移动推送 TPNS

旧版（信鸽）迁移指南

产品文档

腾讯云

【联系我们】

我们致力于为您提供个性化的售前购买咨询服务，及相应的技术售后服务，任何问题请联系 4009100100。

文档目录

旧版（信鸽）迁移指南

概述

最近更新时间：2020-09-04 17:37:39

该文档适用于从移动推送 TPNS 原免费版（https://xg.qq.com）迁移过来的用户。

随着 GDPR 数据隐私规范的落实和国家对数据合规性要求越来越严格，为了给您提供更强大、更优质的推送服务，信鸽团队在腾讯云上推出移动推送 TPNS（Tencent Push Notification，TPNS）版本，相较于免费版本有以下提升：

对比项	信鸽免费版	移动推送 TPNS（专享通道版）
推送速度	共享通道，高峰易堵塞	资源隔离，独享高速通道
数据存储	共享存储空间	App 级数据隔离，更安全
厂商通道	支持华为、小米、魅族、FCM、APNs	支持华为、小米、魅族、FCM、vivo、OPPO、APNs
SDK 合规性	不符合 GDPR 规范	符合 GDPR 规范，联合保活能力可自选开关
限速推送	不支持	支持
全球漫游	不支持	支持
境外推送	不支持	支持，服务全球化部署。符合数据出入境规范
专有客服	不支持	7 X 24 小时专有客服支持
高级功能	不支持	• 支持聚合单推统计 • 智能通道策略 • 通道配额预估 • 消息撤回、覆盖 • 通知栏开启率统计 • 失效/卸载设备数统计 • 更丰富的数据分析、A/B Test、音视频富媒体消息等功能
SDK 稳定性	一般	增加 Crash 监控，提升稳定性

迁移说明

腾讯信鸽（https://xg.qq.com）将于2020年10月31日关闭服务，请尽快迁移至腾讯云移动推送TPNS（https://cloud.tencent.com/product/tpns），我们将全程提供迁移指引。

迁移前准备

最近更新时间：2020-07-07 11:56:42

填写意向迁移月份

为及时配合您的迁移，邀请您填写您的迁移意向和迁移计划！单击 意向收集表 填写。

注册腾讯云账号

在开始使用移动推送 TPNS 服务之前，您需要先注册一个腾讯云账号，详情请参见 注册腾讯云 文档。

创建产品和应用

完成注册腾讯云账号之后，您可以登录 移动推送 TPNS 控制台 创建产品及应用，可参考 创建产品和应用 文档。

购买服务

创建完应用后，您可在服务管理页面，选择购买服务。

- 请尽量保证填写信息的真实性，确保申请通过。
- 体验版应用最多支持1000个设备参与测试，超过限制可能会被终止试用，请勿商用以免造成损失。

购买服务

购买之前可查看 购买指南 预估您所需要购买的版本类型，参考 购买推送服务 完成购买。

Android 迁移指南

最近更新时间：2020-07-06 13:10:31

为保障迁移后的正常使用，请您升级至 Android V1.1.5.5 及以上版本，以下是针对从信鸽平台 4.x 版本迁移到移动推送 TPNS 平台 Android V1.1.5.5 及以上版本的变更说明。

注销信鸽平台推送服务

注意：

- 如果未做以下配置，在【旧版信鸽】和【移动推送 TPNS】两个平台上同时推送时，可能会出现重复消息。
- 如果您的应用确定不再使用【旧版信鸽】进行推送，可忽略上方配置。

如果 App 的推送服务是从信鸽平台（https://xg.qq.com）迁移到移动推送 TPNS 平台，TPNS 版本需要增加以下配置：

- 在 AndroidManifest 上添加的 application 节点内添加以下配置，填写信鸽平台的 ACCESS ID

```
<meta-data
android:name="XG_OLD_ACCESS_ID"
android:value="信鸽平台应用的ACCESS ID" />
```

- 在应用首次覆盖安装时，如您在logcat中看到如下日志打印，即说明 SDK 已成功获取信鸽版本的推送信息，将在推送注册时一并向服务器上报：

```
D/TPush: [PushServiceNetworkHandler] FreeVersionInfo -> AccessId:2100168750, token:0e3baa5e895d47f94e21840a231a0e6b651b7f47, channel:huawei
```

