

**深信服EasyAPP方案SDK包使用说明**

目录

[1 概述 4](#_Toc355961567)

[2 SDK简介 4](#_Toc355961568)

[2.1 SDK的功能模块 4](#_Toc355961569)

[2.2 SDK支持的环境 4](#_Toc355961570)

[2.3 使用SDK的流程 5](#_Toc355961571)

[3 配置VPN网关设备 5](#_Toc355961572)

[3.1 配置应用资源 5](#_Toc355961573)

[3.1.1 配置用户角色 7](#_Toc355961574)

[3.2 配置内网DNS服务 9](#_Toc355961575)

[3.2.1 域名解析流程如图 9](#_Toc355961576)

[3.2.2 配置HOSTS规则 9](#_Toc355961577)

[3.2.3 配置内网域名解析规则 11](#_Toc355961578)

[3.2.1 内网域名解析服务器 12](#_Toc355961579)

[3.2.2 IOS平台特别说明 13](#_Toc355961580)

[4 自挂钩SDK使用介绍 13](#_Toc355961581)

[4.1 接口介绍 13](#_Toc355961582)

[4.2 Android平台SDK包 13](#_Toc355961583)

[4.2.1 Android开发包组成 13](#_Toc355961584)

[4.2.2 Android开发包的使用 13](#_Toc355961585)

[4.3 Android开发包Demo使用示例 14](#_Toc355961586)

[4.4 IOS 平台SDK包 16](#_Toc355961587)

[4.4.1 IOS开发包介绍 16](#_Toc355961588)

[4.4.2 IOS开发包使用步骤 16](#_Toc355961589)

[4.5 IOS开发包Demo使用示例 20](#_Toc355961590)

[4.6 自动注销和断线自动重连说明 23](#_Toc355961591)

[5 iOS&Android平台SDK接口介绍 23](#_Toc355961592)

[5.1 Android平台接口 23](#_Toc355961593)

[5.1.1 Android平台阻塞接口 23](#_Toc355961594)

[5.1.2 4.1.2 Android平台非阻塞接口 27](#_Toc355961595)

[5.2 4.2 iOS平台接口 31](#_Toc355961596)

[5.2.1 4.2.1 iOS平台阻塞接口 31](#_Toc355961597)

[5.2.2 4.2.2 iOS平台非阻塞接口 36](#_Toc355961598)

# 概述

深信服EasyApp方案提供的SDK包是一套供第三方移动应用程序（以下简称APP）调用，以快速实现与深信服SSL VPN 网关创建和销毁VPN加密隧道，完成VPN认证登录和细粒度授权控制的第三方接口套件。

该套件通过对APP自身进行Hook操作，改变APP原有的网络操作行为，业务系统APP应用程序开发人员只需要在APP主程序中加载启动SDK套件即可实现VPN功能，简单易用。

# SDK简介

## SDK的功能模块

本方案提供的IOS&Android平台APP直连 SDK，主要包括三个模块：Hook模块、认证模块、转发数据模块、本地数据防泄密模块。

* **Hook模块** 功能是对网络connect和DNS域名解析操作进行拦截，并进行网络数据重定向，达到自挂钩的效果。
* **认证模块** 完成应用程序的用户认证，其中可能存在多种组合认证的方式，此组合情况依赖于VPN服务端配置，目前SDK只支持用户名密码认证、证书认证和硬件特征码认证。
* **转发模块** 完成数据的转发功能，将客户应用程序数据转发给VPN服务端，或者将VPN服务端下发数据转发给客户应用程序。
* **本地数据防泄密模块** 默认没有开启本地数所防泄密功能。当开启本地防泄密功能以后，此功能可以把应用程序本地存储的文件（包括sqlite3数据库、本地配置文件）加密存储在手机上。以防手机丢失后，手机保存在本地的数据泄露。

## SDK支持的环境

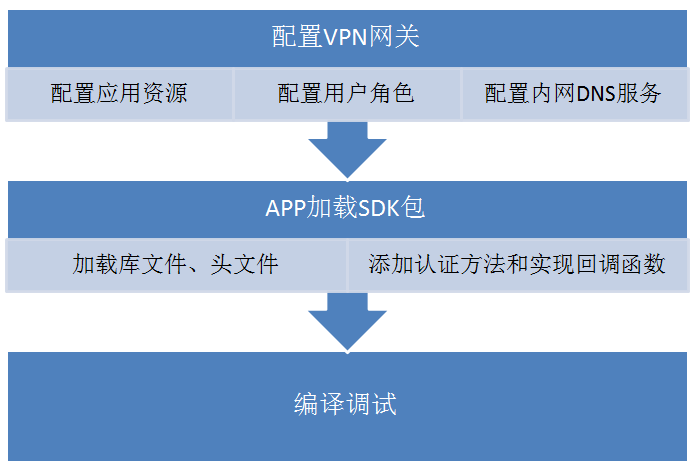
本方案支持Google Android操作系统

版本要求：2.0~4.2

本方案支持苹果IOS操作系统

版本要求：armv6/armv7/armv7s

## 使用SDK的流程



# 配置VPN网关设备

## 配置应用资源

本SDK方案只支持配置为TCP资源实现通过VPN隧道加密的目的，因此当解析到访问的资源IP地址和端口为TCP资源时，才完成自挂钩HOOK操作；对于非配置的IP地址和端口，数据不被HOOK操作，还是按应用原有网络行为操作。因此，在正式使用SDK包前，需要配置好相应的TCP资源，为正式编写代码做好准备。

* **1、确定资源的内网IP地址和端口。**例如，本案例要连接的服务器IP地址是200.200.73.65，端口号为4420；
* **2、新建TCP资源**：SSL VPN设置 > 资源管理 > 新建 > TCP应用。

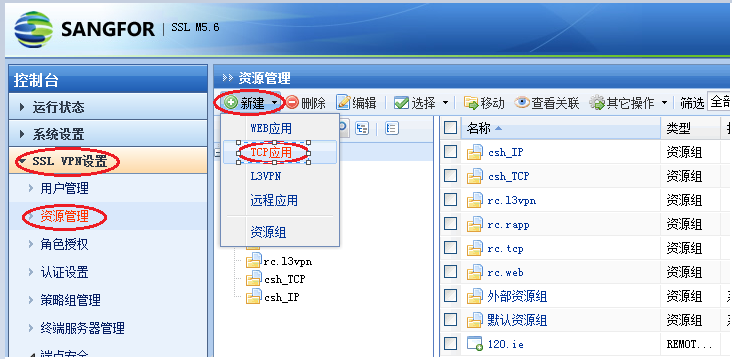


图3.1 新建资源步骤

* **3、填写资源信息**：必须填写“名称、类型、地址”信息。例如：在截图中取资源名为sangforsdk，输入IP地址为200.200.73.65，起始端口和终止端口均为4420；
  + 注：资源地址配置可以选择IP段，即只要我们访问的资源IP属于IP段内，SDK都会解析为TCP资源。

