PUSH报表设计规格

|  |  |  |  |
| --- | --- | --- | --- |
| Prepared by  拟制 | 陆松超 | Date  日期 | 2016-09-03 |
| Reviewed by  评审人 |  | Date  日期 |  |
| Approved by  批准 |  | Date  日期 |  |
| Authorized by  签发 |  | Date  日期 |  |



华为技术有限公司

All rights reserved

版权所有 侵权必究

Revision record 修订记录

| Date  日期 | Revision Version  修订版本 | CR ID / Defect IDCR号 | Section Number修改章节 | Change Description  修改描述 | Author  作者 |
| --- | --- | --- | --- | --- | --- |
| 2016/09/03 | V0.1 |  | 5.1和7.6 | 1. 梳理当前Push报表  2. 设计CP报表录入工具 | L00169780 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Catalog 目 录

[PUSH报表设计规格 1](#_Toc460672893)

[1 Introduction 简介 6](#_Toc460672894)

[1.1 Scope 范围 6](#_Toc460672895)

[1.2 Purpose 目的 6](#_Toc460672896)

[1.3 Definition 定义和术语 6](#_Toc460672897)

[1.4 Identification and diagrams 标识和图例 6](#_Toc460672898)

[2 Overview 概述 7](#_Toc460672899)

[2.1 Description of version 版本描述 7](#_Toc460672900)

[2.2 Service Profile 业务简述 7](#_Toc460672901)

[2.3 Networking and independence of equipment 组网与设备独立性 7](#_Toc460672902)

[2.4 Degsign Specification Constraints 设计规格限制与约束 7](#_Toc460672903)

[3 System overall description 系统总体描述 8](#_Toc460672904)

[3.1 System Context 系统上下文 8](#_Toc460672905)

[3.2 System Function Lists 系统功能列表 8](#_Toc460672906)

[3.3 System Performance Lists 系统性能列表 8](#_Toc460672907)

[4 System architecture design 系统架构设计 9](#_Toc460672908)

[4.1 Function Architecture View 功能架构视图 9](#_Toc460672909)

[4.2 0 Level Logical architecture 0层逻辑架构 9](#_Toc460672910)

[4.3 1 Level Logical Architecture 1层逻辑架构 9](#_Toc460672911)

[4.4 Physical Deployment View 物理部署视图 9](#_Toc460672912)

[4.5 DB Physical Diagram View数据库物化视图 9](#_Toc460672913)

[5 Funcotion Requirements 功能需求 10](#_Toc460672914)

[5.1 联盟CP报表录入工具(Report V1) 10](#_Toc460672915)

[5.1.1 需求描述 10](#_Toc460672916)

[5.1.2 需求分解 10](#_Toc460672917)

[5.1.3 规格约束 10](#_Toc460672918)

[5.1.4 兼容性设计 10](#_Toc460672919)

[5.1.5 实现方案 10](#_Toc460672920)

[5.1.6 数据模型 12](#_Toc460672921)

[5.1.7 接口设计 14](#_Toc460672922)

[5.1.8 异常说明 14](#_Toc460672923)

[5.1.9 性能指标 14](#_Toc460672924)

[5.1.10 软件部署 15](#_Toc460672925)

[5.1.11 升级说明 15](#_Toc460672926)

[6 全球化设计 16](#_Toc460672927)

[6.1 功能A 16](#_Toc460672928)

[6.2 功能B 16](#_Toc460672929)

[7 Availablity可靠性设计 17](#_Toc460672930)

[7.1 流量控制 17](#_Toc460672931)

[7.2 有损服务 17](#_Toc460672932)

[7.3 业务多云 17](#_Toc460672933)

[7.4 业务容灾 17](#_Toc460672934)

[7.5 隔离设计 17](#_Toc460672935)

[7.6 兼容性设计 17](#_Toc460672936)

[7.6.1 Push 现网报表系统总体介绍 17](#_Toc460672937)

[7.6.2 联盟T\_NOTIFICATIONREQ表中的报表原先的处理逻辑 19](#_Toc460672938)

[7.6.3 联盟REPORT表中的报表原先的处理逻辑 21](#_Toc460672939)

[7.6.4 联盟REPORT表中的报表原先的处理逻辑(CRS 部分) 23](#_Toc460672940)

[7.6.5 联盟CP用户数报表 24](#_Toc460672941)

[7.6.6 运营分析数据平台Push KPI报表 24](#_Toc460672942)

[7.6.7 运营分析数据平台Push 用户分析报表 25](#_Toc460672943)

[7.6.8 运营分析数据平台Push 消息分析报表 25](#_Toc460672944)

[7.6.9 运营分析数据平台Push 连接状况分析报表 25](#_Toc460672945)

[7.6.10 现网PUSH CP相关报表数据人工干预方法 25](#_Toc460672946)

[7.6.11 Push话单和接口日志格式 26](#_Toc460672947)

[8 Serviceability可服务性设计 35](#_Toc460672948)

[8.1 业务安装 35](#_Toc460672949)

[8.2 (灰度)升级 35](#_Toc460672950)

[8.3 业务扩容 35](#_Toc460672951)

[8.4 故障定位 35](#_Toc460672952)

[8.5 业务告警 35](#_Toc460672953)

[9 Security安全性设计 36](#_Toc460672954)

[9.1 安全威胁建模 36](#_Toc460672955)

[10 UI设计 37](#_Toc460672956)

[11 Appendix附录 38](#_Toc460672957)

**Keywords 关键词：**Push Report ETL Taland

**Abstract 摘 要：**Push业务平台的报表整体描述

**缩略语清单：**

|  |  |  |
| --- | --- | --- |
| Abbreviations  缩略语 | Full spelling  英文全名 | Chinese explanation  中文解释 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Introduction 简介

## Scope 范围

N/A

## Purpose 目的

N/A

本文主要阅读对象为XXXX等

## Definition 定义和术语

N/A

## Identification and diagrams 标识和图例

N/A

# Overview 概述

## Description of version 版本描述

N/A

## Service Profile 业务简述

N/A

## Networking and independence of equipment 组网与设备独立性

N/A

## Degsign Specification Constraints 设计规格限制与约束

N/A

# System overall description 系统总体描述

## System Context 系统上下文

各接口描述如下：

|  |  |  |
| --- | --- | --- |
| **接口** | **协议** | **接口说明** |
|  |  |  |
|  |  |  |

## System Function Lists 系统功能列表

系统功能列表参考附件：

|  |  |  |  |
| --- | --- | --- | --- |
| 需求编号 | 需求类别 | 需求 | 详细描述 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## System Performance Lists 系统性能列表

# System architecture design 系统架构设计

## Function Architecture View 功能架构视图

N/A

## 0 Level Logical architecture 0层逻辑架构

N/A

## 1 Level Logical Architecture 1层逻辑架构

N/A

N/A

## Physical Deployment View 物理部署视图

N/A

## DB Physical Diagram View数据库物化视图

N/A

# Funcotion Requirements 功能需求

## 联盟CP报表录入工具(Report V1)

### 需求描述

Push业务平台当前的报表系统请参考“7.6.1 Push现网报表系统总体介绍”中的描述，本工具的目的是替换原先现网的“Report Notification”和“Report Message”这两个部件。

原先的CP报表录入工具使用第三方数据处理工具“Talend Open Studio”进行开发，在设计上存在单点故障问题，并且使用Taland Open Studio开发的代码也受限于工具本身的能力。现在用Java重新实现下这两个工具的功能。

### 需求分解

| **编号** | **部件** | **需求分解** |
| --- | --- | --- |
| 1 | Push Report | Report Notification功能是将报表数据录入t\_notificationreq表 (PushDB库) |
| 2 | Push Report | Report Message功能是将报表数据录入report表 (Report库) |

### 规格约束

本工具暂时不解决单点故障问题。

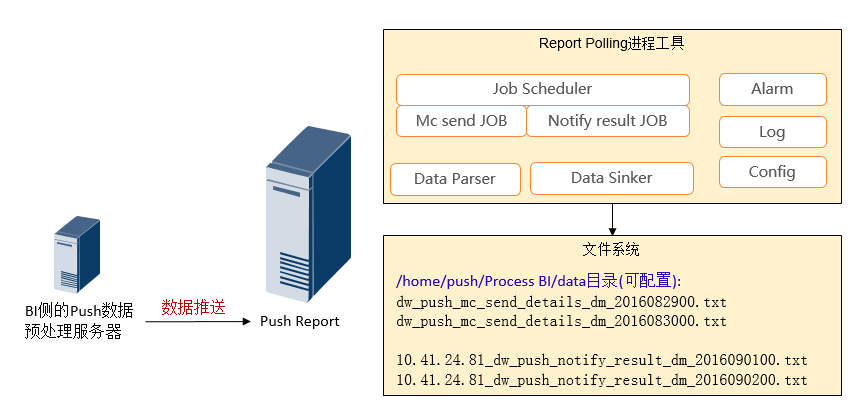
### 兼容性设计

新的工具需要完全兼容旧工具的所有功能:

* 数据解析;
* 数据处理的日志记录；
* 数据处理成功后将成功处理的文件移动到特定目录下；

### 实现方案

1. 工具总体架构



* 配置模块

配置文件使用简单的Java标准properties文件即可，各个任务分别有各自的配置文件格式和解析规则，例如对于notification的配置信息如下：

|  |
| --- |
| taskTimeSpan=10m #这里配置任务调度的时间间隔，10s(S)表示10秒;10m(M)表示10分钟；10h(h) 表  #示10小时，暂时只日志秒、分钟、小时三个配置项；  fileMask=\*dw\_push\_notify\_result\_dm\*.txt #数据目录下解析文件的格式  backupDir=/opt/huawei/hicloud/push/report/BIData/done #配置解析成功后的报表原文件目录  reportFileDir=/home/push/ProcessBI/data #被解析的数据目录  dbHost=10.61.15.185 #类型为MySQL的Data Sinker的数据IP配置  dbName=pushdb  dbPort=3306  tableName=t\_notificationreq  dbUserName=pushdb  dbPassword=OBnjEjG2d1mYVUDmPPlh #注意这里需要采用AES CBC加密 |