自动集成方式

依赖变更

在【app build.gradle】>【dependencies】下，请将以下信鸽 4.x 版本依赖做相应变更。

变更前：

```
implementation 'com.tencent.xinge:xinge:4.3.5-release'
implementation 'com.tencent.wup:wup:1.0.0.E-Release'
implementation 'com.tencent.mid:mid:4.0.7-Release'
```

变更后：

```
// TPNS 推送 [VERSION] 为当前SDK版本号，版本号可在SDK下载页查看
implementation 'com.tencent.tpns:tpns:[VERSION]-release'
```

组件变更

若监听消息自行继承过 XGBaseReceiver 接口类，请在 AndroidManifest 文件下，将该自定义组件注册内容做相应变更。

变更前：

```
<receiver android:name="完整包名.MessageReceiver"
android:exported="true" >
<intent-filter>
<!-- 接收消息透传 -->
<action android:name="com.tencent.android.tpush.action.PUSH_MESSAGE" />
<!-- 监听注册、反注册、设置/删除标签、通知被点击等处理结果 -->
<action android:name="com.tencent.android.tpush.action.FEEDBACK" />
</intent-filter>
</receiver>
```

变更后：

```
<receiver android:name="完整包名.MessageReceiver">
<intent-filter>
<!-- 接收消息透传 -->
<action android:name="com.tencent.android.xg.vip.action.PUSH_MESSAGE" />
<!-- 监听注册、反注册、设置/删除标签、通知被点击等处理结果 -->
<action android:name="com.tencent.android.xg.vip.action.FEEDBACK" />
</intent-filter>
</receiver>
```

说明：
因 XGBaseReceiver 接口类中有方法名变更和接口新增，请注意修改、新增实现对应接口。详情请参见 接口变更说明。

手动集成方式

依赖包变更

请前往 移动推送 TPNS 控制台，下载 Android SDK 压缩包、解压并按照以下步骤替换依赖包文件：

- 删除信鸽 4.x 版本所使用的全部 .jar 文件，使用 libs 目录下的所有 .jar 文件替换。
- 删除信鸽 4.x 版本所使用的全部 .so 文件，在 Other-Platform-SO 目录下，按照当前 .so 支持的平台添加 .so 文件。

