Weline macOS SDK 开发手册

Weline.io

关于本手册

版权声明

信息更新

本文档仅用于为NDA用户提供信息，并且随时可由Weline.io更改或撤回。

免责条款

根据适用法律的许可范围，Weline.io按“原样”提供本文档而不承担任何形式的担保，包括（但不限于）任何隐含的适销性、特殊目的适用性或无侵害性。在任何情况下，Weline.io都不会对最终用户或任何第三方因根据说明文档使用SDK/后台接口造成的任何直接或间接损失或损坏负责，即使Weline.io明确得知这些损失或损坏，这些损坏包括（但不限于）利润损失、业务中断、信誉或数据丢失。

阅读对象

本文的读者对象为和 Weline.io 签署了 NDA（保密协议）的第三方厂商的研发人员。

密级

本文的密级为秘密，禁止向非 NDA 公司或个人开放此文件。

目录

修订记录

日期	修订版本	描述
2019/04/10	V1.0a	完成初稿
2020/04/14	V1.0b	增加 3.12 节，DLT 设置接口
2020/08/15	V1.0c	更新 3.5 节，创建隧道新增多种认证方式支持
2020/11/05	V1.0d	更新 1.2.4 节，启动 sdvnd 服务的命令加-w 参数；更新 3.1 节 添加 APP 启动 sdvnd 服务的代码示例

1 概要

Mac OS X 平台 SDK 底层采用 C/C++语言发开，API 接口以 OC 的形式提供，所以开发者需要在项目中添加 libc++.tbd 库的链接，特别注意使用该 SDK 需要在 Xcode 工程里面去掉应用的 sandbox 限制。

此文档适合公司内部和签署了 NDA 的第三方研发人员阅读。

1.1 术语

标识	说明
sdk 进程	调用 sdk 库的进程
sdvnd 进程	Mac OS X 系统下的虚网的服务进程
partnerId	合作伙伴 ID，由 Weline.io 后台统一授权分配
appId	产品 ID，针对每一款产品授权分配
deviceClass	设备类型编码，用于区分同一个产品 ID 下的多个类型的设备
SN	设备唯一识别码（机器码）
vip	设备的虚拟 ip 地址
“xx”	json 格式字符形式
xx	json 格式的整数形式

1.2 关于 SDK

1.2.1 关于系统权限

由于 Mac OS X 平台虚拟网隧道的建立需要使用系统提供的 tun 设备和进行必要的系统路由表操作，所以需要具备 root 权限。

1.2.2 关于 SDK 的使用

为了方便开发者将 Weline.io 虚网功能集成到应用中，我们提供了两种使用 SDK 的方式:

方式一 ： 以 root 权限运行的应用可以使用 libcmapi_service.a 进行开发
方式二 ： 以非 root 权限运行的应用可以使用 libcmapi_client.a 进行开发

1.2.2.1 root 权限使用 SDK 说明

- SDK 文件说明 文件名 | 说明 —|— libcmapi_service.a|虚拟网核心服务库文件，需要调用进程以 root 权限运行 cmapi_service.h |接口头文件 common_defs.h |错误码等定义文件 main.m |cmapi_service_demo 示例代码
- 使用 SDK 需要添加的系统依赖

```
libc++.tbd
IOKit.framework
SystemConfiguration.framework
CoreFoundation.framework
```

1.2.2.2 非 root 权限使用 SDK 说明

- SDK 文件说明 文件名 | 说明 —|— libcmapi_client.a |虚拟网接口库文件，用来连接和控制 sdvnd 程序，可以以非 root 权限运行 sdvnd |虚拟网核心服务程序，需要以 root 权限运行 net.Weline.io.sdvn.service.plist|Mac OS X 平台服务注册文件，用于将 sdvnd 注册为随系统启动的服务进程 cmapi_client.h |接口头文件 common_defs.h |错误码等定义文件 main.m |cmapi_client_demo 示例代码
- 使用 SDK 需要添加的系统依赖

```
libc++.tbd
```

1.2.3 关于 SDK 的系统操作

- 文件操作 文件路径 | 说明 —|— /etc/sdvn/server.cfg|配置信息文件，由 libcmapi_service.a 或者 sdvnd 程序写入，文件小于 1000 字节 /etc/sdvn/.netinfo.backup|备份信息文件，由 libcmapi_service.a 或者 sdvnd 程序写入，文件小于 200 字节
- 路由表操作

虚拟网功能实现需要对系统路由表进行操作，所以请注意：
1.与其它操作系统路由表的软件(如:VPN 程序)可能产生冲突
2.请勿运行多个 libcmapi_service.a 实例

3.请勿将 libcmapi_service.a 和 sdvnd 同时使用
4.libcmapi_service.a 不具备多实例检测，sdvnd 具备多实例检测

- DNS 操作

虚拟网提供虚拟域名解析功能，如果开启该功能，将会在虚拟网登入成功后修改系统 DNS 设置，虚拟网登出还原 DNS 设置。如何开启该功能参见 SDK 初始化接口。

1.2.4 关于 sdvnd 服务的安装

- 将 sdvnd 打包到应用程序中

在 Xcode 中设置 Copy Files 将 sdvnd 拷贝到 Executables 中

- 修改 **net.Weline.io.sdvn.service.plist** 文件内容，填写正确的路径(请只替换 XXXX)

