如下：

|  |
| --- |
| vi /ztkx/cbpay/timedtask/config/baseConf.properties |

打开该文件后，可看到如下配置信息:

|  |
| --- |
| #主配置文件  #容器名称  CONTAINER\_NAME=TIMEDTASK  #in容器消息监听listener  messagelistener=com.ztkx.cbpay.container.msglistener.TempletMsgListener  #mq重连时间间隔  mq.reconnection.interval=5000  #jms消息超时时间  JMS\_MESSAGE\_OVER\_TIME=6000  #设置对账文件目录  CHECKING\_FILE\_PATH=/ztkx/cbpay/file  #通讯地址配置  tcpserverhost=172.30.12.35  tcpserverport=8811  tcpconnecttimeout=60000  tcpreadtimeout=60000  #和计费交互时使用dubbo用到  applicationname=consumer-hello-word-app  registryaddress=zookeeper://10.1.72.169:2181  serviceversion=1.0.0 |

注：红色标注部分即为待修改部分，含义如下；

tcpserverhost：该参数表示处理请求的discenter的ip地址，

tcpserverport：该参数表示处理请求的discenter的端口号。

此处配置应该与数据库中的协议配置保持一致。

* 查询数据库中的配置

以cbpay用户登录到（discenter容器连接的）数据库，运行以下sql。

|  |
| --- |
| select xmlconf from protocol t where serverid ='CRONTAB\_CLI' |

查询到如下内容：

|  |
| --- |
| <protocol><common id="crontab\_cli" type="tcp" flag="server" inOut="DataIn/DataOut" mode="syn" encoding="GBK" sessionCount="5" connecMode="short"/><request host="127.0.0.1" port="8811" policy="dynamic\_len:start:0,end:5" dataParams="" file="" encryption="" connectTimeout="60000" method=""/><response policy="dynamic\_len:start:0,end:5" dataParams="" readTimeout="60000" /></protocol> |

* 修改配置内容：
* 修改tcpserverhost参数与discenter所在服务器保持一致，如果discenter容器部署在172.30.12.35机器上，则tcpserverhost=172.30.12.35。
* 修改tcpserverport：该参数与查询数据库记录中的port参数的值保持一致（如上所示则为8811）。

1. dbpool.xml文件。

该文件用于保存定时任务的连接池信息。

* 首先使用cbpay用户登录到新环境，然后使用vi命令打开该文件，具体命令如下：

|  |
| --- |
| vi /ztkx/cbpay/timedtask/config/dbpool/dbpool.xml |

* 进入文件后可看到如下配置信息：

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <c3p0-config>  <named-config name="myApp">  <property name="user">cbpay</property>  <property name="password">cbpay</property>  <property name="driverClass">oracle.jdbc.driver.OracleDriver</property>  <property name="jdbcUrl">jdbc:oracle:thin:@172.30.12.22:1521:fund</property>  <property name="initialPoolSize">10</property>  <property name="maxIdleTime">0</property>  <property name="maxIdleTimeExcessConnections">60</property>  <property name="maxPoolSize">30</property>  <property name="minPoolSize">5</property>  <property name="acquireIncrement">1</property>  <property name="idleConnectionTestPeriod">60</property>  <property name="acquireRetryAttempts">30</property>  <property name="acquireRetryDelay">1000</property>  <property name="testConnectionOnCheckout">false</property>  <property name="preferredTestQuery">select \* from dual</property>  </named-config>  </c3p0-config> |

注：红色标注的三项配置即是需要修改的内容，含义如下：

property name="user"标签中配置数据库用户名。

property name="password"配置用户密码。

property name="jdbcUrl"标签中配置数据库的url。

* 修改配置

根据定时任务所连实际数据库的信息进行修改即可。修改完成后保存退出！

## 第三章部署数据库

### 安装数据库

跨境支付系统所使用的数据库为oracle 版本号为（11.2.0.3.0）。安装步骤如下：

待完善！

### 新建数据库

#### 新建数据库实例

待完善！

#### 创建数据库用户cbpay

* 首先使用oracle用户登录到数据库服务器。
* 使用sqlplus命令以dba身份登录到oracle数据，具体命令如下：

|  |
| --- |
| sqlplus "/as sysdba" |

* 创建用户cbpay,具体命令如下：

create user cbpay identified by cbpay;

