上海迅时通信设备有限公司

P2P约会服务SDK使用指南 v3



网址：www.newrocktech.com

论坛：bbs.newrocktech.com

电话：021-61202700

传真：021-61202704

修改记录

文档版本 03 (2016-05-28 )

修改：p2p\_init函数增加log\_type参数。

新增：约会服务消息处理新接口，在P2P约会服务SDK v1.1.1版本及以上版本开始废弃原来的约会服务消息处理接口，使用新接口代替。

新增：获取设备ID等信息接口。

文档版本 02 (2016-03-28 )

修改描述。

新增注册、连接失败错误码查询。

新增加密、调试模式设置。

文档版本 01 (2016-03-08 起草人：zywan, hbchen)

全文。

文档说明

本文适用于迅时P2P约会服务功能开发系列指导手册。

目录

[一、介绍 5](#_Toc452479653)

[1.1 P2P约会服务 5](#_Toc452479654)

[1.2 P2P约会服务SDK 5](#_Toc452479655)

[1.3 术语 5](#_Toc452479656)

[二、P2P约会服务API接口 6](#_Toc452479657)

[2.1 新建约会服务端点 6](#_Toc452479658)

[2.2端点注册 6](#_Toc452479659)

[2.3 端点释放 7](#_Toc452479660)

[2.4 端点信息查询 7](#_Toc452479661)

[2.5 端点注册失败错误码查询 8](#_Toc452479662)

[2.6 新建约会连接 9](#_Toc452479663)

[2.7连接释放 9](#_Toc452479664)

[2.8查询连接信息 10](#_Toc452479665)

[2.9 连接失败错误码查询 10](#_Toc452479666)

[2.10解析数据包 11](#_Toc452479667)

[2.11约会服务消息处理 12](#_Toc452479668)

[2.12约会服务消息处理(新接口) 12](#_Toc452479669)

[2.13状态机处理 13](#_Toc452479670)

[2.14初始化 13](#_Toc452479671)

[2.15设置模式 14](#_Toc452479672)

[2.16获取设备ID和KEY 14](#_Toc452479673)

[三、P2P约会服务SDK使用说明 15](#_Toc452479674)

[3.1 SDK组成 15](#_Toc452479675)

[3.2 demo示例 15](#_Toc452479676)

## 一、介绍

### 1.1 P2P约会服务

在现实的复杂网络环境中，由于NAT、防火墙、私有专用网等原因，导致很多点对点应用端点之间无法建立直接的网络通信，使得企业在融合通信应用中遇到很多连接问题而无法使用，如喂喂移动分机、侬好、远程获取录音等。

通过对NAT行为的分析和测试，我们发现大多数的NAT或防火墙，可以通过在云端部署一套公有的约会服务器(Rendezvous)，通过hole punching来协助实现这种网络环境下的P2P连接问题，我们称这种通过约会服务器来协助实现P2P连接服务的过程为P2P约会服务。

当前P2P约会服务仅适用于UDP协议的通信应用。

### 1.2 P2P约会服务SDK

P2P约会服务SDK是为了方便开发者迅速开发使用P2P约会服务而由迅时开发提供的，SDK封装了P2P约会服务的消息解析和封装，P2P约会服务的通信协议和交互处理，利用SDK提供的API调用接口可以快速开发出支持迅时P2P约会服务功能的端点应用。

### 1.3 术语

本文中使用的一些名词术语约定。

约会服务器（Rendezvous）：指迅时提供的部署在云平台上，为P2P建立连接的端点提供连接协助服务的服务器。

端点（Endpoint）：指要建立点对点通信的应用程序。

P2P Server端点：指主动提供应用服务的应用程序，允许其它端点向我发起连接请求。

P2P Client端点：指使用别的服务端点提供的服务，主动向服务端点发起连接请求。

## 二、P2P约会服务API接口

### 2.1 新建约会服务端点

REND\_EPT\_HANDLE new\_rendezvous\_endpoint(char \*cid, char \*service, char \*oid, char \*ic, char \*key, int udp\_fd);