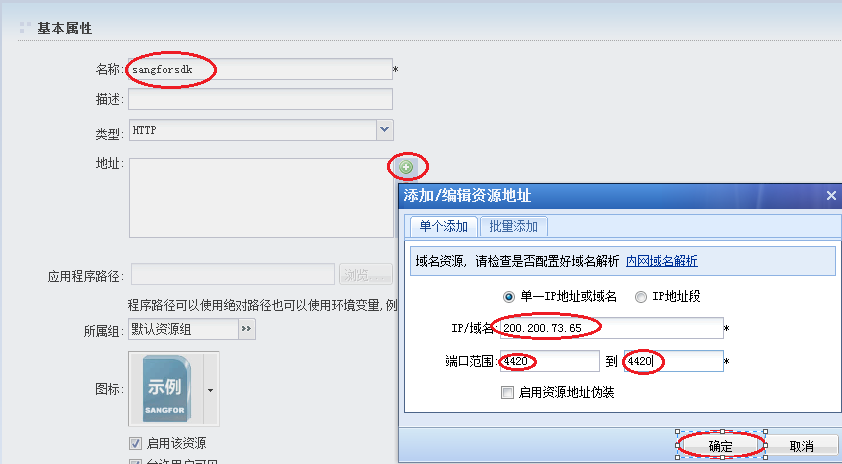


图3.2 输入资源IP地址和端口

* **4、保存配置并选择立即生效**：
  + 注意：为避免编码过程中遗忘，请每次修改配置后马上点击“立即生效”。

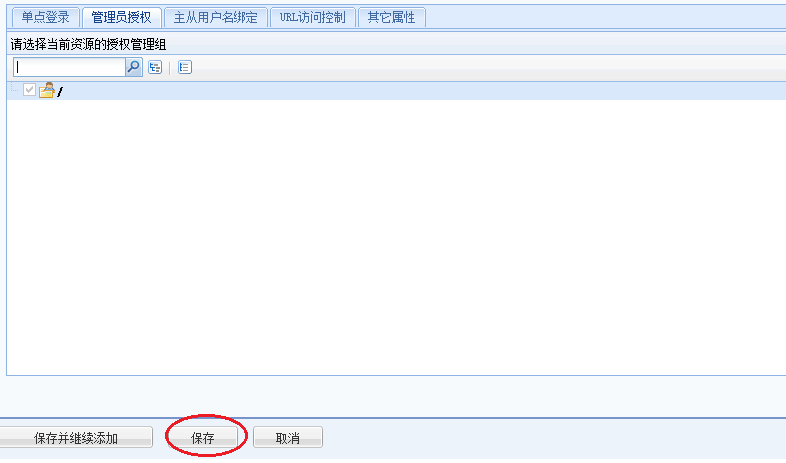


图 3.3 点击保存

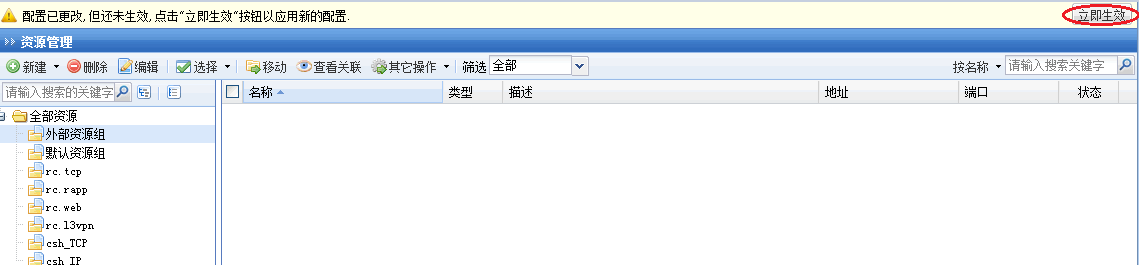


图 3.4 点击立即生效

### 配置用户角色

由于VPN定义中，必须有用户概念，因此资源配置完成后，必须要关联一个角色，VPN才会下发与用户相关联的资源信息和列表信息。

* **1、新建或找到现有角色**：SSL VPN设置> 角色授权。例如：在截图中的qjxRole。

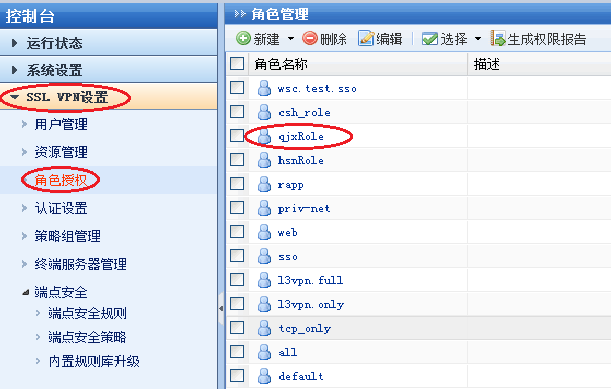


图3.5 选择角色

* **2、将资源与角色关联**：编辑授权资源 > 全部资源 > sangforsdk > 确定。

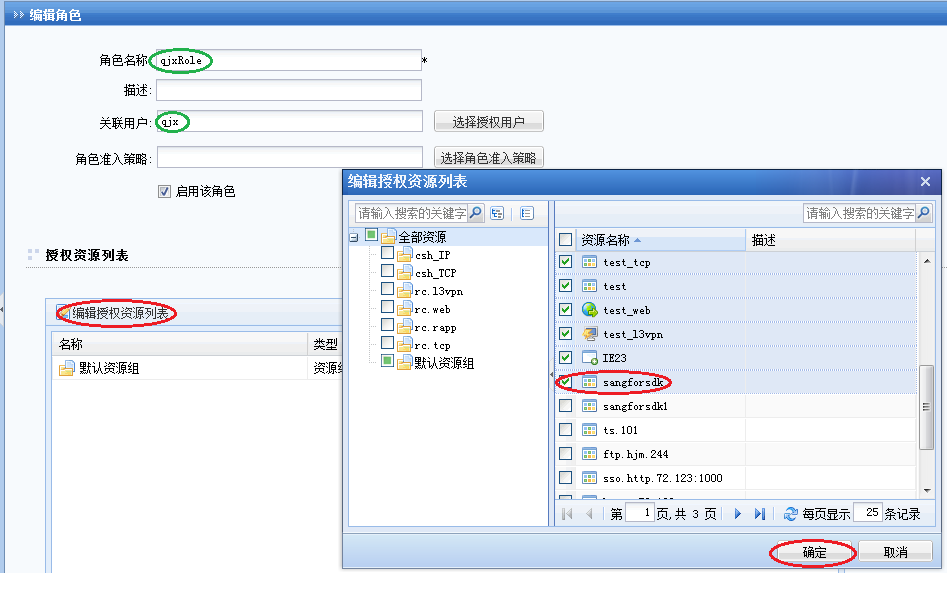


图3.6 将资源与角色关联

* 3、**保存配置并选择立即生效**：
  + 注意：为避免编码过程中遗忘，请每次修改配置后马上点击“立即生效”。

## 配置内网DNS服务

### 域名解析流程如图

应用中的设定的域名需要解析的简图如下，为保障域名能够正常解析，需要配置主机HOST以及内网域名解析服务器。



### 配置HOSTS规则

HOSTS规则，指的是需要将某一域名（主机）映射唯一的IP地址，该规则通过在VPN网关设备上配置，并随配置文件下发给本SDK方案。例如：如下图所示，主机名localhost对应于127.0.0.1的IP地址，主机名bbs对应于200.200.0.20的IP地址等。

* **1、进入网络配置🡪HOST配置选项**
* **2、添加主机映射**
  + 注意：域名（主机）映射表下发至本SDK方案的前提是该域名必须在资源列表中，以主机名（或称域名）新建资源，如图2.10所示，服务端才会给本SDK方案下发主机IP地址与域名配对信息。