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>Disabled</key>
    <false/>
    <key>Label</key>
    <string>net.Weline.io.sdvn.service</string>
    <key>OnDemand</key>
    <false/>
    <key>ProgramArguments</key>
    <array>
        <string>/Applications/XXXX/XXXX.app/Contents/MacOS/sdvnd</string>
    </array>
    <key>RunAtLoad</key>
    <true/>
    <key>WorkingDirectory</key>
    <string>/Applications/XXXX/XXXXX.app/Contents/MacOS</string>
    <key>KeepAlive</key>
    <true/>
</dict>
</plist>
```

- 安装脚本中添加执行命令

```
sudo launchctl load -w /Library/LaunchDaemons/net.Weline.io.sdvn.service.plist
```

2 错误码

2.1 错误码定义

```
typedef enum
{
    CE_SUCC = 0,
    CE_FAIL,
    CE_SOCKET,        // connect fail
    CE_DISCONNECTED,
    CE_VERSION_DISMATCH,
    CE_PROTOCOL,
    CE_TIMEOUT,
    CE_INVALID_USER, // server return error
    CE_INVLIAD_PASS,
    CE_MAX_DEVICE,
    CE_NO_GATEWAY = 10,
    CE_NO_NETWORK,
    CE_DEVICE_NOTEXISTS,
    CE_DEVICE_ONLINED,
    CE_DEVICE_DISABLED,
    CE_USER_UNACTIVE,
    CE_USER_LOCKED,
    CE_INVALID_TUN_DEVICE, // end of server error
    CE_CANCEL,
    CE_REDIRECT,
    CE_AUX_AUTH_DISMATCH = 20,
    CE_INVALID_PARTNERID,
```

```
    CE_INVALID_APPID,
    CE_PENDING,
    CE_STATUS,
    CE_UNINITIALIZED,
    CE_NETWORK_UNREACHABLE,
    CE_INVALID_AUTHORIZATION,
    CE_UNKNOWN_DEVCODE,
    CE_INVALID_SN,
    CE_NO_SN_SELECTED = 30,
    CE_OPERATION_DENIED,
    CE_MEMERY_OUT,
    CE_INVALID_SMS,
    CE_INVALID_TICKET,
    CE_TRY_TOO_MANY_TIMES,
    CE_INVALID_DEIVE_CLASS,
    CE_CALL_THIRD_API_FAIL,
    CE_INVALID_CODE,
    CE_MODULE_LOST,
    CE_GW_AUTH_TIMEOUT,
    CE_CORE_TIMEOUT = 1000,
    CE_INVALID_PARAMETER = 1001,
} CN_ERR;
```

2.2 错误码说明

错误码	值	说明
CE_SUCC	0	成功
CE_FAIL	1	一般无法定位具体错误原因时返回该值
CE_SOCKET	2	网络套接字接口操作失败(一般是系统不支持该操作或者进程权限不够导致)
CE_DISCONNECTED	3	TCP 连接中断
CE_VERSION_DISMATCH	4	版本不匹配
CE_PROTOCOL	5	协议错误
CE_TIMEOUT	6	等待超时
CE_INVALID_USER	7	无效的用户名
CE_INVLIAD_PASS	8	无效的密码
CE_MAX_DEVICE	9	虚网网络中设备数量超限

CE_NO_GATEWAY	10	无可用的虚拟网关
CE_NO_NETWORK	11	无可用的虚拟网
CE_DEVICE_NOTEXISTS	12	设备不存在(设备被管理员删除)
CE_DEVICE_ONLINED	13	设备以及在线
CE_DEVICE_DISABLED	14	设备被禁用
CE_USER_UNACTIVE	15	用户未激活
CE_USER_LOCKED	16	用户被锁定
CE_INVALID_TUN_DEVICE	17	虚拟网卡设备不可用
CE_CANCEL	18	用户主动取消操作
CE_REDIRECT	19	连接被重定向
CE_AUX_AUTH_DISMATCH	20	认证模式不匹配
CE_INVALID_PARTNERID	21	无效的合作伙伴 ID
CE_INVALID_APPID	22	无效的 app ID
CE_PENDING	23	正在加载
CE_STATUS	24	错误的状态(该操作不能在这种状态下进行，如登出状态下获取好友设备信息)
CE_UNINITIALIZED	25	SDK 未成功初始化
CE_NETWORK_UNREACHABLE	26	本地网络不可用
CE_INVALID_AUTHORIZATION	27	无效的认证模式
CE_UNKNOWN_DEVCODE	28	无效的设备类型(只有 Weline.io 授权的设备类型才允许登入)
CE_INVALID_SN	29	无效的设备机器码(只有 Weline.io 授权的设备机器码才允许登入)
CE_NO_SN_SELECTED	30	未选择超级节点(部分操作需要选择超级节点才能进行)
CE_OPERATION_DENIED	31	操作被拒绝(执行了 Weline.io 未授权的操作)
CE_MEMERY_OUT	32	内存不够，操作失败
CE_INVALID_SMS	33	短信口令失效，请重新申请
CE_INVALID_TICKET	34	无效的 ticket
CE_TRY_TOO_MANY_TIMES	35	尝试次数过于频繁

CE_INVALID_DEIVE_CLASS	36	无效的设备类型
CE_CALL_THIRD_API_FAIL	37	与第三方接口对接失败(PC 端和移动端使用)
CE_INVALID_CODE	38	无效的第三方 code（PC 端和移动端使用）
CE_MODULE_LOST	39	模块丢失（内部错误，一般不对外)
CE_CORE_TIMEOUT	1000	连接服务进程超时，可能服务进程未被正常启动
CE_INVALID_PARAMETER	1001	参数错误，如:传入的 JSON 参数格式不正确

3 接口说明

3.1 初始化命令

- 接口定义

