通信协议

# 1 概述

## 背景

原生客户端需要与前置服务器之间进行各种数据交互，为了更好的完成数据的交互，定义一套通信双方都能够理解的接口协议是必不可少的。

## 1.2 目标

明确定义一套原生客户端与前置服务器之间的通信协议，作为客户端与前置服务器之间通讯唯一执行标准，双方严格按照本协议文档定义的接口来实现交互功能。为软件开发人员提供指导和参考。

# 2 设计原则

## 2.1 安全性

数据安全性是指对通讯的数据进行必要的安全保护措施，即使数据包被非法截获，也能够保证数据包不能够被识别；而合法的客户端能够安全的解开通讯数据包。

## 2.2 完整性

数据完整性是指在传输过程确保数据不被未授权的篡改，或者在篡改后能够被迅速发现。同时，定义的数据包能够完成业务功能所需要的各项数据。

## 2.3 扩展性

可扩展性是指定义的数据包接口能够很容易的进行业务数据的扩展，以满足将来的业务升级需要。

## 2.4 兼容性

兼容性是指定义的数据包内容能够向下兼容不同版本的业务。

# 3 通讯包封装

通讯包采用双层封装方式，外层包包含通用的数据信息，内层包指明具体的通信数据及其相关的安全控制策略信息，通讯包数据组装格式采用标准的JSON格式来封装。

通讯包封装格式如下：

Cookies

{

"SESSIONID" : "FSDFS231SFSDFS221"

}

Headers

{

"sessionState" : "100",

"Service-Number" : "123213131312132"

}

Response

// 外层数据包

{

"pkgFlag" : "0",

"errCode" : "0",

"errMsg" : "",

// 内层数据包

"dataPackage" : {

"cryptFlag" : "0",

"hashFlag" : "0",

"hash" : "hash值",

"signatureFlag" : "0",

"signature" : "签名值",

"business" : "{\"xxx\":\"yyyyy\", \”xxx1\”:\”yyyyy1\”}"

}

}

## 3.1 Cookies说明

Cookies包含一个字段，分别如下：

1. SESSIONID

该字段表示服务器的Cookies的sessionId

## 3.2 Headers说明

Headers包含一个字段，分别如下：

1. sessionState

该字段表示服务器返回数据 Headers 的 会话状态。

100：客户端的STATE\_HANDSHAKEFSUCCESS

101：客户端的STATE\_HANDSHAKEFAILED

102：客户端的STATE\_SESSIONTIMEOUT

其他: 客户端的STATE\_SESSIONTIMEOUT

1. Service-Number

该字段表示服务器返回的随机数

## 3.3 外层包说明

外层包Response包含四个字段，分别如下：

1. pkgFlag

该字段表示数据包的类型。

0：表示该数据包由客户端向前置服务器端发送的请求包

1：表示该数据包由前置服务器端向客户端反馈的响应包

1. errCode

该字段表示错误代码，在类型为0的数据包中，该错误码始终为0；在包类型为0的响应包中会有不同的定义，具体根据实际业务来定义。此处暂定为：

0：表示正常响应包

10000+：可选择的状态码

见 [5.4业务客户端-通信状态码]

1. errMsg(可选)

该字段指errCode对应的错误信息，当errCode为0时，该字段可以不存在

1. serviceNumber

该字段表示服务器返回的随机数

1. dataPackage

该字段表示整个内层包信息，具体说明参见3.4内层包说明

## 3.4 内层包说明

内层包包含六个字段，分别如下：

1. cryptFlag

该字段表示business字段的加密类型，其具体含义如下：

0：表示不加密，即明文方式传输

1：表示通过3DES算法加密

2：表示通过AES算法加密

3：表示通过RC5算法加密

4：表示通过默认的公钥加密

1. hashFlag

该字段表示business字段的散列算法类型，其具体含义如下：

0：表示不对business字段进行散列计算

1：MD5表示对business字段进行MD5散列计算

2：SHA1表示对business字段进行SHA1散列计算

1. signatureFlag

该字段表示business字段的签名算法，其具体含义如下：

0：表示不对business字段进行签名

1：表示对business字段进行MD5withRSA算法签名

2：表示对business字段进行SHA1withRSA算法签名

1. signature(可选)

