# 所有服务节点：管理平台、API，说明为什么这样部署，能避免什么问题

## API服务

API服务面向终端SDK，接受终端SDK调用并返回数据，包括：机构、菜单、卡片、JS规则等。

考虑的点：

* 高并发：
  + 终端数量多，且未来还会持续增长
  + 终端有本地缓存、本地数据库，可降低对同一数据的访问频次
* 高可用：为避免单点故障，建议部署大于1台服务器

部署方案：

* 部署4台API服务器，通过Nginx做负载均衡，采用轮询算法，由4台API服务器共同承担终端SDK的访问。随着时间推移，如果发现4台不够，可以很容易地增加服务器分担负载，保证服务稳定
* 为避免Nginx成为瓶颈，设计部署2台Nginx服务器，通过DNS（或LVS）进行负载均衡

## 打点服务

打点服务面向终端SDK，接收终端SDK上报的打点数据，包括：卡片解析、菜单点击等。

考虑的点：

* 高并发：
  + 终端数量多，且未来还会持续增长
  + 终端先将打点数据记录到本地数据库，定时调用打点服务
* 高可用：为避免单点故障，建议部署大于1台服务器

部署方案：

* 部署2台打点服务器，通过Nginx做负载均衡
* 与API服务用同一套Nginx

## 管理平台

管理平台分为客户门户和运营门户。

客户门户面向来自领航平台的产品客户，根据实际需要自定义Logo、菜单、卡片等。

运营门户面向管理员，负责审核客户门户提交的数据。

考虑的点：

* 高并发：
  + 用户数量不多
  + 用户操作有“思考时间”，并发程度较低
* 高可用：为避免单点故障，建议部署大于1台服务器

部署方案：

* 部署2台客户门户服务器、2台运营门户服务器，通过Nginx做负载均衡
* 与API服务用同一套Nginx

# 测试方案

## 启停测试

### API服务

|  |  |  |  |
| --- | --- | --- | --- |
| **测试项目** | API服务 | | |
| **测试内容** | 启动API服务 | | |
| **测试条件** | API服务已停止，数据库连接字符串配置正确，数据库可正常连接 | | |
| **测试序号** | **测试步骤** | **预期结果** | **实际结果** |
|  | 1. 使用登录名isms登录API服务器 2. 进入API项目路径   cd /apps/ctcc-mcs-api   1. 执行Shell脚本：   ./run.sh start prod | 1. 输出：   “启动程序，运行 screen -d -m /usr/bin/java …(JVM参数)… -Dapp.path=/apps/ctcc-mcs-api -Dspring.profiles.active=prod -jar app.jar” | □ OK  □ FAIL |

|  |  |  |  |
| --- | --- | --- | --- |
| **测试项目** | API服务 | | |
| **测试内容** | 停止API服务 | | |
| **测试条件** | API服务已启动 | | |
| **测试序号** | **测试步骤** | **预期结果** | **实际结果** |
|  | 1. 使用登录名isms登录API服务器 2. 进入API项目路径   cd /apps/ctcc-mcs-api   1. 执行Shell脚本：   ./run.sh stop | 1. 输出：   “关闭程序” | □ OK  □ FAIL |

|  |  |  |  |
| --- | --- | --- | --- |
| **测试项目** | API服务 | | |
| **测试内容** | 重启API服务 | | |
| **测试条件** | API服务已启动 | | |
| **测试序号** | **测试步骤** | **预期结果** | **实际结果** |
|  | 1. 使用登录名isms登录API服务器 2. 进入API项目路径   cd /apps/ctcc-mcs-api   1. 执行Shell脚本：   ./run.sh restart prod | 1. 输出：   “启动程序，运行 screen -d -m /usr/bin/java …(JVM参数)… -Dapp.path=/apps/ctcc-mcs-api -Dspring.profiles.active=prod -jar app.jar” | □ OK  □ FAIL |