AndroidManifest 文件变更

1. 在 Android 配置文件 AndroidManifest.xml 中，删除所有信鸽 4.x 版本使用组件和权限，删除内容具体如下：

```
<application>
<!-- 【必须】 信鸽receiver广播接收 -->
<receiver android:name="com.tencent.android.tpush.XGPushReceiver"
android:process=":xg_service_v4" >
<intent-filter android:priority="0x7fffffff" >
<!-- 【必须】 信鸽SDK的内部广播 -->
<action android:name="com.tencent.android.tpush.action.SDK" />
<action android:name="com.tencent.android.tpush.action.INTERNAL_PUSH_MESSAGE" />
<!-- 【必须】 系统广播：开屏和网络切换 -->
<action android:name="android.intent.action.USER_PRESENT" />
<action android:name="android.net.conn.CONNECTIVITY_CHANGE" />
<!-- 【可选】 一些常用的系统广播，增强信鸽service的复活机会，请根据需要选择。当然，您也可以添加APP自定义的一些广播让启动service -->
<action android:name="android.bluetooth.adapter.action.STATE_CHANGED" />
<action android:name="android.intent.action.ACTION_POWER_CONNECTED" />
<action android:name="android.intent.action.ACTION_POWER_DISCONNECTED" />
</intent-filter>
</receiver>
<!-- 【可选】APP实现的Receiver，用于接收消息透传和操作结果的回调，请根据需要添加 -->
<!-- YOUR_PACKAGE_PATH.CustomPushReceiver需要改为自己的Receiver： -->
<receiver android:name="com.qq.xgdemo.receiver.MessageReceiver"
android:exported="true" >
<intent-filter>
<!-- 接收消息透传 -->
<action android:name="com.tencent.android.tpush.action.PUSH_MESSAGE" />
<!-- 监听注册、反注册、设置/删除标签、通知被点击等处理结果 -->
<action android:name="com.tencent.android.tpush.action.FEEDBACK" />
</intent-filter>
</receiver>
<!-- 【注意】 如果被打开的activity是启动模式为SingleTop，SingleTask或SingleInstance，请根据通知
```

```
的异常自查列表第8点处理-->
<activity
android:name="com.tencent.android.tpush.XGPushActivity"
android:exported="false" >
<intent-filter>
<!-- 若使用AndroidStudio，请设置android:name="android.intent.action"-->
<action android:name="" />
</intent-filter>
</activity>

<!-- 【必须】 信鸽service -->
<service
android:name="com.tencent.android.tpush.service.XGPushServiceV4"
android:exported="true"
android:persistent="true"
android:process=":xg_service_v4" />
<!-- 云控相关 -->
<receiver android:name="com.tencent.android.tpush.cloudctr.network.CloudControlDownloadRec
eiver">
<intent-filter>
<action android:name="com.tencent.android.xg.vip.action.cloudcontrol.action.DOWNLOAD_FILE_FI
NISH" />
</intent-filter>
</receiver>
<service android:name="com.tencent.android.tpush.cloudctr.network.CloudControlDownloadServi
ce" />
<!-- 【必须】 提高service的存活率 -->
<service
android:name="com.tencent.android.tpush.rpc.XGRemoteService"
android:exported="true">
<intent-filter>
<!-- 【必须】 请修改为当前APP包名 .PUSH_ACTION, 如demo的包名为：com.qq.xgdemo -->
<action android:name="当前应用的包名.PUSH_ACTION" />
</intent-filter>
</service>

<!-- 【必须】 【注意】authorities修改为 包名.AUTH_XGPUSH, 如demo的包名为：com.qq.xgdemo-->
<provider
android:name="com.tencent.android.tpush.XGPushProvider"
android:authorities="当前应用的包名.AUTH_XGPUSH"
android:exported="true"/>
<!-- 【必须】 【注意】authorities修改为 包名.TPUSH_PROVIDER, 如demo的包名为：com.qq.xgdemo-
->
<provider
android:name="com.tencent.android.tpush.SettingsContentProvider"
```

```
android:authorities="当前应用的包名.TPUSH_PROVIDER"
android:exported="false" />
<!--【必须】【注意】authorities修改为 包名.TENCENT.MID.V4, 如demo的包名为：com.qq.xgdemo--
>
<provider
android:name="com.tencent.mid.api.MidProvider"
android:authorities="当前应用的包名.TENCENT.MID.V4"
android:exported="true" >
</provider>
<!--【必须】请将YOUR_ACCESS_ID修改为APP的AccessId，“21”开头的10位数字，中间没空格 -->
<meta-data
android:name="XG_V2_ACCESS_ID"
android:value="YOUR_ACCESS_ID" />
<!--【必须】请将YOUR_ACCESS_KEY修改为APP的AccessKey，“A”开头的12位字符串，中间没空格 --
>
<meta-data
android:name="XG_V2_ACCESS_KEY"
android:value="YOUR_ACCESS_KEY" />
</application>
<!--【必须】信鸽SDK所需权限 -->
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="android.permission.VIBRATE" />
<!--【常用】信鸽SDK所需权限 -->
<uses-permission android:name="android.permission.RECEIVE_USER_PRESENT" />
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.WRITE_SETTINGS" />
<!--【可选】信鸽SDK所需权限 -->
<uses-permission android:name="android.permission.RESTART_PACKAGES" />
<uses-permission android:name="android.permission.BROADCAST_STICKY" />
<uses-permission android:name="android.permission.KILL_BACKGROUND_PROCESSES" />
<uses-permission android:name="android.permission.GET_TASKS" />
<uses-permission android:name="android.permission.READ_LOGS" />
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BATTERY_STATS" />
```

2. 请参见新版本 SDK 集成 重新添加组件和权限。

接口变更

与 4.x 对比，部分 API 接口做了以下变更。

- 删除带账号注册的 API，设置账号只能通过 bindAccount 或 appendAccount 来设置。

```
// 删除以下API
XGPushManager.registerPush(Context context, String account)
XGPushManager.registerPush(Context context, String account, final XGIOperateCallback callback)
XGPushManager.registerPush(Context context, String account,String url, String payload, String oth
erToken, final XGIOperateCallback callback)
```

- 账号绑定和注册推送功能分开，bindAccount 和 appendAccount 不再带有注册功能，推荐在 registerPush 成功的回调里，调用 bindAccount 或 appendAccount。
- 继承 XGPushBaseReceiver 时需要多实现以下两个函数。

```
/**
* 设置帐号结果处理函数
*/
public abstract void onSetAccountResult(Context context, int errorCode,
String operateName);
/**
* 删除帐号结果处理函数
*/
public abstract void onDeleteAccountResult(Context context, int errorCode,
String operateName);
```

