

# 设备(门禁主机)

网络 SDK 编程指南

V5.2

# 声 明

非常感谢您购买我公司的产品，如果您有什么疑问或需要请随时联系我们。

- 我们已尽量保证手册内容的完整性与准确性，但也不免出现技术上不准确、与产品功能及操作不相符或印刷错误等情况，如有任何疑问或争议，请以我司最终解释为准。
- 产品和手册将实时进行更新，恕不另行通知。
- 本手册中内容仅为用户提供参考指导作用，请以 SDK 实际内容为准。

# 目 录

声 明 .....	I
目 录 .....	II
1 SDK 简介 .....	1
2 SDK 版本更新 .....	4
3 函数调用顺序 .....	9
3.1 SDK 接口调用主要流程 .....	9
3.2 门参数/读卡器参数配置流程 .....	10
3.3 计划配置流程 .....	11
3.3.1 门状态计划配置流程 .....	12
3.3.2 读卡器验证方式计划配置流程 .....	13
3.3.3 卡权限计划模板配置流程 .....	14
3.4 布防获取门禁事件流程 .....	15
3.5 获取和下发卡号功能流程 .....	16
3.6 获取和下发人脸参数流程 .....	17
3.7 获取和下发指纹参数流程 .....	18
3.8 远程开门控制流程 .....	19
3.9 手动抓拍流程 .....	19
3.10 联动抓拍流程 .....	20
4 函数调用实例 .....	21
4.1 门参数配置示例代码 .....	21
4.2 读卡器参数配置示例代码 .....	23
4.3 门状态计划配置示例代码 .....	25
4.4 读卡器验证方式计划配置示例代码 .....	29
4.5 卡权限计划模板配置示例代码 .....	33
4.6 布防获取门禁事件示例代码 .....	36
4.7 获取和下发卡号相关联参数示例代码 .....	38
4.8 远程开门示例代码 .....	44
4.9 联动抓拍示例代码 .....	46
5 功能接口介绍 .....	50
5.1 通用接口介绍 .....	50
5.2 门禁主机功能接口 .....	50
5.3 指纹门禁一体机功能接口 .....	58
6 函数说明 .....	63
6.1 SDK 初始化 .....	63
6.1.1 初始化 SDK NET_DVR_Init .....	63
6.1.2 释放 SDK 资源 NET_DVR_Cleanup .....	63
6.2 SDK 本地功能 .....	63
SDK 本地参数配置 .....	63
6.2.1 获取 SDK 本地参数 NET_DVR_GetSDKLocalCfg .....	63
6.2.2 设置 SDK 本地参数 NET_DVR_SetSDKLocalCfg .....	64
连接和接收超时时间及重连设置 .....	64
6.2.3 设置网络连接超时时间和连接尝试次数 NET_DVR_SetConnectTime .....	64

6.2.4	设置重连功能 NET_DVR_SetReconnect .....	65
6.2.5	设置接收超时时间 NET_DVR_SetRecvTimeOut .....	65
	多网卡绑定 .....	65
6.2.6	获取所有 IP，用于支持多网卡接口 NET_DVR_GetLocalIP .....	65
6.2.7	设置 IP 绑定 NET_DVR_SetValidIP .....	66
	SDK 版本、状态和能力 .....	66
6.2.8	获取 SDK 的版本号和 build 信息 NET_DVR_GetSDKBuildVersion .....	66
6.2.9	获取当前 SDK 的状态信息 NET_DVR_GetSDKState .....	66
6.2.10	获取当前 SDK 的功能信息 NET_DVR_GetSDKAbility .....	66
	SDK 启用写日志 .....	67
6.2.11	启用写日志文件 NET_DVR_SetLogToFile .....	67
	异常消息回调 .....	67
6.2.12	注册接收异常、重连等消息的窗口句柄或回调函数 NET_DVR_SetExceptionCallBack_V30 .....	67
	获取错误信息 .....	69
6.2.13	返回最后操作的错误码 NET_DVR_GetLastError .....	69
6.2.14	返回最后操作的错误码信息 NET_DVR_GetErrorMsg .....	69
6.3	用户注册 .....	70
6.3.1	激活设备 NET_DVR_ActivateDevice .....	70
6.3.2	通过解析服务器，获取设备的动态 IP 地址和端口号 NET_DVR_GetDVRIPByResolveSvr_EX .....	70
6.3.3	用户注册设备 NET_DVR_Login_V40 .....	70
6.3.4	用户注销 NET_DVR_Logout .....	71
6.4	获取设备能力集 .....	71
6.4.1	获取设备能力集 NET_DVR_GetDeviceAbility .....	71
6.4.2	获取门禁总能力集 NET_DVR_STDXMLConfig .....	72
6.5	布防、撤防 .....	73
	设置报警等信息上传的回调函数 .....	73
6.5.1	注册报警回调函数 NET_DVR_SetDVRMessageCallBack_V31 .....	73
	布防撤防 .....	74
6.5.2	建立报警上传通道，获取报警等信息 NET_DVR_SetupAlarmChan_V41 .....	74
6.5.3	撤销报警上传通道 NET_DVR_CloseAlarmChan_V30 .....	74
6.6	远程参数配置 .....	74
	通用参数配置 .....	74
6.6.1	获取设备的配置信息 NET_DVR_GetDVRConfig .....	74
6.6.2	设置设备的配置信息 NET_DVR_SetDVRConfig .....	75
	门禁相关参数配置 .....	76
6.6.3	获取设备的配置信息 NET_DVR_GetDVRConfig .....	76
6.6.4	设置设备的配置信息 NET_DVR_SetDVRConfig .....	78
6.6.5	批量获取配置信息 NET_DVR_GetDeviceConfig .....	80
6.6.6	批量设置配置信息 NET_DVR_SetDeviceConfig .....	81
	RS485/防区/触发器配置和控制 .....	83
6.6.7	获取 NET_DVR_GetDVRConfig .....	83
6.6.8	设置 NET_DVR_SetDVRConfig .....	83
6.6.9	设置触发器 NET_DVR_SetAlarmHostOut .....	84
6.6.10	对防区布防 NET_DVR_AlarmHostSetupAlarmChan .....	84