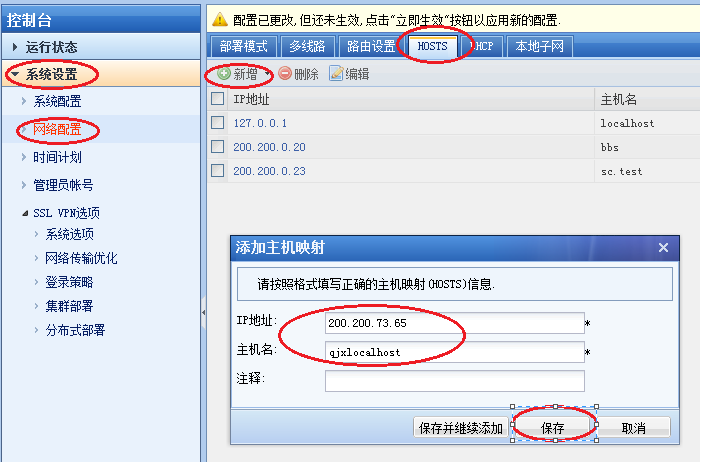


图3.9 内网主机IP地址与域名配对

* **3、以主机名（域名）新建资源:**请参考3.1，必须新建为TCP资源。

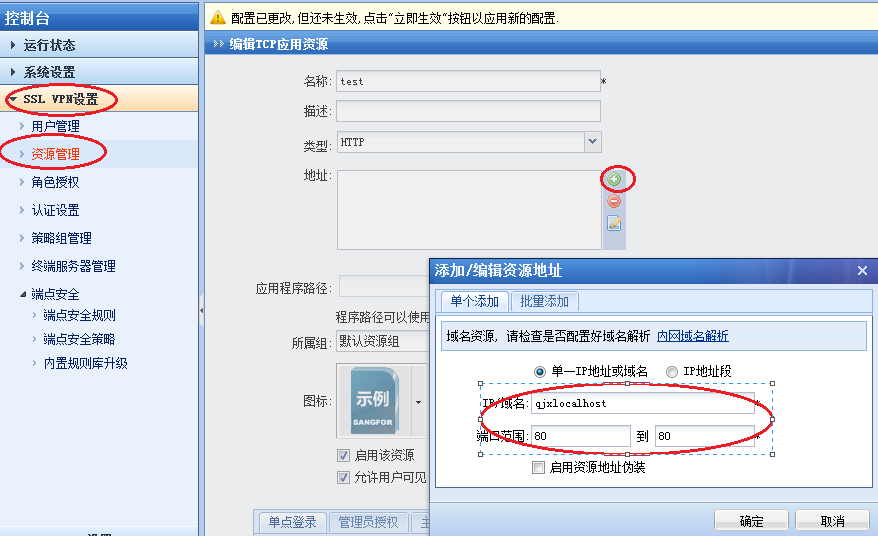


图 3.10 以主机名（域名）新建资源

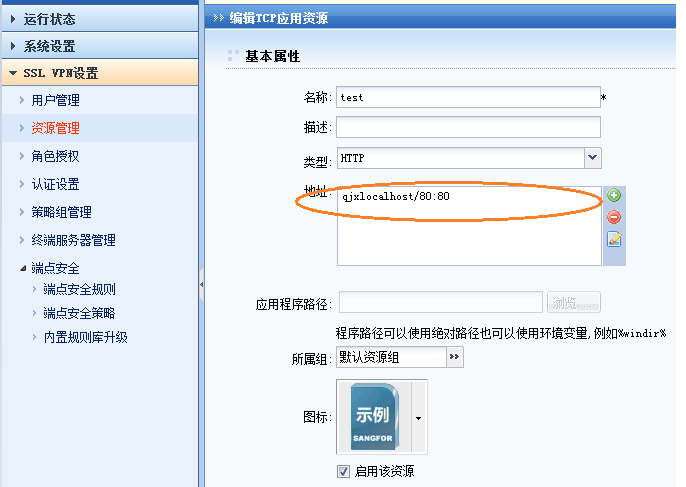


图3.11 以主机名（域名）新建资源后截图

* 4、**保存配置并选择立即生效**：
  + 注意：为避免编码过程中遗忘，请每次修改配置后马上点击“立即生效”。

### 配置内网域名解析规则

内网域名解析指的是，需要设置某些域名采用内网DNS进行解析。根据该配置规则，只有当要解析的域名满足配置规则或通配符的字符串的时候，才开启内网域名解析，否则转入外网域名解析流程。例如：应用需要解析[www.sangfor.com](http://www.sangfor.com) 和 [www.baidu.com](http://www.baidu.com)两个地址，如下图配置了\*.sangfor；则www.sangfor.com 的解析会交给内网DNS。而www.baidu.com没有配置，则会交给外网DNS 如8.8.8.8。

* **1、进入“内网域名解析”选型**
* **2、取消“接入计算机使用此DNS服务作为首选DNS服务器”选型**
* 注意：该选型的复选框必须取消，否则VPN服务器不会下发该规则至应用。
* **3、新增域名解析规则**
* 4、**保存配置并选择立即生效**：
  + 注意：为避免编码过程中遗忘，请每次修改配置后马上点击“立即生效”。

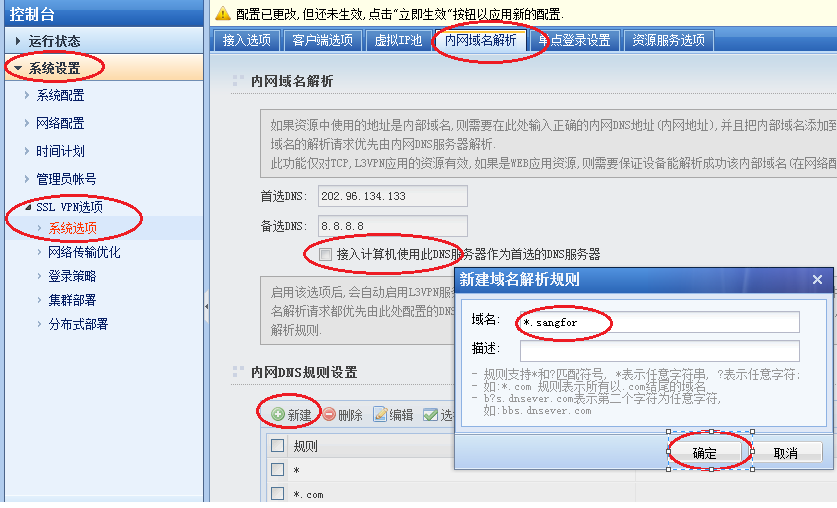


图 3.8 内网域名解析配置规则

### 内网域名解析服务器

系统设置 > SSL VPN选项 > 系统选项 > 内网域名解析 > 编辑输入DNS服务器IP地址，如图2.7所示。

注意：不得选择“接入计算机使用此DNS服务器作为首选的DNS服务器”选型，即复选框不得打钩。否则内网DNS配置不会下发。

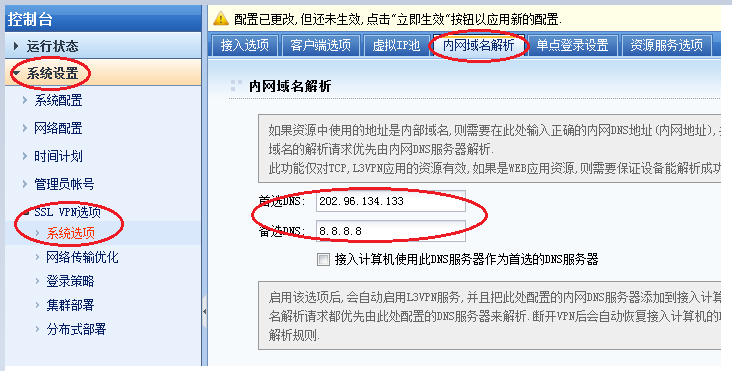


图 3.7 配置内网域名解析服务器

### IOS平台特别说明

由于IOS平台的封闭性， IOS无法获取到设备上的DNS域名解析服务器的IP地址，所以IOS平台的SDK会获取内网域名解析的DNS IP，作为HOST主机解析失败的备用地址。因此，IOS平台必须配置内网域名解析服务器。

# 自挂钩SDK使用介绍

## 接口介绍

IOS和Android平台各自提供两套用户认证使用接口，一套是阻塞的接口，一套是非阻塞的接口。想了解IOS和Android平台接口原型请具体查看第5章。

* 注意：**非阻塞的接口调用必须在主线程中**，否则非主线程的操作都会返回失败，并将错误提示字符串设置为非主线程操作错误。

## Android平台SDK包

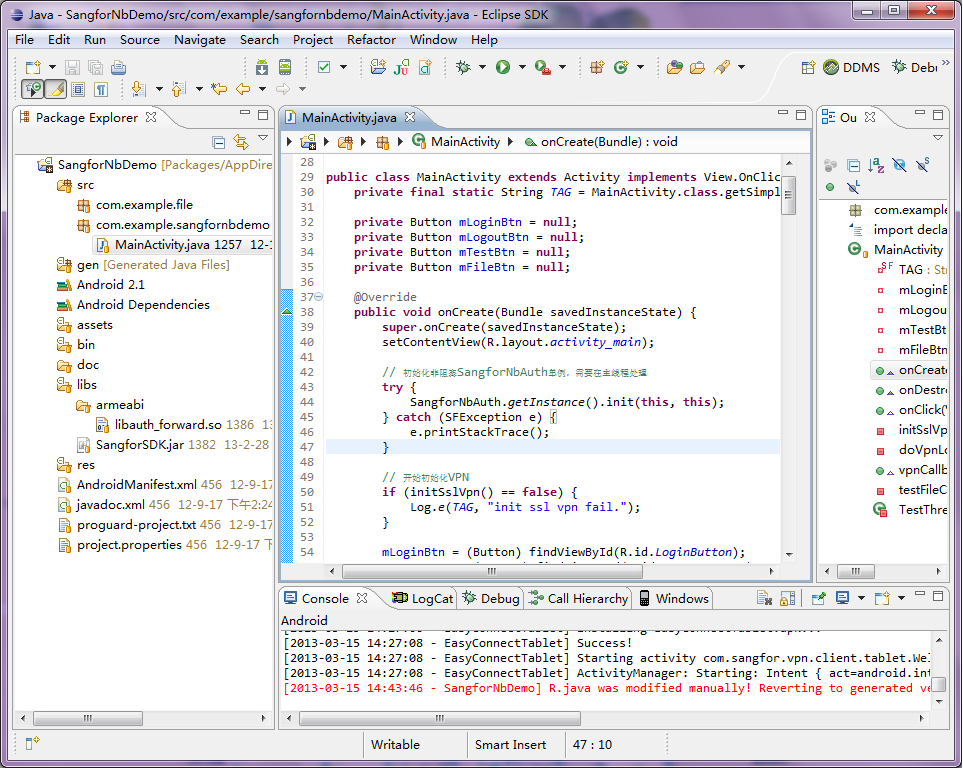
### Android开发包组成

Android平台SangforHook库包包含一个so文件和一个jar包：

* **libauth\_forward.so**文件：为编译的SDK库文件，需要加载到工程当中；
* **SangforHook.jar**包：封装的用户接口包，包含两套接口，一套是阻塞接口，另一套是非阻塞接口。

### Android开发包的使用

* **1、将jar导入工程当中**。在android工程中找到对应的libs目录，拷贝SangforHook.jar包到libs目录下，
* **2、so导入到工程中**。在libs目录下新建armeabi目录，拷贝libauth\_forward.so到libs/armeabi目录，如下图所示：



* **3、实现认证回调函数**，具体请看Demon实例。

## Android开发包Demo使用示例

下面为伪代码实现的示例，黑色和灰色部分为APP应用自身代码，蓝色部分为本方案需要添加的代码，绿色部分为备注文档。

**//当MainActivity.java APP应用程序主进程函数中，初始化SDK实例（该步骤为必要步骤）**

public void onCreate(Bundle savedInstanceState)

{

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

**// 开始初始化SDK实例……**

SangforAuth.getInstance().init(this, this);