```
-(NSUInteger) initWithPartnerId:(NSString *)PartnerId
                          AppId:(NSString *)AppId
                    deviceClass:(NSUInteger)deviceClass
                     dns_divert:(BOOL)dns_divert;
```

- 参数说明：

@PartnerId 企业或者伙伴 ID，由 Weline.io 授权分配
@AppId App ID，由 Weline.io 授权分配
@deviceClass 设备类型代码，由 Weline.io 授权分配
@dns_divert 是否开始虚拟域名解析功能，TRUE 开启，FALSE 不开启，如无虚拟域名解析需求建议不开启

- 返回值说明 值 | 说明 —|—| CE_SUCC | 返回此值表示初始化成功 CE_CORE_TIMEOUT | 返回此值表示 sdvnd 进程未正常启动，请 APP 再次启动服务 CE_INVALID_PARAMETER | 返回此值表示传入了错误的参数

- **APP 启动 sdvnd 服务的代码示例**

```
-(void)launchctlload:(NSString *)path{
    NSAppleEventDescriptor *eventDescriptor = nil;
      NSAppleScript *script = nil;
        NSString *scriptSource = [NSString stringWithFormat:@"do shel
```

```
l script \"launchctl load -w %@\" with administrator privileges",pat
h];
      if (scriptSource)
      {
          script = [[NSAppleScript alloc] initWithSource:scriptSourc
e];
          if (script)
          {
              eventDescriptor = [script executeAndReturnError:nil];
              if (eventDescriptor)
              {
                  NSLog(@"%@", [eventDescriptor stringValue]);
              }
          }
      }
}
```

3.2 获取虚拟网络登录状态信息

- 接口定义

```
-(NSString *) getStatus;
```

- 返回字段说明

```
{
    "result": CE_SUCC,   //返回结果
    "status": CCS_XXX,  // 当前登录状态，定义见 CM_CONNECTION_STAT
    "disconnect_reason": xxx,   // 连接断开原因 (CCS_DISCONNECTED 和 CC
S_WAIT_RECONNECTING 状态有值)，定义见 DISCONNECT_REASON
    "pre_disconnect_reason": xxx, // 前一次认证失败的原因 (非 CS_CONNEC
TED 状态有值)
    "pre_auth_step": xxx,         // 前一次认证失败的故障点 (非 CS_CONNE
CTED 状态有值)
    "pre_auth_code": xxx,         // 前一次认证失败返回的错误代码 (非 CS_
CONNECTED 状态有值)
    "data": {
        "duration": xx,       // seconds from established(登录时长)
        "network_status": {   // if CCS_CONNECTED
            "rx_bytes": "xxxx", // uint64 converted to string
            "rx_speed": xxx,    // uint rx speed (bytes)
            "tx_bytes": "xxxx", // uint64 converted to string
            "tx_speed": xxx,    // uint tx speed (bytes)
            "dlt_rx_bytes": "xxxx", // uint64 converted to string
```

```
            "dlt_rx_speed": xxx,    // uint rx speed (bytes)
            "dlt_tx_bytes": "xxxx", // uint64 converted to string
            "dlt_tx_speed": xxx,    // uint tx speed (bytes)
            "latency" : xxx     // 网络延时 单位 ms
        }
    }
}
```

- 登录状态

```
typedef enum
{
    CCS_UNKNOWN = 0, //未知
    CCS_PREPARE,     //已经就绪（内部初始化使用）
    CCS_CONNECTING,  //正在登入
    CCS_CONNECTED,   //登入状态，虚网隧道已激活
    CCS_DISCONNECTED,//登出状态，虚网隧道未激活
    CCS_AUTHTICATED, //认证完成，虚网隧道未激活
    CCS_UNACTIVE, // 设备未激活，用于非终端类型设备
    CCS_ACTIVING, // 设备激活中
    CCS_DISCONNECTING,  // 正在断开连接
    CCS_WAIT_RECONNECTING, //正在等待重连
} CM_CONNECTION_STAT;
```

- 登录状态说明 登录状态 | 值| 状态说明 —|—|—| CCS_UNKNOWN | 0 | 未知状态 CCS_PREPARE
- | 1 |程序内部初始化使用 CCS_CONNECTING
- | 2 | 正在登入，有两种情况会导致进入该状态: 1.主动调用 Build Tunnle；2.断线重连 CCS_CONNECTED
- | 3 |登入成功，该状态下虚拟隧道已激活，可以获取到所有与虚网相关信息和状态信息 CCS_DISCONNECTED
- | 4 |登出成功，该状态下虚拟隧道断开，虚网不可用，只能获取少量信息(如:历史账号等) CCS_AUTHTICATED
- | 5 |认证完成，该状态为 CCS_CONNECTED 的过渡状态，该状态下 Weline.io 账号密码身份认证通过，但是虚拟隧道可能未激活，网络列表和好友设备列表可能未下发完成，该状态下可以获取部分虚网相关的基本信息(如:ticket，vip，domain，name 等)，对于移动端设备（iOS 和 Android）上层收到该状态可以开始配置 VPN CCS_UNACTIVE
- | 6 |账号未激活，该状态表明未配置任何可用的 Weline.io 账号。Weline.io 账号配置有两种途径:1.用户主动输入账号密码(主要用于 PC 端

和移动端)；2.特殊设备通过设备 SN 自动注册（该状态主要为特殊设备而设计） CCS_ACTIVING

- | 7 |账号激活中，该状态表明特殊设备正在自动注册账号，如果长期处于该状态：1.请先检查设备是否有可用互联网；2.检查设备 SN 是否在 Weline.io 授权范围内（该状态主要为特殊设备而设计） CCS_DISCONNECTING
- | 8 |正在登出，该状态表明正在销毁虚拟隧道，释放相关资源，还原 DNS，路由表等信息 CCS_WAIT_RECONNECTING
- | 9 |等待重连，断线重连(如:网络中断，切换网络环境等)会导致进入该状态，该状态为 CCS_CONNECTING 的过渡状态，如果不关心可以等效为 CCS_CONNECTING
- 设备处于**登出**状态的原因定义

