goolink
cloud communication

GooLink API v1.5

GooLink API 是浪涛互动开发的设备端 SDK，用于接入 GooLink 云服务；采用调用接口和使用回调函数的方式实现控制信息和音视频流的交互。

API 接口函数：

glnk_get_version
glnk_destroy
glnk_push_alarm
glnk_get_state_to_server
glnk_init
glnk_sendvideodata
glnk_sendaudiodata

unsigned long glnk_get_version()
说明：
－获取 GooLink
SDK 版本号
返回值：
－GooLink
SDK 版本号，例如版本 1.0.0，用 16 进制
表示为 0x100

intglnk_init(InitNetParam* netparam);
说明：初始化 GooLink
SDK
－netparam 基本功能参数设置；
－upnpparamsupnp 映射设置；
－upnpparam_sizeupnp 配套数
返回值：0 初始化成功，非 0 为失败

```
typedefstructInitNetParam
{
    unsigned char       dev[8];   // 本地网络名称
    unsigned short      localTCPport;   // 本地传输 TCP 端口
    unsigned short      localUDPport;   // 本地传输 UDP 端口
    unsigned char       udid[32];        //   GooLink 全局 ID
    unsigned char channelnum;      // 视频通道数 1-64
```

```
    unsigned char       issupportsubvideostream;    // 是否支持视频子码流
    unsigned char maxsession;       // 可以支持的最大连接数 1-16
}InitNetParam;

typedefstructUpnpParam
{
    char       protocol[8];              // 映射的协议类型
    char       description[48];          // 描述
    unsigned short      intern_port; // 内网端口
    unsigned short      extern_port; // 外网端口
}UpnpParam;
```

intglnk_destroy()
说明：释放 GooLink SDK 资源，反初始化

voidglnk_push_alarm(PushAlarm alarm)
说明：推送离线告警
- alarm，告警信息

```
typedefstruct_PushAlarm
{
    int16_t alarm_type; //告警类型
    int16_t channel; //告警通道号
    GooTime @mestamp;//告警时间
}PushAlarm;
```

-目前支持的告警类型有

```
typedefenum _PushAlarmType
{
    PAT_VIDEO_FRAME       = 0,          // Video frame detection
    PAT_DEVICE_RESTART = 1,          // Device restart
    PAT_MOTION_DETECT    = 2,          // Motion Detection
    PAT_VIDEO_LOSS        = 3,          // Video Loss
    PAT_DISK_FULL         = 4,          // Disk full
    PAT_BLIND_DETECT     = 5,          // Blind detection
    PAT_SD_ERROR          = 6,          // SD card error
    PAT_ADDR_CONFLICT    = 7,          // Address confliction
    PAT_INFRARED          = 8,          // Infrared alarm
    PAT_VIDEO_ALARM       = 9,          // Video alarm
    PAT_AUDIO_ALARM       = 10,         // Audio alarm
    PAT_TEMPERATURE       = 11,         // Temperature alarm
    PAT_FUME              = 12,         // Fume sensor alarm
    PAT_INVASION          = 13,         // Invasion alarm
```

```
    PAT_CALLING_ALARM    = 14,    // USER call Alarm
    PAT_BREAKDOOR_ALARM= 15,              // Door break alarm
    PAT_IMAGEMOVE_ALARM =    16, //影像移動报警
    PAT_SOUND_ALARM = 17, //聲音报警
    PAT_DEVICEMOVE_ALARM = 18, //機器移動报警
    PAT_ENERGYREMOVE_ALARM = 19, //電源拔除报警
    PAT_SDCARDREMOVE_ALARM = 20, //SDCARD 拔除觸發
    PAT_SDCARDFULL_ALARM = 21, //SDCARD 容量已滿觸發
    PAT_IO01_ALARM = 22,    //I /O 报警
    PAT_IO02_ALARM = 23, // I/O 1 报警
    PAT_IO03_ALARM =24,// I/O 2 报警
    PAT_IO04_ALARM =25, // I/O 3 报警
    PAT_IO05_ALARM =26,    //I/O 4 报警
    PAT_IO06_ALARM =27, // I/O 5 报警
    PAT_IO07_ALARM =28, //    I/O 6 报警
    PAT_IO08_ALARM =29,    // I/O 7 报警
    PAT_IO09_ALARM =30,    //I/O 8 报警
    PAT_IO10_ALARM =31,    //    I/O 9 报警
    PAT_IO11_ALARM =32,    //    I/O 10 报警
    PAT_IO12_ALARM =33,    //    I/O 11 报警
    PAT_IO13_ALARM =34,    //    I/O 12 报警
    PAT_IO14_ALARM =35,    //    I/O 13 报警
    PAT_IO15_ALARM =36,    //    I/O 14 报警
    PAT_IO16_ALARM =37,    //    I/O 15 报警
    PAT_IO17_ALARM =38    //    I/O 16 报警
} PushAlarmType;

intglnk_get_state_to_server()
? 说明：查询和 Goolink 服务器连接状态
? 参数：
返回值：
返回连接服务器状态：5 为连接成功非 5 为没有连接上服务器


intglnk_sendvideodata( unsigned char    channel,
    unsigned char ismainorsub,
    charisIFrame,
    void* videoData,
    unsignedintvideoDataLen);
说明：将视频数据进行封装
参数：
- channel：视频通道（IPC 默认为 0）
- ismainorsub,主或子码流  0 为主码流，1 为次码流
```