对于当前而且系统需要提供两个配置文件：

conf/jobs/notify\_result\_report.properties

conf/jobs/mc\_send\_report.properties

* 日志模块

采用LogBack具体参考Hiboard中的配置，提供error日志，handle日志

Run日志记录所有处理错误的信息；

|  |
| --- |
| 这里存放系统运行过程中的异常堆栈信息； |

Operation日志记录处理过的报表原始文件信息；

|  |
| --- |
| 时间|被处理的文件名称|错误码|处理结果详情  示例：  2016-09-03 13:15:13 | dw\_push\_mc\_send\_details\_dm\_2016083000.txt | 0 | OK |

Alarm日志

|  |
| --- |
| 这里存放数据库操作异常告警日志；  告警ID|事件类型|告警级别|告警清楚类别|告警名称|告警发生时间  示例：  00001|DB|Important|ACAD|DB Connection Failure|2016-12-13 13:13:42 |

* 告警模块

暂时只实现如果数据入库模块连接数据库时延很长或者数据库访问失败时，记录到Alarm日志中。

* 数据解析模块

暂时只实现类似CVS的解析器，用于将文件系统的文件解析到内存中。

* 数据录入模块

暂时只实现基于MySQL的数据录入模块，将数据录入到对应的表中；

* 任务调度模块

负责任务的调度

1. Notify result报表业务流程

Notify Result JOB按照配置的定时周期，定时report目录下的数据录入到t\_notificationreq表中。无论录入成功或者失败，都是记录Operation日志，并且将处理过的数据统一挪动到data/done目录下。运行过程有异常记录run日志，如果连接数据库有错误记录告警日志。

1. MC Send报表业务流程

### 数据模型

1. Notification报表原始数据格式说明

|  |  |  |
| --- | --- | --- |
| 序号 | 字段名称 | 说明 |
| 1 | 时间戳 | 报表的日期精确到年月日，例如：20160902 |
| 2 | AppId | 华为Push内部的AppId |
| 3 | RequestId | 对应t\_notificationreq中的requestid |
| 4 | 发送数 |  |
| 5 | 到达数 |  |
| 6 | 点击数 |  |

示例：

20160902|2000001434|1495825957|15|5|

20160902|2000001434|1495825979|28|16|

20160902|2000001434|1495828185|2|2|

20160902|0000003358|1495828607299678420000000|2|0|

20160902|0000003358|1495828758299691820000000|2|0|

20160902|2000001434|1495828806|3|0|

20160902|2000001434|1495828817|4|2|

20160902|2000001434|1495829164|6|4|0

20160902|0000003358|1495829243299730420000000|2|0|

20160902|2000001434|1495829472|2|0|

1. Message报表原始数据格式说明

|  |  |  |
| --- | --- | --- |
| 序号 | 字段名称 | 说明 |
| 1 | 时间戳 | 报表的日期精确到年月日，例如：2016-08-31 |
| 2 | AppID | 华为Push内部的AppId |
| 3 | 应用名称 | 例如：天天聊(hicloud.hotalk) |
| 4 | 所有用户数 |  |
| 5 | 消息总数 |  |
| 6 | 同步类型消息总数 |  |
| 7 | 同步消息成功数 |  |
| 8 | 同步消息不在线消息数 |  |
| 9 | 同步消息发送失败数 |  |
| 10 | 异步类型消息总数 |  |
| 11 | 异步消息成功数 |  |
| 12 | 异步覆盖消息数 |  |
| 13 | 异步消息超时数 |  |
| 14 | 异步缓存消息数 |  |
| 15 | 新增用户数 |  |
| 16 | 丢失用户数 |  |
| 17 | 净增用户数 |  |
| 18 | 营销消息数 | 原先的处理上这块有问题需要补齐营销消息数 |

示例：

2016-08-31|0000000013000001|同步(android.ds)|42004961|4294|0|0|0|0|4294|2506|1625|163|0|403520|368911|34609|0

2016-08-31|0000001001000001|应用市场|103266427|5329717|0|0|0|0|5329717|2235238|2700543|393936|0|1265582|1157763|107819|0

2016-08-31|0000001002000001|天天聊(hicloud.hotalk)|4|||||||||||0|0|0|0

2016-08-31|0000001003000001|天天浏览器(tiantianmini)|||||||||||||||0

2016-08-31|0000001004000001|同步(hisync)|||||||||||||||0

2016-08-31|0000001005000001|华为cloud+|119320|||||||||||791|1138|-347|0

2016-08-31|0000001006000001|天天聊(huawei.hotalk)|1912|||||||||||12|25|-13|0

2016-08-31|0000001007000001|天天电话(phoneplus)|29676|||||||||||102|171|-69|0

2016-08-31|0000001008000001|天天浏览器(tiantianlady)|||||||||||||||0

2016-08-31|0000001009000001|天天聊(hotalk)|11390|||||||||||79|71|8|0

2016-08-31|0000001010000001|天天聊(freetalk)|82|||||||||||1|4|-3|0

2016-08-31|0000001011000001|天天聊(message)|2989897|||||||||||39013|43852|-4839|0

2016-08-31|0000001012000001|华为PushDemo|1|||||||||||0|0|0|0

2016-08-31|0000001014000001|天天秀|1351|||||||||||9|17|-8|0

2016-08-31|0000001015000001|华为桌面|||||||||||||||0

2016-08-31|0000001016000001|天天新闻|1|||||||||||0|0|0|0

2016-08-31|0000001017000001|天天聊(talk)|||||||||||||||0

2016-08-31|0000001018000001|华为PayDemo|||||||||||||||0

2016-08-31|0000001019000001|华为推送APK|1474540|14957974|0|0|0|0|14957974|13787365|1075396|95213|0|812087|807304|4783|0

2016-08-31|0000001020000001|华为云服务(cloudservice)|1|||||||||||0|0|0|0

2016-08-31|0000001021000001|闪推|3|||||||||||0|0|0|0

2016-08-31|0000001022000001|contact+|||||||||||||||0

1. Report表说明

|  |
| --- |
| CREATE TABLE report (  time timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',  appId varchar(20) NOT NULL,//华为Push内部的appId  provider varchar(128) DEFAULT NULL,//华为Push内部的provider Id  userAll decimal(15,0) DEFAULT '0',//所有用户数  msgAll decimal(15,0) DEFAULT '0',//消息总数  synMsg decimal(15,0) DEFAULT '0',//同步类型消息总数  synSucc int(10) DEFAULT '0', //同步消息成功数  synOffline int(10) DEFAULT '0',// 同步消息不在线消息数  synFail int(10) DEFAULT '0',//同步消息发送失败数  asynMsg decimal(15,0) DEFAULT '0', //异步类型消息总数  asynSucc int(10) DEFAULT '0', //异步消息成功数  asynCover int(10) DEFAULT '0',//异步覆盖消息数  asynTout int(10) DEFAULT '0', //异步消息超时数  asynTBSend int(10) DEFAULT '0',//异步缓存消息数  userAdd decimal(15,0) DEFAULT '0',//新增用户数  userLose decimal(15,0) DEFAULT '0',//丢失用户数  userNetAdd decimal(15,0) DEFAULT '0',//净增用户数  hiadMsg int(11) NOT NULL DEFAULT '0',//营销消息数  PRIMARY KEY (time,appId)  ) ENGINE=InnoDB DEFAULT CHARSET=utf8; |

1. T\_notificationreq表说明

|  |
| --- |
| CREATE TABLE `t\_notificationreq` (  `requestId` varchar(32) NOT NULL,  `devAppId` varchar(32) NOT NULL,  `targetUserType` int(4) NOT NULL DEFAULT '0',  `userType` tinyint(4) DEFAULT '-1',  `userId` varchar(32) NOT NULL,  `createTime` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',  `sendTime` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',  `expireTime` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',  `notificationTitle` varchar(255) CHARACTER SET utf8mb4 DEFAULT NULL,  `allowPeriods` varchar(512) DEFAULT NULL,  `fromType` int(1) DEFAULT '1',  `pushType` int(1) DEFAULT '2',  `msgType` int(1) DEFAULT '0',  `doings` int(1) DEFAULT '1',  `currentState` int(1) DEFAULT '0',  `location` varchar(1024) DEFAULT '',  `tokens` text,  `tags` varchar(1024) DEFAULT '',  `excludeTags` varchar(1024) DEFAULT '',  `message` varchar(10240) CHARACTER SET utf8mb4 DEFAULT NULL,  `deviceTokenNum` int(11) DEFAULT '0',  `msgCount` int(11) DEFAULT '0',  `succCount` int(11) DEFAULT '0',  `clickCount` int(11) DEFAULT '0',  `flag` int(10) DEFAULT '0',  `threadName` varchar(32) DEFAULT NULL,  PRIMARY KEY (`requestId`,`devAppId`),  KEY `index\_createTime` (`createTime`) USING BTREE,  KEY `index\_sendTime` (`sendTime`) USING BTREE,  KEY `index\_notificationTitle` (`notificationTitle`(191)) USING BTREE,  KEY `index\_pushType` (`pushType`) USING BTREE,  KEY `index\_devappid` (`devAppId`),  KEY `index\_userId` (`userId`) USING BTREE,  KEY `index\_currentState` (`currentState`)  ) ENGINE=InnoDB DEFAULT CHARSET=utf8; |

### 接口设计

BI推送到Push Report服务器上的文件系统目录下，Push Report将数据录入到MySQL数据库中。

### 异常说明

N/A

### 性能指标

1. 现网实际数据

Notification的录入文件大小平均在4M左右，现网最大为60M左右；

Message的录入文件大小平均在1M左右，现网最大不超过1M；

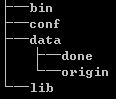
以上两份数据一般BI会每天推送1份到Push进行处理。

现网t\_notificationreq表的数据量在170万左右，report表的数据量在200万左右。

1. 性能指标

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 业务名称 | 环境配置 | 预置条件 | 性能指标 | 备注 |
| 通知消息统计 | 4C 32G普通虚拟机;  JDK 1.7 | t\_notificationreq表预置250万数据;  当个文件在4M； | 数据单个文件入库性能不超过2s； |  |
| 业务消息统计 | 4C 32G普通虚拟机;  JDK 1.7 | Report表预先预置250万数据;  当个文件在1M； | 数据单个文件入库性能不超过2s； |  |