```
typedef enum
{
    DR_UNSET = 0,
    DR_BY_USER,
    DR_MISVERSION,
    DR_NETWORK,
    DR_MISSING_INFO,
    DR_INVALID_USER,
    DR_INVALID_PASS,
    DR_DEVICE_DELETED,
    DR_DEVICE_ONLINE,
    DR_DEVICE_DISABLED,
    DR_MAX_DEVICE = 10,
    DR_NO_NETWORK,
    DR_KO_USER_REMOVED,
    DR_KO_DEVICE_REMOVED,
    DR_KO_DEVICE_DELETED,
    DR_TUN_DEVICE,
    DR_DTYPE_CHANGED,
    DR_NETWORK_UNUSABLE,
    DR_USER_RELOGIN,
    DR_ALGO_TYPE_CHANGED,
    DR_AUX_AUTH_DISMATCH = 20,
    DR_INVALID_AUTHORIZATION,
    DR_PROTOCOL,
    DR_NETWORK_BROKEN,
    DR_KICKOUT_BY_AS,
```

```
    DR_INVALID_SMS,
    DR_INVALID_TICKET,
    DR_TRY_TOO_MANY_TIMES,
    DR_INVALID_DEVICE_CLASS,
    DR_CALL_THIRD_API_FAIL,
    DR_INVALID_CODE = 30,
    DR_WAKE_UP,
} DISCONNECT_REASON;
```

- 设备处于登出状态的原因说明 名称 |值| 描述 |隧道行为| —|—|—|—| DR_BY_USER
- |1| 1.通过 Logout 接口登出 2.设备刚刚开机，但未设置自动登入|停止自动重连 DR_MISVERSION
- |2|版本太低(被限制登录)|停止自动重连 DR_NETWORK_TIMEOUT
- |3| 与后台服务器的网络连接超时|自动重连 DR_MISSING_INFO
- |4|未配置过任何账号信息或者设备账号信息丢失|停止自动重连 DR_INVALID_USER
- |5|无效的用户名|停止自动重连 DR_INVALID_PASS
- |6|无效的密码，或者口令|停止自动重连 DR_DEVICE_DELETED
- |7|设备被管理端删除|停止自动重连 DR_DEVICE_ONLINE
- |8|已有一台该 SN 的设备在线|停止自动重连 DR_DEVICE_DISABLED
- |9| 设备被限制登入|停止自动重连 DR_MAX_DEVICE
- |10| 设备数量超限|停止自动重连 DR_NO_NETWORK
- |11| 无可用的虚拟网络（一般不会发生）|停止自动重连 DR_KO_USER_REMOVED
- |12| 用户被某个虚拟网络管理者踢出|停止自动重连 DR_KO_DEVICE_REMOVED
- |13| 设备被某个虚拟网络管理者踢出|停止自动重连 DR_KO_DEVICE_DELETED
- |14| 设备被某个虚拟网络管理者删除|停止自动重连 DR_TUN_DEVICE
- |15| 系统无可用 tun 设备，或者 tun0 设备其他程序占用(限 Linux 平台) |自动重连 DR_DTYPE_CHANGED

- |16| 设备被升级为超级节点(限 Linux 平台) |自动重连 DR_NETWORK_UNUSABLE
- |17| 无可用互联网（如设备网线被拔） |自动重连 DR_USER_RELOGIN
- |18| 用户执行了重新登入 |自动重连 DR_ALGO_TYPE_CHANGED
- |19| 隧道算法被更改|自动重连 DR_AUX_AUTH_DISMATCH
- |20| 认证模式不匹配|停止自动重连 DR_INVALID_AUTHORIZATION
- |21| 无效的授权|停止自动重连 DR_PROTOCOL
- |22| 协议错误|停止自动重连 DR_NETWORK_BROKEN
- |23| 网络中断，如切换 WIFI 等|停止自动重连 DR_KICKOUT_BY_AS
- |24| 设备被后台限制登录|停止自动重连 DR_INVALID_SMS
- |25| 短信口令已过期，需要重新申请短信口令|停止自动重连，并要求用户重新申请并输入短信口令 DR_INVALID_TICKET
- |26| 无效的 ticket|停止自动重连，并要求用户重新申请并输入短信口令 DR_TRY_TOO_MANY_TIMES
- |27| 登录尝试次数过于频繁|停止自动重连 DR_INVALID_DEVICE_CLASS
- |28|无效的设备类型|停止自动重连 DR_CALL_THIRD_API_FAIL
- |29|与第三方接口对接失败|停止自动重连 DR_INVALID_CODE
- |30|无效的第三方 code|停止自动重连 DR_WAKE_UP|31|系统休眠，导致网络中断，从休眠唤醒后将自动重连|自动重连
- 说明

1.此接口一般用于登入成功之后获取虚拟隧道延时、速度、流量等信息用

2.虚拟网络登录状态也可以使用下面的 getBaseInfo 接口获取

3.status 字段的值为 CONNECTION_STAT 枚举类型中的值，设备登入、登出、取消登入等操作是否成功都需要依据该字段的值来判断，而状态发生变化会通过异步事件通知

4.reason 字段只在 status 为 CCS_DISCONNECTED 时有效

3.3 获取基本信息

- 接口定义