- 继承 XGPushBaseReceiver 的实现类，在 AndroidManifest 文件配置时，前缀命名规则为 com.tencent.android.xg.vip.action.，区别于 4.x 版本的 com.tencent.android.tpush.action.。
  TPNS 版本正确配置：

```
<receiver android:name="com.tencent.android.xg.cloud.demo.MessageReceiver">
<intent-filter>
<!-- 接收消息透传 -->
<action android:name="com.tencent.android.xg.vip.action.PUSH_MESSAGE" />
<!-- 监听注册、反注册、设置/删除标签、通知被点击等处理结果 -->
<action android:name="com.tencent.android.xg.vip.action.FEEDBACK" />
</intent-filter>
</receiver>
```

厂商通道集成变更

厂商通道变更只需做以下变更：

- 删除 gradle 依赖（自动集成）

```
// 删除以下对应的厂商依赖
implementation 'com.tencent.xinge:mipush:4.3.2-xiaomi-release'
implementation 'com.tencent.xinge:xgmz:4.3.2-meizu-release'
implementation 'com.tencent.xinge:xghw:4.3.2-huawei-release'
// fcm
implementation 'com.tencent.xinge:fcm:4.3.2-release'
implementation 'com.google.firebase:firebase-messaging:17.3.1'
```

替换为以下对应的厂商依赖，[VERSION] 为当前 SDK 版本号，版本号可在 SDK 下载页查看。

```
implementation 'com.tencent.tpns:xiaomi:[VERSION]-release'
implementation 'com.tencent.tpns:meizu:[VERSION]-release'
implementation 'com.tencent.tpns:huawei:[VERSION]-release'
//fcm
implementation 'com.tencent.tpns:fcm:[VERSION]-release'
implementation 'com.google.firebase:firebase-messaging:17.6.0'
```

- 手动集成的方式需要替换 .jar 文件
  前往 移动推送 TPNS 控制台，下载 Android SDK 压缩包、并解压目录 Other-Push-jar 下，找到对应厂商通道所需 .jar 文件，替换信鸽 4.x 版本使用的厂商通道 .jar 文件。

代码混淆保留变更

如果您的项目中使用 proguard 等工具做了代码混淆，请将以下混淆保留选项：

变更前：

```
-keep public class * extends android.app.Service
-keep public class * extends android.content.BroadcastReceiver
-keep class com.tencent.android.tpush.** {*;}
-keep class com.tencent.mid.** {*;}
-keep class com.qq.taf.jce.** {*;}
-keep class com.tencent.bigdata.** {*;}
```

变更后：

```
-keep public class * extends android.app.Service
-keep public class * extends android.content.BroadcastReceiver
-keep class com.tencent.android.tpush.** {*;}
-keep class com.tencent.bigdata.baseapi.** {*;}
-keep class com.tencent.bigdata.mqttchannel.** {*;}
-keep class com.tencent.tpns.dataacquisition.** {*;}
```

iOS 迁移指南

最近更新时间：2020-07-06 13:10:36

版本迁移指南

如您接入的是信鸽平台的2.x的版本，请参考 SDK 集成文档 接入。
为保障迁移后的正常使用，请您升级至 iOS V1.2.5.3 及以上版本，以下是针对从信鸽平台的 3.x 版本迁移到移动推送 TPNS iOS V1.2.5.3 及以上版本的变更说明：

自动集成方式

Pod名称变更

通过 Cocoapods 下载地址：
`pod 'TPNS-iOS'`

托管仓库变更

因仓库地址变更为腾讯工蜂首次通过 pod 下载需要注册登录 工蜂地址，并在【账户】菜单栏中设置账号和密码，然后在 Terminal 中设置腾讯工蜂的账号和密码。后续即可正常使用，当前 PC 不需要再次登录。