6.6.11	对防区撤防 NET_DVR_AlarmHostCloseAlarmChan.....	84
	获取所有防区/触发器及外接模块.....	85
6.6.12	启动长连接远程配置 NET_DVR_StartRemoteConfig.....	85
6.6.13	逐个获取查找到的结果信息 NET_DVR_GetNextRemoteConfig .....	86
6.6.14	关闭长连接配置接口所创建的句柄, 释放资源 NET_DVR_StopRemoteConfig .....	87
	用户配置.....	87
6.6.15	获取设备用户配置 NET_DVR_GetAlarmDeviceUser.....	87
6.6.16	设置设备用户配置 NET_DVR_SetAlarmDeviceUser .....	88
	长连接参数配置.....	88
6.6.17	启动长连接远程配置 NET_DVR_StartRemoteConfig.....	88
6.6.18	发送长连接数据 NET_DVR_SendRemoteConfig .....	90
6.6.19	关闭长连接配置接口所创建的句柄, 释放资源 NET_DVR_StopRemoteConfig .....	91
6.7	远程控制.....	91
6.7.1	远程控制 NET_DVR_RemoteControl.....	91
6.8	门禁控制.....	92
6.8.1	门禁控制 NET_DVR_ControlGateway .....	92
6.9	就地控制器相关.....	92
6.9.1	就地控制器管理 NET_DVR_STDXMLConfig.....	92
6.9.2	就地控制器控制 NET_DVR_STDXMLConfig.....	93
6.9.3	就地控制器搜索及状态查询 NET_DVR_StartRemoteConfig.....	94
6.10	USB 设备管理.....	95
6.10.1	USB 设备管理 NET_DVR_STDXMLConfig .....	95
6.11	可视对讲呼叫功能.....	96
6.11.1	启动长连接远程配置 NET_DVR_StartRemoteConfig.....	96
6.11.2	发送长连接数据 NET_DVR_SendRemoteConfig .....	98
6.11.3	关闭长连接配置接口所创建的句柄, 释放资源 NET_DVR_StopRemoteConfig .....	98
6.12	考勤相关功能.....	99
6.12.1	考勤相关配置 NET_DVR_STDXMLConfig.....	99
6.13	手动抓拍.....	101
6.13.1	单帧数据捕获并保存成 JPEG 图片 NET_DVR_CaptureJPEGPicture.....	101
6.13.2	单帧数据捕获并保存成 JPEG 存放在指定的内存空间中 NET_DVR_CaptureJPEGPicture_NEW .....	101
6.13.3	网络触发抓拍 NET_DVR_ContinuousShoot .....	101
6.14	底图功能.....	102
6.14.1	底图上传 NET_DVR_PicUpload .....	102
6.14.2	获取图片上传的进度 NET_DVR_GetPicUploadProgress.....	102
6.14.3	获取远程上传的状态 NET_DVR_GetPicUploadState .....	102
6.14.4	关闭上传句柄, 释放资源 NET_DVR_CloseUploadHandle.....	103
6.14.5	远程控制 NET_DVR_RemoteControl.....	103
6.15	设备维护管理.....	103
	在线状态检测.....	103
6.15.1	设备在线状态检测 NET_DVR_RemoteControl.....	103
	远程升级.....	104
6.15.2	设置远程升级时网络环境 NET_DVR_SetNetworkEnvironment.....	104
6.15.3	远程升级 NET_DVR_Upgrade .....	104

6.15.4	获取远程升级的进度 NET_DVR_GetUpgradeProgress .....	104
6.15.5	获取远程升级的状态 NET_DVR_GetUpgradeState .....	105
6.15.6	关闭远程升级句柄，释放资源 NET_DVR_CloseUpgradeHandle .....	105
	恢复设备默认参数 .....	105
6.15.7	恢复设备默认参数 NET_DVR_RestoreConfig .....	105
6.15.8	完全恢复出厂默认参数 NET_DVR_RemoteControl .....	105
	导入/导出配置文件 .....	106
6.15.9	导出配置文件 NET_DVR_GetConfigFile_V30 .....	106
6.15.10	导出配置文件 NET_DVR_GetConfigFile .....	106
6.15.11	导入配置文件 NET_DVR_SetConfigFile_EX .....	106
6.15.12	导入配置文件 NET_DVR_SetConfigFile .....	107
	远程重启 .....	107
6.15.13	重启设备 NET_DVR_RebootDVR .....	107
7	错误代码及说明 .....	108
7.1	网络通讯库错误码 .....	108
7.2	RTSP 通讯库错误码 .....	111
7.3	软解码库错误码 .....	112
8	附录:结构体 .....	114