### 软件部署



其中bin目录下存放的是进程启动和停止的shell脚本；

conf目录下存放配置文件;

lib目录下存放java的Jar包；

data目录下是默认的处理文件目录；

### 升级说明



现网升级时需要停止report机器上的如上两个Crontab任务，然后启动新的报表录入工具。

# 全球化设计

## 功能A

N/A

## 功能B

N/A

# Availablity可靠性设计

## 流量控制

## 有损服务

## 业务多云

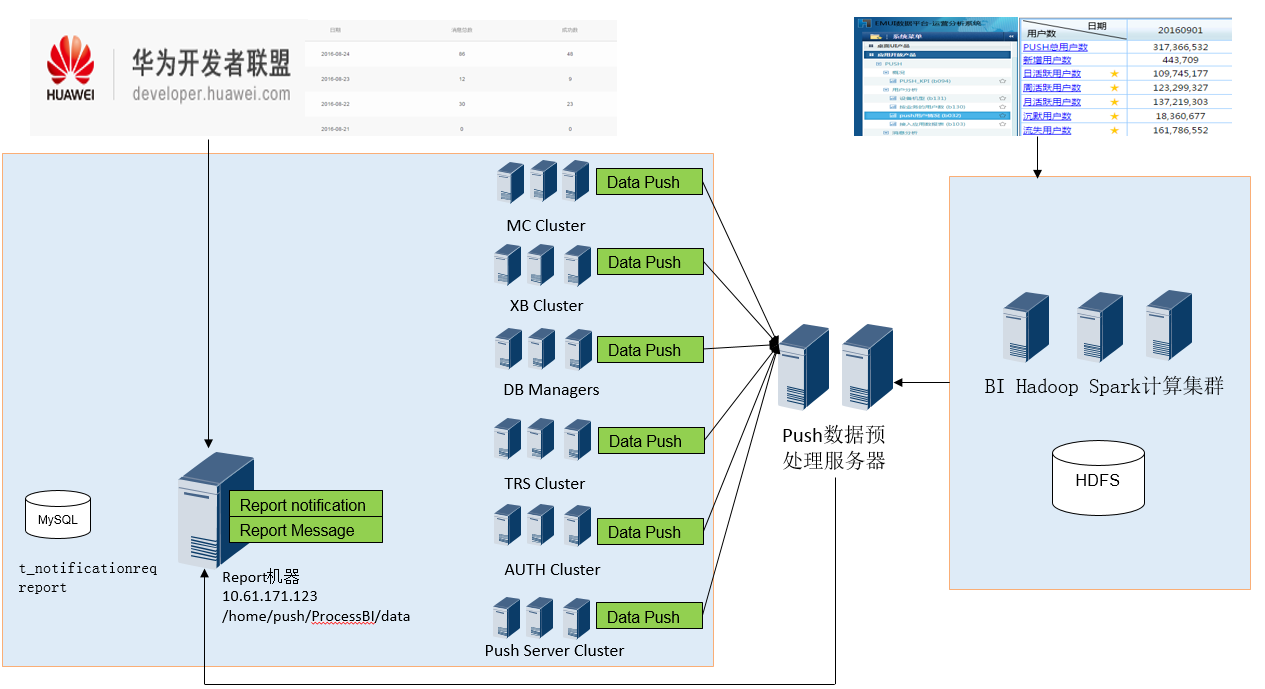
## 业务容灾

## 隔离设计

## 兼容性设计

### Push 现网报表系统总体介绍

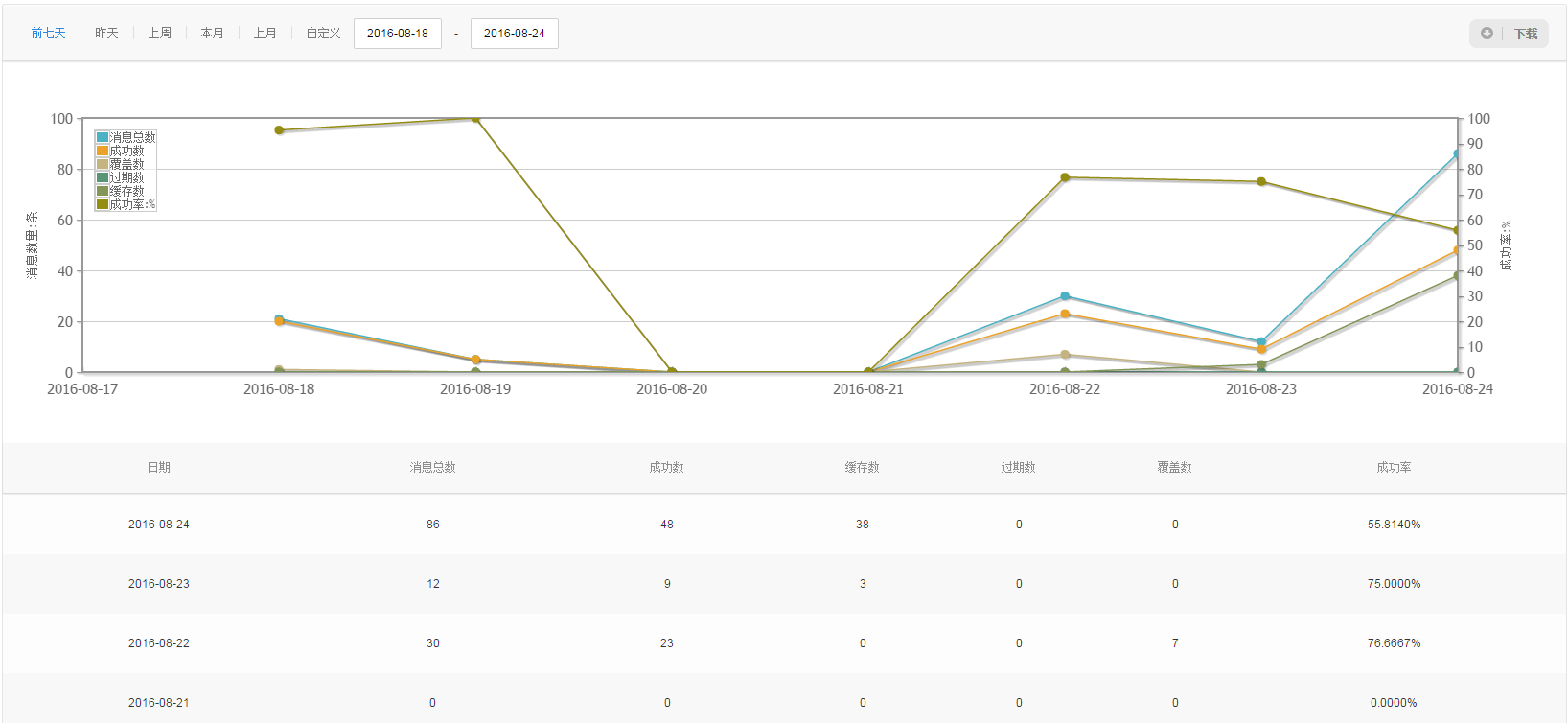
原先PUSH报表处理的整体视图：



*图 I: 华为Push报表整体架构示意图*

原先的报备分为两大块：

* 开发者联盟上为最终CP展示的报表



*图 I: 华为开发者联盟Push报表示例*

* EMUI数据平台上的运营分析系统报表

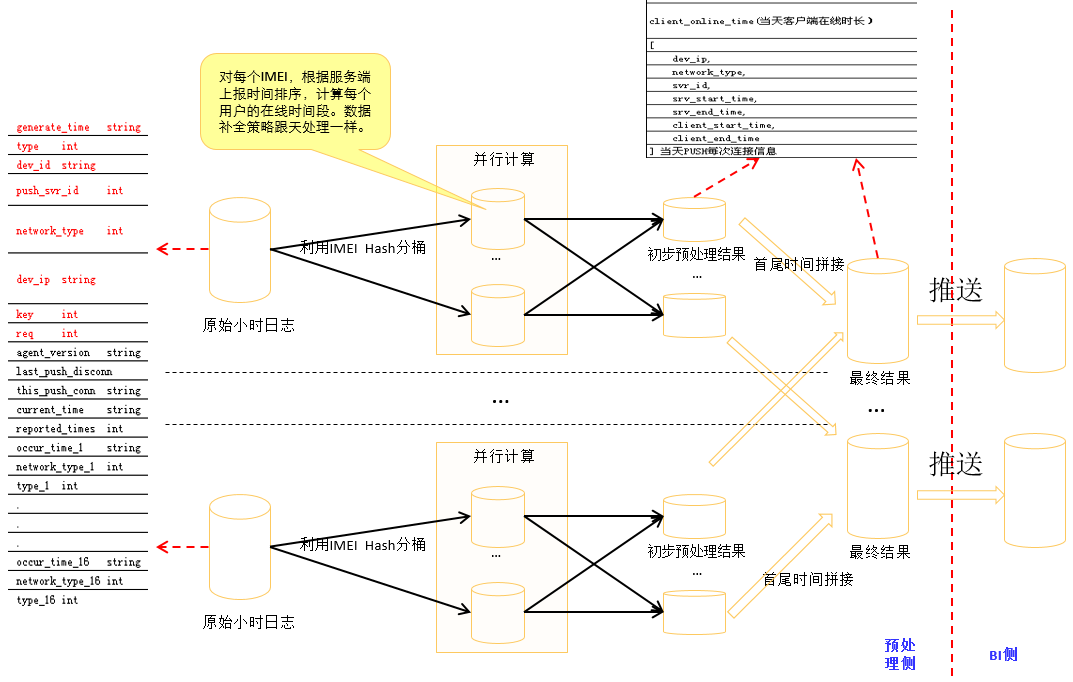


*图 I: EMUI数据平台示例*

在Push的MC/XB/DB Manager/TRS/AUTH/Push Server/CRS侧都会部署Data Push工具并使用Linux操作系统的Crontab任务调度模块定时将产生的日志推送到Push数据预处理服务器(部署在BI侧)，经过预处理后的数据会发送到BI的并行计算hadoop集群进行最终的报表计算。