### 打点服务

|  |  |  |  |
| --- | --- | --- | --- |
| **测试项目** | 打点服务 | | |
| **测试内容** | 启动打点服务 | | |
| **测试条件** | 打点服务已停止，数据库连接字符串配置正确，数据库可正常连接 | | |
| **测试序号** | **测试步骤** | **预期结果** | **实际结果** |
|  | 1. 使用登录名isms登录打点服务器 2. 进入打点项目路径   cd /apps   1. 执行Shell脚本：   ./run.sh start prod | 1. 输出：   “启动程序，运行 screen -d -m /usr/bin/java …(JVM参数)… -Dapp.path=/apps -Dspring.profiles.active=prod -jar app.jar” | □ OK  □ FAIL |

|  |  |  |  |
| --- | --- | --- | --- |
| **测试项目** | 打点服务 | | |
| **测试内容** | 停止打点服务 | | |
| **测试条件** | 打点服务已启动 | | |
| **测试序号** | **测试步骤** | **预期结果** | **实际结果** |
|  | 1. 使用登录名isms登录打点服务器 2. 进入打点项目路径   cd /apps   1. 执行Shell脚本：   ./run.sh stop | 1. 输出：   “关闭程序” | □ OK  □ FAIL |

|  |  |  |  |
| --- | --- | --- | --- |
| **测试项目** | 打点服务 | | |
| **测试内容** | 重启打点服务 | | |
| **测试条件** | 打点服务已启动 | | |
| **测试序号** | **测试步骤** | **预期结果** | **实际结果** |
|  | 1. 使用登录名isms登录打点服务器 2. 进入打点项目路径   cd /apps   1. 执行Shell脚本：   ./run.sh restart prod | 1. 输出：   “启动程序，运行 screen -d -m /usr/bin/java …(JVM参数)… -Dapp.path=/apps -Dspring.profiles.active=prod -jar app.jar” | □ OK  □ FAIL |

### 运营门户服务

|  |  |  |  |
| --- | --- | --- | --- |
| **测试项目** | 运营门户服务 | | |
| **测试内容** | 启动运营门户服务 | | |
| **测试条件** | 运营门户服务已停止，数据库连接字符串配置正确，数据库可正常连接 | | |
| **测试序号** | **测试步骤** | **预期结果** | **实际结果** |
|  | 1. 使用登录名isms登录运营门户服务器 2. 进入运营门户项目路径   cd /apps/ctcc-mcs-operate   1. 执行Shell脚本：   ./run.sh start prod | 1. 输出：   “启动程序，运行 screen -d -m /usr/bin/java …(JVM参数)… -Dapp.path=/apps/ctcc-mcs-operate -Dspring.profiles.active=prod -jar app.jar” | □ OK  □ FAIL |

|  |  |  |  |
| --- | --- | --- | --- |
| **测试项目** | 运营门户服务 | | |
| **测试内容** | 停止运营门户服务 | | |
| **测试条件** | 运营门户服务已启动 | | |
| **测试序号** | **测试步骤** | **预期结果** | **实际结果** |
|  | 1. 使用登录名isms登录运营门户服务器 2. 进入运营门户项目路径   cd /apps/ctcc-mcs-operate   1. 执行Shell脚本：   ./run.sh stop | 1. 输出：   “关闭程序” | □ OK  □ FAIL |

|  |  |  |  |
| --- | --- | --- | --- |
| **测试项目** | 运营门户服务 | | |
| **测试内容** | 重启运营门户服务 | | |
| **测试条件** | 运营门户服务已启动 | | |
| **测试序号** | **测试步骤** | **预期结果** | **实际结果** |
|  | 1. 使用登录名isms登录运营门户服务器 2. 进入运营门户项目路径   cd /apps/ctcc-mcs-operate   1. 执行Shell脚本：   ./run.sh restart prod | 1. 输出：   “启动程序，运行 screen -d -m /usr/bin/java …(JVM参数)… -Dapp.path=/apps/ctcc-mcs-operate -Dspring.profiles.active=prod -jar app.jar” | □ OK  □ FAIL |