# 1 SDK 简介

设备网络 SDK 是基于设备私有网络通信协议开发的，为嵌入式网络硬盘录像机、NVR、视频服务器、网络摄像机、网络球机、解码器、多屏控制器、报警主机等网络产品服务的配套模块，用于远程访问和控制设备软件的二次开发。本文档主要介绍门禁主机相关的功能，设备型号如下：

门禁主机包括但不限于以下产品型号：DS-K2601(-G)、DS-K2602(-A/-G)、DS-K2604(-G)。

指纹门禁一体机包括但不限于以下产品型号：

DS-K1T105C(-C)、DS-K1T105E(-C)、DS-K1T105M(-C)、DS-K1T200CF(-C)、DS-K1T200EF(-C)、DS-K1T200MF(-C)、DS-K1T300CF(-C)、DS-K1T300EF(-C)、DS-K1T300MF(-C)。

设备网络 SDK 主要功能：门禁主机参数配置、获取和设置卡参数、门禁控制、手动抓拍以及布防获取设备上传的报警信息等功能。

**注意：**该文档附录中结构体说明仅供参考，更多说明及更新请参见《设备网络 SDK 使用手册.chm》。

设备网络 SDK 包含网络通讯库、软解码库、硬解码库等功能组件，我们提供 Windows 和 Linux 两个版本的 SDK，各自所包含的组件如下：

表 1.1 Windows SDK 组件

网络通讯库	外部接口	HCNetSDK.h	头文件	
		HCNetSDK.lib	LIB 库文件	
		HCNetSDK.dll	DLL 库文件	
	核心组件	HCCore.lib	LIB 库文件	
		HCCore.dll	DLL 库文件	
组件库	设备配置核心组件	HCCoreDevCfg.dll	DLL 库文件	HCNetSDKCom 文件夹
	预览组件	HCPreview.lib	LIB 库文件	HCNetSDKCom 文件夹
		HCPreview.dll	DLL 库文件	HCNetSDKCom 文件夹
	回放组件	HCPlayBack.dll	DLL 库文件	HCNetSDKCom 文件夹
	语音组件	HCVoiceTalk.dll	DLL 库文件	HCNetSDKCom 文件夹
	报警组件	HCArm.lib	LIB 库文件	HCNetSDKCom 文件夹
		HCArm.dll	DLL 库文件	HCNetSDKCom 文件夹
	显示组件	HCDisplay.dll	DLL 库文件	HCNetSDKCom 文件夹
	行业应用管理配置组件	HCIndustry.dll	DLL 库文件	HCNetSDKCom 文件夹
	维护管理配置组件	HCGeneralCfgMgr.lib	LIB 库文件	HCNetSDKCom 文件夹
		HCGeneralCfgMgr.dll	DLL 库文件	HCNetSDKCom 文件夹
RTSP 通讯库		StreamTransClient.dll	DLL 库文件	HCNetSDKCom 文件夹
转封装库		SystemTransform.dll	DLL 库文件	HCNetSDKCom 文件夹
语音对讲库		AudioIntercom.dll	DLL 库文件	HCNetSDKCom 文件夹

		OpenAL32.dll	DLL 库文件	HCNetSDKCom 文件夹
字符转码库		libiconv2.dll	DLL 库文件	HCNetSDKCom 文件夹
模拟能力集		LocalXml.zip	XML 文件包	
软解码库		plaympeg4.h	头文件	
		PlayCtrl.lib	LIB 库文件	
		PlayCtrl.dll	DLL 库文件	
		AudioRender.dll	DLL 库文件	
		SuperRender.dll	DLL 库文件	
硬解码库		DsSdk.dll	DLL 库文件	HCNetSDKCom 文件夹

表 1.2 Linux SDK 组件

网络通讯库	外部接口	HCNetSDK.h	头文件	
		libhcnetsdk.so	SO 库文件	
	核心组件	libHCCore.so	SO 库文件	
组件库	设备配置核心组件	libHCCoreDevCfg.so	SO 库文件	HCNetSDKCom 文件夹
	预览组件	libHCPreview.so	SO 库文件	HCNetSDKCom 文件夹
	回放组件	libHCPlayBack.so	SO 库文件	HCNetSDKCom 文件夹
	语音组件	libHCVoiceTalk.so	SO 库文件	HCNetSDKCom 文件夹
	报警组件	libHCAAlarm.so	SO 库文件	HCNetSDKCom 文件夹
	显示组件	libHCDisplay.so	SO 库文件	HCNetSDKCom 文件夹
	行业应用管理配置组件	libHCIndustry.so	SO 库文件	HCNetSDKCom 文件夹
	维护管理配置组件	libHCGeneralCfgMgr.so	SO 库文件	HCNetSDKCom 文件夹
hpr 库		libhpr.so	SO 库文件	
RTSP 通讯库		libStreamTransClient.so	SO 库文件	HCNetSDKCom 文件夹
转封装库		libSystemTransform.so	SO 库文件	HCNetSDKCom 文件夹
字符转码库		libiconv2.so	SO 库文件	HCNetSDKCom 文件夹
软解码库		LinuxPlayM4.h	头文件	
		PlayM4.h	头文件	
		libMPCtrl.so	SO 库文件	
		libPlayCtrl.so	SO 库文件	