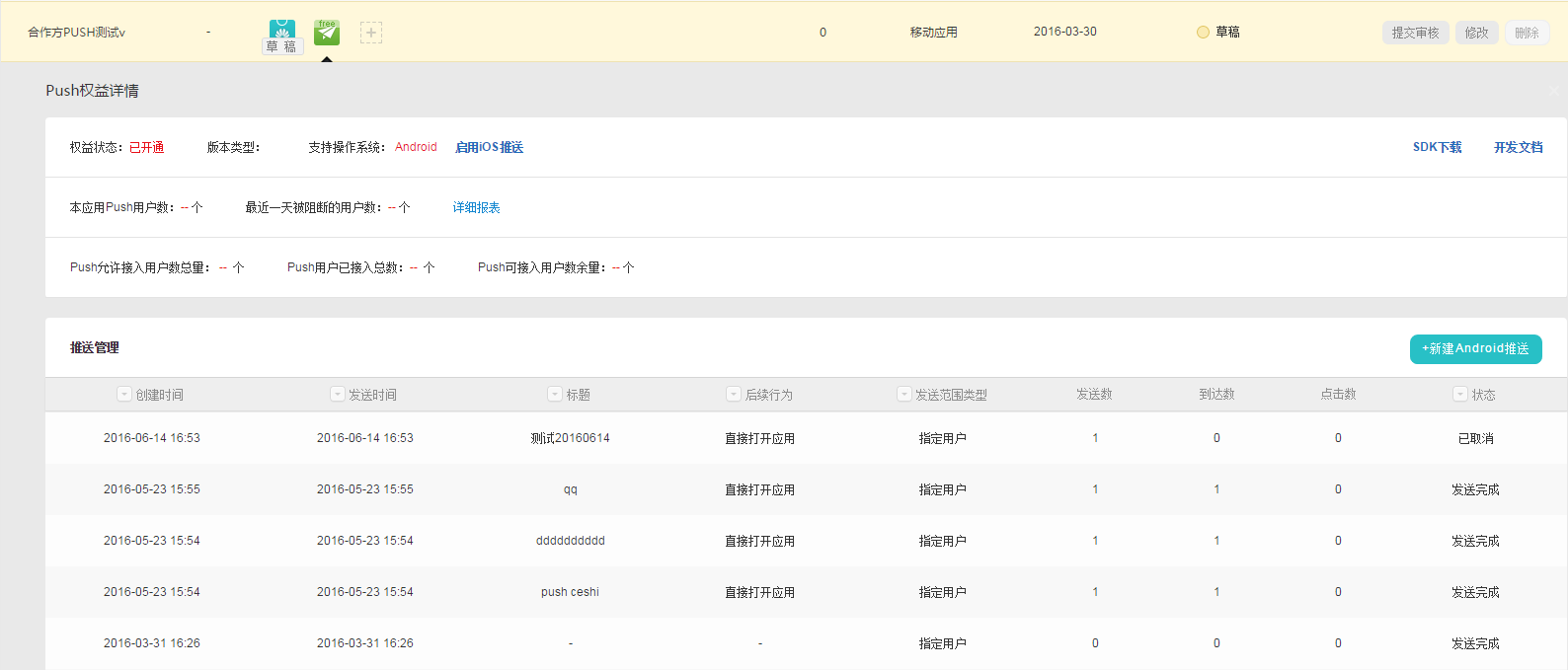
计算结果一部分是直接在报表系统(EMUI数据平台)进行展示处理，另外一部分给CP的数据会推送到Push Report的机器上，在这个Report机器上有两个Crontab的定时任务：report notification和report message分别将数据录入t\_notificationreq表和report表供开发者联盟进行调用。

Push数据预处理的架构图如下：



*图 I: PUSH报表数据预处理服务器*

### 联盟T\_NOTIFICATIONREQ表中的报表原先的处理逻辑



|  |  |  |
| --- | --- | --- |
| 报备项 | 业务含义 | 备注 |
| 发送数 |  |  |
| 到达数 |  |  |
| 点击数 |  |  |

1. 数据的产生

数据分为三个部分，一个是XB产生的数据，另外一个是MC产生的数据，最后是DBManager的数据.

* 1. XB产生的数据

|  |
| --- |
| <appender name=*"notificationLog"*  class=*"ch.qos.logback.core.rolling.RollingFileAppender"*>  <File>${catalina.home}/logs/notification/push\_portal\_notification.log  </File>  <encoder>  <pattern>%m%n</pattern>  </encoder>  <rollingPolicy class=*"ch.qos.logback.core.rolling.TimeBasedRollingPolicy"*>  <fileNamePattern>${catalina.home}/logs/notification/push\_portal\_notification.log.%d{yyyy-MM-dd}  </fileNamePattern>  </rollingPolicy>  </appender>  // 注意：以上这个数据的产生是在ForDeveloperService.notificationSend,  // ForPortalService.portalNotificationSend  // ForPortalService.portalNormalPush  <appender name=*"notificationLogError"*  class=*"ch.qos.logback.core.rolling.RollingFileAppender"*>  <File>${catalina.home}/logs/notification/push\_portal\_notification\_error.log  </File>  <encoder>  <pattern>%m%n</pattern>  </encoder>  <rollingPolicy class=*"ch.qos.logback.core.rolling.TimeBasedRollingPolicy"*>  <fileNamePattern>${catalina.home}/logs/notification/push\_portal\_notification\_error.log.%d{yyyy-MM-dd}  </fileNamePattern>  </rollingPolicy>  </appender>  // 注意：暂时代码中没有了，但是配置中还有  <appender name=*"notificationDevLog"*  class=*"ch.qos.logback.core.rolling.RollingFileAppender"*>  <File>${catalina.home}/logs/notification/push\_developer\_notification.log  </File>  <encoder>  <pattern>%m%n</pattern>  </encoder>  <rollingPolicy class=*"ch.qos.logback.core.rolling.TimeBasedRollingPolicy"*>  <FileNamePattern>${catalina.home}/logs/notification/push\_developer\_notification.log.%d{yyyy-MM-dd}  </FileNamePattern>  </rollingPolicy>  </appender>  //注意：这块现网也没有使用，代码中还是有的，有些废弃的代码 |

现网上PUSH侧生成如下几份日志文件：

* push\_portal\_notification.log

格式请参见“7.6.7 报备话单格式”章节的“XB notification log”

* push\_portal\_notification\_error.log

这块现网上还是有数据，但是在当前版本的代码里面是没有了，需要再调查下。

日志样例：

1482676145|2000000976|2016-08-26 14:15:00|2016-08-26 14:15:00|2016-08-28 14:15:00

* push\_developer\_notification.log

这块现网上是数据为空的，代码中有废弃的代码。

* 1. MC产生的数据

格式请参见“7.6.7 报备话单格式”章节的“MC bin log”

* 1. DB Manager产生的数据

格式请参见“7.6.7 报备话单格式”章节的“MC bin log”

1. 数据传送到BI

* 现网XB机器的推送工具部署

10 \* \* \* \* /opt/huawei/hicloud/push/DataPushHourBj/DataPush4cron.sh 1>>

/opt/huawei/hicloud/push/DataPushHourBj/log/localhost.log 2>&1

|  |  |  |
| --- | --- | --- |
| 命令 | 执行任务的内容 | 备注 |
| DataPush4cron.sh | 将数据推送的BI的机器 | /opt/huawei/hicloud/push/DataPushHourBj/DataPush4cron.sh |

* 现网MC机器的推送工具部署

15 \* \* \* \* /opt/huawei/hicloud/push/DataPushHourBj/DataPush4cron.sh 1>> /opt/huawei/hicloud/push/DataPushHourBj/log/localhost.log 2>&1

|  |  |  |
| --- | --- | --- |
| 命令 | 执行任务的内容 | 备注 |
| DataPush4cron.sh | 将数据推送的BI的机器 | /opt/huawei/hicloud/push/DataPushHourBj/DataPush4cron.sh |

* 现网DB Manager机器的推送工具部署

15 \* \* \* \* /opt/huawei/hicloud/push/HIBI-Push/DataPushHourBj/DataPush4cron.sh 1>>

/opt/huawei/hicloud/push/HIBI-Push/DataPushHourBj/log/localhost.log 2>&1

|  |  |  |
| --- | --- | --- |
| 命令 | 执行任务的内容 | 备注 |
| DataPush4cron.sh | 将数据推送的BI的机器 | /opt/huawei/hicloud/push/HIBI-Push/DataPushHourBj/DataPush4cron.sh |

1. BI计算结果推送到PUSH

BI计算结果首先会将数据推送到Push的中转机器，接着定时Crontab任务会将这些数据再推送到Report机器上的配置目录下，在Report机器上的定时入库任务会定时将这些数据解析出来并入库。