新增支持 Carthage 导入

在 Cartfile 文件中指明依赖的第三方库：
`github "xingePush/carthage-TPNS-iOS"`

手动导入

SDK包下载

请前往 移动推送 TPNS 控制台 下载 iOS SDK 压缩包、解压。

工程文件变更

变更前：

`XGPush.h` 、 `libXG-SDK.a`

变更后：

`XGPush.h` 、 `libXG-SDK-Cloud.a` 、 `XGMTACloud.framework`

依赖系统库变更

在 Build Phases 下，请将以下信鸽 3.x 版本系统依赖库做相应变更。

变更前：

* **CoreTelephony**.framework
* **SystemConfiguration**.framework
* **UserNotifications**.framework
* **libXG-SDK**.a
* **libz**.tbd
* **libsqlite3**.0.tbd

变更后：

* XGMTACloud.framework
* CoreTelephony.framework
* SystemConfiguration.framework
* UserNotifications.framework
* libXG-SDK-Cloud.a
* libz.tbd
* CoreData.framework
* CFNetwork.framework
* libc++.tbd

接口变更

与信鸽版本对比，部分 API 接口做了以下变更：

不推荐再调用 reportXGNotification* 数据统计上报接口，SDK 已内部自动处理。

变更前：

```
/**
收到推送的回调
@param application UIApplication 实例
@param userInfo 推送时指定的参数
@param completionHandler 完成回调
*/
- (void)application:(UIApplication *)application
didReceiveRemoteNotification:(NSDictionary *)userInfo
fetchCompletionHandler:(void (^)(UIBackgroundFetchResult))completionHandler
{
[[XGPush defaultManager] reportXGNotificationInfo:userInfo];
```

```
completionHandler(UIBackgroundFetchResultNewData);
}
// iOS 10 新增回调 API
// App 用户点击通知
// App 用户选择通知中的行为
// App 用户在通知中心清除消息
// 无论本地推送还是远程推送都会走这个回调
#if __IPHONE_OS_VERSION_MAX_ALLOWED >= __IPHONE_10_0
- (void)xgPushUserNotificationCenter:(UNUserNotificationCenter *)center
didReceiveNotificationResponse:(UNNotificationResponse *)response
withCompletionHandler:(void (^)(void))completionHandler
{
[[XGPush defaultManager] reportXGNotificationResponse:response];
completionHandler();
}

// App 在前台弹通知需要调用这个接口
- (void)xgPushUserNotificationCenter:(UNUserNotificationCenter *)center
willPresentNotification:(UNNotification *)notification
withCompletionHandler:(void (^)(UNNotificationPresentationOptions))completionHandler
{
[[XGPush defaultManager] reportXGNotificationInfo:notification.request.content.userInfo];
completionHandler(UNNotificationPresentationOptionBadge | UNNotificationPresentationOptionSo
und | UNNotificationPresentationOptionAlert);
}
#endif
```

变更后：(如果没有自定义处理需求，则不再需要实现以下回调)

```
/**
收到推送的回调
@param application UIApplication 实例
@param userInfo 推送时指定的参数
@param completionHandler 完成回调
*/
- (void)application:(UIApplication *)application
didReceiveRemoteNotification:(NSDictionary *)userInfo
fetchCompletionHandler:(void (^)(UIBackgroundFetchResult))completionHandler
{
completionHandler(UIBackgroundFetchResultNewData);
}
// iOS 10 新增回调 API
// App 用户点击通知
// App 用户选择通知中的行为
// App 用户在通知中心清除消息
```

```
// 无论本地推送还是远程推送都会走这个回调
#if __IPHONE_OS_VERSION_MAX_ALLOWED >= __IPHONE_10_0
- (void)xgPushUserNotificationCenter:(UNUserNotificationCenter *)center
didReceiveNotificationResponse:(UNNotificationResponse *)response
withCompletionHandler:(void (^)(void))completionHandler
{
completionHandler();
}

// App 在前台弹通知需要调用这个接口
- (void)xgPushUserNotificationCenter:(UNUserNotificationCenter *)center
willPresentNotification:(UNNotification *)notification
withCompletionHandler:(void (^)(UNNotificationPresentationOptions))completionHandler
{
completionHandler(UNNotificationPresentationOptionBadge | UNNotificationPresentationOptionSo
und | UNNotificationPresentationOptionAlert);
}
#endif
```