本版本的设备网络 SDK 开发包中包含以上各个组件, **HCNetSDK.dll、HCCore.dll 必须加载**(对于 Linux SDK, 即 libhcnetsdk.so、libHCCore.so), 其他组件, 用户可以根据需要选择其中的一部分或者全部, 以下将对各个组件在 SDK 中的作用和使用条件分别说明。

- **网络通讯库**: 设备网络 SDK 的主体, 主要用于网络客户端与各类产品之间的通讯交互, 负责远程功能



调控, 远程参数配置及码流数据的获取和处理等。设备网络 SDK V5.0 针对产品应用业务进行细化, 对之前版本的 SDK 的功能模块进行组件化, 其中外部接口 (HCNetSDK.dll) 仍然保持和设备网络 SDK V4.x 版本保存一致(向下兼容), 其他单独的业务功能 (预览、回放等) 可以加载单独的模块组件, 多个业务功能也可以组合使用。**更新 SDK 时, HCNetSDK.dll、HCCore.dll 以及 HCNetSDKCom 文件夹下的功能组件库文件都需要更新加载, 且 HCNetSDKCom 文件夹名不能修改。**

- **hpr 库:** 网络通讯库的依赖库, Linux SDK 使用时和网络通讯库同时加载。
- **RTSP 通讯库:** 支持 RTSP 传输协议的网络库。当需要对支持 RTSP 协议的产品进行取流等操作时就必须加载该项组件。
- **转封装库:** 库的功能可以分为两种: 一种是将标准码流转换成采用我们公司封装格式的码流。当用户需要对支持 RTSP 协议的产品捕获采用本公司封装格式的码流数据时 (即当设置 [NET\\_DVR\\_RealPlay\\_V40](#) 接口中的回调函数捕获数据或者调用 [NET\\_DVR\\_SetRealDataCallBack](#) 接口捕获数据时) 必须加载该组件。另一种功能是将标准码流转换成其他格式的封装, 如 3GPP、PS 等。例如, 当用户需要对支持 RTSP 协议的产品实时捕获指定封装格式的码流数据 (对应的 SDK 接口为 [NET\\_DVR\\_SaveRealData](#)) 时必须加载该项组件。
- **语音对讲库:** 用于语音对讲时通过声卡采集数据并按照指定的编码格式编码码流或者解码播放音频码流数据 (不带封装格式的码流数据)。V4.2.2.5 及以前版本 SDK 均采用 windows API 实现相关功能。之后版本默认使用语音对讲库的方式, 通过接口 [NET\\_DVR\\_SetSDKLocalCfg](#) 可以选择之前的 windows API 模式。OpenAL32.dll 为依赖库, 语音对讲库模式下必须加载。**CVR 暂不支持语音对讲功能。**
- **字符转换库:** 电脑字符集和设备字符集不一致时, SDK 内部需要进行字符编码转换, SDK 默认使用 libiconv 库进行类型转换。如果用户不想使用 libiconv 编码库, 可以调用 [NET\\_DVR\\_SetSDKLocalCfg](#) (类型: [NET\\_SDK\\_LOCAL\\_CFG\\_TYPE\\_BYTE\\_ENCODE](#)) 设置字符转码回调函数, 将用户自己的字符编码接口告知 SDK, 然后 SDK 将使用用户提供的字符编码接口进行字符串处理。
- **模拟能力集:** 如果需要获取设备能力集 ([NET\\_DVR\\_GetDeviceAbility](#)), 建议调用 [NET\\_DVR\\_SetSDKLocalCfg](#) 启用模拟能力集, 此时需要加载 LocalXml.zip (要求和网络通讯库放在同一个目录下)。
- **软解码库:** 主要用于对实时码流数据进行解码显示 (实现预览功能) 和对录像文件进行回放解码等。用户如果需要在 SDK 内部进行对实时流和录像码流播放显示时 (即 [NET\\_DVR\\_RealPlay\\_V40](#) 接口的第二个结构体参数的播放句柄设置成有效句柄时) 必须加载该组件, 而如果用户仅需要用网络通讯库捕获到数据后再外部自行处理就不需要加载该组件, 这种情况下用户在外部分行解码将更灵活, 可参见软解码库函数说明《播放器 SDK 编程指南》。**Linux 64 位系统不支持软解码功能, 预览、回放等窗口句柄传空, 仅支持只取流不解码。**
- **硬解码库:** 需在配备硬解码卡 (MD 卡) 的前提下使用, 通过解码卡的解码与输出功能实现实时流的解码显示及向监视器上矩阵输出的功能。Windows64 位或者 Linux 系统下无该硬解码库。对于智能交通摄像机, 不需要使用该库文件。

## 2 SDK 版本更新

### Version 5.2.5.40 (build 20161102)

- 更新原因：经济型指纹门禁产品 V1.0 DS-K1T803F, DS-K1A801F
- 新增考勤相关参数配置功能（对应接口 [NET\\_DVR\\_STDXMLConfig](#)），不同参数配置对应不同命令，详见该接口说明。

### Version 5.2.5.40 (build 20161102)

- 更新原因：视频门禁一体机 DS-K1T500S, DS-K1T500SF V1.0
- 新增可视通话信令交互功能（对应接口 [NET\\_DVR\\_StartRemoteConfig](#)）  
命令：NET\_DVR\_VIDEO\_CALL\_SIGNAL\_PROCESS。
- 新增采集指纹信息功能（对应接口 [NET\\_DVR\\_StartRemoteConfig](#)）  
命令：NET\_DVR\_CAPTURE\_FINGERPRINT\_INFO。