注：第一个cbpay为数据库用户名，第二个cbpay表示该用户的密码！

* 为用户cbpay赋权限，具体命令如下：

grant dba to cbpay;

grant connect to cbpay;

grant resource to cbpay;

* 使用exit命令退出。具体如下：

|  |
| --- |
| exit |

#### 新建表空间

* 使用sqlplus命令，以cbpay用户进入命令模式，具体如下：

|  |
| --- |
| sqlplus cbpay/cbpay |

注：第一个cbpay为数据库用户，第二个cbpay为cbpay用户的密码！

* 新建表空间，一次输入以下两条sql命令，如下所示：

create tablespace CBPAY\_JOURNAL datafile '/oradata/cbpay/cbpay\_journal01.dbf' size 500m AUTOEXTEND off;

create tablespace CBPAY\_CONF datafile '/oradata/cbpay/cbpay\_conf01.dbf' size 500m AUTOEXTEND off;

* 使用exit命令退出。具体如下：

|  |
| --- |
| exit |

### 导入数据

1. 数据导出和导入

* 首先以oracle用户登录到原数据
* 用cbpay用户从原数据库中导出数据，命令如下：

expcbpay/cbpay file=/home/oracle/cbpaydata/cbpay.dmp

注：第一个cbpay为数据库用户名，第二个cbpay表示该数据库用户的密码，/home/oracle/cbpaydata/cbpay.dmp为存放导出数据的文件。

* 将文件上传到新服务器

可使用sftp命令，也可使用ftp工具等实现。在此不再赘述。

* 使用exit命令退出原数据库服务器。具体如下：

|  |
| --- |
| exit |

* 以oracle用户登录到新数据库服务器。
* 以cbpay用户将数据导入数据库中，命令如下：

imp cbpay/cbpay@fund file=/home/oracle/cbpaydata/cbpay.dmp FULL=Y

* 使用exit命令退出。具体如下：

|  |
| --- |
| exit |

1. 数据验证

* 使用PLSQL Developer（或其他数据库工具）连接到到数据库，用户名cbpay，密码cbpay。案例如下：
* 使用如下sql命令查询数据库中的数据，具体如下：