- isIFrame：是否为 I 帧 0 为 p 帧，1 为 I 帧
- videoData：原始 H264 数据
- videoDataLen：原始 H264 数据长度

返回值：
成功放入缓冲的数据长度

intglnk_sendaudiodata(unsigned char channel,
    void* audioData,
    unsigned intaudioDataLen)；
说明：将音频数据进行封装
参数：
- channel：音频通道，IPC 默认为 0
- audioData：音频数据
- audioDataLen：音频数据长度

返回值：
成功放入缓冲的数据长度

回调是 GooLink SDK 主动调用的函数，函数有客户自行编写。
回调函数必须是非阻塞的，如遇到流或者控制命令，必须先复制到另外的缓冲区后再处理。
回调函数同一返回 1 为成功，0 为失败。
回调函数名及格式：

GLNK_PwdAuth_Callback	//用户名密码校验
GLNK_GetDevInfo_Callback	//获取设备消息
GLNK_RTVideoOpen_Callback	//打开实时视频
GLNK_RTVideoClose_Callback	//关闭实时视频
GLNK_PTZOpen_Callback	//打开云台
GLNK_PTZCmd_Callback	//云台控制
GLNK_PTZClose_Callback	//关闭云台
GLNK_AudioEncodeOpen_CallBack	//开启音频编码
GLNK_AudioEncodeClose_CallBack	//关闭音频编码
GLNK_AudioDecodeOpen_CallBack	//
GLNK_AudioDecode_CallBack	//
GLNK_AudioDecodeClose_CallBack	//

booleanGLNK_PwdAuth_Callback(char* username, char* pwd);

说明：用户名密码校验（不可阻塞）
参数：
- username：用户名
- pwd：密码

voidGLNK_GetDevInfo_Callback(GLNK_V_DeviceInfo* devinfo);
说明：获取设备信息（不可阻塞）
参数：
- devinfo : 设备信息

```
typedefstruct _GLNK_V_DeviceInfo
{
    char companyId[GLNKPACKET_STR_LEN_16];      // 公司描述
    char productId[GLNKPACKET_STR_LEN_16];      // 产品描述
    char name[GLNKPACKET_STR_LEN_16];           // 名称
    char softwareVersion[GLNKPACKET_STR_LEN_16]; // 软件版本
    GLNK_Date    manufactureDate;       // 版本日期
    unsigned char channelNum;           // 视频通道数
    unsigned char alarmType;            // 报警类型
    unsigned char reserve1;             // 保留
    unsigned char reserve2;             // 保留
} GLNK_V_DeviceInfo;
```

intGLNK_RTVideoOpen_Callback(unsigned char channel,unsigned char ismainorsub, GLNK_VideoDataFormat* videoinfo);
说明：打开实时视频（不可阻塞）
参数：
- channel：通道号
- ismainorsub：主或子码流
- videoinfo：视频信息

```
typedefstruct _GLNK_VideoDataFormat
{
    unsigned int codec;             //编码方式
    unsigned int bitrate;           //比特率, bps
    unsigned short width;           //图像宽度
    unsigned short height;          //图像高度
    unsigned char framerate;        //帧率, fps
    unsigned char colorDepth;       //should be 24 bits
    unsigned char frameInterval;    //I 帧间隔
    unsigned char reserve;
```