### Version 5.2.5.10 (build 20160817)

- 更新原因：分布式三层架构门禁产品 V1.0
- 远程升级接口 [NET\\_DVR\\_Upgrade\\_V40](#) 扩展，支持升级就地控制器，pInbuffer 为 4 字节设备编号，10001 表示就地控制器 1，10002 表示就地控制器 2，依次类推。
- 设备协议接入能力集 AccessProtocolAbility 新增<zoneInfo> <alarmoutInfo>节点；  
网络报警主机 XML 能力集 AlarmHostAbility 中新增是否不支持配置防区关联触发器节点  
<noZoneRelatedTrigger>;  
zoneType 节点取值新增 doorEmergencyOpenProtectionZone,doorEmergencyShutdownProtectionZone;  
ModuleType 取值新增 1DoorController,2DoorsController,4DoorsController;  
（对应接口 [NET\\_DVR\\_GetDeviceAbility](#)，能力类型：DEVICE\_ABILITY\_INFO。）
- 防区参数结构体 [NET\\_DVR\\_ALARMIN\\_PARAM](#) 中 byType（防区类型）新增取值：10-门禁紧急开门防区，11-门禁紧急关门防区；byModuleType（模块类型）新增取值：7-1 门就地控制器、8-2 门就地控制器、9-4 门就地控制器。
- 门禁报警报警 COMM\_ALARM\_ACS 各报警主类型下次类型新增，详见：[NET\\_DVR\\_ACS\\_ALARM\\_INFO](#)  
主类型“报警”下面新增次类型 0x415~0x417；主类型“异常”下新增次类型 0x413~0x41a；主类型“操作”下新增次类型 0x410~0x416；主类型“事件”下新增次类型 0x51~0x5e。  
主机事件信息结构体 [NET\\_DVR\\_ACS\\_EVENT\\_INFO](#) 扩展：新增就地控制器编号字段 wLocalControllerID，网口 ID 字段 byInternetAccess，防区类型字段 byType。
- 新增就地控制器管理功能（对应接口 [NET\\_DVR\\_STDXMLConfig](#)）：  
能力集：GET /ISAPI/AccessControl/localController/capabilities；  
添加：PUT /ISAPI/AccessControl/localController/addDevice；  
获取：GET /ISAPI/AccessControl/localController/localControllerID/<ID>;  
设置：PUT /ISAPI/AccessControl/localController/localControllerID/<ID>;  
删除单个：DELETE /ISAPI/AccessControl/localController/localControllerID/<ID>;  
删除全部：DELETE /ISAPI/AccessControl/localController。
- 门参数配置结构体：[NET\\_DVR\\_DOOR\\_CFG](#) 扩展：使用 12 个保留字节新增字段 byLeaderCardMode、

- byUseLocalController、wLocalControllerID、wLocalControllerDoorNumber、wLocalControllerStatus、byLockInputCheck、byLockInputType、byDoorTerminalMode、byOpenButton。
- 读卡器参数配置结构体: [NET\\_DVR\\_CARD\\_READER\\_CFG](#) 扩展: 使用 8 个保留字节新增字段 byUseLocalController、byRes1、wLocalControllerID、wLocalControllerReaderID、wCardReaderChannel。
  - 新增就地控制器控制功能 (对应接口 [NET\\_DVR\\_STDXMLConfig](#)):  
能力集: GET /ISAPI/AccessControl/localController/control/capabilities;  
设置: PUT /ISAPI/AccessControl/localController/control/localControllerID/<ID>。
  - 新增搜索在线就地控制器, 获取就地控制器状态功能 (复用接口 [NET\\_DVR\\_StartRemoteConfig](#)):  
命令: NET\_DVR\_GET\_ONLINE\_LOCAL\_CONTROLLER,  
NET\_DVR\_GET\_LOCAL\_CONTROLLER\_STATUS。
  - 新增 USB 设备管理功能 (对应接口 [NET\\_DVR\\_STDXMLConfig](#)):  
获取 USB 设备状态: GET /ISAPI/AccessControl/USBStatus/USBNumber/<ID>;  
获取 USB 设备状态能力: GET /ISAPI/AccessControl/USBStatus/capabilities;  
USB 控制: PUT /ISAPI/AccessControl/USBControl/USBNumber/<ID>;  
获取 USB 控制能力: GET /ISAPI/AccessControl/USBControl/capabilities;  
获取 USB 控制进度: GET /ISAPI/AccessControl/USBControlProgress/USBNumber/<ID>;  
获取 USB 控制进度能力: GET /ISAPI/AccessControl/USBControlProgress/capabilities。
  - 新增 V50 的卡参数配置命令 NET\_DVR\_GET\_CARD\_CFG\_V50、NET\_DVR\_SET\_CARD\_CFG\_V50  
复用接口 [NET\\_DVR\\_StartRemoteConfig](#)、[NET\\_DVR\\_SendRemoteConfig](#)
  - 新增网络参数配置 V50 命令, 支持双监听: NET\_DVR\_GET\_NETCFG\_V50、NET\_DVR\_SET\_NETCFG\_V50  
复用接口 [NET\\_DVR\\_GetDVRConfig](#)、[NET\\_DVR\\_SetDVRConfig](#)
  - 新增事件及卡号联动参数配置(V50)、卡权限周计划参数配置 (V50)、卡权限假日计划参数配置 (V50)、卡权限假日组参数配置 (V50)、卡权限计划模板参数配置 (V50) 命令  
对应获取命令: NET\_DVR\_GET\_EVENT\_CARD\_LINKAGE\_CFG\_V50、  
NET\_DVR\_GET\_CARD\_RIGHT\_WEEK\_PLAN\_V50、NET\_DVR\_GET\_CARD\_RIGHT\_HOLIDAY\_PLAN\_V50、  
NET\_DVR\_GET\_CARD\_RIGHT\_HOLIDAY\_GROUP\_V50、NET\_DVR\_GET\_CARD\_RIGHT\_PLAN\_TEMPLATE\_V50  
对应设置命令: NET\_DVR\_SET\_EVENT\_CARD\_LINKAGE\_CFG\_V50、  
NET\_DVR\_SET\_CARD\_RIGHT\_WEEK\_PLAN\_V50、NET\_DVR\_SET\_CARD\_RIGHT\_HOLIDAY\_PLAN\_V50、  
NET\_DVR\_SET\_CARD\_RIGHT\_HOLIDAY\_GROUP\_V50、NET\_DVR\_SET\_CARD\_RIGHT\_PLAN\_TEMPLATE\_V50  
复用接口 [NET\\_DVR\\_GetDeviceConfig](#)、[NET\\_DVR\\_SetDeviceConfig](#)
  - 防区参数结构体 [NET\\_DVR\\_ALARMIN\\_PARAM](#) 扩展, byType 新增取值: 10-门禁紧急开门防区, 11-门禁紧急关门防区; byModuleType 新增取值: 7-1 门就地控制器、8-2 门就地控制器、9-4 门就地控制器。  
触发器参数结构体 [NET\\_DVR\\_ALARMOUT\\_PARAM](#) 扩展, byModuleType 新增取值: 6-1 门就地控制器、7-2 门就地控制器、8-4 门就地控制器。  
获取子系统防区、触发器、搜索/注册外接模块功能复用如下接口:  
[NET\\_DVR\\_StartRemoteConfig](#)、[NET\\_DVR\\_GetNextRemoteConfig](#)、[NET\\_DVR\\_StopRemoteConfig](#)