消息格式变更

与信鸽版本对比，终端收到消息格式变更如下：

通知消息

变更前：

```
{
aps = {
alert = {
body = "通知内容";
subtitle = "通知副标题";
title = "通知标题";
};
badge = 1;
category = "iOS通知category";
"mutable-content" = 1; // 开启后，推送详情中会有「抵达」和「点击」数据上报
sound = "自定义通知音效.mp3";
};
custom1 = bar; // 自定义下发的参数string/JSON
custom2 = {
bang = whiz;
}; // 自定义下发的参数string/JSON
```

```
xg = {
bid = 0;
guid = 16625039711;
msgid = 2273893660;
token = c0999df7375868b9742de4b35387e49332b8c43b682482a7261278f1292eda95;
ts = 1585709566;
};
"xg_media_resources" = "富媒体链接如(https://img2.woyaogexing.com/2018/01/24/5248092370629ca4!400x400_big.jpg)";
}
```

变更后：

```
{
aps = {
alert = {
body = "通知内容";
"launch-image" = "";
"loc-args" = "<null>";
"loc-key" = "";
subtitle = "通知副标题";
"subtitle-loc-args" = "<null>";
"subtitle-loc-key" = "";
title = "通知标题";
"title-loc-args" = "<null>";
"title-loc-key" = "";
};
"badge_type" = "-1"; // 角标类型，-1不变，-2自动加一，也可以自定义为大于0的值
category = "iOS通知category";
"mutable-content" = 1; // 开启后，推送详情中会有「抵达」和「点击」数据上报
sound = "自定义通知音效.mp3";
"thread-id" = "";
};
custom = "{\"附加参数key1\":\"附加参数value1\",\"附加参数key2\":\"附加参数value2\"}"; // 可用于应用业务逻辑处理
xg = {
bid = 390178628;
groupId = "pt:tpns_20200401";
guid = 338502;
msgid = 390178628;
msgtype = 1;
pushTime = 1585716731;
source = 1;
targettype = 2;
ts = 1585716731;
```

```
xgToken = 00c30e0aeddff1270d8816ddc594606dc184;
};
"xg_media_resources" = "富媒体链接如(https://img2.woyaogexing.com/2018/01/24/5248092370629ca4!400x400_big.jpg)";
}
```

静默消息

变更前：

```
{
aps = {
content = "内容";
"content-available" = 1;
title = "标题";
};
custom1 = bar1; // 自定义下发的参数string/JSON
custom2 = (
bang,
whizfgg
); // 自定义下发的参数string/JSON
xg = {
bid = 0;
guid = 16625039711;
msgid = 2274434534;
token = c0999df7375868b9742de4b35387e49332b8c43b682482a7261278f1292eda95;
ts = 1585709604;
};
}
```

变更后：

```
{
aps = {
alert = "<null>";
"badge_type" = 0;
category = "";
"content-available" = 1;
sound = "";
"thread-id" = "";
};
custom = "{\"附加参数key1\":\"附加参数value1\",\"附加参数key2\":\"附加参数value2\"}"; // 可用于应用业务逻辑处理
xg = {
bid = 389886288;
```

```
groupId = "pt:tpns_20200331";
guid = 338502;
msgid = 389886288;
msgtype = 2;
pushTime = 1585644263;
source = 1;
targettype = 2;
ts = 1585644263;
xgToken = 00c30e0aeddff1270d8816ddc594606dc184;
};
}
```

抵达数据上报集成

为了实现抵达数据上报和富媒体消息的功能，SDK 提供了 Service Extension 接口，可供客户端调用，从而可以监听消息的到达和发送富媒体消息，需要您实现此接口，接入指南请参见 通知服务扩展的使用说明。

> 注意：
> 如果未集成此接口，则统计数据中消息『抵达数』与『点击数』一致。

测试

指定单设备推送

变更前：

信鸽平台使用 Device Token（长度为64位）对指定设备进行推送，获取 Device Token 代码示例:

```
//获取 APNS 生成的 Token
[[XGPushTokenManager defaultTokenManager] deviceTokenString];
```

变更后：
移动推送 TPNS 平台使用 TPNS Token(长度为36位)对指定设备进行推送，获取 TPNS Token 代码示例：

```
//获取 TPNS 生成的 Token
[[XGPushTokenManager defaultTokenManager] xgTokenString];
```

注销信鸽平台推送服务

注意：

- 如果未做以下配置，在【旧版信鸽】和【移动推送 TPNS 】两个平台上同时推送时，可能会出现重复消息。
- 如果您的应用确定不再使用【旧版信鸽】进行推送，可忽略上方配置。