该字段表示business字段的签名结果，不签名时该字段可以不存在

1. hash(可选)

该字段表示business字段的散列计算结果，不做散列计算时该字段可以不存在

1. business

真正的业务数据，明文情况下为完整的JSON格式的数据，并依据上面前四个字段定义的类型进行加工。

# 4 通信管理

## 4.1 密钥协商

### 4.1.1.客户端请求协议

{

"pkgFlag" : "0",

"errCode" : "0",

"errMsg" : "",

"dataPackage" : {

"cryptFlag" : "0",

"signatureFlag" : "0",

"hashFlag" : "0",

"business" :"SEagQcWihn6uuzbgv6hlq4tVuakog=="

}

}

business字段是明文通过公钥加密后，再转成base64得到的结果，其明文内容如下：

{

"cr" : "2F36720E9EC0477E874698242E29A71C ",

"shake-flag" : "0",

"user-guide" : "0",

"sys-version" : "1.3",

"sys-client" : "Android",

"sys-screenh" : "480",

"sys-screenw" : "320",

"sys-serial" : "DEVICEID|$|15|$|d30a47e83239fb1#$#MAC|$|12|$|353a34390000",

"ser-version" : "1.3",

"var-city" : ""

}

cr：表示客户端随机数

shake-flag：

user-guide：

sys-version：系统版本号

sys-client：客户端类型(Android,iPhone,iPad等)

sys-screenh：客户端设备屏幕高度(像素)

sys-screenw：客户端设备屏幕宽度(像素)

sys-serial：设备序列号

ser-version：

var-city：终端当前所在城市

### 4.1.2服务器响应协议

Cookies

{

"SESSIONID" : "FSDFS231SFSDFS221"

}

Headers

{

"sessionState" : "100",

"Service-Number" : "123213131312132"

}

Response

{

"pkgFlag" : 0,

"errCode" : 0,

"errMsg" : 0,

"dataPackage" : {

"cryptFlag" : 0,

"hashFlag" : 2,

"signatureFlag" : 0,

"hash" : "vyGp6PvFo4RvsFtPoIWeCReyIC8=",

"business" : "{}"