### 客户门户服务

|  |  |  |  |
| --- | --- | --- | --- |
| **测试项目** | 客户门户服务 | | |
| **测试内容** | 启动客户门户服务 | | |
| **测试条件** | 客户门户服务已停止，数据库连接字符串配置正确，数据库可正常连接 | | |
| **测试序号** | **测试步骤** | **预期结果** | **实际结果** |
|  | 1. 使用登录名isms登录客户门户服务器 2. 进入客户门户项目路径   cd /apps/ctcc-mcs-client   1. 执行Shell脚本：   ./run.sh start prod | 1. 输出：   “启动程序，运行 screen -d -m /usr/bin/java …(JVM参数)… -Dapp.path=/apps/ctcc-mcs-client -Dspring.profiles.active=prod -jar app.jar” | □ OK  □ FAIL |

|  |  |  |  |
| --- | --- | --- | --- |
| **测试项目** | 客户门户服务 | | |
| **测试内容** | 停止客户门户服务 | | |
| **测试条件** | 客户门户服务已启动 | | |
| **测试序号** | **测试步骤** | **预期结果** | **实际结果** |
|  | 1. 登录客户门户服务器 2. 进入客户门户项目路径   cd /apps/ctcc-mcs-client   1. 执行Shell脚本：   ./run.sh stop | 1. 输出：   “关闭程序” | □ OK  □ FAIL |

|  |  |  |  |
| --- | --- | --- | --- |
| **测试项目** | 客户门户服务 | | |
| **测试内容** | 重启客户门户服务 | | |
| **测试条件** | 客户门户服务已启动 | | |
| **测试序号** | **测试步骤** | **预期结果** | **实际结果** |
|  | 1. 使用登录名isms登录客户门户服务器 2. 进入客户门户项目路径   cd /apps/ctcc-mcs-client   1. 执行Shell脚本：   ./run.sh restart prod | 1. 输出：   “启动程序，运行 screen -d -m /usr/bin/java …(JVM参数)… -Dapp.path=/apps/ctcc-mcs-client -Dspring.profiles.active=prod -jar app.jar” | □ OK  □ FAIL |

## 负载均衡测试

|  |  |  |  |
| --- | --- | --- | --- |
| **测试项目** | API服务 | | |
| **测试内容** | API服务的负载均衡 | | |
| **测试条件** | 集群内全部API服务均已启动 | | |
| **测试序号** | **测试步骤** | **预期结果** | **实际结果** |
|  | 1. 使用登录名isms登录API服务器 2. 进入API项目路径   cd /apps/ctcc-mcs-api/   1. 执行命令，实时查看日志：   tail –f logs/ing/sys.log   1. 重复1-3步，对集群内每台服务器进行操作 2. 使用Postman工具，对API服务发送10次请求：   （http://host:port）/api/v1/systemTime | 1. 10次请求均响应成功 2. 集群内每台服务器都能记录到至少1条日志 | □ OK  □ FAIL |

|  |  |  |  |
| --- | --- | --- | --- |
| **测试项目** | API服务 | | |
| **测试内容** | API服务的高可用 | | |
| **测试条件** | 集群内只启动一个API服务，其余停止 | | |
| **测试序号** | **测试步骤** | **预期结果** | **实际结果** |
|  | 1. 使用Postman工具，对API服务发送10次请求：   （http://host:port）/api/v1/systemTime | 1. 10次请求均响应成功 | □ OK  □ FAIL |