如果 App 的推送服务是从 信鸽平台 迁移到移动推送 TPNS 平台， 需要调用 `TPNS SDK(1.2.5.3+)` 的接口将设备信息在信鸽平台中进行反注册。

接口

```
// 信鸽平台的 accessId(支持信鸽 SDK V2、V3版本)
@property uint32_t freeAccessId;
```

用法

- 引入头文件: `XGForFreeVersion.h`
- 在 `startXGWithAppID:appKey:delegate:` 之前调用此接口，参考示例：

```
[XGForFreeVersion defaultForFreeVersion].freeAccessId = 2200262432;
[[XGPush defaultManager] startXGWithAppID: <#your tpns access ID#>appKey:<#your tpns access key#> delegate:<#your delegate#>];
```

API 迁移指南

最近更新时间：2020-07-06 13:10:43

本文主要介绍了信鸽版本到移动推送 TPNS 版本的接口迁移说明，包括 V3 和 V2 的推送接口、账号接口和标签接口的差异。

请求域名地址变动说明

请求参数变动说明

协议字段	字段含义说明	变动说明
openapi.xg.qq.com	域名	请根据购买的集群选择对应的域名地址： 1. 广州集群：api.tpns.tencent.com 2. 中国香港集群：api.tpns.hk.tencent.com 3. 新加坡集群：api.tpns.sgp.tencent.com

V3 接口协议变动说明

移动推送 TPNS 版本 V3 接口协议格式 对比 信鸽版本V3接口协议格式基本相同。
协议中部分字段格式以及命名有变化，具体差异如下：

鉴权方式

信鸽版本使用 `AppId + SecretKey` 进行 `Basic Auth` 鉴权。(信鸽版本鉴权说明)

移动推送 TPNS 版本使用 `AccessId + SecretKey` 进行 `Basic Auth` 鉴权。(移动推送 TPNS 版本鉴权说明)

> 说明：
> **移动推送 TPNS 版本没有对应 AppId 字段，需要使用对应的应用 id `AccessId` 和密钥 `SecretKey` 进行鉴权。**

推送接口

移动推送 TPNS 版本推送接口协议 格式和 信鸽版本 基本相同, 主要区别如下:

请求参数变动说明

协议字段	字段含义说明	信鸽版	移动推送 TPNS 版
custom_content	Android 推送自定义参数	字段格式： json	字段格式：需要序列化为 json string
custom_content	iOS 推送自定义参数	字段格式： json	字段格式：需要序列化为 json string
push_id	账号列表推送和设备列表推送时，需要填写的推送任务ID	账号列表推送和设备列表推送时， 第一次推送该值填0，系统会创建 对应的推送任务，并且返回对应的 pushid：123，后续推送push_id填 123(同一个文案）表示使用与123 id 对应的文案进行推送	不再支持该字段对应功能

账号绑定接口

移动推送 TPNS 版本账号绑定协议格式 和 信鸽版本 完全相同，无需特别改动。

账号查询接口

移动推送 TPNS 版本账号查询协议格式 和 信鸽版本 基本相同，主要区别如下：

响应参数变动说明

协议字段	字段含义说明	变动说明
ret_code	操作返回码	字段名变更为 retCode
err_msg	操作响应消息	字段名变更为 errMsg

标签绑定接口

移动推送 TPNS 版本标签绑定协议格式 和 信鸽版本 基本相同，主要区别如下:

请求参数变动说明

协议字段	字段含义说明	信鸽版	移动推送 TPNS 版

协议字段	字段含义说明	信鸽版	移动推送 TPNS 版
tag_token_list	当进行标签和设备批量绑定/解绑时，提供需要绑定/解绑的标签设备列表，operator_type=9,10时必填	字段格式：[["tag1","token1"]，["tag2","token2"]]，每个对里面标签在前，token在后， 列表中每个元素为jsonArray	字段格式：[{"tag":"tag123", "token":"token123"}]，列表中每个原始为jsonObject

返回码

移动推送 TPNS 版本错误码是一套全新的返回码，和信鸽版本不同。
信鸽版本返回码定义参考：信鸽版本返回码
移动推送 TPNS 版本返回码定义参考：移动推送 TPNS 版本返回码