}

}

**客户端协商密钥操作：**

1. **服务器SessionId**

Cookies 的SESSIONID。

1. **服务器随机数**

Headers 的 Service-Number

1. **组装 对称密钥**

结合 客户端随机数 组装对称密钥。

## 4.2 握手会话状态管理

### 4.2.1 客户端握手会话状态标识

STATE\_NULLSESSION = 0, //!< 未建立握手会话

STATE\_HANDSHAKEING = 1, //!< 正在握手会话

STATE\_HANDSHAKEFAILED = 2, //!< 握手会话失败

STATE\_SESSIONTIMEOUT = 3, //!< 握手会话超时，需要重新握手会话

STATE\_HANDSHAKEFSUCCESS = 4, //!< 握手会话成功

### 4.2.2 服务器端响应协议及会话状态标识

1、响应报文的headers ：

{

"sessionState" : 100

}

2、服务器会话状态标识

服务器 100 = 客户端的STATE\_HANDSHAKEFSUCCESS

服务器 101 = 客户端的STATE\_HANDSHAKEFAILED

服务器 102 = 客户端的STATE\_SESSIONTIMEOUT

服务器 其他= 客户端的STATE\_SESSIONTIMEOUT

## 4.3 Hash验证

Hash验证是指针对指定的文件计算hash值，并将计算后的hash值发送至前置服务器，最后由前置服务器返回比对结果。

此处的验证协议适合已有的文件列表验证和菜单文件验证，其他类型的hash验证可以通过适当修改business字段即可实现。

### 4.3.1 客户端发送数据报文

{

"pkgFlag" : "0",

"errCode" : "0",

"errMsg" : "",

"dataPackage" : {

"cryptFlag" : "1",

"hashFlag" : "0",

"signatureFlag" : "0",

"business" : "LQTOVu7/rf3KNnODHevrzIpGQ3ab9uvLQ/AbBQxGxO5mHvxS447p+6R9MxIcLX+2ou1Gv9nsmhyv7hq5SFXdjn4qCiwY0LMbVS41dc00y7"

}

}

business字段明文内容如下：

{

"type" : "MD5",

"url" : "menu.xml",

"hash" : "2F36720E9EC0477E874698242E29A71C"

}

type:验证的hash算法

url：表示待验证的文件的相对路径

hash：表示待验证的文件根据type提供的算法计算的hash值

### 4.3.2 服务器返回报文

{

"pkgFlag" : "0",

"errCode" : "0",

"errMsg" : "",

"dataPackage" : {

"cryptFlag" : "1",

"hashFlag" : "0",

"signatureFlag" : "0",

"business" : "LQTOVu7/rf3KNnODHevrzIpGQ3ab9uvLQ/AbBQxGxO5mHvxS447p+6R9MxIcLX+2ou1Gv9nsmhyv7hq5SFXdjn4qCiwY0LMbVS41dc00y7"

}

}

business字段明文内容如下：

{

"result" : "1",

"hash" : "2F36720E9EC0477E874698242E29A71C"

}

result字段说明：

0：表示维持原状

1：表示需要更新对应的文件

hash字段说明：表示待更新文件在服务器上计算出的hash值，当无需更新时，可以不存在。

## 4.4 普通交易请求

无

# 5业务客户端-通信状态码

提供给业务客户端的通信状态码,统一由 PTComPackage 的 errCode 和errMsg 提供。

## 5.1框架自定义状态码

CommunicationSuccess = 0, //!< 连接成功

SessionStateError = 1, //!< 会话状态错误

NetworkUnreachableError = 2, //!< 网络不通

ExceptionError = 3, //!< 未知的异常错误.

## 5.2网络请求状态码

500

404等

## 5.3外层包状态码

1、如果网络请求正确，使用外层数据包的 errCode 、 errMsg

0：请求结果正确

其它：请求结果其它错误码

## 5.4业务客户端状态码汇总

PTComPackage 的外层数据包 errCode :

1、 前置框架定义的状态码0：请求成功

2、 框架状态码1：会话状态错误

3、 框架状态码2：网络不通

4、 框架状态码3：未知的异常错误

5、 服务器状态码500：

6、 服务器错误404：

7、 前置框架自定义状态码 10001： 自定义状态码1

8、 前置框架自定义状态码 10002： 自定义状态码2

# 6 Ionic JS端与原生通信对外层通信包的扩展

## 6.1 新增 ptSessionState 字段

### 6.1.1 字段功能及扩展原因

字段功能:JS段保存会话状态标识.

扩展原因:由于ionic开发的应用是web主体型应用, JS端是完整的app应用,需要JS端保存会话状态标识.

### 6.1.2 添加ptSessionState时机

在使用JS桥通信时，由原生客户端接收到前置外层数据包，并解密获取明文的外层数据包之后，返回给JS桥通信结果之前

### 6.1.3 扩展的外层数据包格式

// 外层数据包

{

"pkgFlag" : "0",

"errCode" : "0",

"errMsg" : "",

// 内层数据包

"dataPackage" : {

"cryptFlag" : "0",

"hashFlag" : "0",

"hash" : "hash值",

"signatureFlag" : "0",

"signature" : "签名值",

"business" : "{\"xxx\":\"yyyyy\", \”xxx1\”:\”yyyyy1\”}"

},

"ptSessionState" : "4"

}