# 异常处理部分，要明确到具体应用、接口程序名称、IP、端口，异常表现出的现象、如何诊断、诊断为异常的依据定义、恢复异常的操作步骤等

## API服务

|  |  |  |  |
| --- | --- | --- | --- |
| **异常现象** | **诊断** | **诊断步骤** | **解决办法** |
| SDK请求超时 | 检查API服务是否启动 | 1. 使用登录名isms登录API服务器 2. 输入命令   ps –ef|grep java   1. 查看是否有“cctc-mcs-api” 2. 重复1-3步，对集群内每台服务器进行操作 | 启动API服务 |
| 检查Nginx服务是否启动 | 1. 使用登录名root登录Nginx服务器 2. 输入命令   ps –ef|grep nginx   1. 查看是否有结果（除grep之外） 2. 重复1-3步，对集群内每台服务器进行操作 | 启动Nginx服务 |
| 检查MySQL服务是否启动 | 1. 使用登录名root登录MySQL服务器 2. 输入命令   ps –ef|grep mysql   1. 查看是否有结果（除grep之外） 2. 重复1-3步，对集群内每台服务器进行操作 | 启动MySQL服务 |
| 检查API服务器磁盘是否已满 | 1. 使用登录名isms登录API服务器 2. 输入命令   df -h   1. 查看根挂载点已用百分比是否达到100% 2. 重复1-3步，对集群内每台服务器进行操作 | 删除三天前的日志 |
| 检查Nginx服务器磁盘是否已满 | 1. 使用登录名root登录Nginx服务器 2. 输入命令   df -h   1. 查看根挂载点已用百分比是否达到100% 2. 重复1-3步，对集群内每台服务器进行操作 | 删除三天前的日志 |
| 检查MySQL服务器磁盘是否已满 | 1. 使用登录名root登录MySQL服务器 2. 输入命令   df -h   1. 查看根挂载点已用百分比是否达到100% 2. 重复1-3步，对集群内每台服务器进行操作 | 要么扩容，要么迁移数据 |
| 检查Nginx服务器的error日志是否存在错误 | 1. 使用登录名root登录Nginx服务器 2. 进入Nginx日志路径   cd /usr/local/nginx/logs   1. 输入命令   tail -f error.log   1. 查看是否存在错误  * no live upstreams while connecting to upstream * Connection TimeOut  1. 重复1-4步，对集群内每台服务器进行操作 | 1. 修改nginx配置文件，增加 proxy\_set\_header Connection "KeepAlive" 2. 修改run.sh， 启动命令加参数 -Dserver.tomcat.max-threads=20000 -Dserver.tomcat.max-Connections=20000 |
|  |  |  |  |

## 打点服务

|  |  |  |  |
| --- | --- | --- | --- |
| **异常现象** | **诊断** | **诊断步骤** | **解决办法** |
| SDK打点请求超时 | 检查打点服务是否启动 | 1. 使用登录名isms登录打点服务器 2. 输入命令   ps –ef|grep java   1. 查看是否有结果 2. 重复1-3步，对集群内每台服务器进行操作 | 启动打点服务 |
| 检查Nginx服务是否启动 | 1. 使用登录名root登录Nginx服务器 2. 输入命令   ps –ef|grep nginx   1. 查看是否有结果（除grep之外） 2. 重复1-3步，对集群内每台服务器进行操作 | 启动Nginx服务 |
| 检查MySQL服务是否启动 | 1. 使用登录名root登录MySQL服务器 2. 输入命令   ps –ef|grep mysql   1. 查看是否有结果（除grep之外） 2. 重复1-3步，对集群内每台服务器进行操作 | 启动MySQL服务 |
| 检查打点服务器磁盘是否已满 | 1. 使用登录名isms登录打点服务器 2. 输入命令   df -h   1. 查看根挂载点已用百分比是否达到100% 2. 重复1-3步，对集群内每台服务器进行操作 | 要么扩容，要么迁移数据 |
| 检查Nginx服务器磁盘是否已满 | 1. 使用登录名root登录Nginx服务器 2. 输入命令   df -h   1. 查看根挂载点已用百分比是否达到100% 2. 重复1-3步，对集群内每台服务器进行操作 | 删除三天前的日志 |
| 检查MySQL服务器磁盘是否已满 | 1. 使用登录名root登录MySQL服务器 2. 输入命令   df -h   1. 查看根挂载点已用百分比是否达到100% 2. 重复1-3步，对集群内每台服务器进行操作 | 要么扩容，要么迁移数据 |
|  |  |  |  |