**// ……完成SDK实例初始化**

**// 以下为APP自身的初始化过程，与SDK无关，用省略号代替**

**……**

}

**// 该函数主要是配置SSLVPN网关的地址和端口参数（该步骤为必要步骤）**

private boolean initSslVpn()

{

SangforAuth sfAuth = SangforAuth.getInstance();

long host = ipToLong("200.200.75.161");

int port = 443;

sfAuth.vpnInit(host, port);

return true;

}

**// 如果APP需要关心SSLVPN隧道建立过程中异常时返回的状态信息，还需要实现一个SDK的回调函数。（该步骤为非必要步骤，可以不实现）**

@Override public void vpnCallback(int vpnResult, int authType) {

SangforAuth sfAuth = SangforAuth.getInstance();

switch (vpnResult) {**//具体异常事件请详见DEMO工程**

case IVpnDelegate.RESULT\_VPN\_INIT\_FAIL:

case IVpnDelegate.RESULT\_VPN\_INIT\_SUCCESS:

case IVpnDelegate.RESULT\_VPN\_AUTH\_FAIL:

case IVpnDelegate.RESULT\_VPN\_AUTH\_SUCCESS:

case IVpnDelegate.RESULT\_VPN\_AUTH\_LOGOUT:

}

}

**// 登陆SSLVPN和注销SSLVPN过程应该要像下面一样调用SDK提供的接口。**

public void onClick(View v) {

if (v.equals(mLoginBtn)) {

doVpnLogin(IVpnDelegate.AUTH\_TYPE\_PASSWORD);

} else if (v.equals(mLogoutBtn)) {

SangforNbAuth.getInstance().vpnLogout();

**// 注销步骤为非必要步骤，可以不实现，不显式调用注销即可，则可以通过后台超时机制自动注销会话。**

}

}

## IOS 平台SDK包

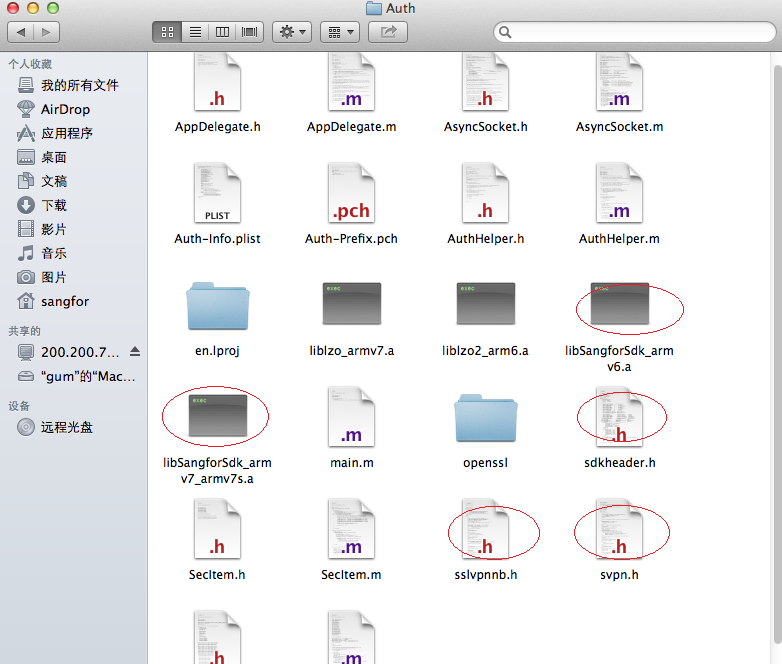
### IOS开发包介绍

IOS平台主代理开发包文件中包含：三个头文件、三个库文件，分别是：

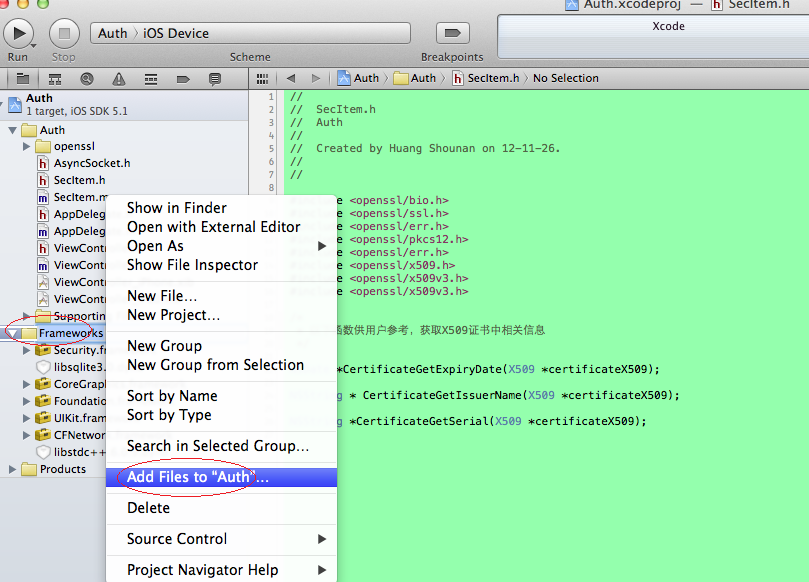
* **sdkheader.h**文件：定义了一些包括认证标识码、错误码、回调函数类型等数据类型；
* **svpn.h**文件：声明了以阻塞方式认证的函数接口；
* **sslvpnnb.h**文件：声明了以非阻塞方式认证的函数接口；
* **libSangforSdk\_v6.a**文件：提供的armv6 架构的SDK库文件；
* **libSangforSdk\_v7.a**文件：提供的armv7 架构的SDK库文件；
* **libSangforSdk\_7s.a**文件：提供的armv7s 架构的SDK库文件；

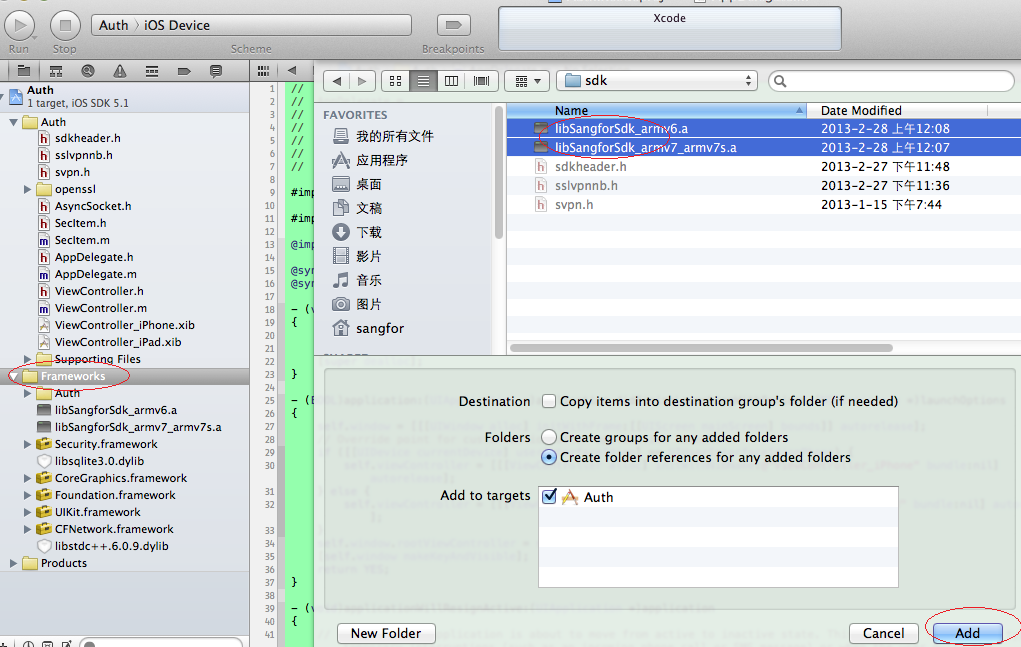
### IOS开发包使用步骤

* 注意：由于SDK是采用C++的写法编译的库，所以如果在新建工程当中未使用.cpp后缀文件时，需要链接标准C++库，将libstdc++.6.0.9.dylib加入工程当中；同时还需要添加一个CFNetwork.framework动态库。
* **1、将svpn.h、sslvpnnb.h、sdkheader.h、libSangforSdk\_v6.a、libSangforSdk\_v7.a和libSangforSdk\_7s.a文件拷贝到工程目录当中**，如下图中的DEMO示例工程Auth所示：



* **2、将svpn.h、sslvpnnb.h、sdkheader.h、libSangforSdk\_v6.a、libSangforSdk\_v7.a和libSangforSdk\_7s.a文件添加到工程中**，具体步骤如下图所示：