```
-(NSString *) getBaseInfo;
```

- 返回字段：

```
{
    "result": CE_SUCC,       //返回结果
    "users": [              // 曾经成功登录过的账号列表（所有状态有值）
        { "account": "xxx" }  // 账号
    ],
    "information": {
        "version": "x.x.x.x",  // 当前版本（所有状态有值）
        "status": CCS_XXX,  // 当前状态(所有状态有值)，参见上面的 CM_CONNECTION_STAT 定义
        "disconnect_reason": xxx,   // 连接断开原因(CCS_DISCONNECTED 状态有值)，参见上面的 DISCONNECT_REASON 定义
        "pre_disconnect_reason": xxx, // 前一次认证失败的原因（非 CS_CONNECTED 状态有值)
        "pre_auth_step": xxx,         // 前一次认证失败的故障点（非 CS_CONNECTED 状态有值)
        "pre_auth_code": xxx,         // 前一次认证失败返回的错误代码（非 CS_CONNECTED 状态有值)
        "account" : "xxx",  // 最后一次成功登录的账号（所有状态有值，为登录过任何账号时为空)
        "uid": "xxx",       // 账号对应的用户唯一编号（CCS_AUTHTICATED 和 CCS_CONNECTED 状态有值)
        "device_id": "xxx",  // 当前设备的唯一编号（成功登录过后，所有状态有值)
        "domain" : "xxx",    // 当前设备的虚拟域名（CCS_AUTHTICATED 和 CCS_CONNECTED 状态有值)
        "ad_domain"; "xxx",  // 当前设备的虚拟 AD 域名（CCS_AUTHTICATED 和 CCS_CONNECTED 状态有值)
        "interface": "xxx",  // 网卡名称(CCS_AUTHTICATED 和 CCS_CONNECTED 状态有值)
        "vip" : "x.x.x.x",    // 当前的虚拟 IP（CCS_AUTHTICATED 和 CCS_CONNECTED 状态有值)
        "vmask" : "x.x.x.x",  // 虚拟掩码（CCS_AUTHTICATED 和 CCS_CONNECTED 状态有值)
        "priv_ip": "x.x.x.x",  // 设备的内网地址（CCS_AUTHTICATED 和 CCS_CONNECTED 状态有值)
        "priv_port": xxx,      // 内网私有端口  (CCS_AUTHTICATED 和 CCS_CONNECTED 状态有值)
        "timestamp" : xxx,  // 建立连接时的时间戳（CCS_AUTHTICATED 和 CCS_CONNECTED 状态有值)
        "ticket" : "xxx",   // 当前登录的票据（CCS_AUTHTICATED 和 CCS_CONNECTED 状态有值)
```

```
        "netid" : "xxx",    // 当前所在网络的 Id (CCS_CONNECTED 状态有值)
        "snid" : "xxx",       //当前选择的 SN Id (CCS_CONNECTED 状态有值)
        "as_host" : "xxxx"   //AS 域名(所有状态有值)
        "name": "xxxxx"      //设备名称
    },
    "tunnel": {        //虚拟网卡配置
        "interface": "xxxx", // CCS_CONNECTED 状态有值
        "mtu": xxx, // 所有状态有值
    },
    "option" : {
        "autologin" : true | false, // 程序启动时，自动登录 (所有状态有值)
        "dlt": true | false,  // DLT 设置 (所有状态有值)
        "st": true | false, // ST 设置 (所有状态有值)
        "snmode": true | false  // 设备当前是否为 SN (CCS_AUTHTICATED 和 CCS_CONNECTED 状态有值)
        "vip_feature": xxx, // VIP_FEATURE 组合值(CCS_AUTHTICATED 和 CCS_CONNECTED 状态有值)
        "algo_level": xxx,  // 算法等级设置 (所有状态有值)
        "partner_feature": xxx, // 内部使用
    }
    "api_gw_info": {  //应用网关域名(CCS_AUTHTICATED 和 CCS_CONNECTED 状态有值)
        "domain": "xxxx"
    }
}
```

- VIP_FEATURE 组合（会员用户特殊权限）