} GLNK_VideoDataFormat;

intGLNK_RTVideoClose_Callback(unsigned char channel, unsigned char ismainorsub);
说明：关闭实时视频（不可阻塞）
参数：
- channel：通道号
- ismainorsub：主或子码流

intGLNK_PTZOpen_Callback(unsigned int channel);
说明：打开云台（不可阻塞）
参数：
- channel：通道号

intGLNK_PTZCmd_Callback(GLNK_PTZControlCmdptzcmd,unsignedint channel, GLNK_ControlArgData* arg);
? 说明：云台控制（不可阻塞）
? 参数：
- ptzcmd：云台命令
- channel：通道号
- arg：额外参数

// ControlArgData
// GLNK_PTZ_MV_STOP ：无
// GLNK_PTZ_ZOOM_DEC ：arg1，步长
// GLNK_PTZ_ZOOM_INC ：arg1，步长
// GLNK_PTZ_FOCUS_INC ：arg1，步长
// GLNK_PTZ_FOCUS_DEC ：arg1，步长
// GLNK_PTZ_MV_UP ：arg1，水平速度; arg2，垂直速度; arg3，步长
// GLNK_PTZ_MV_DOWN ：arg1，水平速度; arg2，垂直速度; arg3，步长
// GLNK_PTZ_MV_LEFT ：arg1，水平速度; arg2，垂直速度; arg3，步长
// GLNK_PTZ_MV_RIGHT ：arg1，水平速度; arg2，垂直速度; arg3，步长
// GLNK_PTZ_IRIS_INC ：arg1，步长
// GLNK_PTZ_IRIS_DEC ：arg1，步长
// GLNK_PTZ_AUTO_CRUISE : arg1, 1 = 开始巡航, 0 = 停止巡航; arg2，水平速度; arg3，垂直速度
// GLNK_PTZ_GOTO_PRESET : arg1，预置点编号
// GLNK_PTZ_SET_PRESET : arg1，预置点编号
// GLNK_PTZ_CLEAR_PRESET : arg1，预置点编号，如果为 0xFFFFFFFF 标识清除全部
// GLNK_PTZ_ACTION_RESET
// GLNK_PTZ_MV_LEFTUP ：arg1，水平速度; arg2，垂直速度; arg3，步长

```
//     GLNK_PTZ_MV_LEFTDOWN   ：arg1，水平速度; arg2，垂直速度; arg3，步长
//     GLNK_PTZ_MV_RIGHTUP    ：arg1，水平速度; arg2，垂直速度; arg3，步长
//     GLNK_PTZ_MV_RIGHTDOWN ：arg1，水平速度; arg2，垂直速度; arg3，步长
//     GLNK_PTZ_CLEAR_TOUR    ：arg1，线路编号
//     GLNK_PTZ_ADD_PRESET_TO_TOUR : arg1，预置点编号; arg2，线路编号
//     GLNK_PTZ_DEL_PRESET_TO_TOUR : arg1，预置点编号; arg2，线路编号

typedefenum _GLNK_PTZControlCmd
{
      GLNK_PTZ_MV_STOP          = 0,      // 停止运动
      GLNK_PTZ_ZOOM_DEC         = 5,
      GLNK_PTZ_ZOOM_INC         = 6,
      GLNK_PTZ_FOCUS_INC        = 7,      //焦距
      GLNK_PTZ_FOCUS_DEC        = 8,
      GLNK_PTZ_MV_UP            = 9,      //向上
      GLNK_PTZ_MV_DOWN          = 10,     //向下
      GLNK_PTZ_MV_LEFT          = 11,     //向左
      GLNK_PTZ_MV_RIGHT         = 12,     //向右
      GLNK_PTZ_IRIS_INC         = 13,     //光圈
      GLNK_PTZ_IRIS_DEC         = 14,     //
      GLNK_PTZ_AUTO_CRUISE      = 15,     //自动巡航
      GLNK_PTZ_GOTO_PRESET      = 16,     //跳转预置位
      GLNK_PTZ_SET_PRESET       = 17,     //设置预置位点
      GLNK_PTZ_CLEAR_PRESET = 18,     //清除预置位点
      GLNK_PTZ_ACTION_RESET = 20,     //PTZ 复位
      GLNK_PTZ_MV_LEFTUP        = 21,
      GLNK_PTZ_MV_LEFTDOWN      = 22,
      GLNK_PTZ_MV_RIGHTUP       = 23,
      GLNK_PTZ_MV_RIGHTDOWN = 24,
      GLNK_PTZ_CLEAR_TOUR       = 25,
      GLNK_PTZ_ADD_PRESET_TO_TOUR    = 26,
      GLNK_PTZ_DEL_PRESET_TO_TOUR    = 27
} GLNK_PTZControlCmd;
```