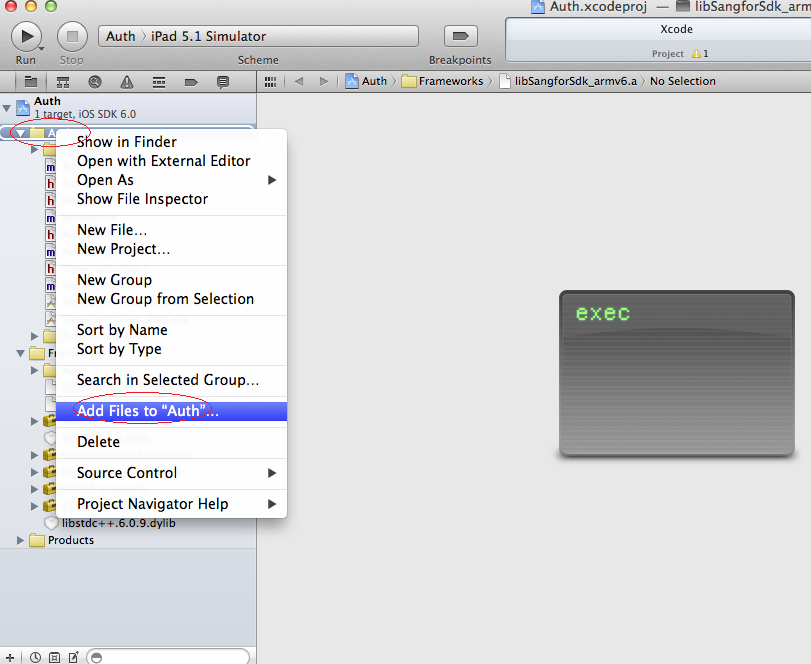
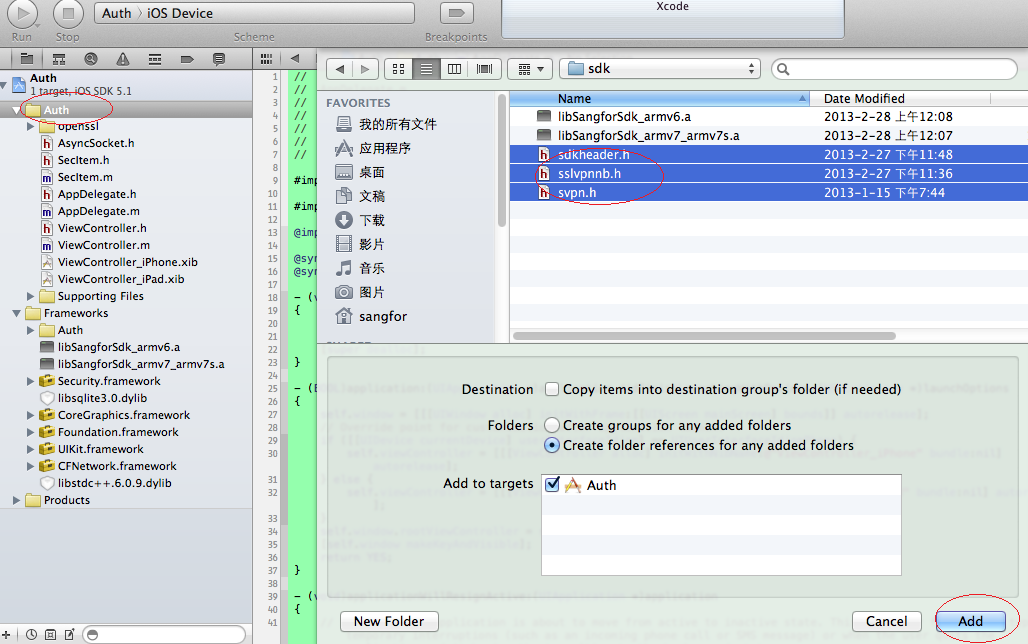


图 步骤2 添加SDK接口头文件（1）



* **3、如果用户需要同时访问多个加载了EasyApp SDK的APP应用程序，则需要在工程中AppDelegate中实现-(void)applicationWillEnterBackground:(UIApplication\*) application 函数，（一般新建工程中会自动实现）**；如果只有一个APP应用程序，则不需要实现此函数。
* **4、完成认证和回调函数**，请具体查看IOS平台Demon使用实例。

## IOS开发包Demo使用示例

下面为伪代码实现的示例，灰色部分为APP应用自身代码，蓝色部分为本方案需要添加的代码，绿色部分为备注文档。

**/\*第一 初始化\*/**

- (void)viewDidLoad

{

[super viewDidLoad];

/\*说明 此句为SVPN认证的VPN地址初始化接口调用，已经直接将C接口封装为AuthHelper类的方法，方便用户使用，如果想了解实现细则，可以直接查看AuthHelper类的实现方法

\* 功能 初始化SVPN的服务器的IP地址和端口值

\* 注意事项 此接口只需要初始化一次,设置的委托必须实现委托方法onCallBack

\*/

self.helper = [AuthHelper getInstance];

[helper setSvpnHostAndPort:vpnIp port:port delegate:self];

//以下为APP自身的初始化过程，与SDK无关。

**……**

}

**/\*第二 VPN状态获取（如果不需要关注VPN状态可以不实现此回调函数）\*/**

/\*说明 初始化和认证的需要实现的回调函数，当初始化成功或失败，认证成功或者失败的时候，该回调函数会被调用来给用户通知结果，回调结果具体查看vpnErrno的值；如果回调通知认证成功，且下一个authType类型不为空类型时，则回调会提示用户设置下一个认证类型的参数，重新开始下一个认证，类型的值在sdkheader.h中有说明

\* 功能 通知vpn初始化成功或者失败，认证成功或者失败的结果

\* 注意事项 在给AuthHelper设置的委托中需要实现此方法，同时也可以在第一次

\*/

- (void) onCallBack:(const VPN\_RESULT\_NO)vpnErrno authType:(const int)authType

{

switch (vpnErrno)

{

case RESULT\_VPN\_INIT\_FAIL: // vpn 初始化失败

break;

case RESULT\_VPN\_AUTH\_FAIL: // vpn 认证失败

break;

case RESULT\_VPN\_INIT\_SUCCESS: // vpn 初始化成功

break;

case RESULT\_VPN\_AUTH\_SUCCESS:// vpn 认证成功，authType具体值表示下一个需要的认证

break;

case RESULT\_VPN\_AUTH\_LOGOUT: // vpn 注销成功

break;

case RESULT\_VPN\_NONE:

break;

default:

break;

}

}

**/\*第三 设置认证参数，开始登录SVPN\*/**

-(IBAction)buttonPressed:(id)sender

{

/\* 说明 此句为SVPN认证设置参数的接口调用，已经直接将C接口封装为AuthHelper类的方法，方便用户使用，如果想了解实现细则，可以直接查看AuthHelper类的实现方法

\* 功能 开始登陆SVPN服务器，开始SVPN的用户名密码认证

\* 注意事项 设置认证参数可以在开始认证前或者开始认证之后，如果开始认证后发现没有设置对应的认真参数，还会以回调函数的形势通知用户设置，此处具有很强的灵活性，方便用户选择性的使用\*/

[helper setUserNamePassword:userName1 password:password1];

/\*说明 此句为SVPN认证开启认证的接口调用，已经直接将C接口封装为AuthHelper类的方法，方便用户使用，如果想了解实现细则，可以直接查看AuthHelper类的实现方法

\* 功能 开始登陆SVPN服务器，开始SVPN的用户名密码认证

\* 注意事项 此接口调用必须在setSvpnHostAndPort调用成功后调用，否则可能会直接失败，认证的结果将会以回调函数的方式通知用户\*/

[helper loginVpn:SSL\_AUTH\_TYPE\_PASSWORD];

**// 以下为APP自身的初始化过程，与SDK无关。**

**……**

}

第四步，注销登陆SVPN

-(IBAction)buttonPressed:(id)sender

{

if (islogout) {

/\*\*说明 此句为SVPN注销SVPN用户的接口调用，已经直接将C接口封装为AuthHelper类的方法，方便用户使用，如果想了解实现细则，可以直接查看AuthHelper类的实现方法

\* 功能 注销登陆的SVPN用户

\*/

[helper logoutVpn];

//注销步骤为非必要步骤，可以不实现，不显式调用注销即可，则可以通过后台超时机制自动注销会话。

}

**//以下为APP自身的初始化过程，与SDK无关，用省略号代替。**

**……**

}

## 自动注销和断线自动重连说明

**自动注销：**由于VPN网关设备处可以设置用户超时时间，即如果用户在一定时间内与VPN服务端没有任何数据交互，则VPN网关设备会自动的注销用户，则需要用户再次登录上线。

**断线自动重连**：用户登录后，超时时间内的意外的网络掉线，SDK会监测到并进行自动重连操作，用户无需任何操作，且无感知。

**主动注销**：当用户主动注销登陆时，SDK不会自动重连。

# iOS&Android平台SDK接口介绍

## Android平台接口

### Android平台阻塞接口

阻塞接口，认证过程采用阻塞方式，适用用单一认证或者一次性把认证参数设定完再进行认证，被动注销时也会通过回调接口通知。

SangforAuth获得SangforAuth的单例。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | SangforAuth getInstance() | | | |
| 功能 | 获取SangforVpn单实例 | | | |
| 前置条件 | 无 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| 无 | 无 | 无 | 无 |
|  |  |  |  |
| 返回值 | SangforAuth | | | |
| 说明 | 获取SangforAuth单实例，用于Vpn认证 | | | |