## 运营门户

|  |  |  |  |
| --- | --- | --- | --- |
| **异常现象** | **诊断** | **诊断步骤** | **解决办法** |
| 浏览器访问运营门户时，报错 | 检查运营门户是否启动 | 1. 使用登录名isms登录运营门户服务器 2. 输入命令   ps –ef|grep java   1. 查看是否有“cctc-mcs-operate” 2. 重复1-3步，对集群内每台服务器进行操作 | 启动运营门户服务 |
| 检查Nginx服务是否启动 | 1. 使用登录名root登录Nginx服务器 2. 输入命令   ps –ef|grep nginx   1. 查看是否有结果（除grep之外） 2. 重复1-3步，对集群内每台服务器进行操作 | 启动Nginx服务 |
| 检查MySQL服务是否启动 | 1. 使用登录名root登录MySQL服务器 2. 输入命令   ps –ef|grep mysql   1. 查看是否有结果（除grep之外） 2. 重复1-3步，对集群内每台服务器进行操作 | 启动MySQL服务 |
| 检查运营门户服务器磁盘是否已满 | 1. 使用登录名isms登录运营门户服务器 2. 输入命令   df -h   1. 查看根挂载点已用百分比是否达到100% 2. 重复1-3步，对集群内每台服务器进行操作 | 删除三天前的日志 |
| 检查Nginx服务器磁盘是否已满 | 1. 使用登录名root登录Nginx服务器 2. 输入命令   df -h   1. 查看根挂载点已用百分比是否达到100% 2. 重复1-3步，对集群内每台服务器进行操作 | 删除三天前的日志 |
| 检查MySQL服务器磁盘是否已满 | 1. 使用登录名root登录MySQL服务器 2. 输入命令   df -h   1. 查看根挂载点已用百分比是否达到100% 2. 重复1-3步，对集群内每台服务器进行操作 | 要么扩容，要么迁移数据 |
|  |  |  |  |

## 客户门户

|  |  |  |  |
| --- | --- | --- | --- |
| **异常现象** | **诊断** | **诊断步骤** | **解决办法** |
| 浏览器访问客户门户时，报错 | 检查客户门户是否启动 | 1. 使用登录名isms登录客户门户服务器 2. 输入命令   ps –ef|grep java   1. 查看是否有“cctc-mcs-client” 2. 重复1-3步，对集群内每台服务器进行操作 | 启动客户门户服务 |
| 检查Nginx服务是否启动 | 1. 使用登录名root登录Nginx服务器 2. 输入命令   ps –ef|grep nginx   1. 查看是否有结果（除grep之外） 2. 重复1-3步，对集群内每台服务器进行操作 | 启动Nginx服务 |
| 检查MySQL服务是否启动 | 1. 使用登录名root登录MySQL服务器 2. 输入命令   ps –ef|grep mysql   1. 查看是否有结果（除grep之外） 2. 重复1-3步，对集群内每台服务器进行操作 | 启动MySQL服务 |
| 检查客户门户服务器磁盘是否已满 | 1. 使用登录名isms登录客户门户服务器 2. 输入命令   df -h   1. 查看根挂载点已用百分比是否达到100% 2. 重复1-3步，对集群内每台服务器进行操作 | 删除三天前的日志 |
| 检查Nginx服务器磁盘是否已满 | 1. 使用登录名root登录Nginx服务器 2. 输入命令   df -h   1. 查看根挂载点已用百分比是否达到100% 2. 重复1-3步，对集群内每台服务器进行操作 | 删除三天前的日志 |
| 检查MySQL服务器磁盘是否已满 | 1. 使用登录名root登录MySQL服务器 2. 输入命令   df -h   1. 查看根挂载点已用百分比是否达到100% 2. 重复1-3步，对集群内每台服务器进行操作 | 要么扩容，要么迁移数据 |
|  |  |  |  |