|  |
| --- |
| select t.\*, t.rowid from b\_merchant\_info t |

注：查看数据库中是否存在数据以及数据是否存在乱码问题

如果验证数据无问题，则数据迁移成功！

## 第四章启动服务

### 启动activemq

Activemq服务用于系统内部进行消息的交互，启动脚本为activemq。

* 启动activemq服务，命令如下：

|  |
| --- |
| /ztkx/cbpay/activemq/bin/activemq start |

* 如需停止activemq服务，命令如下：

|  |
| --- |
| /ztkx/cbpay/activemq/bin/activemq stop |

### 启动b2bic

b2bic用于和平安银行交互时发送和接收报文，并对报文进行加解密

（1）b2bic启动脚本为startup.sh，关闭脚本为close.sh。使用startup.sh命令启动b2bic服务，具体用法如下：

|  |
| --- |
| /ztkx/cbpay/b2bic/startup.sh |

* 如需停止b2bic服务，使用close.sh，具体用法如下：

|  |
| --- |
| /ztkx/cbpay/b2bic/close.sh |

（2）修改配置；

启动b2bic之后，进入页面配置信息。

1. 首先打开浏览器，在浏览器中输入如下url。

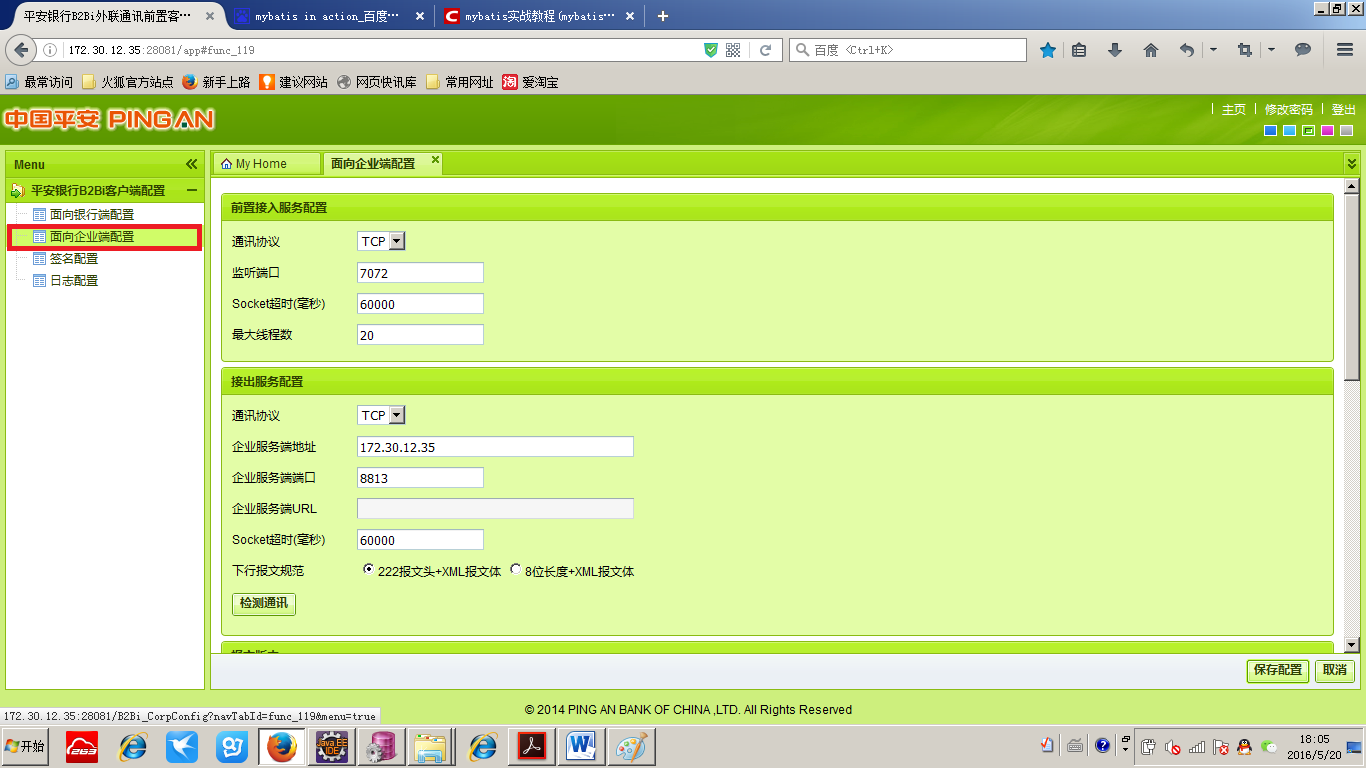
|  |
| --- |
| http://172.30.12.33:28081/app#func\_119 |