SangforAuth 初始化SangforAuth实例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | void init(Context context, IVpnDelegate delegate) throws SFException | | | |
| 功能 | 设置上下文对象context, 设置回调接口delegate | | | |
| 前置条件 | 实现各个回调接口，同时需要在主线程中调用该函数 | | | |
| 后置条件 | 需要处理关注下异常的处理 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| Context | Context | in | 上下文对象 |
| Delegate | IVpnDelegate | in | 回调接口 |
| 返回值 | Void | | | |
| 说明 | 连接vpn前设置实现回调接口，该接口主要用户通知用户是否被动注销了，其它功能基本可忽略； | | | |
| 备注 | 抛出：  com.sangfor.vpn.SFException：任一参数为null或者在非主线程中调用； | | | |

SangforAuth 初始化vpn

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | boolean vpnInit(long host, int port) | | | |
| 功能 | 初始化vpn、设定vpn的地址和端口 | | | |
| 前置条件 | 无 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| Host | long | in | 连接vpn地址 |
| Port | int | in | 连接vpn端口 |
| 返回值 | boolean | | | |
| 说明 | 初始化vpn，成功返回true，失败返回false | | | |

SangforAuth 设置认证参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | boolean setLoginParam(String key, String value) | | | |
| 功能 | 认证之前通过该接口设定认证参数，key值在IVpnDelegate中定义，value为key值对应的数据 | | | |
| 前置条件 | 认证之前需要调用该接口设定参数 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| key | String | in | 认证参数key值，定义在IVpnDelegate |
| value | String | in | Key对应的认证参数 |
| 返回值 | Boolean | | | |
| 说明 | 设定成功返回true，失败返回false，该接口可重复调用，将需要的认证参数一次性设定完 | | | |

SangforAuth 用户认证

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | boolean vpnLogin(int authType) | | | |
| 功能 | 根据认证类型进行用户认证 | | | |
| 前置条件 | 先要设定认证参数 | | | |
| 后置条件 | 如果此次认证成功，且需要下一个认证，则会回调告知需要下一个认证 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| authType | int | in | 认证类型 |
| 无 |  |  |  |
| 返回值 | 成功认证返回true，失败返回false | | | |
| 说明 | 成功返回true失败返回false | | | |

SangforAuth 取消认证

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | boolean [vpnCancelLogin](file:///C:/Users/csh/Desktop/SDK/doc/com/sangfor/vpn/auth/SangforAuth.html#vpnCancelLogin())() | | | |
| 功能 | 取消登录 | | | |
| 前置条件 | 正在登录vpn | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| 无 |  |  |  |
| 无 |  |  |  |
| 返回值 | Boolean | | | |
| 说明 | 成功返回true失败返回false | | | |

SangforAuth 主动用户注销

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | boolean [vpnLogout](file:///C:/Users/csh/Desktop/SDK/doc/com/sangfor/vpn/auth/SangforAuth.html#vpnLogout())() | | | |
| 功能 | 主动用户注销 | | | |
| 前置条件 | 用户已经登入了 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
|  |  |  |  |
|  |  |  |  |
| 返回值 | Boolean | | | |
| 说明 | 当用户认证通过后，如果想退出使用vpn则需要调用该接口 | | | |

SangforAuth 获取错误信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | [vpnGeterr](file:///C:/Users/csh/Desktop/SDK/doc/com/sangfor/vpn/auth/SangforAuth.html#vpnGeterr())() | | | |
| 功能 | 当调用vpn接口时出错，可通过该接口获取具体信息 | | | |
| 前置条件 | 发生了调用失败 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| 无 | 无 | 无 |  |
|  |  |  |  |
| 返回值 | String 注意该返回值的编码格式是UTF-8编码 | | | |
| 说明 | 当调用vpn接口失败时可获取错误信息，需要及时获取，如果继续操作错误信息会被其它覆盖掉 | | | |

接口IVpnDelegate 回调函数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | void vpn[Callback](file:///C:/Users/csh/Desktop/SDK/doc/com/sangfor/vpn/IVpnDelegate.html#vpnCallback(int, int))(int vpnResult, int authType) | | | |
| 功能 | vpn操作过程中回调通知结果 | | | |
| 前置条件 | 无 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| vpnResult | int | in | Vpn操作结果 |
| authType | int | in | 当前认证类型 |
| 返回值 | 无 | | | |
| 说明 | 操作vpn过程中会回调，主要用在用户被动注销时通知logout | | | |

查询VPN状态接口

|  |  |
| --- | --- |
| 名称 | [vpnQueryStatus](file:///C:\Documents%20and%20Settings\Administrator\%E6%A1%8C%E9%9D%A2\EasyApp\Packages\AppDirectConnect\android\doc\com\sangfor\vpn\auth\SangforAuth.html#vpnQueryStatus())() |
| 功能 | 查询VPN状态 |
| 前置条件 | 在主线程调用 |
| 后置条件 | 无 |
| 返回值 | VPN状态，状态码在IVpnDelegate中 |
| 说明 |  |

设置需要加密文件路径的规则

|  |  |
| --- | --- |
| 名称 | void setCryptFileRule(String rules) |
| 功能 | 设置需要加密文件路径的规则。可以使用\*和?模式匹配符号。如/sdcard/\*;等，可以有多条规则，但每一规则之前用分号隔开 |
| 前置条件 | 在主线程调用 |
| 后置条件 | 无 |
| 返回值 | 无 |
| 说明 | 需要在vpnInit之前调用，在之后调用无效果。如果在vpnInit之前没有调用此接口，将不会启动本地防泄密功能 |

设置加密文件路径排除规则

|  |  |
| --- | --- |
| 名称 | void setExCrptFileRule(String rules) |
| 功能 | 设置加密文件路径排除规则，同上可以使用\*和?的模式匹配。可以有多条，每一条之前用分号隔开。 |
| 前置条件 | 在主线程调用 |
| 后置条件 | 无 |
| 返回值 | 无 |
| 说明 | 如果同时设置了加密规则和排除规则，先匹配排除规则 |

### 4.1.2 Android平台非阻塞接口

非阻塞接口，用户认证过程中均采用非阻塞方式，所以需要注意调用下面标注是否为异步接口，如果是则调用成功并不代表成功了，结果会通过回到函数通知。

SangforNbAuth获得SangforNbAuth的单例。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | SangforNbAuthgetInstance() | | | |
| 功能 | 获取SangforNbAuth单实例 | | | |
| 前置条件 | 无 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| 无 | 无 | 无 | 无 |
|  |  |  |  |
| 返回值 | SangforNbAuth 实例 | | | |
| 说明 | 获取SangforNbAuth单实例，用于Vpn认证 | | | |

SangforNbAuth初始化SangforNbAuth实例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | void init(Context context, IVpnDelegate delegate) throws SFException | | | |
| 功能 | 设置上下文对象context, 设置回调接口delegate | | | |
| 前置条件 | 实现各个回调接口，同时需要在主线程中调用该函数 | | | |
| 后置条件 | 需要处理关注下异常的处理 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| Context | Context | in | 上下文对象 |
| Delegate | IVpnDelegate | in | 回调接口 |
| 返回值 | Void | | | |
| 说明 | 连接vpn前设置实现回调接口，该接口主要用户通知用户是否被动注销了，其它功能基本可忽略； | | | |
| 备注 | 抛出：  com.sangfor.vpn.SFException：任一参数为null或者在非主线程中调用； | | | |

SangforNbAuth初始化vpn

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | boolean vpnInit(long host, int port) | | | |
| 功能 | 初始化vpn、设定vpn的地址和端口 | | | |
| 前置条件 | 无 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| Host | long | in | 连接vpn地址 |
| Port | int | in | 连接vpn端口 |
| 返回值 | boolean 成功调用返回true，失败调用返回false | | | |
| 说明 | 异步接口，最终结果由回调函数通知 | | | |

SangforNbAuth设置认证参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | boolean setLoginParam(String key, String value) | | | |
| 功能 | 认证之前通过该接口设定认证参数，key值在IVpnDelegate中定义，value为key值对应的数据 | | | |
| 前置条件 | 认证之前需要调用该接口设定参数 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| key | String | in | 认证参数key值，定义在IVpnDelegate |
| value | String | in | Key对应的认证参数 |
| 返回值 | boolean 设定成功返回true，失败返回false | | | |
| 说明 | 该接口可重复调用，将需要的认证参数一次性设定完 | | | |

SangforNbAuth用户认证

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | boolean vpnLogin(int authType) | | | |
| 功能 | 根据认证类型进行用户认证 | | | |
| 前置条件 | 先要设定认证参数 | | | |
| 后置条件 | 如果此次认证成功，且需要下一个认证，则会回调告知需要下一个认证 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| authType | int | in | 认证类型 |
| 无 |  |  |  |
| 返回值 | boolean 成功调用返回true，失败调用返回false， | | | |
| 说明 | 异步接口，最终结果由回调函数通知 | | | |