```
typedef enum
{
    VIP_FEATURE_DLT = 0x1,  // 能否关闭 DLT
    VIP_FEATURE_ST = 0x2,  // 能否关闭 sT
    VIP_FEATURE_CHANGE_ALGO = 0x4,  // 能否更改算法
    VIP_FEATURE_GW_BCAST = 0x8,  // 内部使用
    VIP_FEATURE_GW_MCAST = 0x10,  // 内部使用
} VIP_FEATURE;
```

- algo_level 算法等级列表如下 level | label | algo+hmac —|—|— 1 | No Encryption | NULL+NULL 2 | RC4 + MD5-HMAC | rc4 + hmac-md5 3 | AES256 + SHA1-HMAC | aes256 + hmac-sha1

- 说明

1.users 会包含该设备所以成功使用过的历史账号(被删除过的除外)，对于只有一个账号的设备不用理会该字段

2.account 是最近一次登录过的账号，如果 autologin 字段值为 1，sdvnd 进程启动就会使用此账号自动登入

3.domain、vip、timestamp、netid、snid 等这些字段只在登录状态为 CCS_CONNECTED 时有效

4.建议在调用初始化接口以及登入成功之后调用此接口来获取获取虚网基本信息

3.4 取消登入过程

登入过程耗时受网络环境、DNS 解析速度等因素影响，所以时间长短不定，如果耗时过长，可以选择取消

- 接口定义

```
-(NSUInteger) cancelLogon;
```

- 返回值说明 值 | 说明 —|—| CE_SUCC | 返回此值表示当前状态已经为 CCS_DISCONNECTED CE_STATUS | 未达到预期状态，在终止之前状态已经转换为 CCS_CONNECTED CE_PENDING | 表示正在取消登入，需要通过异步事件来获取状态变化，状态变化过程为 CCS_CONNECTING -> CCS_DISCONNECTED

3.5 创建虚网隧道

- 接口定义

```
-(NSUInteger) logonWithAccount:(NSString *)account Password:(NSStrin
g *)password AuthMethod:(int)auth_method;
```

- 参数说明

```
@account        Weline.io 账号
@password       Weline.io 账号密码
@auth_method    认证类型，参见 AUTH_METHOD 定义
```

- 认证类型定义

```
typedef enum
{
    AUTH_METHOD_NONE = 0x0,
    AUTH_METHOD_ACCPWD = 0x1,    //静态密码认证
    AUTH_METHOD_SMS = 0x2,       //短信口令认证
    AUTH_METHOD_RECONNECT_BY_TICKET = 0x3,  //令牌认证
    AUTH_METHOD_THIRD_CODE = 0x4, //Mac OSX 平台未使用
```

```
    AUTH_METHOD_PHONE_TOKEN = 0x65, //Mac OSX 平台未使用
} AUTH_METHOD;
```

- 返回值说明 值 | 说明 —|—| CE_SUCC | 返回此值表示当前状态已经为 CCS_CONNECTED CE_STATUS | 无法建立隧道，设备可能处于 CCS_UNACTIVE 或者 CCS_ACTIVING 状态 CE_PENDING | 表示正在登入，可以通过异步事件来获取状态变化，状态变化过程可能为: 1. 登入成功 ： CCS_CONNECTING -> CCS_AUTHENTICATED -> CCS_CONNECTED2. 登入失败 ： CCS_CONNECTING -> CCS_DISCONNECTED 注 ： 登入失败，可查看事件中 dr 字段获取失败原因
- 其他说明

由于短信口令密码是一次性的，所以新增 AUTH_METHOD 方式为 3 的令牌认证。

用户首次使用短信口令登录成功后，后台会下发一个临时令牌，在令牌有效期内客户端可以使用 AUTH_METHOD 方式为 3 的令牌方式进行免密登录，登录成功后会更新令牌，如果长时间不登录，令牌过期(DR_INVALID_TICKET)将需要重新输入短信口令登录。

3.6 获取网络信息

获取设备所有网络信息及当前所在网络的 id

- 接口定义

```
-(NSString *) getNetworks;
```

- 返回字段说明：

```
{
    "result" : CE_SUCC,  //返回值
    "netid": "xxx",      //当前的网络 ID
    "data" : [
        {
          "id" : "xxx",    //网络 ID
          "name" : "xxx",  //网络名称
          "owner" : "xxx"  //网络属主帐号
        }
    ]
}
```

- 返回值 值 | 说明 —|—| CE_SUCC | 成功，返回的 Json 串中包含网络列表数据 CE_STATUS | 当前状态不是 CCS_CONNECTED

3.7 获取好友设备信息

获取设备当前所在网络的其他在线好友设备的信息

- 接口定义