创建一个新的约会服务端点，由需要使用约会服务的应用程序在其使用的UDP端口上调用创建。

首次没有cid和key需要约会服务器下发的，需要发送空注册请求，调用此接口时cid和key为空即可。

参数说明：

|  |  |  |
| --- | --- | --- |
| 输入参数 | 参数类型 | 说明 |
| cid | char \* | 端点id，为NULL时约会服务器下发id |
| service | char \* | 服务类型，端点提供的服务类型 |
| oid | char \* | 服务对象id，当同一个服务有多个实例时需要提供实例对象id |
| ic | char \* | 邀请码，用于安全，拥有邀请码的端点才可以使用我提供的服务 |
| key | char \* | 密钥，用于端点和约会服务器之间的数据通信加密 |
| udp\_fd | int | 使用约会服务的UDP插口 |
|  |  |  |
|  |  |  |

返回说明：

|  |  |
| --- | --- |
| 类型 | 说明 |
| REND\_EPT\_HANDLE | 新建端点标识 |

### 2.2端点注册

int rendezvous\_endpoint\_reg(REND\_EPT\_HANDLE endpoint);

端点向约会服务器发起注册。

参数说明：

|  |  |  |
| --- | --- | --- |
| 输入参数 | 类型 | 说明 |
| endpoint | REND\_EPT\_HANDLE | 端点标识 |

返回说明：

|  |  |
| --- | --- |
| 类型 | 说明 |
| int | 0 ：注册请求成功  其它：注册请求失败 |

### 2.3 端点释放

void free\_rendezvous\_endpoint(REND\_EPT\_HANDLE endpoint);

释放创建的端点占用的资源，在端点不再使用或udp端口关闭后可释放端点，端点释放后不可用。

参数说明：

|  |  |  |
| --- | --- | --- |
| 输入 | 类型 | 说明 |
| endpoint | REND\_EPT\_HANDLE | 端点标识 |

### 2.4 端点信息查询

int get\_rendezvous\_endpoint(REND\_EPT\_HANDLE endpoint, int \*status, char \*cid, char \*ed, char \*ped);

查询端点信息和状态。

参数说明：

|  |  |  |
| --- | --- | --- |
| 输入 | 类型 | 说明 |
| endpoint | REND\_EPT\_HANDLE | 端点标识 |
| status | int \* | 返回查询端点状态  ENDPOINT\_INITIAL,  ENDPOINT\_REGISTERING  ENDPOINT\_REGISTER\_FAIL  ENDPOINT\_REGISTER\_OK |
| cid | char \* | 返回查询端点id |
| ed | char \* | 返回查询端点内网IP:PORT |
| ped | char \* | 返回查询端点公网IP:PORT |

返回说明：

|  |  |
| --- | --- |
| 类型 | 说明 |
| int | 0 ：获取信息成功  其它：客户端不存在或获取失败 |

### 2.5 端点注册失败错误码查询

int get\_rendezvous\_endpoint\_error(REND\_EPT\_HANDLE endpoint, int \*status, int \*error\_code);

注册失败时查询端点的注册状态及错误码。

在应用端点的cid和key是由约会服务器下发的情况下，如果端点保存了cid和key，并且长时间没有注册使用(如应用退出)，端点下次注册使用之前保存的cid和key，则可能cid和key在约会服务器侧已失效而导致约会服务器回应403拒绝，此时需要应用端点重新发起空注册请求，重新下发cid和key。

参数说明：

|  |  |  |
| --- | --- | --- |
| 输入 | 类型 | 说明 |
| endpoint | REND\_EPT\_HANDLE | 端点标识 |
| status | int \* | 返回查询端点状态  ENDPOINT\_INITIAL,  ENDPOINT\_REGISTERING  ENDPOINT\_REGISTER\_FAIL  ENDPOINT\_REGISTER\_OK |
| error\_code | int \* | 返回查询端点错误码  400消息格式不对或缺少参数  403加解密失败或cid不存在 |

返回说明：

|  |  |
| --- | --- |
| 类型 | 说明 |
| int | 0 ：获取状态信息成功  其它：客户端不存在或获取失败 |

### 2.6 新建约会连接

REND\_CONN\_HANDLE new\_rendezvous\_connection(REND\_EPT\_HANDLE my\_endpoint, char \*tid, char \*service, char \*toid, char \*tic);