SangforNbAuth取消认证

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | boolean [vpnCancelLogin](file:///C:/Users/csh/Desktop/SDK/doc/com/sangfor/vpn/auth/SangforAuth.html#vpnCancelLogin())() | | | |
| 功能 | 取消登录 | | | |
| 前置条件 | 正在登录vpn | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| 无 |  |  |  |
| 无 |  |  |  |
| 返回值 | boolean 成功返回true失败返回false | | | |
| 说明 | 异步接口，成功取消由回调函数通知 | | | |

SangforNbAuth主动用户注销

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | boolean [vpnLogout](file:///C:/Users/csh/Desktop/SDK/doc/com/sangfor/vpn/auth/SangforAuth.html#vpnLogout())() | | | |
| 功能 | 主动用户注销 | | | |
| 前置条件 | 用户已经登入了 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
|  |  |  |  |
|  |  |  |  |
| 返回值 | boolean 成功调用返回true，失败调用返回false | | | |
| 说明 | 当用户认证通过后，如果想退出使用vpn则需要调用该接口，异步接口，logout结果由回调函数通知 | | | |

SangforNbAuth获取错误信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | [vpnGeterr](file:///C:/Users/csh/Desktop/SDK/doc/com/sangfor/vpn/auth/SangforAuth.html#vpnGeterr())() | | | |
| 功能 | 当调用vpn接口时出错，可通过该接口获取具体信息 | | | |
| 前置条件 | 发生了调用失败 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| 无 | 无 | 无 |  |
|  |  |  |  |
| 返回值 | String ，注意该返回值的编码格式是UTF-8编码 | | | |
| 说明 | 当调用vpn接口失败时可获取错误信息，需要及时获取，如果继续操作错误信息会被其它覆盖掉 | | | |

接口IVpnDelegate 回调函数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | void vpn[Callback](file:///C:/Users/csh/Desktop/SDK/doc/com/sangfor/vpn/IVpnDelegate.html#vpnCallback(int, int))(int vpnResult, int authType) | | | |
| 功能 | vpn操作过程中回调通知结果 | | | |
| 前置条件 | 无 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| vpnResult | int | in | Vpn操作结果 |
| authType | int | in | 当前认证类型 |
| 返回值 | 无 | | | |
| 说明 | 操作vpn过程中会回调，主要用在用户被动注销时通知logout | | | |

查询VPN状态接口

|  |  |
| --- | --- |
| 名称 | [vpnQueryStatus](file:///C:\Documents%20and%20Settings\Administrator\%E6%A1%8C%E9%9D%A2\EasyApp\Packages\AppDirectConnect\android\doc\com\sangfor\vpn\auth\SangforAuth.html#vpnQueryStatus())() |
| 功能 | 查询VPN状态 |
| 前置条件 | 在主线程调用 |
| 后置条件 | 无 |
| 返回值 | VPN状态，状态码在IVpnDelegate中 |
| 说明 |  |

设置需要加密文件路径的规则

|  |  |
| --- | --- |
| 名称 | void setCryptFileRule(String rules) |
| 功能 | 设置需要加密文件路径的规则。可以使用\*和?模式匹配符号。如/sdcard/\*;等，可以有多条规则，但每一规则之前用分号隔开 |
| 前置条件 | 在主线程调用 |
| 后置条件 | 无 |
| 返回值 | 无 |
| 说明 | 需要在vpnInit之前调用，在之后调用无效果。如果在vpnInit之前没有调用此接口，将不会启动本地防泄密功能 |

设置加密文件路径排除规则

|  |  |
| --- | --- |
| 名称 | void setExCrptFileRule(String rules) |
| 功能 | 设置加密文件路径排除规则，同上可以使用\*和?的模式匹配。可以有多条，每一条之前用分号隔开。 |
| 前置条件 | 在主线程调用 |
| 后置条件 | 无 |
| 返回值 | 无 |
| 说明 | 如果同时设置了加密规则和排除规则，先匹配排除规则 |

## 4.2 iOS平台接口

### 4.2.1 iOS平台阻塞接口

阻塞接口，认证过程采用阻塞方式，适用用单一认证或者一次性把认证参数设定完再进行认证，被动注销时也会通过回调接口通知。

vpn\_init初始化VPN信息，设置回调函数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | Int vpn\_init(const VPN\_CALL\_BACK callBank, const uint32\_t vpn, const uint16\_t port); | | | |
| 功能 | 初始化VPN信息 | | | |
| 前置条件 | 无 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| callback | VPN\_CALL\_BACK | in | 初始化成功或者失败的回调函数 |
| vpn | unt32\_t | In | vpn的Ip地址，必须为网络序 |
| port | Unt16\_t | In | vpn的端口值，必须为网络序 |
| 返回值 | 0 表示初始化预成功，-1表示失败 | | | |
| 说明 | 当认证初始化成后，callBack都会回调，注意查看回调函数的错误码，判断认证初始化是否真正成功。 | | | |

vpn\_set\_login\_param 设置认证参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | void vpn\_set\_login\_param(const char \*key, const char \*value); | | | |
| 功能 | 设置认证参数，key的参数设置请具体参看sdkheader.h文件 | | | |
| 前置条件 | 无 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| key | const char\* | In | 必须为sdkheader中标准的key值 |
| value | const char\* | In | 必须为正确的有效值，否者认证可能会失败 |
| 返回值 | 无 | | | |
| 说明 | 设置认证参数 | | | |

vpn\_login登陆VPN，开始认证

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | int vpn\_login(cons tint authType) | | | |
| 功能 | 登陆VPN,开始当前认证 | | | |
| 前置条件 | 必须在调用vpn\_init后才能正确调用，否则可能会失败 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| authType | int | in | 当前开始认证类型 |
| 返回值 | 0表示认证预成功，-1表示出错误 | | | |
| 说明 | 必须在vpn\_init后调用，否者可能会出错,。注意：要察看当前认证成功、失败或者认证成功或者失败结束，必须对回调函数中的返回的参数值进行判断，才能加以确定，具体请看Demon实例。 | | | |

vpn\_logout 注销VPN登陆

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | int vpn\_logout() | | | |
| 功能 | 注销VPN登陆 | | | |
| 前置条件 | 无 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| 无 | 无 | 无 | 无 |
| 返回值 | 0表示预注销成功，-1表示注销失败 | | | |
| 说明 | vpn\_logout与vpn\_login须配对使用 | | | |

vpn\_quit 退出VPN

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | Int vpn\_quit() | | | |
| 功能 | 退出VPN登陆 | | | |
| 前置条件 | 必须先调用vpn\_init | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| 无 | 无 | 无 | 无 |
| 返回值 | 0表示成功，-1表示失败 | | | |
| 说明 | 注意vpn\_quit与vpn\_quit的匹配使用，调用vpn\_quit前必须先调用vpn\_init | | | |

vpn\_query\_state查询VPN状态信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | VPN\_STATUS vpn\_query\_status() | | | |
| 功能 | 查询VPN的状态码 | | | |
| 前置条件 | 无 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| 无 | 无 | 无 | 无 |
| 返回值 | 返回当前VPN的状态码，可以具体查看sdkheader.h头文件中的状态定义信息 | | | |
| 说明 | 无 | | | |

set\_dns\_server设置DNS域名解析服务器地址

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | void set\_dns\_server(const char\*dnsServer); | | | |
| 功能 | 设置DNS域名解析服务器地址 | | | |
| 前置条件 | 该设置必须在vpn\_init调用前设置才有效，否则后续设置不回被加载 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| dnsServer | const char \* | in | 必须为有效的IP地址字符串，若为无效的IP地址字符串，设置的时候会被丢弃 |
| 返回值 | 无 | | | |
| 说明 | 此调用必须在vpn\_init前调用才有效，在vpn\_init后调用可能会失效 | | | |

vpn\_geterr获取错误提示字符串信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | const char\* vpn\_geterr() | | | |
| 功能 | 当调用出现错误时候，可以调用该接口获取错误提示字符串信息 | | | |
| 前置条件 | 当出现调出现错误的时候 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| 无 | 无 | 无 | 无 |
| 返回值 | NULL表示获取失败，非NULL返回查询的错误提示字符串 | | | |
| 说明 | 可能会出现获取失败的情况 | | | |

vpn\_query\_hardid 获取设备硬件特征码信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | const char \*vpn\_query\_hardid() | | | |
| 功能 | 获取硬件特征码信息字符串 | | | |
| 前置条件 | 无 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| 无 | 无 | 无 | 无 |
| 返回值 | NULL表示获取失败，非NULL表示获取成功 | | | |
| 说明 | 可能会出现获取失败的情况 | | | |

get\_version(void) 获取SDK的版本号

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | const char \* get\_version(void) | | | |
| 功能 | 获取SDK的版本号 | | | |
| 前置条件 | 无 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| 无 | 无 | 无 | 无 |
| 返回值 | 非NULL表示成功返回版本号信息 | | | |
| 说明 | 无 | | | |