### Version 5.1.6.10 (build 20160223)

- 更新原因: 人员通道门禁主机 DS-K2602-A V1.1
- 新增下发平台认证结果功能 (对应接口 [NET\\_DVR\\_SetDVRConfig](#))  
命令: NET\_DVR\_SET\_PLATFORM\_VERIFY\_CFG。
- 新增人数统计参数配置 (对应接口 [NET\\_DVR\\_GetDVRConfig](#) 和 [NET\\_DVR\\_SetDVRConfig](#))  
命令: NET\_DVR\_GET\_PERSON\_STATISTICS\_CFG、NET\_DVR\_SET\_PERSON\_STATISTICS\_CFG。

- 新增屏幕字符串显示配置（对应接口 [NET\\_DVR\\_GetDVRConfig](#) 和 [NET\\_DVR\\_SetDVRConfig](#)）  
命令：NET\_DVR\_GET\_ACS\_SCREEN\_DISPLAY\_CFG、NET\_DVR\_SET\_ACS\_SCREEN\_DISPLAY\_CFG。
- 新增人员通道闸门时间参数配置（对应接口 [NET\\_DVR\\_GetDVRConfig](#) 和 [NET\\_DVR\\_SetDVRConfig](#)）  
命令：NET\_DVR\_GET\_GATE\_TIME\_CFG、NET\_DVR\_SET\_GATE\_TIME\_CFG。
- [NET\\_DVR\\_ACS\\_EXTERNAL\\_DEV\\_CFG](#)(门禁主机串口外设参数)扩展，使用 2 个保留字节新增参数：  
wDevDetailType(外设详细设备型号)，byACSDevType(设备类型)新增取值：8-指纹头、9-语音模块。
- [NET\\_DVR\\_ID\\_CARD\\_INFO\\_ALARM](#)(身份证刷卡信息上传)使用 9 个保留字节新增参数：dwPicDataLen(图片数据大小)、pPicData(图片数据)、byCardType(卡类型)。
- [NET\\_DVR\\_FINGER\\_PRINT\\_STATUS](#)(指纹状态参数)，使用 36 个保留字节新增参数：  
byErrorMsg[ERROR\_MSG\_LEN](下发错误信息)、dwCardReaderNo(指纹读卡器编号)，  
byCardReaderRecvStatus(指纹读卡器状态)新增取值类型：2-该指纹模组不在线、3-重试或指纹质量差、  
4-内存已满、5-已存在该指纹、6-已存在该指纹 ID、7-非法指纹 ID、8-该指纹模组无需配置。
- 门禁主机能力集([ACS\\_ABILITY](#))扩展：
  - 1) 新增节点：<PlatformVerifyCfg>(平台认证能力)、<PersonStatisticsCfg>(人数统计能力)、  
<ScreenDisplayCfg>(屏幕字符串显示参数能力)、<GateTimeCfg>(人员通道闸门时间参数能力)。
  - 2) <ExternalDevCfg>(串口外设参数能力)，新增子节点：<DevDetailType>(详细设备型号)，  
<ACSDevType>(设备类型)新增取值：fingerPrintModule(指纹头)、voiceModule(语音模块)。