1. PUSH统计数据入库

这块代码目前在版本配置库上已经没有了，需要重新开发和维护。

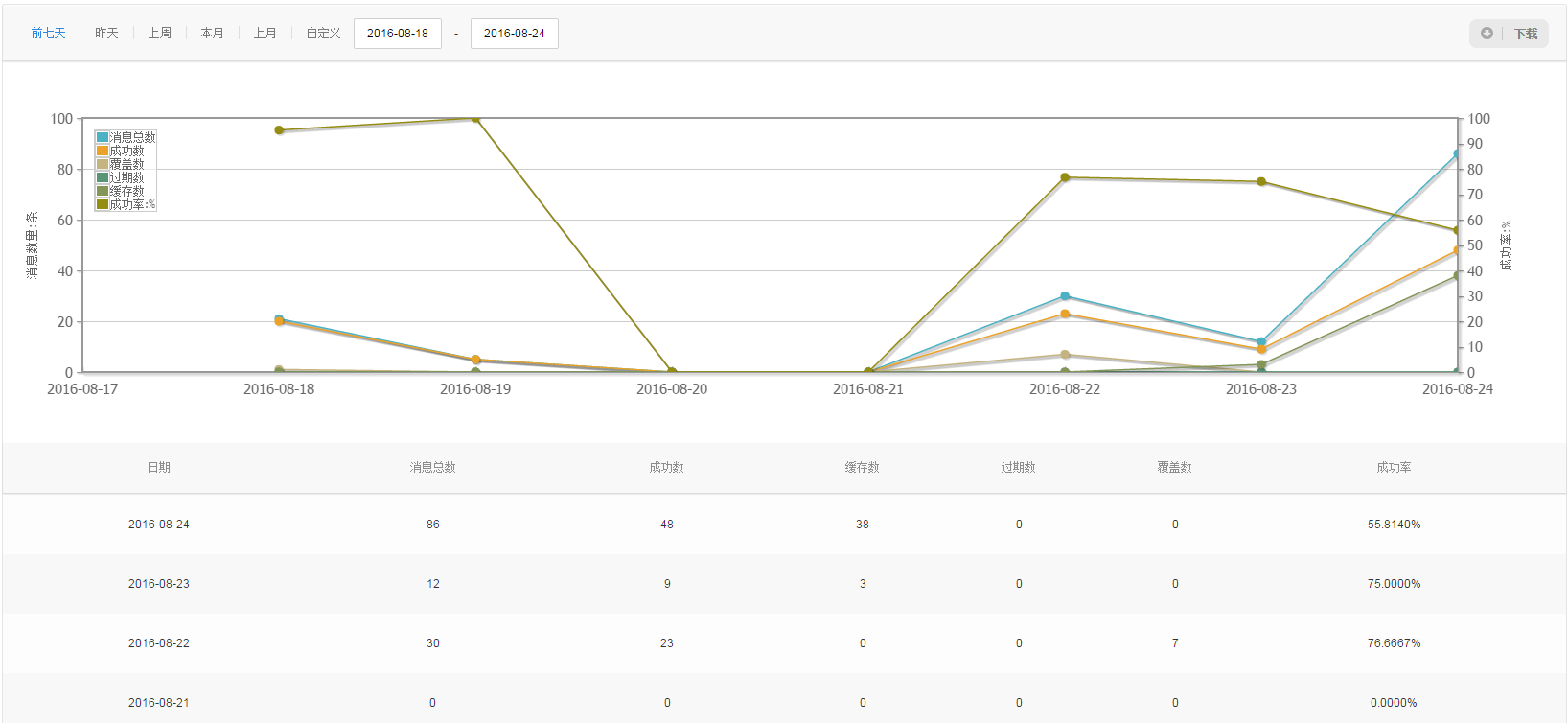
注意：现网的配置路径和机器为10.61.171.123

/opt/huawei/hicloud/push/report/BIData/report\_notification/report\_notification/push/report\_notification\_0\_1/contexts

1. CP查询统计信息

通过XB在网关开放的接口查询t\_notificationreq表中的内容；

### 联盟REPORT表中的报表原先的处理逻辑



|  |  |  |
| --- | --- | --- |
| 报备项 | 业务含义 | 备注 |
| 消息总数 | 发送的总消息数 |  |
| 成功数 | 同步消息成功数 + 异步消息成功数 |  |
| 缓存数 | 异步缓存消息数 |  |
| 过期数 | 异步过期消息数 |  |
| 覆盖数 | 异步覆盖消息数 |  |

1. 数据的产生

数据分为两个部分，一个是MC产生的数据，另外一个是DBManager的数据

格式请参见“7.6.7 报备话单格式”章节的“XB notification log”

1. 数据传送到BI

现网数据的传输仍然是通过DataPush工具完成，在各个MC机器和DB Manager机器上都会有Crontab的调度任务在执行。

* 现网MC机器的推送工具部署

15 \* \* \* \* /opt/huawei/hicloud/push/DataPushHourBj/DataPush4cron.sh 1>> /opt/huawei/hicloud/push/DataPushHourBj/log/localhost.log 2>&1

|  |  |  |
| --- | --- | --- |
| 命令 | 执行任务的内容 | 备注 |
| DataPush4cron.sh | 将数据推送的BI的机器 | /opt/huawei/hicloud/push/DataPushHourBj/DataPush4cron.sh |

* 现网DB Manager机器的推送工具部署

15 \* \* \* \* /opt/huawei/hicloud/push/HIBI-Push/DataPushHourBj/DataPush4cron.sh 1>>

/opt/huawei/hicloud/push/HIBI-Push/DataPushHourBj/log/localhost.log 2>&1

|  |  |  |
| --- | --- | --- |
| 命令 | 执行任务的内容 | 备注 |
| DataPush4cron.sh | 将数据推送的BI的机器 | /opt/huawei/hicloud/push/HIBI-Push/DataPushHourBj/DataPush4cron.sh |

1. BI计算结果推送到PUSH

同“7.6.1”章节的机制。

1. PUSH统计数据入库

这块代码目前在版本配置库上已经没有了，需要重新开发和维护。

注意：现网的配置路径和机器为10.61.171.123

/opt/huawei/hicloud/push/report/BIData/report\_user\_message/report\_user\_message/push/report\_user\_message\_2\_0/contexts

1. CP查询统计信息

联盟通过PUSH在网关上注册的接口查询report表

|  |
| --- |
| CREATE TABLE report (  time timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',  appId varchar(20) NOT NULL,//华为Push内部的appId  provider varchar(128) DEFAULT NULL,//华为Push内部的provider Id  userAll decimal(15,0) DEFAULT '0',//所有用户数  msgAll decimal(15,0) DEFAULT '0',//消息总数  synMsg decimal(15,0) DEFAULT '0',//同步类型消息总数  synSucc int(10) DEFAULT '0', //同步消息成功数  synOffline int(10) DEFAULT '0',//同步消息发送时设备不在线的情况的消息数量  synFail int(10) DEFAULT '0',//同步消息发送失败数  asynMsg decimal(15,0) DEFAULT '0', //异步类型消息总数  asynSucc int(10) DEFAULT '0', //异步消息成功数  asynCover int(10) DEFAULT '0',//异步覆盖消息数  asynTout int(10) DEFAULT '0', //异步消息超时数  asynTBSend int(10) DEFAULT '0',//异步缓存消息数  userAdd decimal(15,0) DEFAULT '0',//新增用户数  userLose decimal(15,0) DEFAULT '0',//丢失用户数  userNetAdd decimal(15,0) DEFAULT '0',//净增用户数  hiadMsg int(11) NOT NULL DEFAULT '0',//营销消息数  PRIMARY KEY (time,appId)  ) ENGINE=InnoDB DEFAULT CHARSET=utf8; |

### 联盟REPORT表中的报表原先的处理逻辑(CRS 部分)

这块现网数据量不大，后续需要逐步下线。

1. 数据的产生

CRS部件的原始数据是通过配置logback.xml来实现的

|  |
| --- |
| <appender name=*"statPushLogger"*  class=*"ch.qos.logback.core.rolling.RollingFileAppender"*>  <File>${catalina.home}/logs/stat/statistic\_push.log</File>  <encoder>  <pattern>%d{yyyy-MM-dd HH:mm:ss SSS}%m%n</pattern>  </encoder>  <rollingPolicy class=*"ch.qos.logback.core.rolling.TimeBasedRollingPolicy"*>  <fileNamePattern>${catalina.home}/logs/stat/statistic\_push.log.%d{yyyy-MM-dd}  </fileNamePattern>  </rollingPolicy>  </appender> |

格式请参见“7.6.7 报备话单格式”章节的“CRS bin log”

1. 数据传送到BI

现网数据的传输仍然是通过DataPush工具完成，在各个CRS机器上都会有Crontab的调度任务在执行。

* 现网CRS机器的推送工具部署

40 1 \* \* \* /opt/huawei/logs/HIBI-Push/DataPush/DataPush4cron.sh 1>> /opt/huawei/logs/HIBI-Push/DataPush/log/localhost.log 2>&1

|  |  |  |
| --- | --- | --- |
| 命令 | 执行任务的内容 | 备注 |
| DataPush4cron.sh | 将数据推送的BI的机器 | /opt/huawei/logs/HIBI-Push/DataPush/DataPush4cron.sh |

1. BI计算结果推送到PUSH

同“7.6.1”章节的机制。

1. PUSH统计数据入库

这块代码目前在版本配置库上已经没有了，需要重新开发和维护。

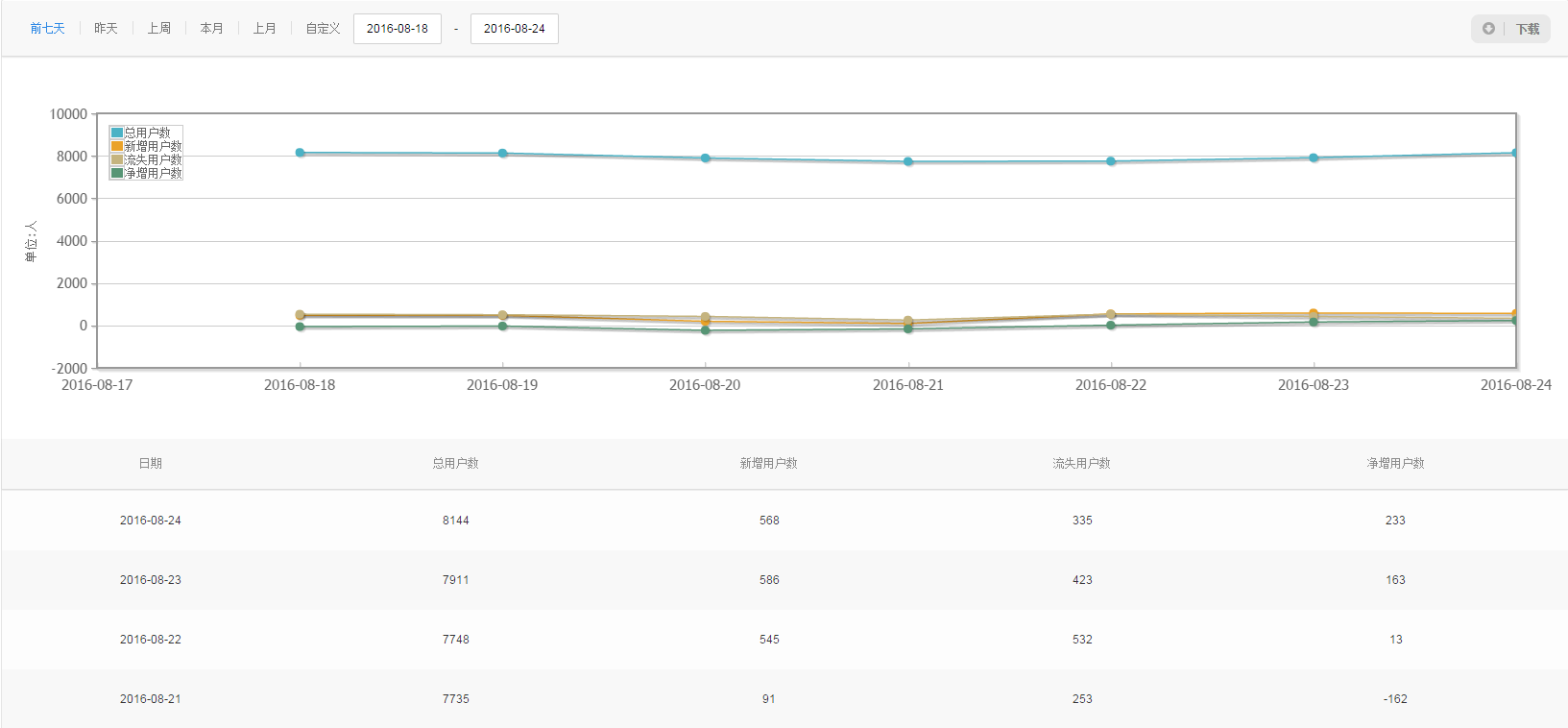
注意：现网的配置路径和机器为10.61.171.123

/opt/huawei/hicloud/push/report/BIData/report\_user\_message/report\_user\_message/push/report\_user\_message\_2\_0/contexts