设置需要加密文件路径的规则

|  |  |
| --- | --- |
| 名称 | void set\_file\_crypt\_rules (const char \*rules) |
| 功能 | 设置需要加密文件路径的规则。可以使用\*和?模式匹配符号。如/sdcard/\*;等，可以有多条规则，但每一规则之前用分号隔开 |
| 前置条件 | 在主线程调用 |
| 后置条件 | 无 |
| 返回值 | 无 |
| 说明 | 需要在vpnInit之前调用，在之后调用无效果。如果在vpnInit之前没有调用此接口，将不会启动本地防泄密功能 |

设置加密文件路径排除规则

|  |  |
| --- | --- |
| 名称 | void set\_file\_exclude\_crypt\_rules(const char \*rules); |
| 功能 | 设置加密文件路径排除规则，同上可以使用\*和?的模式匹配。可以有多条，每一条之前用分号隔开。 |
| 前置条件 | 在主线程调用 |
| 后置条件 | 无 |
| 返回值 | 无 |
| 说明 | 如果同时设置了加密规则和排除规则，先匹配排除规则 |

### 4.2.2 iOS平台非阻塞接口

非阻塞接口，用户认证过程中均采用非阻塞方式，所以需要注意调用下面标注是否为异步接口，如果是则调用成功并不代表成功了，结果会通过回到函数通知。

ssl\_vpn\_init初始化VPN信息，设置回调函数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | Int ssl\_vpn\_init(const VPN\_CALL\_BACK callBank, const uint32\_t vpn, const uint16\_t port); | | | |
| 功能 | 初始化VPN信息 | | | |
| 前置条件 | 无 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| callback | VPN\_CALL\_BACK | in | 初始化成功或者失败的回调函数 |
| vpn | unt32\_t | In | vpn的Ip地址，必须为网络序 |
| port | Unt16\_t | In | vpn的端口值，必须为网络序 |
| 返回值 | 0 表示初始化预成功，-1表示初始化失败 | | | |
| 说明 | 当认证初始化成功或者失败后，callBack都会回调，注意查看回调函数的参数，来判断认证初始化是否真正成功。 | | | |
| 备注 | 注意：该非阻塞接口必须在主线程中调用，否者调用会返回错误，同时要判断回调函数中的回调参数错误码，来判断初始是否真正化成功。 | | | |

ssl\_vpn\_set\_login\_param 设置认证参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | void ssl\_vpn\_set\_login\_param(const char \*key, const char \*value); | | | |
| 功能 | 设置认证参数，key的参数设置请具体参看sdkheader.h文件 | | | |
| 前置条件 | 无 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| key | const char\* | In | 必须为sdkheader中标准的key值 |
| value | const char\* | In | 必须为正确的有效值，否者认证可能会失败 |
| 返回值 | 无 | | | |
| 说明 | 设置认证参数 | | | |
| 备注 | 注意：该非阻塞接口必须在主线程中调用，否者调用会返回错误 | | | |

ssl\_vpn\_login登陆VPN，开始认证

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | int ssl\_vpn\_login(cons tint authType) | | | |
| 功能 | 登陆VPN,开始当前认证 | | | |
| 前置条件 | 必须在调用ssl\_vpn\_init后才能正确调用，否则可能会失败 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| authType | int | in | 当前开始认证类型 |
| 返回值 | 0表示认证预成功，-1表示出错误 | | | |
| 说明 | 必须在ssl\_vpn\_init后调用，否者可能会出错,。注意：要察看当前认证成功、失败或者认证成功或者失败结束，必须对回调函数中的返回的参数值进行判断，才能加以确定，具体请看Demon实例。 | | | |
| 备注 | 注意：该非阻塞接口必须在主线程中调用，否者调用会返回错误 | | | |

ssl\_vpn\_logout 注销VPN登陆

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | int ssl\_vpn\_logout() | | | |
| 功能 | 注销VPN登陆 | | | |
| 前置条件 | 无 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| 无 | 无 | 无 | 无 |
| 返回值 | 0表示注销预成功，-1表示注销失败 | | | |
| 说明 | ssl\_vpn\_logout与ssl\_vpn\_login须配对使用 | | | |
| 备注 | 注意：该非阻塞接口必须在主线程中调用，否者调用会返回错误，同时是否真正注销成功需要对注销回调函数的错误码进行判断。 | | | |

ssl\_vpn\_quit 退出VPN

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | Int ssl\_vpn\_quit() | | | |
| 功能 | 退出VPN登陆 | | | |
| 前置条件 | 必须先调用ssl\_vpn\_init | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| 无 | 无 | 无 | 无 |
| 返回值 | 0表示成功，-1表示失败 | | | |
| 说明 | 注意ssl\_vpn\_quit与ssl\_vpn\_quit的匹配使用，调用ssl\_vpn\_quit前必须先调用ssl\_vpn\_init | | | |
| 备注 | 注意：该非阻塞接口必须在主线程中调用，否者调用会返回错误 | | | |

ssl\_vpn\_query\_state查询VPN状态信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | VPN\_STATUS ssl\_vpn\_query\_status() | | | |
| 功能 | 查询VPN的状态码 | | | |
| 前置条件 | 无 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| 无 | 无 | 无 | 无 |
| 返回值 | 返回当前VPN的状态码，可以具体查看sdkheader.h头文件中的状态定义信息 | | | |
| 说明 | 无 | | | |
| 备注 | 注意：该非阻塞接口必须在主线程中调用，否者调用会返回错误 | | | |

ssl\_set\_dns\_server设置DNS域名解析服务器地址

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | void ssl\_set\_dns\_server(const char\*dnsServer); | | | |
| 功能 | 设置DNS域名解析服务器地址 | | | |
| 前置条件 | 该设置必须在ssl\_vpn\_init调用前设置才有效，否则后续设置不回被加载 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| dnsServer | const char \* | in | 必须为有效的IP地址字符串，若为无效的IP地址字符串，设置的时候会被丢弃 |
| 返回值 | 无 | | | |
| 说明 | 此调用必须在ssl\_vpn\_init前调用才有效，在ssl\_vpn\_init后调用可能会失效 | | | |
| 备注 | 注意：该非阻塞接口必须在主线程中调用，否者调用会返回错误 | | | |

ssl\_vpn\_geterr获取错误提示字符串信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | const char\* ssl\_vpn\_geterr() | | | |
| 功能 | 当调用出现错误时候，可以调用该接口获取错误提示字符串信息 | | | |
| 前置条件 | 当出现调出现错误的时候 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| 无 | 无 | 无 | 无 |
| 返回值 | NULL表示获取失败，非NULL返回查询的错误提示字符串 | | | |
| 说明 | 可能会出现获取失败的情况，注意该返回值的编码格式是UTF-8编码 | | | |
| 备注 | 注意：该非阻塞接口必须在主线程中调用，否者调用会返回错误 | | | |

ssl\_vpn\_query\_hardid 获取设备硬件特征码信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | const char \*ssl\_vpn\_query\_hardid() | | | |
| 功能 | 获取硬件特征码信息字符串 | | | |
| 前置条件 | 无 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| 无 | 无 | 无 | 无 |
| 返回值 | NULL表示获取失败，非NULL表示获取成功 | | | |
| 说明 | 可能会出现获取失败的情况 | | | |
| 备注 | 注意：该非阻塞接口必须在主线程中调用，否者调用会返回错误 | | | |

ssl\_get\_version(void) 获取SDK的版本号

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | const char \* ssl\_get\_version(void) | | | |
| 功能 | 获取SDK的版本号 | | | |
| 前置条件 | 无 | | | |
| 后置条件 | 无 | | | |
| 参数 | 参数名 | 类型 | 方向 | 说明 |
| 无 | 无 | 无 | 无 |
| 返回值 | 非NULL表示成功返回版本号信息 | | | |
| 说明 | 无 | | | |

设置需要加密文件路径的规则

|  |  |
| --- | --- |
| 名称 | void set\_file\_crypt\_rules (const char \*rules) |
| 功能 | 设置需要加密文件路径的规则。可以使用\*和?模式匹配符号。如/sdcard/\*;等，可以有多条规则，但每一规则之前用分号隔开 |
| 前置条件 | 在主线程调用 |
| 后置条件 | 无 |
| 返回值 | 无 |
| 说明 | 需要在vpnInit之前调用，在之后调用无效果。如果在vpnInit之前没有调用此接口，将不会启动本地防泄密功能 |

设置加密文件路径排除规则

|  |  |
| --- | --- |
| 名称 | void set\_file\_exclude\_crypt\_rules(const char \*rules); |
| 功能 | 设置加密文件路径排除规则，同上可以使用\*和?的模式匹配。可以有多条，每一条之前用分号隔开。 |
| 前置条件 | 在主线程调用 |
| 后置条件 | 无 |
| 返回值 | 无 |
| 说明 | 如果同时设置了加密规则和排除规则，先匹配排除规则 |