### Version 5.1.3.32 (build 20151021)

- 更新原因：人员通道门禁主机 DS-K2602-A V1.0
- 新增串口外设参数配置功能（对应接口 [NET\\_DVR\\_GetDVRConfig](#) 和 [NET\\_DVR\\_SetDVRConfig](#)），命令：  
NET\_DVR\_GET\_ACS\_EXTERNAL\_DEV\_CFG、NET\_DVR\_SET\_ACS\_EXTERNAL\_DEV\_CFG。
- 新增人员通道参数配置功能（对应接口 [NET\\_DVR\\_GetDVRConfig](#) 和 [NET\\_DVR\\_SetDVRConfig](#)），命令：  
NET\_DVR\_GET\_PERSONNEL\_CHANNEL\_CFG、NET\_DVR\_SET\_PERSONNEL\_CHANNEL\_CFG。
- 新增报警信息上传类型（对应接口 [NET\\_DVR\\_SetDVRMessageCallBack V31](#)）：  
COMM\_ID\_INFO\_ALARM(身份证刷卡信息)、  
COMM\_PASSNUM\_INFO\_ALARM(通行人数信息)。
- [NET\\_DVR\\_ACS\\_ALARM\\_INFO](#)(门禁主机报警信息)扩展：
  - 1) dwMinor(报警次类型)新增取值，异常次类型：MINOR\_CHANNEL\_CONTROLLER\_OFF、  
MINOR\_CHANNEL\_CONTROLLER\_RESUME，事件次类型：MINOR\_CARD\_PLATFORM\_VERIFY。
  - 2) [NET\\_DVR\\_ACS\\_EVENT\\_INFO](#)(事件信息)使用 1 个保留字节新增参数：byCardReaderKind(读卡器类型)
- 门禁主机能力集([ACS\\_ABILITY](#))扩展，新增节点：<ExternalDevCfg>(串口外设参数配置能力)、  
<PersonnelChannelCfg>(人员通道参数配置能力)。
- 新增设备类型：DS\_K2602\_AX(人员通道主机)。

### Version 5.1.3.10 (build 20150720)

- 更新原因：指纹门禁一体机 DS-K1TXXX V1.0
- 新增卡号关联用户配置功能（对应接口 [NET\\_DVR\\_GetDeviceConfig](#) 和 [NET\\_DVR\\_SetDeviceConfig](#)），命令：  
NET\_DVR\_GET\_CARD\_USERINFO\_CFG、NET\_DVR\_SET\_CARD\_USERINFO\_CFG。
- [NET\\_DVR\\_SINGLE\\_PLAN\\_SEGMENT](#)(计划参数)中 byVerifyMode(验证方式)新增取值：5-指纹、6-指纹+密码、  
7-指纹或刷卡、8-指纹+刷卡、9-指纹+刷卡+密码（无先后顺序）。
- [NET\\_DVR\\_EVENT\\_LINKAGE\\_INFO](#)(事件联动参数)中 wSubEventType(事件次类型)新增取值：



- 1) 新增设备事件次类型: EVENT\_ACS\_SD\_CARD\_FULL、EVENT\_ACS\_LINKAGE\_CAPTURE\_PIC。
- 2) 新增门事件次类型: EVENT\_ACS\_REMOTE\_CAPTURE\_PIC、EVENT\_ACS\_DOORBELL\_RINGING。
- [NET\\_DVR\\_ACS\\_ALARM\\_INFO](#)(门禁主机报警信息)扩展:
  - 1) 使用 8 个保留字节新增参数: dwPicDataLen(图片数据大小)、pPicData(图片数据缓冲区)。
  - 2) dwMinor(报警信息次类型)新增取值, 报警次类型: 0x40d、0x40e, 操作次类型: 0x40b, 事件次类型: 0x25~0x31。
  - 门禁主机能力集(ACS\_ABILITY)扩展:
    - 1) 新增节点: <RealteUserInfo>(卡号关联用户配置能力)、<ContinuousShootCfg>(网络触发连拍能力)、<PictureCfg>(底图能力)。
    - 2) <CardReaderVerifyTypePlan>(读卡器验证方式计划能力)中<verifyType>(验证方式)新增取值: fingerPrint, fingerPrintAndPasswd, fingerPrintorCard, fingerPrintAndCard, fingerPrintAndCardAndPasswd。
    - 3) <AcsHostCfg>(门禁主机参数配置能力)中新增子节点: <showCapPic>(是否支持 LCD 屏上显示抓拍图片)、<showCardNo>(是否支持 LCD 屏上显示卡号)、<showUserInfo>(是否支持 LCD 屏上显示用户信息)、<overlayUserInfo>(抓拍的图片是否支持叠加用户信息)、<voicePrompt>(是否支持语音提示)、<uploadCapPic>(是否支持联动抓拍后上传图片)、<saveCapPic>(是否支持抓拍保存图片)。
    - 4) <EventLinkage>(事件及卡号联动能力)中<SubEventNameList>(事件次类型)的子节点<subEventName>(事件名称)新增取值: SDCardFull、LinkageCapturePic、RemoteCapturePic、DoorBellRing。
  - 新增设备类型: DS\_K1TXXX(指纹一体机)。