注：172.30.12.33为新环境的ip地址

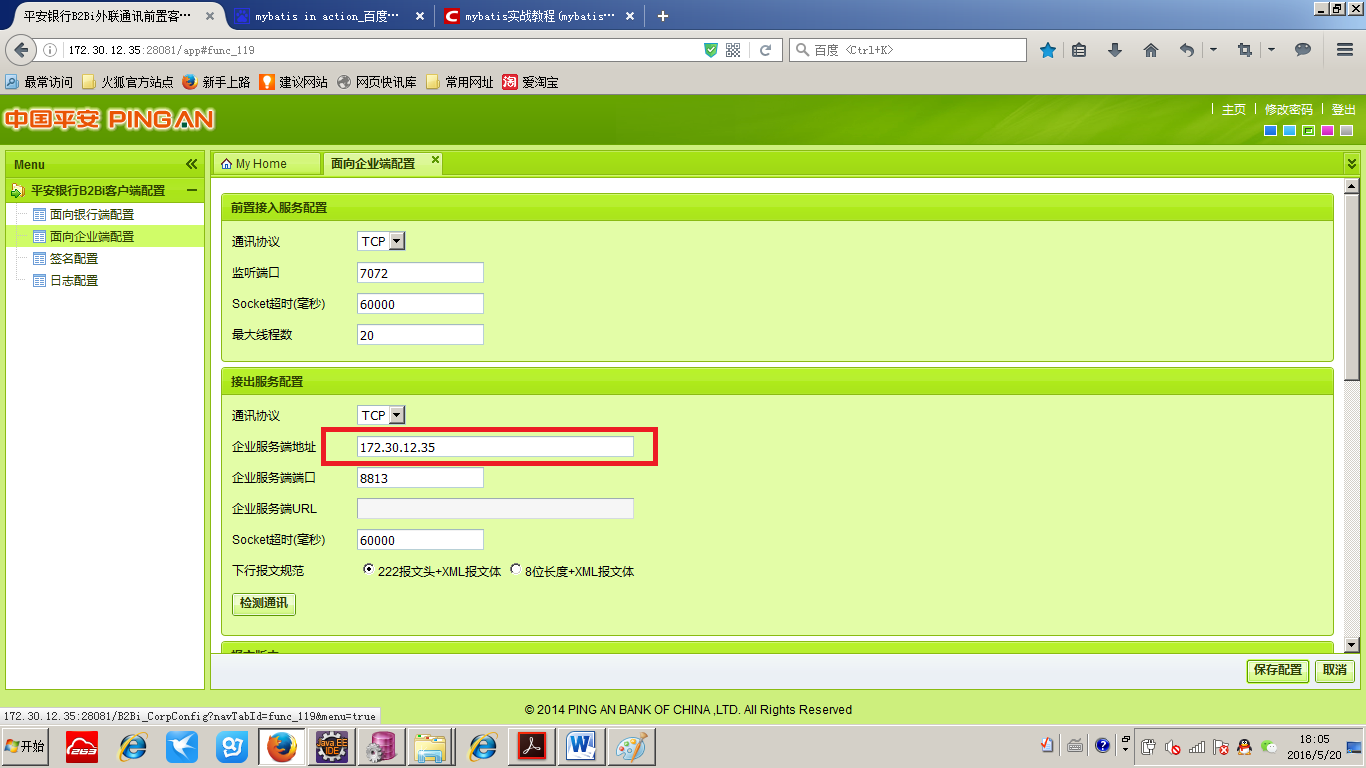
1. 输入用户名（admin）,密码（888888），登录系统。登录界面如下：



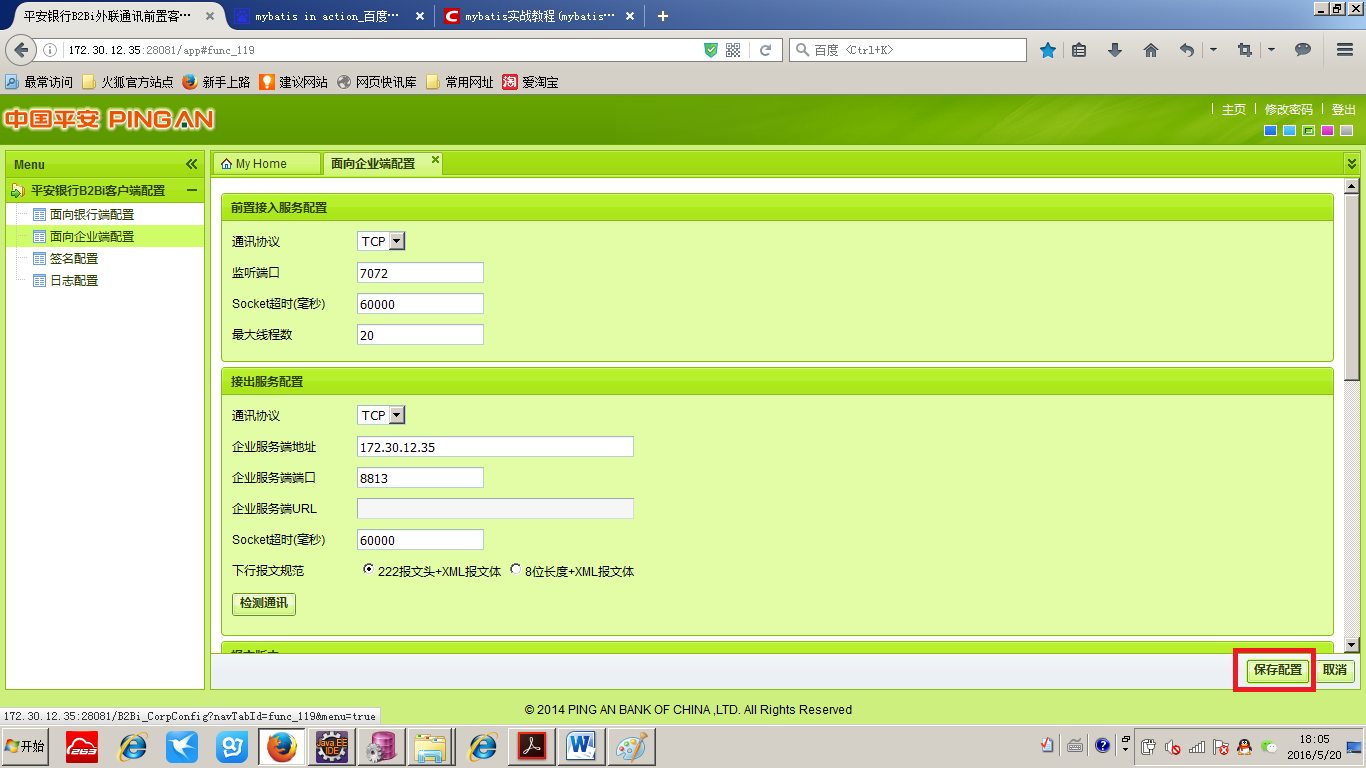
1. 登录后进入首页，进入“面向企业端配置”，如下图所示：



1. 修改“企业服务端地址”修改为新环境IP地址，如下图所示；



1. 保存配置（单击右下方”保存配置”按钮）并退出，如下图所示：



### 启动主服务

#### 启动in容器

in容器启动脚本为startIn.sh，停止脚本为stopIn.sh。

* 使用startIn.sh命令启动in容器，具体用法如下：

|  |
| --- |
| /ztkx/cbpay/CrossBorderPay/bin/startIn.sh |

* 如需停止in容器服务，使用stopIn.sh，具体用法如下：

|  |
| --- |
| /ztkx/cbpay/CrossBorderPay/bin/stopIn.sh |

#### 启动out容器

out容器启动脚本为startOut.sh，停止脚本为stopOut.sh。

* 使用startOut.sh命令启动out容器，具体用法如下：

|  |
| --- |
| /ztkx/cbpay/CrossBorderPay/bin/startOut.sh |

* 如需停止out容器服务，使用stopOut.sh，具体用法如下：

|  |
| --- |
| /ztkx/cbpay/CrossBorderPay/bin/stopOut.sh |

#### 启动discenter容器

discenter容器启动脚本为startDis.sh，停止脚本为stopDis.sh。

* 使用startDis.sh命令启动discenter容器，具体用法如下：

|  |
| --- |
| /ztkx/discenter/discenter/bin/startDis.sh |

* 如需停止discenter容器服务，使用stopDis.sh，具体用法如下：

|  |
| --- |
| /ztkx/discenter/discenter/bin/stopDis.sh |

### 启动console服务

Console服务的启动脚本为startCon.sh，停止脚本为stopCon.sh。

* 使用startCon.sh命令启动console服务，具体用法如下：

|  |
| --- |
| /ztkx/discenter/discenter/bin/startCon.sh |

* 如需停止console服务，使用stopCon.sh，具体用法如下：

|  |
| --- |
| /ztkx/discenter/discenter/bin/stopCon.sh |

### 启动timedtask定时任务

Timedtask定时任务启动脚本为startTimerTask.sh，停止脚本为stopTimerTask.sh。