新建约会连接。

参数说明：

|  |  |  |
| --- | --- | --- |
| 输入 | 类型 | 说明 |
| my\_endpoint | REND\_EPT\_HANDLE | 端点标识 |
| tid | char \* | 要连接的对方端点id |
| service | char \* | 要连接的对方端点服务 |
| toid | char \* | 要连接的对方端点服务对象id |
| tic | char \* | 要连接的对方端点的邀请码 |

返回说明：

|  |  |
| --- | --- |
| 类型 | 说明 |
| REND\_CONN\_HANDLE | 新建连接标识 |

### 2.7连接释放

void free\_rendezvous\_connection(REND\_CONN\_HANDLE conn);

连接建立成功后，拿走连接通道信息后可将它释放，以免占用内存。

注意：连接成功或失败后60秒会自动释放。请及时获取连接信息。

参数说明：

|  |  |  |
| --- | --- | --- |
| 输入 | 类型 | 说明 |
| conn | REND\_CONN\_HANDLE | 连接标识 |

### 2.8查询连接信息

int get\_rendezvous\_connection(REND\_CONN\_HANDLE conn, int \*status, char \*r\_ed, char \*r\_ped);

查询连接信息和状态。

参数说明：

|  |  |  |
| --- | --- | --- |
| 输入 | 类型 | 说明 |
| conn | REND\_CONN\_HANDLE | 连接标识 |
| status | int \* | 连接状态  CONNECTION\_INIT,  CONNECTION\_REQUEST,  CONNECTION\_PING,  CONNECTION\_FAILED,  CONNECTION\_OK |
| r\_ed | char \* | 对方内网IP:PORT，不通时为空串 |
| r\_ped | char \* | 对方公网IP:PORT，不通时为空串 |

返回说明：

|  |  |
| --- | --- |
| 类型 | 说明 |
| int | 0 ：获取信息成功  1 ：获取失败 |

### 2.9 连接失败错误码查询

int get\_rendezvous\_connection\_error(REND\_CONN\_HANDLE conn, int \*status, int \*error\_code);

查询连接的状态及状态码，连接失败时状态码有意义。

参数说明：

|  |  |  |
| --- | --- | --- |
| 输入 | 类型 | 说明 |
| conn | REND\_CONN\_HANDLE | 连接标识 |
| status | int \* | 返回查询连接状态  ENDPOINT\_INITIAL,  ENDPOINT\_REGISTERING  ENDPOINT\_REGISTER\_FAIL  ENDPOINT\_REGISTER\_OK |
| error\_code | int \* | 返回查询连接错误码  400消息格式不对或未注册  403连接拒绝，邀请码不对  408 连接超时 |

返回说明：

|  |  |
| --- | --- |
| 类型 | 说明 |
| int | 0 ：获取信息成功  其它：客户端不存在或获取失败 |

### 2.10解析数据包

int decode\_rendezvous\_packet(char \*udp\_message, struct sockaddr\_in \*peer\_addr, rendezvous\_packet\_t \*packet);

解析UDP数据包，判断是否为约会服务消息数据。

参数说明：

|  |  |  |
| --- | --- | --- |
| 输入 | 类型 | 说明 |
| udp\_message | char \* | udp消息内容 |
| peer\_addr | struct sockaddr\_in \* | 收到udp消息的对方地址 |
| packet | rendezvous\_packet\_t \* | 存放解析后的协议报文 |

返回说明：

|  |  |
| --- | --- |
| 类型 | 说明 |
| int | 0 ：是约会服务消息数据  其它：不是 |

### 2.11约会服务消息处理

int rendezvous\_message\_handle(int udp\_fd, rendezvous\_packet\_t \*packet);

处理收到的约会服务消息数据，更新端点或连接的状态。

注意：此接口在P2P约会服务SDK v1.1.1版本及以上版本中将不再支持使用。

参数说明：

|  |  |  |
| --- | --- | --- |
| 输入 | 类型 | 说明 |
| udp\_fd | int | 收到数据包的udp插口 |
| packet | rendezvous\_packet\_t \* | 约会服务消息数据包 |

返回说明：

|  |  |
| --- | --- |
| 类型 | 说明 |
| int | 0 ：处理成功  其它：处理失败 |

### 2.12约会服务消息处理(新接口)

int handle\_rendezvous\_packet(int udp\_fd, char \*udp\_message, struct sockaddr\_in \*peer\_addr);