intGLNK_PTZClose_Callback(unsigned int channel);

说明：关闭云台（不可阻塞）

参数：

- channel：通道号

intGLNK_AudioEncodeOpen_CallBack(unsigned char channel, GLNK_AudioDataFormat*

audioinfo);（不可阻塞）
说明：开启音频编码
参数：
- channel：通道号
- audioinfo：音频信息

```
//音频数据格式
typedefstruct _GLNK_AudioDataFormat
{
    unsigned intsamplesRate;     //每秒采样
    unsigned int bitrate;        //比特率, bps
    unsigned short waveFormat;   //编码格式
    unsigned short channelNumber; //音频通道 1 单通道 2 双通道一般为单通道。
    unsigned short blockAlign;   //块对齐, channelSize * (bitsSample/8)
    unsigned short bitsPerSample; //每采样比特数
    unsigned short frameInterval; //帧间隔, 单位 ms
    unsigned short reserve;

} GLNK_AudioDataFormat;
```

intGLNK_AudioEncodeClose_CallBack(unsigned char channel);
说明：关闭音频编码（不可阻塞）
参数：
- channel：通道号

intGLNK_AudioDecodeOpen_CallBack(unsigned char channel, GLNK_AudioDataFormat *audioinfo);
说明：开启音频解码（不可阻塞）
参数：
- channel：通道号
- audioinfo：递交给设备的音频解码信息

intGLNK_AudioDecode_CallBack(unsigned char channel, char* buffer, unsigned int length);
说明：提交音频数据（不可阻塞）
参数：
- channel：通道号
- buffer：音频数据地址
- length：数据长度

intGLNK_AudioDecodeClose_CallBack(unsigned char channel);
说明：关闭音频解码（不可阻塞）

参数：

- channel：通道号

intGLNK_ResetUsrPsword_Callback(char* username, char* oldpwd, char* newpwd);

说明：重设密码（不可阻塞）

参数：

- Username：用户名
- Oldpwd：旧密码
- Newpwd：新密码

intGLNK_SetRecordConfigure_Callback(GLNK_V_RecordChgRequest *Req);

说明：设置录像参数（不可阻塞）

- Req：录像参数数据结构

```
//开启录像的模式联合体
typedefenum _GLNK_RECORD_CHG_CMD
{
    GLNK_RECORD_CLOSE = 0x00,录像关闭
    GLNK_RECORD_OPEN   = 0x01,录像开启
    GLNK_RECORD_OPEN_AUTO_BY_TIME = 0x02 根据录像时间开启
}GLNK_RECORD_CHG_CMD;

    typedefstruct _GLNK_V_RecordChgRequest
    {
        uint32_t   command;      //开启录像的模式
    GLNK_DateTimestartTime;//自动录像的开始时间
    GLNK_DateTimeendTime;//自动录像的结束时间
    }GLNK_V_RecordChgRequest;
```

intGLNK_SearchWifi_Callback(char **buf);

说明：搜索 wifi 信息（不可阻塞）

参数：

- Buf：搜索放回的数据（wifi 信息）

返回值：

- 成功为 1
- 失败为 0

Wifi 信息数据结构

```
typedefstruct _GLNK_V_WifiInfo
{
    char name[GLNKPACKET_STR_LEN_32];//wifi 的名字
```

```
    char ssid[GLNKPACKET_STR_LEN_32];//wifi 的 ssid
    GLNK_WifiSignalLevel level;//wifi 信号的强度等级
}GLNK_V_WifiInfo;
```