* 使用startTimerTask.sh命令启动定时任务，具体用法如下：

|  |
| --- |
| /ztkx/cbpay/timedtask/bin/startTimerTask.sh |

* 如需停止定时任务，使用stopTimerTask.sh，具体用法如下：

|  |
| --- |
| /ztkx/cbpay/timedtask/bin/stopTimerTask.sh |

### 启动收银台服务

收银台服务启动脚本为startcashier.sh，停止脚本为stopcashier.sh。

* 使用startcashier.sh命令启动收银台任务，具体用法如下：

|  |
| --- |
| /ztkx/discenter/discenter/bin/startcashier.sh |

* 如需停止收银台，使用stopTimerTask.sh，具体用法如下：

|  |
| --- |
| /ztkx/discenter/discenter/bin/stopcashier.sh |

## 第五章注意事项

### 超时问题

当使用activemq传输消息时，由于测试环境没有时间同步，所以当消息在不同的机器之间交互时，可能因为机器之间的时间差，导致消息未经处理直接进入死信队列！

需要修改的配置文件如下：

1. discenter容器

baseConf.properties文件

* 首先使用discenter用户登录到主应用的服务器上。
* 使用vi命令打开配置文件，具体如下：

|  |
| --- |
| vi /ztkx/discenter/discenter/config/baseConf.properties |

打开文件后可看到如下配置：

|  |
| --- |
| #主配置文件  #初始化上下文池大小  contextPoolSize=50  #容器名称  CONTAINER\_NAME=IN  #集群节点  cluster=cluster1  #in容器消息监听listener  messagelistener=com.ztkx.cbpay.container.msglistener.TempletMsgListener  #mq重连时间间隔  mq.reconnection.interval=5000  #容器总线程池大小  threadPoolSize=50  #jms消息超时时间  JMS\_MESSAGE\_OVER\_TIME=60000  #是否需要将原始报文传入数据库的标识  TRANS\_LOG\_SWITCH=on  #是否需要更新交易日志  TRANS\_TIMESTAMP\_SWITCH=on  test35:/ztkx/cbpay/CrossBorderPay/config/in\_conf>vi /ztkx/cbpay/CrossBorderPay/config/in\_conf/baseConf.properties  #主配置文件  #初始化上下文池大小  contextPoolSize=50  #容器名称  CONTAINER\_NAME=IN  #集群节点  cluster=cluster1  #in容器消息监听listener  messagelistener=com.ztkx.cbpay.container.msglistener.TempletMsgListener  #mq重连时间间隔  mq.reconnection.interval=5000  #容器总线程池大小  threadPoolSize=50  #jms消息超时时间  JMS\_MESSAGE\_OVER\_TIME=60000  #是否需要将原始报文传入数据库的标识  TRANS\_LOG\_SWITCH=on  #是否需要更新交易日志  TRANS\_TIMESTAMP\_SWITCH=on |

注：红色标注部分即为待修改参数，该参数含义如下：

JMS\_MESSAGE\_OVER\_TIME：该参数表示通过activmq传输消息的过期时间；

* 修改配置
* 修改超时时间使其大于两台机器（即discenter容器和in容器所在的两台机器）的时间差即可；
* 注：如果该discenter下挂在了多个in容器，则超时时间必须大于discenter容器所在机器和任意一个in容器所在机器的时间差。

1. in容器

baseConf.properties文件

* 首先使用cbpay用户登录到主应用的服务器上。
* 使用vi命令打开配置文件，具体如下：

|  |
| --- |
| vi vi /ztkx/cbpay/CrossBorderPay/config/in\_conf/baseConf.properties |

打开文件后可看到如下配置：

|  |
| --- |
| #主配置文件  #初始化上下文池大小  contextPoolSize=50  #容器名称  CONTAINER\_NAME=DISCENTER  #in容器消息监听listener  messagelistener=com.ztkx.cbpay.container.msglistener.DiscenterMsgListener  #mq重连时间间隔  mq.reconnection.interval=5000  #容器总线程池大小  threadPoolSize=50  #jms消息超时时间  JMS\_MESSAGE\_OVER\_TIME=60000  #discenter和app的交易超时时间(3分钟)  message\_over\_Time=180000  #消息缓冲池  message\_buffer\_pool\_size=300 |

注：红色标注部分即为待修改参数，该参数含义如下：

JMS\_MESSAGE\_OVER\_TIME：该参数表示通过activmq传输消息的过期时间。

* 修改配置

修改超时时间使其大于两台机器（即in容器和它所连的discenter容器所在的两台机器）的时间差即可；

### Jdk问题

目前跨境支付所有程序都需运行在jdk 1.7.0 上，java\_home在用户的启动脚本设定，以后如果出现需要其他版本jdk支持的程序，需在启动脚本中指定具体路径（后期注意）。