```
-(NSString *) getDevices;
```

- 返回字段：

```
{
    "result": CE_SUCC,   //返回结果
    "devices": [  // 设备列表
     {
        "id": "xx", // Id
        "owner": "xxx", // 设备属主帐号
        "userid": "xxx", // User Id
        "domain": "xxx", // 设备域名
        "ad_domain": "xxx", // 设备 AD 域名
        "name": "xx", // 设备名称
        "device_class": xxx, // 设备类型
        "vip": "x.x.x.x",
        "pubip": "x.x.x.x",
        "ver": "x.x.x.x",
        "feature": xx,
        "admin_userid": "xx", // 设备管理者 id
        "admin_token": "xx", // 设备管理者 token
        "dlt": {
            "dlt_status": true|false,  //dlt 开关
            "algo_level": xx,          //隧道算法，参见算法定义
            "class": xx,
            "peer_ip": "x.x.x.x",
            "peer_port": xx
        }
     }
    ],
    "smartnodes": [  // 节点列表
     {
        "id": "xx", // 节点 Id
        "owner": "xxx", // 节点属主帐号
        "userid": "xxx", // User Id
        "domain": "xxx",
        "name": "xxx",
        "device_class": xxx, // 设备类型
```

```
        "vip": "x.x.x.x",
        "pubip": "x.x.x.x",
        "privip": "x.x.x.x",
        "ver": "x.x.x.x",
        "feature": xx,
        "admin_userid": "xx", // 设备管理者 id
        "admin_token": "xx", // 设备管理者 token
        "dlt": {
            "class": xx,
            "peer_ip": "x.x.x.x",
            "peer_port": xx
        },
        "subnet": [
        { "net": "x.x.x.x", "mask": "x.x.x.x" }
        ]
    }
   ]
}
```

- 返回值 值 | 说明 —|—| CE_SUCC | 成功，返回的 Json 串中包含设备列表数据 CE_STATUS | 当前状态不是 CCS_CONNECTED

- 设备种类

```
enum device_type
{
    DT_UNKOWN = 0,
    DT_DEVICE = 1,     //普通设备
    DT_SMARTNODE = 50, //超级节点设备
};
```

- 设备属性

```
typedef enum _DEVICE_FEATURE
{
    DF_ADCONTROLLER = 1,    //AD 域控制器
    DF_ACCESS_INTERNET = 2, //访问互联网
    DF_ACCESS_SUBNET = 4,   //访问子网
    DF_HIDDEN_DEVICE = 8    //设备不可见
    DF_CHECK_OWNER = 16    //设备控制时校验管理者令牌
} DEVICE_FEATURE;
```

- DLT 类型

```
typedef enum _CM_DLT_CLASS
{
```

```
    SC_NONE = 0x0, //00 no dlt
    SC_TX = 0x1,   //01 unilateral dlt just for tx
    SC_RX = 0x2,   //10 unilateral dlt just for rx
    SC_BOTH = 0x3,  //11 bilateral dlt
} CM_DLT_CLASS;
```

3.8 登出操作

- 接口定义

```
-(NSUInteger) Logout;
```

- 返回值 值 | 说明 —|—| CE_SUCC | 成功，返回此值表示当前状态已经为 CCS_DISCONNECTED CE_PENDING | 表示正在登出，可以通过异步事件来获取状态变化，状态变化过程为: CCS_DISCONNECTING -> CCS_DISCONNECTED

3.9 选择超级节点

超级节点具有 IP 层网络转发功能，设备如果选择了超级节点，那么该设备的 DNS 解析和上网流量可以通过节点来转发

- 接口定义

```
-(NSUInteger) selectSNode:(NSString *) Id;
```

- 参数说明

@Id 好友设备 Id，可以通过 getDevices 接口获取到当前虚拟网络中的所以设备列表

- 返回值 值 | 说明 —|—| CE_SUCC | 成功，网络流量导向该节点 CE_STATUS | 当前状态不是 CCS_CONNECTED
- 说明:

1. 该操作只有在登录状态为 CCS_CONNECTED 时有效
2. 智能节点 id 信息包含在好友设备信息里面，可以通过 UC_GET_DEVICES 获取
3. 选择该超级节点进行上网时，节点必须在线
4. 选择之后会将选择的超级节点 ID 保存到配置文件
5. 操作时 id 如果为空，这表示为取消选择，如果选个多个节点，节点 id 用"," 分隔

3.10切换网络

设备有多个网络，每个网络都有不同的好友设备供其访问，设备可以自行选择所需要的网络

- 接口定义

```
-(NSUInteger) switchNetwork:(NSString *) Id;
```

- 参数说明

@Id　　虚拟网络 ID，可以通过 getNetworks 接口获取所以网络列表

- 说明:

1.该操作只有在登录状态为 CCS_CONNECTED 时有效
2.设备第一次登入虚网会在其所属账号的默认网络中

- 返回值 值 | 说明 —|—| CE_PENDING | 正在切换网络，通过异步事件来通知网络变化 CE_NO_NETWORK | 该 id 所表示的网络不存在 CE_DISCONNECTED | 本地网络中断

3.11删除历史账号

- 接口定义

```
-(NSUInteger) removeUser:(NSString *) account;
```

- 返回字段

@account　　某个历史账号，可以通过 getBaseInfo 接口获取到所有登入过的历史账号列表

- 返回值 值 | 说明 —|—| CE_SUCC | 操作成功 CE_FAIL | 操作失败
- 说明:

1.该操作只有在登录状态为 CCS_DISCONNECTED 时有效
2.删除某个账号后，与这个账号相关的所以本地配置信息都会删除(如:ST 开关状态，DLT 开关，选择的超级节点 ID 等)

3.12DLT 设置

- 接口定义