Wifi 强度等级联合体

```
typedefenum _GLNK_WifiSignalLevel
{
    GLNK_Signal_Strong = 0,      //信号强
    GLNK_Signal_Mid,             //信号一般
    GLNK_Signal_Weak             //信号弱
}GLNK_WifiSignalLevel;
```

Int GLNK_WifiConfig_Callback(GLNK_V_WifiConfigRequest *Req);

说明：配置 wifi（不可阻塞）结果已经返回给客户端

参数：

Req：wifi 配置的数据结构（需要配置参数）

Wifi 配置的数据结构

```
typedefstruct _GLNK_V_WifiConfigRequest
{
    char name[GLNKPACKET_STR_LEN_32]; //wifi 的名字
    char ssid[GLNKPACKET_STR_LEN_32];//wifi 的 ssid
    char password[GLNKPACKET_STR_LEN_32];//wifi 的密码
}GLNK_V_WifiConfigRequest;
```

IntGLNK_VideoFileSearch_CallBack(GLNK_V_SearchFileRequest*SearchFileInfo, GLNK_V_FileInfo **ptr, int *size)；

说明：录像文件搜索（可阻塞）

参数：

- SearchFileInfo：录像文件搜索信息
- Ptr：搜索到的录像文件的信息
- Size：ptr 的大小（按字节计算）

录像文件搜索信息数据结构

```
typedefstruct _GLNK_V_SearchFileRequest
{
    uint32_t deviceId;           //设备编号（默认为 0）
    uint32_t  channelMask;       //通道掩码，需要搜索哪些通道的录像，将该位置 1. 0xFFFFFFFF 表示所有通道
    GLNK_DateTimestartTime;   //查询起始时间
    GLNK_DateTimeendTime;   //查询终止时间
    uint32_t    recordTypeMask;   //录像类型掩码  0x01 = 开关量告警录像, 0x02 = 移动侦测录像, 0x04 = 常规录像, 0x08 = 手动录像，  0xFF = 全部录像
```

```
    uint32_t    index;   //文件索引，设为 0
    uint32_t  count;   //返回数据个数，默认为 10
    uint32_t    reserve;   //
} GLNK_V_SearchFileRequest;
```

搜索到的录像文件的信息的数据结构

```
typedefstruct _GLNK_V_FileInfo
{
    char fileName[260];//          文件名，包括路径
    uint32_t          deviceId;// 设备 ID（默认为 0）
    uint32_t          length;// 文件总长度
    uint32_t          frames;//总帧数
    GLNK_DateTime     startTime;// 开始时间
    GLNK_DateTime     endTime;//结束时间
    uint8_t           channel;//   录像通道号
    uint8_t           recordType;//录像类型掩码  0x01 =  开关量告警录像, 0x02 =  移动侦测录像,0x04 =  常规录像, 0x08 =  手动录像
    uint8_t           reserve[2];//        保留
} GLNK_V_FileInfo;
```

打开需要回放的录像文件

```
int32_t          GLNK_RecordOpen_CallBack(char*recordname,          int32_t          *recordfd, GLNK_VideoDataFormat* videoinfo, GLNK_AudioDataFormat* audioinfo);
```

说明：打开需要下载的录像文件（不可阻塞）

参数：

- recordname：录像文件名字
- recordfd：返回的打开的录像文件的描述符
- videoinfo：录像视频数据格式
- audioinfo：录像音频数据格式
-

视频数据格式

```
typedefstruct _GLNK_VideoDataFormat
{
    unsigned int codec;             //编码方式
    unsigned int bitrate;           //比特率, bps
    unsigned short width;               //图像宽度
    unsigned short height;              //图像高度
    unsigned char framerate;            //帧率, fps
    unsigned char colorDepth;           //should be 24 bits
    unsigned char frameInterval;     //I 帧间隔
    unsigned char reserve;

} GLNK_VideoDataFormat;
```

```
//音频数据格式
typedefstruct _GLNK_AudioDataFormat
{
    unsigned intsamplesRate;     //每秒采样
    unsigned int bitrate;        //比特率, bps
    unsigned short waveFormat;   //编码格式
    unsigned short channelNumber; //音频通道号单通道 1  双通道 2
    unsigned short blockAlign;   //块对齐, channelSize * (bitsSample/8)
    unsigned short bitsPerSample; //每采样比特数
    unsigned short frameInterval; //帧间隔, 单位 ms
    unsigned short reserve;

} GLNK_AudioDataFormat;
```