1. CP查询统计信息

同7.6.2章节的处理。

### 联盟CP用户数报表



用户报表的作用如下，假设用户张三在华为开发者联盟上注册了一个账户，并且该账户下注册了两个应用app1和app2。王五是app1的用户，第一次使用时王五手机上的push agent模块会首先访问Push的TRS模块，TRS模块会记录请求和应答的日志。App1底层的push agent模块收到TRS的应答消息后会连接到指定的push server，并发送登陆消息到该push server并转发到auth模块，此时auth模块会记录登陆信息，auth模块接着想app1发送登陆成功应答。App1最后发送获取push token请求。

|  |  |  |
| --- | --- | --- |
| 报备项 | 业务含义 | 备注 |
| 总用户数 |  |  |
| 新增用户数 |  |  |
| 流失用户数 |  |  |
| 净增用户数 |  |  |

1. 数据的产生
2. 数据传送到BI
3. BI计算结果推送到PUSH
4. PUSH统计数据入库
5. CP查询统计信息

### 运营分析数据平台Push KPI报表

1. 数据的产生
2. 数据传送到BI
3. BI计算结果推送到PUSH
4. PUSH统计数据入库
5. CP查询统计信息

### 运营分析数据平台Push 用户分析报表

1. 数据的产生
2. 数据传送到BI
3. BI计算结果推送到PUSH
4. PUSH统计数据入库
5. CP查询统计信息

### 运营分析数据平台Push 消息分析报表

1. 数据的产生
2. 数据传送到BI
3. BI计算结果推送到PUSH
4. PUSH统计数据入库
5. CP查询统计信息

### 运营分析数据平台Push 连接状况分析报表

1. 数据的产生
2. 数据传送到BI
3. BI计算结果推送到PUSH
4. PUSH统计数据入库
5. CP查询统计信息

### 现网PUSH CP相关报表数据人工干预方法

|  |
| --- |
| STEP 1:  登陆到现网的report机器IP为10.61.171.123  STEP 2:  到目录  /opt/huawei/hicloud/push/report/BIData/report\_notification/report\_notification/push/report\_notification\_0\_1/contexts  下查看配置文件  其中：backupDir=/opt/huawei/hicloud/push/report/BIData/done  已经计算好入库的通知数据  /opt/huawei/hicloud/push/report/ProcessBI/datadone 已经计算好的MSG报表数据  reportFileDir=/home/push/ProcessBI/data   PUSH侧将这个目录下的数据放到数据库表里面  STEP 3:  将/opt/huawei/hicloud/push/report/BIData/done目录下需要重新入库的文件放到/home/push/ProcessBI/data目录下重新入库  注意：入库实际是一个crontab的任务 |

### Push话单和接口日志格式

1. XB notification log

|  |  |  |
| --- | --- | --- |
| 序号 | 字段名称 | 说明 |
| 1 | 流程唯一标识 | 当前填写的requestId |
| 2 | appid | PUSH给每一个应用分配的唯一ID，10位 |
| 3 | 推送任务创建时间 | 格式如：2016-05-16 00:00:52，精确到秒 |
| 4 | 推送任务发送时间 | 格式如：2016-05-16 00:00:52，精确到秒 |
| 5 | 推送任务过期时间 | 格式如：2016-05-16 00:00:52，精确到秒 |

流程唯一标识|appid|推送任务创建时间|推送任务发送时间|推送任务过期时间

日志样例：

147617731778492665000000|0000003358|2016-05-16 00:00:52|2016-05-16 00:00:52|2016-05-18 00:00:52

147617752478494285000000|0000003358|2016-05-16 00:01:37|2016-05-16 00:01:37|2016-05-18 00:01:37

1. MC bin log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 说明 | | |
| 1 | 时间戳 | 发生的时间，格式如：2012-08-27 22:29:44:422，精确到毫秒 | | |
| 2 | 日志类型 | 类型名称 | 类型含义 | 备注信息 |
| push cancel | 第三方调用CRS主动取消缓存消息 | Since CRS |
| push request | 首送push请求 | Since CRS |
| push response | push应答 | Since CRS |
| push timeout | push发送等待应答超时 | **Deprecated** |
| rsd request | 重发Push请求 | Since CRS |
| timeout rsd | 超时后立即重发 | **Deprecated** |
| timeout delete | 过期删除 | Since MessageCenter |
| replace | 被覆盖 | Since MessageCenter |
| 3 | ProviderID | Push消息发送方的帐号信息，当前为16位appId，当BI计算时，应截取前10位计算，以免后6位中多用户信息不同造成同一app计算出不同统计信息的问题。 | | |
| 4 | 消息优先级 | * 0：同步消息，需要等待终端的同步应答 * 1：异步消息 | | |
| 5 | 是否缓存 | * 0：发送失败不需要缓存重发 * 1：发送失败需要在有效期内缓存重发 | | |
| 6 | MsgID | 针对每条Push消息分配的全局唯一标志 | | |
| 7 | OriginalMsgID | 首次发送时该字段为空，重发时首次发送时的MsgID  (Since MessageCenter 该字段为空) | | |
| 8 | LastMsgID | 首次发送，该字段为空，重发时上次发送时的MsgID  针对存储的时候如果如果被覆盖，则在这个字段体现。被第六个字段对应的MsgID对应的消息覆盖，其他情况为空？  当为覆盖话单的时候，该字段表示newMsgID | | |
| 9 | DeviceToken | Push消息的接受方标识，特定终端上的特定终端应用，当前对前16位的deviceId做aescbc加密 | | |
| 10 | 是否群发 | * 0：非群发，即单发 * 1：代表群发 | | |
| 11 | RequestID | 由调用端生成并保证唯一性  群发的所有消息用同一个RequestID | | |
| 12 | 目标PushServer | 终端连接的PushServer的IP地址 | | |
| 13 | 发送结果 | * 0：成功 * 1：设备不在线 * 2：成功发送到设备，对应的客户端应用不存在，通常表示已经卸载 * 11：过期删除 * 12：无路由 * 13：设备在其他大区 * 14：内部错误 比如同步消息的时候无PushServer连接？异步可以缓存。 * 15: 被覆盖消息 | | |
| 14 | flag | 假在线标识，为1时，上一条BI不计算  \* 当消息下发时，先到内存数据库中查询该设备之前是否有消息发送，即到内存数据库中查询该设备是否有记录消息发送时间  \* 如果没有消息发送时间，则在内存数据库中写入该消息发送时间，话单中flag字段不填值   * \* 如果有消息发送时间，则将内存数据库中的时间，与当前时间比较，如果差值大于Config.msg\_time的值，则话单中flag字段填1；否则不填值 | | |
| 15 | Sha256 deviceId | * deviceId的sha256形式，不会改变，便于搜索某个deviceId的消息 | | |

样例如下：

2016-08-26 14:34:18 538|push request|2000000750000001|1|1|76BA249CB6431D1E|||63pKi+RdjFps3IcS5sFpd5I7D52V17MwCwdevLqW3ti/fC+bIXfQya12JC8ImiEk0000000000000000|0|1472238613397430|118.194.60.226|||1jTpOsB+6HVVJkqsX3lq0onY6VSHzRxOy7vrw72CWJ8=

2016-08-26 14:34:18 538|push request|2000000750000001|1|1|76BA2485AC030A1E|||GmyOHaGJWvpAKPrT8Y+BZCTrRdeDK0aqmEu+l5zTc96mF3fMBOg/N6XYuGnNXf2j0000000000000000|0|1472207063275930||||sXTzV8Eq7HybOw0YsgqjjKnSNUYBCRnA/bi9k2Z8+VE=

2016-08-26 14:34:18 538|push response||||76BA2485AC030A1E|||||||12||

2016-08-26 14:34:18 538|push request|0000003690000001|1|1|76BA246C33031E1E|||mnb+NI3IwPfSPfevrmmThjjTSlaO6iO8bvfLXvBXNgyJQdeUIEeUaI10my7BdiRT0000003690000001|0|ae5f082a07aad6f664c284a18a4a404a||||c+Mt7wDiHp0kCfsBm7/Jn85gf8t/hKct2RCMPwF5ZKs=

2016-08-26 14:34:18 538|push response||||76BA246C33031E1E|||||||12||

2016-08-26 14:48:52.0|timeout delete|2000000750000001|1||763B0D0330C30C28|||OQ0xTS6pf4kOvoyZ8wl+vEUXHMV0PaDF1YdsHNiSfjWPNm4E3YGSovYQeodPIljT0000000000000000||14720816962044540||11||zb1IONAU5DWRuuvweJuIVsTbmlK+c+XC9i8dLV8okAE=