处理收到UDP消息，判断是否P2P消息，是的话则解析处理并更新端点或连接的状态。

注意：此接口在v1.1.1版本及以上版本中开始支持，用于替代上面的旧接口。

参数说明：

|  |  |  |
| --- | --- | --- |
| 输入 | 类型 | 说明 |
| udp\_fd | int | 收到数据包的udp插口 |
| udp\_message | char \* | udp消息内容 |
| peer\_addr | struct sockaddr\_in \* | 收到udp消息的对端地址 |

返回说明：

|  |  |
| --- | --- |
| 类型 | 说明 |
| int | 0 ：是约会服务消息数据  其它：不是 |

### 2.13状态机处理

int rendezvous\_status\_handle();

用于维护和更新端点和连接的状态，需要周期调用。

注意：此函数必须保证周期调用，建议周期为500毫秒~1秒，否则P2P连接很慢或不可用。

返回说明：

|  |  |
| --- | --- |
| 类型 | 说明 |
| int | 0 ：成功  其它：处理失败 |

### 2.14初始化

int p2p\_init(char \*log\_path, char \*app\_name, P2P\_LOG\_TYPE log\_type, int log\_level, char \*server, int server\_port);

在使用p2p约会服务之前调用，用于初始化一些必要参数。

注意：需要判断初始化是否成功，如果初始化失败则不要使用P2P约会功能。

参数说明：

|  |  |  |
| --- | --- | --- |
| 输入 | 类型 | 说明 |
| log\_path | char \* | 日志文件目录， |
| app\_name | char \* | 应用名，决定日志文件名 |
| log\_type | P2P\_LOG\_TYPE | 日志类型(日志输出到文件或屏幕)：  P2P\_LOG\_TYPE\_FILE  P2P\_LOG\_TYPE\_SCREEN |
| log\_level | int | 日志等级，越大日志越多 |
| server | char \* | 约会服务器地址，可以是域名，为空则是默认 |
| server\_port | int | 约会服务器端口，为0时则使用默认的7042 |

返回说明：

|  |  |
| --- | --- |
| 类型 | 说明 |
| int | 0 ：成功  其它：失败 |

### 2.15设置模式

void set\_p2p\_option(int crypt, int debug);

用于运行模式及是否加密，生产环境下不要开启debug模式。

默认使用加密和关闭调试。

参数说明：

|  |  |  |
| --- | --- | --- |
| 输入 | 类型 | 说明 |
| crypt | int | 0：不加密  1：消息加密 |
| debug | int | 0: 关闭调试  1: 调试模式 |

### 2.16获取设备ID和KEY

int get\_deviceId\_key(char \*id, char \*key);

获取设备ID和KEY，设备上需要安装运行云平台组件程序，如果设备上没有云组件或云组件程序(pluginmgr)没有运行，则会获取失败。

参数说明：

|  |  |  |
| --- | --- | --- |
| 输入 | 类型 | 说明 |
| id | char \* | 用于保存获取到的设备ID的值 |
| key | char \* | 用于保存获取到的设备KEY的值 |

返回说明：

|  |  |
| --- | --- |
| 类型 | 说明 |
| int | 0 ：获取成功  -1：获取失败 |

## 三、P2P约会服务SDK使用说明

### 3.1 SDK组成

P2P约会服务SDK主要包含下列文件：

lib目录——各平台下的库文件：

lib/wroc/libp2pnat.so

lib/x86/libp2pnat.so

…

include目录——头文件：

p2p\_api.h

使用SDK时只需要包含p2p\_api.h即可。

demo目录——DEMO示例：

build.sh

comm.c

Makefile

p2pdemo.c

当前SDK仅包含C语言版本，适用于linux系统和嵌入式linux系统。

### 3.2 demo示例

Demo示例请参考p2pdemo.c文件，其包含了P2P约会服务SDK使用的代码参考示例。

编译：

./build.sh x86|wroc|mx60|mx8|om400

运行：

./p2pdemo –d –l 5 下发并注册端点

./p2pdemo –d –l 5 -c peer\_a 下发并注册端点后尝试连接，peer\_a改为要连接端点的cid