读取回放视频数据

int32_t GLNK_RecordReadFrame_CallBack(int32_t recordfd, unsigned char *streamframetype, uint32_t *timestamp, int32_t *videoframeindex, void** streamdata, uint32_t *streamdatalen);、

说明：读取回放视频数据（不可阻塞）

- recordfd：读取的录像文件的描述符
- streamframetype：录像视频数据格式
- timestamp：时间戳
- videoframeindex 视频帧序号
- streamdata 码流数据
- streamdatalen 数据长度

返回值成功返回 1，失败返回 0

关闭录像回放的读取的文件

int32_t GLNK_RecordClose_CallBack(int32_t recordfd);

说明：读取回放视频数据（不可阻塞）

- recordfd：读取的录像文件的描述符

打开需要下载的录像文件

int32_t GLNK_OpenDownLoadRecord_Callback(char* recordname, int32_t *recordfd, char mode, uint32_t offset);

说明：打开需要下载的录像文件（不可阻塞）

参数：

- recordname：录像文件名字
- recordfd：返回的打开的录像文件的描述符
- mode：打开的模式 1----普通下载 2---断点续传

- offset：断点续传的文件偏移量

int32_t GLNK_CloseDownLoadRecord_Callback(int32_t recordfd);
说明：关闭下载的录像文件（不可阻塞）
- recordfd：需要关闭的录像文件描述符
-

int32_t GLNK_ReadDownLoadRecord_Callback(int32_t recordfd, char* data, int32_t datalen, int32_t *start_pos, int32_t *end_pos);
说明：读取录像文件（不可阻塞）
参数：
- Recordfd：读取的录像文件描述符
- Data：数据缓存区
- Datalen：缓存区长度
- start_pos：读取前录像文件的当前偏移位置
- end_pos：读取后的文件偏移位置

int32_t GLNK_GetVide_AudioConfig_Callback (int32_t channel, TLV_V_VA_GET_CONFIG_RSP *req);
说明：获取音视频码流帧率（不可阻塞）
参数：
- channel：需要获取音视频码流帧率相关通道的通道号
- req：返回获取到的音视频码流帧率（不需要再分配内存）

```
typedefstruct _TLV_V_VA_GET_CONFIG_RSP
{
    unsigned char       channel;    //通道号
    unsigned char   isOpenAudio;   //是否打开音频
    unsigned char reverse[2];   //保留
    OWSP_VIDEO_RESOLUTION mainStreamResolution;   //主码流分辨率
    unsigned int   mainStreamFrameRate;    //主码流帧率
    unsigned intmainStreamBitRate;        //主码流比特率

    OWSP_VIDEO_RESOLUTION subStreamResolution;//次码流分辨率
    unsigned intsubStreamFrameRate;//次码流帧率
    unsigned intsubStreamBitRate;//次码流比特率
}TLV_V_VA_GET_CONFIG_RSP;
```

int32_t GLNK_SetVide_AudioConfig_Callback(int32_t *result, TLV_V_VA_SET_CONFIG_REQ *req);

说明：设置视频音视码流帧率（不可阻塞）
参数：
- result：返回设置的结果（不需要再分配内存）
- req：设置音视频码流帧率的数据结构
-

```
typedefstruct _TLV_V_VA_SET_CONFIG_RSP
{
 unsigned char        channel;     //通道号
 unsigned char    isOpenAudio;    //是否打开音频
 unsigned char reverse[2];    //保留
 OWSP_VIDEO_RESOLUTION mainStreamResolution;    //主码流分辨率
 unsigned int    mainStreamFrameRate;     //主码流帧率
 unsigned intmainStreamBitRate;         //主码流比特率

 OWSP_VIDEO_RESOLUTION subStreamResolution;//次码流分辨率
 unsigned intsubStreamFrameRate;//次码流帧率
 unsigned intsubStreamBitRate;//次码流比特率
}TLV_V_VA_SET_CONFIG_RSP;
```