```
-(NSUInteger) setDLT:(BOOL)enable;
```

- 参数说明

@enable　　true 为打开 DLT，false 为关闭 DLT，DLT 设置默认是开启状态

- 返回值 值 | 说明 —|—| CE_SUCC | 操作成功 CE_OPERATION_DENIED | 操作被拒绝，无权限执行此操作

- 说明:

1.DLT 当前的设置状态可以在基本信息里面拿到
2.DLT 模块用于帮助设置之间进行点对点的通讯，以便提升虚拟网隧道传输速度，DLT 设置默认为 TRUE

4 获取事件

- 接口定义

```
-(NSString *) readEvents:(int)timeout;
```

- 说明

1.每次调用都会从消息队列中取出一个消息，如果消息队列为空，则返回 nil
2.消息队列最大消息容量为 100，如果超出会丢掉最先进入的消息

4.1 事件的 json 格式

- 事件定义

```
typedef enum {
    EC_STATUS_CHANGE = 1,      // 虚拟网登录状态状态变化
    EC_DEVICE_CHANGE = 2,      // 好友设备信息变化
    EC_SWITCHNET_COMPLETE = 3, // 网络切换完成事件
    EC_UPDATE = 4,             // 版本更新事件
    EC_NETWORK_CHANGE = 5,     // 虚拟网网络变化
    EC_DEVICE_REGISTE = 6,     // 设备激活事件
    EC_ROUTE_CHANGE = 7,       //for Android and iOS 虚拟网路由变化
    EC_LAST
} CM_EVENT_CLASS;
```

- 通用格式

```
{
    "event": xx,  // 事件类型
    "timestamp": {  // 事件时间戳
        "seconds": xx,
        "useconds": xx
    },
     "message": { // 事件消息
         ...
```

```
    }
}
```

4.1.1 虚拟网登录状态变化通知

- 消息格式

```
"message": {
    "from": xx,
    "to": xx
    "dr": xx
}
```

- 说明

1. from 为前一个状态，to 为变化后的状态，值为 CONNECTION_STAT 类型
2. dr 为 DISCONNECT_REASON 类型，只有 to 为 CCS_DISCONNECTED 时有值

- 几种关键的状态变化说明 | from 值 | to 值 | 说明 | |–|–|–| |CS_DISCONNECTED 4 |CS_CONNECTING 2 |开始登入| |CS_CONNECTING 2 |CS_AUTHENTICATED 5 |身份认证完成| |CS_AUTHENTICATED 5 |CS_CONNECTED 3 |登入成功，虚网隧道建立完成| |CS_CONNECTED 3 |CS_DISCONNECTING 8 |开始登出| |CS_DISCONNECTING 8 |CS_DISCONNECTED 4 |登出完成| |CS_DISCONNECTING 8 |CS_WAIT_RECONNECTING 9 |等待自动重连(一般真实网络中断导致虚拟隧道断开会进行自动重连) |CS_WAIT_RECONNECTING 9 |CS_CONNECTING 2 |开始自动重连| |CS_CONNECTING 3 |CS_WAIT_RECONNECTING 9 |登入失败，等待一定时间再重试(可能由于网络原因导致无法成功登入)|

4.1.2 好友设备状态信息变化通知

- 消息格式

```
"message": {
    "status": "online" | "offline | DLT"
    "device_type": DT_DEVICE |DT_SMARTNODE | DT_GATEWAY
    "id": "xx", // Id
    "owner": "xxx", // 设备属主帐号, smartnode or device
    "userid": "xxx", // User Id, smartnode or device
    "domain": "xxx", // 设备域名
    "ad_domain": "xxx", // 设备 AD 域名, device
    "name": "xx", // 设备名称
    "device_class": xxx, // 设备类型, smartnode or device
    "vip": "x.x.x.x",
    "pubip": "x.x.x.x", // smartnode or device
```

```
    "privip": "x.x.x.x", // smartnode or device
    "ver": "x.x.x.x" // smartnode or device
    "feature": xx,   //属性值
    "admin_userid": "xx", // 设备管理者 id
    "admin_token": "xx", // 设备管理者 token
    "dlt": {  // DLT 信息
        "class": xx,  // CM_DLT_CLASS
        "peer_ip": "x.x.x.x",
        "peer_port": xx
    },
    "subnet": [  // smartnode or gateway
        { "net": "x.x.x.x", "mask": "x.x.x.x" }
    ]
}
```

- 说明

1. 该事件只会在登录状态为 CCS_CONNECTED 时发生
2. 收到该消息通知后也可以通过 getDevices 接口来获取最新的好友设备列表
3. 网络变化时，所有好友设备信息都被清空，所以 DLT 隧道都被销毁

4.1.3 虚拟网网络变化

- 消息格式 无消息本体，只是一个通知消息
- 说明

1. 该消息通知只会在登录状态为 CCS_CONNECTED 时发生
2. 网络变化之后，可以通过 getNetworks 接口来获取新的网络列表
3. 网络变化之后，可以通过 getDevices 接口来获取新的好友设备列表

4.1.4 虚拟网网络切换完成通知

- 消息格式

```
"message": {
    "result": x,  // 为 0 时表示切换成功，否则为错误代码
    "id": "xxx"  // network id
}
```