1. CRS bin log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 说明 | | |
| 1 | 时间戳 | 发生的时间，格式如：2012-08-27 22:29:44:422，精确到毫秒 | | |
| 2 | 日志类型 | 类型名称 | 类型含义 | 备注信息 |
| push cancel | 第三方调用CRS主动取消缓存消息 | Since CRS |
| push request | 首送push请求 | Since CRS |
| push response | push应答 | Since CRS |
| push timeout | push发送等待应答超时 | **Deprecated** |
| rsd request | 重发Push请求 | Since CRS |
| timeout rsd | 超时后立即重发 | **Deprecated** |
| timeout delete | 过期删除 | Since MessageCenter |
| replace | 被覆盖 | Since MessageCenter |
| 3 | ProviderID | Push消息发送方的帐号信息 | | |
| 4 | 消息优先级 | * 0：同步消息，需要等待终端的同步应答 * 1：异步消息 | | |
| 5 | 是否缓存 | * 0：发送失败不需要缓存重发 * 1：发送失败需要在有效期内缓存重发 | | |
| 6 | MsgID | 针对每条Push消息分配的全局唯一标志 | | |
| 7 | OriginalMsgID | 首次发送时该字段为空，重发时首次发送时的MsgID  (Since MessageCenter 该字段为空) | | |
| 8 | LastMsgID | 首次发送，该字段为空，重发时上次发送时的MsgID  针对存储的时候如果如果被覆盖，则在这个字段体现。被第六个字段对应的MsgID对应的消息覆盖，其他情况为空？  当为覆盖话单的时候，该字段表示newMsgID | | |
| 9 | DeviceToken | Push消息的接受方标识，特定终端上的特定终端应用 | | |
| 10 | 是否群发 | * 0：非群发，即单发 * 1：代表群发 | | |
| 11 | RequestID | 由调用端生成并保证唯一性  群发的所有消息用同一个RequestID | | |
| 12 | 目标PushServer | 终端连接的PushServer的IP地址 | | |
| 13 | 发送结果 | * 0：成功 * 1：设备不在线 * 2：成功发送到设备，对应的客户端应用不存在，通常表示已经卸载 * 7：下发通知栏消息给sdk，由Push弹通知栏， PushAPK不允许弹通知栏 * 8：PushAPK不允许弹通知栏 * 9：其他应用不允许弹通知栏 * 11：过期删除 * 12：无路由 * 13：设备在其他大区 * 14：内部错误 比如同步消息的时候无PushServer连接？异步可以缓存。 * 15: 被覆盖消息 | | |
| 14 | flag | 假在线标识，为1时，上一条BI不计算  \* 当消息下发时，先到内存数据库中查询该设备之前是否有消息发送，即到内存数据库中查询该设备是否有记录消息发送时间  \* 如果没有消息发送时间，则在内存数据库中写入该消息发送时间，话单中flag字段不填值   * \* 如果有消息发送时间，则将内存数据库中的时间，与当前时间比较，如果差值大于Config.msg\_time的值，则话单中flag字段填1；否则不填值 | | |
| 15 | Sha256 deviceId |  | | |

日志样例：

2016-05-16 00:00:00 529|push request|1|0|0|6015100000003538|||b0ZrfBeLXLnEXq+Rg8eWl+ngAN5+thS6i5Ldg6LSR+c4vavY+QPLSSQAG4oxY4ms0000001019000001|0||118.194.56.74|||oznu8hRW88ar1n3QM86r8AP2dt9vs3USB0aBeXO+15A=

2016-05-16 00:00:01 017|push response||||6015100000003538|||||||0||oznu8hRW88ar1n3QM86r8AP2dt9vs3USB0aBeXO+15A=

注意：

正常情况下request和response成对出现

CRS中请求话单第三位记录的是providerid（与MC话单的区别）

CRS中请求话单不记录第11位

CRS中同一条消息每一次发送都会修改msgid

写入BI表名：ODS\_PUSH\_CRS\_LOG\_DM

1. TRS user request log

请求日志格式：

|  |  |  |
| --- | --- | --- |
| 序号 | 字段名称 | 说明 |
| 1 | 日期时间 | 精确到毫秒 yyyy-MM-dd HH:mm:ss SSS |
| 2 | 请求标识 | Req |
| 3 | 加密deviceId | deviceId的aes/cbc加密 |
| 4 | mcc | 移动国家码 |
| 5 | mnc | 移动网络码 |
| 6 | belongId | 大区ID，归属地编号 |
| 7 | agentVersion | Pushagent的版本号 |
| 8 | Info | 终端相关信息 |
| 9 | agentVersion | 应用包名，代表渠道 |
| 10 | Mode | 终端设备型号信息 |
| 11 | [intelligent](http://www.iciba.com/intelligent) | 是否支持Polling模式参数下发，1为支持 |
| 12 | requestIp | 客户端请求IP |
| 13 | sha256加密imei |  |

请求日志样例：

2016-05-16 20:53:28 940|Req|/yM7I/WKuImNEw/HtTQWvUmUIPYbP35eQz428AFzaMrWXwWWPrcZtXA0Zffb16OF|460|00|1|2611|2613|android|HUAWEI NXT-AL10|1|192.168.68.206|9MvU80xVlHB3jiwxl8BWgKKxWz5LuLe3KAEx59uUS+4=

写入BI表名：ODS\_PUSH\_TRS\_REQ\_LOG\_DM

1. TRS user response log

响应日志格式：

|  |  |  |
| --- | --- | --- |
| 序号 | 字段名称 | 说明 |
| 1 | 日期时间 | 精确到毫秒 yyyy-MM-dd HH:mm:ss SSS |
| 2 | 响应标识 | Rsp |
| 3 | 加密deviceId | deviceId的aes/cbc加密 |
| 12 | Result | 响应信息，json格式，包含客户端需要的各种参数 |
| 13 | sha256加密imei |  |

响应日志样例：

2016-05-16 20:53:28 941|Rsp|iTXBilwPTw2vvAh5YUHNdl3DWQBV5U71H4gIz/hOmwPelSrk1iQNHELrD5lRwj8u|{"wifi\_serverRec2\_min":10,"publicKey":"MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDiCa5gkvCb+/dHAcgN1WMm0ItA\rY1njDoy6bPCE+oHZI439lmjP14PH7n2xtKsuybPbzPAGwuXq4doRz+wB8JiOUjNQ\rVI88zNzDDhdV3pxQlFgk61VojWtVBH2H45qMPMbMs4HdVs0Qcida2IhXOi6eAyRK\rp3PApI7e/ta1FHYKiwIDAQAB","socketConnectReadOut":20,"PushID":"b15sXsdAJZ3E5+GYXSbrbVrM0QAfK8+ADIp+DNNyBTufFGItaXNaLNRW7W+pBMYY","tokenRegTimeOut":30,"wifi\_serverRec4\_min":300,"fir3gHb":102,"firstQueryTRSHourTimes":2,"serverRec1\_min":3,"connTrsItval":300,"ConnRange":600,"push1StartInt":6,"serverRec2\_min":10,"WifiKeepConnTime":300,"wifi\_serverRec1\_min":3,"noNetHeartbeat":7200,"chkFailConnSvr\_time":172800,"g3MinHeartbeat":300,"wifiMaxHeartbeat":570,"serverContinueConn\_time":300,"pushLeastRun\_time":60,"socketConnTimeOut":90,"serverIp":"118.194.56.47","firWifiHb":22,"biPkgs":"com.hotalk","serverRec5\_min":300,"wifiFlowcVlomes2":30000,"serverRec4\_min":300,"wifiFlowcVlomes3":3000,"flowcVlomes1":9000,"wifiFlowcVlomes1":90000,"flowcVlomes2":3000,"KeepConnTime":300,"wifiFirstQueryTRSHourTimes":100,"push3StartInt":300,"flowcInterval3":86400,"flowcInterval2":604800,"flowcVlomes3":600,"maxQueryTRSDayTimes":1,"pollingId":204396852,"wifiMaxQueryTRSDayTimes":10,"flowcInterval1":2592000,"cloneCheckItval":600,"USE\_SSL":3,"wifi\_serverRec3\_min":30,"grpNum":0,"srvMaxFail\_times":6,"push2StartInt":60,"result":0,"pollingPort":5224,"pollingIp":"122.11.38.32","ReConnInterval":300,"g3MaxHeartbeat":600,"wifiFlowcInterval1":2592000,"wifiFlowcInterval2":604800,"wifiFlowcInterval3":86400,"belongId":"1","wifiMinHeartbeat":300,"wifi\_serverRec5\_min":300,"serverRec3\_min":30,"trsValid\_min":14400,"firstQueryTRSDayTimes":6,"heartBeatRspTimeOut":10,"wifiFirstQueryTRSDayTimes":200,"WifiReConnInterval":300,"push4StartInt":600,"connTrsErrItval":1800,"connSuccItval":3,"serverPort":5223,"MaxConnTimes":3,"trsValid\_max":86400,"maxQTRS\_times":6}|9MvU80xVlHB3jiwxl8BWgKKxWz5LuLe3KAEx59uUS+4=

写入BI表名：ODS\_PUSH\_TRS\_RSP\_LOG\_DM

JSON格式：

|  |
| --- |
| {  "wifi\_serverRec2\_min": 10,  "publicKey": "MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDiCa5gkvCb+/dHAcgN1WMm0ItA\rY1njDoy6bPCE+oHZI439lmjP14PH7n2xtKsuybPbzPAGwuXq4doRz+wB8JiOUjNQ\rVI88zNzDDhdV3pxQlFgk61VojWtVBH2H45qMPMbMs4HdVs0Qcida2IhXOi6eAyRK\rp3PApI7e/ta1FHYKiwIDAQAB",  "socketConnectReadOut": 20,  "PushID": "b15sXsdAJZ3E5+GYXSbrbVrM0QAfK8+ADIp+DNNyBTufFGItaXNaLNRW7W+pBMYY",  "tokenRegTimeOut": 30,  "wifi\_serverRec4\_min": 300,  "fir3gHb": 102,  "firstQueryTRSHourTimes": 2,  "serverRec1\_min": 3,  "connTrsItval": 300,  "ConnRange": 600,  "push1StartInt": 6,  "serverRec2\_min": 10,  "WifiKeepConnTime": 300,  "wifi\_serverRec1\_min": 3,  "noNetHeartbeat": 7200,  "chkFailConnSvr\_time": 172800,  "g3MinHeartbeat": 300,  "wifiMaxHeartbeat": 570,  "serverContinueConn\_time": 300,  "pushLeastRun\_time": 60,  "socketConnTimeOut": 90,  "serverIp": "118.194.56.47",  "firWifiHb": 22,  "biPkgs": "com.hotalk",  "serverRec5\_min": 300,  "wifiFlowcVlomes2": 30000,  "serverRec4\_min": 300,  "wifiFlowcVlomes3": 3000,  "flowcVlomes1": 9000,  "wifiFlowcVlomes1": 90000,  "flowcVlomes2": 3000,  "KeepConnTime": 300,  "wifiFirstQueryTRSHourTimes": 100,  "push3StartInt": 300,  "flowcInterval3": 86400,  "flowcInterval2": 604800,  "flowcVlomes3": 600,  "maxQueryTRSDayTimes": 1,  "pollingId": 204396852,  "wifiMaxQueryTRSDayTimes": 10,  "flowcInterval1": 2592000,  "cloneCheckItval": 600,  "USE\_SSL": 3,  "wifi\_serverRec3\_min": 30,  "grpNum": 0,  "srvMaxFail\_times": 6,  "push2StartInt": 60,  "result": 0,  "pollingPort": 5224,  "pollingIp": "122.11.38.32",  "ReConnInterval": 300,  "g3MaxHeartbeat": 600,  "wifiFlowcInterval1": 2592000,  "wifiFlowcInterval2": 604800,  "wifiFlowcInterval3": 86400,  "belongId": "1",  "wifiMinHeartbeat": 300,  "wifi\_serverRec5\_min": 300,  "serverRec3\_min": 30,  "trsValid\_min": 14400,  "firstQueryTRSDayTimes": 6,  "heartBeatRspTimeOut": 10,  "wifiFirstQueryTRSDayTimes": 200,  "WifiReConnInterval": 300,  "push4StartInt": 600,  "connTrsErrItval": 1800,  "connSuccItval": 3,  "serverPort": 5223,  "MaxConnTimes": 3,  "trsValid\_max": 86400,  "maxQTRS\_times": 6  } |