int32_t GLNK_SetNetWorkConfig_Callback(int32_t *result, TLV_V_Network *req);
说明：设置相关网络参数（不可阻塞）
参数：
- result：返回设置的结果（不需要再分配内存）
- req：设置网络参数的数据结构
-

```
typedefstruct _TLV_V_Network
{
      unsigned intdeviceId;     //设备 id
      unsigned char    hostIP[4]; //有线网卡 ip 地址
      unsigned char       hostName[32];//有线网卡主机名
      unsigned char    gateway[4];//有线网卡网关
      unsigned char    dnsServer[4];//有线网卡 dns 服务器 ip
      unsigned char    dnsServer2[4];//有线网卡 dns 服务器 2ip
      unsigned char    subnetMask[4];//有线网卡子网掩码

      unsigned char    wifiHostIP[4];//wifiip 地址
      unsigned char    wifiHostName[32];//wifi 主机名
      unsigned char    wifiGateway[4];//wifi 网关
      unsigned char    wifiDnsServer[4];//wifidns 服务器
      unsigned char    wifiDnsServer2[4];// wifidns 服务器 2
      unsigned char    wifiSubnetMask[4];//wifi 子网掩码
      unsigned char    wifiMac[8]; //wifi Mac 地址
      unsigned char    mac[8];//有线 mac 地址
```

```
    unsigned char   wifiIPMode;//wifi 连接模式
    unsigned char   IPMode; //有线连接模式
    unsigned char   reverse[2];
} TLV_V_Network;
```

int32_t GLNK_GetNetWorkConfig_Callback(TLV_V_Network *req);

说明：获取相关网络参数（不可阻塞）

参数：

- req：获取网络参数的数据结构(不需再分配内存)

int32_t GLNK_DeviceUpDate_Callback(int32_t type,char *buffer, int32_t length);

说明：设备升级回调函数（不可阻塞）参数：

- type：
  - 1-----请求升级判断 flash 空间大小，打开文件，返回文件描述符参数 length 是将要接收文件的总长度，如果是模拟网页的升级方式，buffer 存储返回的升级 URL
  - 2—将接收到的数据写到 flash 中 buffer 数据开头指针，length 数据长度
  - 3-----接收完成关闭文件，开始升级
  - 4---升级失败关闭文件
- Buffer：数据指针
- Length: 数据长度

返回值：成功为 1，失败为 0

int32_t GLNK_GetVersionFirmware_Callback(char* appbuf, char* solbuf, char* date, char* hardware);

说明：获取设备固件版本信息（不可阻塞）

参数：

- appbuf：app 名字不能超过 20 个字节（根据不同的 app 相应写死名字，如菲扬-→FeiYang）
- solbuf：方案商名字不能超过 20 个字节（用公司名字的全拼，如浪涛->LangTao）
- date：设备软件版本更新的日期不能超过 20 个字节
- hardware：硬件版本---方案商自己定义不能超过 64 个字节

返回值：成功为 1，失败为 0

int32_t GLNK_Lock_Callback(char * passwd, int channel, int type)

说明：上锁解锁接口（不可阻塞）

参数：

Passwd：上锁解锁密码

Channel：上锁解锁通道

Type：1-----解锁 0----上锁

int32_t CloudStorage_UpLoadFile_Callback(char *filename,
int32_t* filenamelen,
int32_t* filesize,
int32_t *filefd)

说明：获取上传文件名字
参数：

- filename：文件名字
- filenamelen：名字的长度
- filesize：文件的大小
- filefd：文件的描述符

int32_t Get_Cloud_UpLoadFile_Result_Callback(char *filename, int32_t result)

说明：上传文件的结果 1 为成功 0 为失败

- filename：文件名字
- result：上传结果 1 为成功 0 为失败

int32_t GLNK_ResetPswdOfModif_Callback(char * oldpasswd, char* newpasswd)
说明：重新设置开锁密码

- oldpasswd：旧密码
- newpasswd：新密码