V2 接口协议变动说明

移动推送 TPNS 版本不再支持V2 协议接口

V2 版本对应的V3 版本接口参考如下：

V2接口	V2接口url	V3 接口	V3 接口url	接口定义说明
全量推送	/v2/push/all_device	推送接口	/v3/push/app	参考 推送接口文档
标签推送	/v2/push/tags_device	推送接口	/v3/push/app	参考 推送接口文档
账号群推	/v2/push/account_list	推送接口	/v3/push/app	参考 推送接口文档
设备单推	/v2/push/single_device	推送接口	/v3/push/app	参考 推送接口文档
账号单推	/v2/push/single_account	推送接口	/v3/push/app	参考 推送接口文档
超大批量账号推送	/v2/push/account_list_multiple	不支持，可使用号码包推送替代	-	-

V2接口	V2接口url	V3 接口	V3 接口url	接口定义说明
超大批量设备推送	v2/push/device_list_multiple	不支持，可使用号码包推送替代	-	-
批量新增标签	/v2/tags/batch_set	标签绑定接口	/v3/device/tag	参考 标签接口文档
批量删除标签	/v2/tags/batch_del	标签绑定接口	/v3/device/tag	参考 标签接口文档

迁移文档FAQ

最近更新时间：2020-07-06 13:10:49

信鸽平台（https://xg.qq.com）的数据可以迁移到移动推送 TPNS 平台吗？

信鸽平台和移动推送 TPNS 平台是两个独立的平台，数据互不相通，不支持数据迁移。

迁移移动推送 TPNS 版本初期，做全量推送是否需要在2个平台操作？

因信鸽平台和移动推送 TPNS 平台是两个独立的平台，当升级移动推送 TPNS 版后，若您的 App 没有强制升级策略，App 覆盖需要一定时间，在新版本 App 覆盖量不足时，做全量推送，需要在信鸽和移动推送 TPNS 两个平台上都操作一次，为了避免重复推送，请您按照以下方法配置：

1. 接入 Android V1.1.5.5 及以上版本，并 注销 Android 信鸽平台推送服务。
2. 接入 iOS V1.2.5.3 及以上版本，并 注销 iOS 信鸽平台推送服务。

如何预估日联网设备月峰值？

日联网设备月峰值，指单日连接移动推送 TPNS 服务器的去重终端设备量（即移动推送 TPNS 可触达的设备量）在一个月内的最高值，一般是在 DAU（每日 App 前台活跃设备数）的3到5倍或 MAU（每月 App 前台活跃设备数）的1/3这个范围内，可以按照这个范围进行预估，平台支持即时升级套餐，超量后可以在 控制台 上直接升级。

移动推送 TPNS 版本的服务端 SDK 支持哪些语言？

目前仅支持 Java SDK，其他语言的服务端 SDK 暂时未上线。

移动推送 TPNS 版本支持哪些厂商通道？

目前 Android 侧支持小米、华为、魅族、vivo、OPPO 等国内主流厂商通道集成，境外支持 Google FCM 通道；iOS 支持 APNs 通道。

移动推送 TPNS 本如何添加子账号？

可以对子账号/协作者进行授权，授权后子账号/协作者可协助您使用移动推送 TPNS 服务，授权方法如下：

1. 进入访问管理控制台的 策略 页面。

2. 目前移动推送 TPNS 提供2种权限角色，您可在策略页面，搜索关键字 TPNS 进行查找。

3. 找到需要授权的预设策略，单击右侧操作列的【关联用户/组】，如下图所示：

4. 在弹出的关联用户/用户组窗口，单击【切换用户组】>【用户】/【用户组】，如下图所示：

5. 勾选要关联的用户/用户组，单击【确定】，完成关联用户操作。

预设策略说明

预设策略	描述	权限

预设策略	描述	权限
QcloudTPNSFullAccess	移动推送 TPNS 全局读写访问权限 一般分配给推送管理员使用	• 查看数据统计 • 创建产品 • 编辑产品信息 • 查询产品列表 • 删除产品 • 创建应用 • 编辑应用信息 • 删除应用 • 查询应用列表 • 更新厂商通道信息 • 更新推送证书 • 创建推送 • 查看推送记录 • SDK 下载
QcloudTPNSReadOnlyAccess	移动推送 TPNS 只读访问权限 一般分配给运营人员使用	• 查看数据统计 • 查询产品列表 • 查询应用列表 • 创建推送 • 查看推送记录 • SDK 下载