1. Auth login log

每个字段的定义如下（竖线分隔，对于日志类型没有意义的字段为空）。

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 字段名称 | 说明 | |
| 1 | 时间戳 | 发生的时间，格式如：2012-08-27 22:29:44:422，精确到毫秒 | |
| 2 | 日志类型 | 1 ：登入 ， 0：登出 | |
| 3 | deviceId | 客户端标示，采用与BI协商的密钥加密 | |
| 4 | Pushserver id | Pushserver id来标示唯一的pushserver | |
| 5 | 网络类型 |  | |
| 6 | 客户端ip | 客户端上报的ip地址 | |
| 7 | Session.key | 客户端连接标示的key | |
| 8 | Session.seq | 客户端连接标示的seq | |
| 9 | agentVersion | 客户端版本号 | |
| 10 | lastDisconnectTime | 上次连接断开的时间 | |
| 11 | currentConnectTime | 本次连接连接时间 | |
| 12 | currentTime | 当前时间 | |
| 13 | eventTimes | 上报事件的个数 | |
| 14-62 | event | 发生时间 |  |
| 网络类型 |  |
| 网络事件 | 0：断开 1：连接 |
| 63 | Sha256 deviceId | 采用sha256 hash过的deviceid | |

样例：

2016-05-16 17:06:52 386|1|CiBKa2N0gz+FuGeUrXCVd0wKptzrKT9yUbwiiMxqVQ1y8Ea/7fr3SPbjVOj5vCoy|200|0|117.136.40.20|1946365|1772|2551|2016-05-16 16:42:22 658|2016-05-16 17:06:51 634|2016-05-16 17:06:51 647|2|2016-05-16 16:42:25 771|0|0|2016-05-16 17:06:51 346|0|1|||||||||||||||||||||||||||||||||||||||||||LzTtRbw5sQjLYmd1GnpUpV4xGqo/VHor8JjyHg81iKk=

写入的BI表名称: ODS\_PUSH\_AUTH\_LOG\_DM

1. Auth logout log

每个字段的定义如下（竖线分隔，对于日志类型没有意义的字段为空）。

|  |  |  |
| --- | --- | --- |
| 序号 | 字段名称 | 说明 |
| 1 | 时间戳 | 发生的时间，格式如：2012-08-27 22:29:44:422，精确到秒 |
| 2 | 日志类型 | 1 ：登入 ， 0：登出 |
| 3 | deviceId | 客户端标示，采用与BI协商的密钥加密 |
| 4 | Pushserver id | Pushserver id来标示唯一的pushserver |
| 5 | null |  |
| 6 | null |  |
| 7 | Session.key | 客户端连接标示的key |
| 8 | Session.seq | 客户端连接标示的seq |
| 9 | Sha256 deviceId | 采用sha256 hash过的deviceid |

样例：

2016-05-16 17:06:52|0|lOXlS/v5FegkLcMcdJlu2t3g1FHJEaSHdNnUjMNu7lIs8+Gd7wpeyRbpQiIRRLkp|162|||1889771|1452|SVDd66uKKXPho1+RIhKi78ajBzSpK/f7cVR8Ug2JQtw=

写入的BI表名称: ODS\_PUSH\_AUTH\_LOG\_DM

1. Auth token apply log

每个字段的定义如下（竖线分隔，对于日志类型没有意义的字段为空）。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 说明 | | |
| 1 | 时间戳 | 发生的时间，格式如：2012-08-27 22:29:44:422，精确到毫秒 | | |
| 2 | 日志类型 | 类型名称 | 类型含义 | 备注信息 |
| register | 申请token |  |
| uninstall | 注销token |  |
| 3 | deviceId | 客户端标示，采用与BI协商的密钥加密 | | |
| 4 | appId | 申请token的应用标示，16位 | | |
| 5 | 包名 | 申请token的应用包名 | | |
| 6 | channel | 应用的渠道 | | |
| 7 | Sha256 deviceId | 采用sha256 hash过的deviceid | | |

样例：

2016-05-16 20:13:48 006|register|EfdK76JsF3HeOZ+n/gq50E0oQHQJjxOb++zu/qMS123Dk/sB0khXL6tM1hewQWRe|0000002108000001|com.android.mediacenter||kQolB3uGpm2tWoXtsB0hGzwD9+8+SzPgFe15ju/vrxs=

写入的BI表名：ODS\_PUSH\_DEVICE\_TOKEN\_DM

1. Auth token release log

每个字段的定义如下（竖线分隔，对于日志类型没有意义的字段为空）。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 说明 | | |
| 1 | 时间戳 | 发生的时间，格式如：2012-08-27 22:29:44:422，精确到毫秒 | | |
| 2 | 日志类型 | 类型名称 | 类型含义 | 备注信息 |
| register | 申请token | register |
| uninstall | 注销token | uninstall |
| 3 | deviceId | 客户端标示，采用与BI协商的密钥加密 | | |
| 4 | appId | 申请token的应用标示，16位 | | |
| 5 | null |  | | |
| 6 | null |  | | |
| 7 | Sha256 deviceId | 采用sha256 hash过的deviceid | | |

样例：

2016-05-16 20:13:47 771|uninstall|KWeSyKLSnGs0yKqlxayyGC8uJ2ZuVSeIvvTqCVh0C0Paazgw1Tjt7Nffh+ky3zTX|2000000736000001|||iiBi3dvFaLJcefGHNOMgqqpklfj7A6spCLvr4Rr9vdI=

写入的BI表名：ODS\_PUSH\_DEVICE\_TOKEN\_DM

1. XB interface log

日志格式：

|  |  |  |
| --- | --- | --- |
| 序号 | 字段名称 | 说明 |
| 1 | 日期时间 | 发生的时间，格式如：2016-05-16 18:00:01 614，精确到毫秒 |
| 2 | 级别 | 目前都填写的：INFO |
| 3 | 请求响应标识 | request/response |
| 4 | 接口类型 | 目前都填写的：TCP |
| 5 | 接口名称 | 填写记录了接口日志的接口名称，目前只有：openpush.openapi.notification\_send |
| 6 | 源端标识 | 当前未填写 |
| 7 | 宿端标识 | 当前未填写 |
| 8 | 流程唯一标识 | 当前未填写 |
| 9 | 事务ID | 当前填写的requestId |
| 10 | ReturnCode | 0标识成功，其余均是失败 |
| 11 | ReturnInfo | 失败时的错误码描述 |
| 12 | 扩展字段 | 当前未填写 |

日志样例：

2016-05-16 18:00:01 614|INFO|request|TCP|openpush.openapi.notification\_send||||147702182686290265000000|||

2016-05-16 18:00:01 616|INFO|response|TCP|openpush.openapi.notification\_send||||147702182686290265000000|0||

注意：

request和response正常情况下成对出现，日志中第9位“流程唯一标识”唯一标记request/response对

写入BI表名：ODS\_PUSH\_XB\_IF\_LOG\_DM

1. MC interface log

日志格式：

|  |  |  |
| --- | --- | --- |
| 序号 | 字段名称 | 说明 |
| 1 | 流程唯一标识 | 当前填写的requestId |
| 2 | appid | PUSH给每一个应用分配的唯一ID，10位 |
| 3 | 推送任务创建时间 | 格式如：2016-05-16 00:00:52，精确到秒 |
| 4 | 推送任务发送时间 | 格式如：2016-05-16 00:00:52，精确到秒 |
| 5 | 推送任务过期时间 | 格式如：2016-05-16 00:00:52，精确到秒 |

流程唯一标识|appid|推送任务创建时间|推送任务发送时间|推送任务过期时间

日志样例：

147617731778492665000000|0000003358|2016-05-16 00:00:52|2016-05-16 00:00:52|2016-05-18 00:00:52

147617752478494285000000|0000003358|2016-05-16 00:01:37|2016-05-16 00:01:37|2016-05-18 00:01:37

写入BI表名：ODS\_PUSH\_PORTAL\_NOTIFICATION\_DM

1. Push Server Connection log

每个字段的定义如下（‘，’分隔，对于日志类型没有意义的字段为空）。

|  |  |  |
| --- | --- | --- |
| 序号 | 字段名称 | 说明 |
| 1 | Pushserver ip | Pushserver ip地址 |
| 2 | 时间 | 时间点，精度为秒 |
| 3 | 连接数 | 时间点对应的此pushserver的连接数 |

样例

118.194.60.236, 2014-12-18 16:45:00, 661388

写入BI的表名：ODS\_PUSH\_CONNECT\_INFO\_DS

# Serviceability可服务性设计

## 业务安装

## (灰度)升级

## 业务扩容

## 故障定位

## 业务告警

# Security安全性设计

## 安全威胁建模

# UI设计

# Appendix附录