### Version 5.1.1.21 (build 20150616)

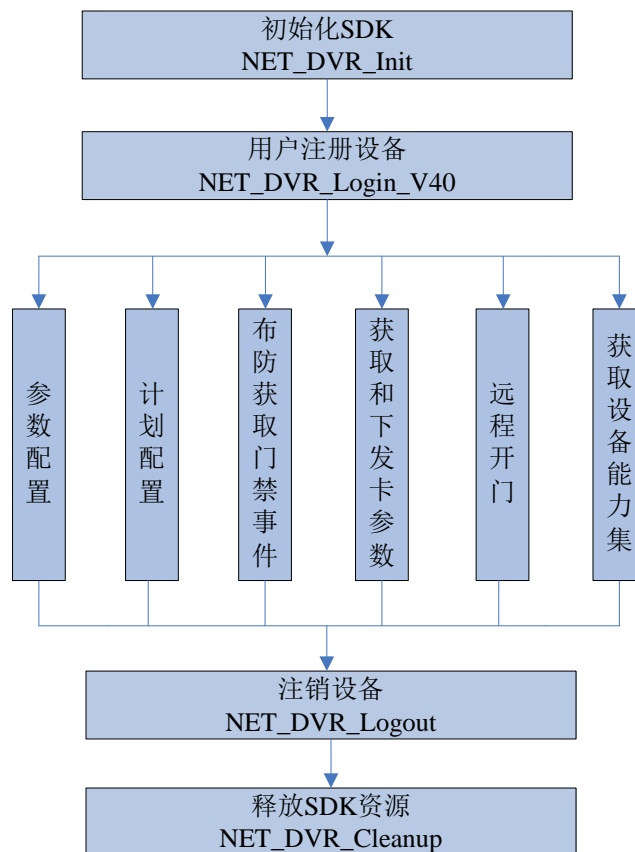
- 更新原因: 新增接口
- 新增手机关联门权限配置功能 (对应接口 [NET\\_DVR\\_GetDVRConfig](#) 和 [NET\\_DVR\\_SetDVRConfig](#)), 命令: NET\_DVR\_GET\_PHONE\_DOOR\_RIGHT\_CFG、NET\_DVR\_SET\_PHONE\_DOOR\_RIGHT\_CFG。
- 新增事件/卡号联动配置功能 (对应接口 [NET\\_DVR\\_GetDVRConfig](#) 和 [NET\\_DVR\\_SetDVRConfig](#)), 命令: NET\_DVR\_GET\_EVENT\_CARD\_LINKAGE\_CFG、NET\_DVR\_SET\_EVENT\_CARD\_LINKAGE\_CFG。
- 新增门禁主机参数配置功能 (对应接口 [NET\\_DVR\\_GetDVRConfig](#) 和 [NET\\_DVR\\_SetDVRConfig](#)), 命令: NET\_DVR\_GET\_ACS\_CFG、NET\_DVR\_SET\_ACS\_CFG。
- 新增指纹管理配置功能 (对应接口 [NET\\_DVR\\_StartRemoteConfig](#) 和 [NET\\_DVR\\_SendRemoteConfig](#)), 命令: NET\_DVR\_GET\_FINGERPRINT\_CFG、NET\_DVR\_SET\_FINGERPRINT\_CFG。
- 新增删除指纹参数功能 (对应接口 [NET\\_DVR\\_RemoteControl](#)), 命令: NET\_DVR\_DEL\_FINGERPRINT\_CFG。
- 新增卡密码开门使能配置功能 (对应接口 [NET\\_DVR\\_StartRemoteConfig](#) 和 [NET\\_DVR\\_SendRemoteConfig](#)), 命令: NET\_DVR\_GET\_CARD\_PASSWD\_CFG、NET\_DVR\_SET\_CARD\_PASSWD\_CFG。
- [NET\\_DVR\\_SINGLE\\_PLAN\\_SEGMENT](#)(计划参数)中 byVerifyMode(验证方式)新增取值: 3-刷卡、4-刷卡或密码。
- [NET\\_DVR\\_ACS\\_WORK\\_STATUS](#)(工作状态), byDoorStatus(门状态)取值修改为: 1-休眠、2-常开状态、3-常闭状态、4-普通状态, byCardReaderVerifyMode(读卡器当前验证方式)取值修改为: 1-休眠、2-刷卡+密码、3-刷卡、4-刷卡或密码。
- [NET\\_DVR\\_ALARMHOST\\_NETPARAM](#)(上传中心网络参数)扩展:
  - 1) byReportProtocol(报告协议)新增取值: 3- ehome。
  - 2) byDevID[ACCOUNTNUM\_LEN\_V40]扩展为 byDevID[ACCOUNTNUM\_LEN\_32], 即长度从 9 字节扩展为 32 字节。
- [NET\\_DVR\\_PHONECFG](#)(电话配置)中 byPhonePerssion(号码权限)新增类型: 0x4 表示支持门操作控制。
- [NET\\_DVR\\_CARD\\_READER\\_CFG](#)(读卡器参数配置)扩展:

- 1) byCardReaderType(读卡器类型)新增取值：6~32。
- 2) 使用 1 个保留字节新增参数：byFingerPrintCheckLevel(指纹识别等级)。
  - [NET\\_DVR\\_CARD\\_CFG](#)(卡参数)中 byCardType(卡类型)新增取值：8- 解除卡。
  - [NET\\_DVR\\_DOOR\\_CFG](#)(门参数)使用 8 个保留字节新增参数：
    - byUnlockPassword[UNLOCK\_PASSWORD\_LEN](解除码)。
  - [NET\\_DVR\\_ACS\\_PARAM\\_TYPE](#)(清空门禁主机参数)中 dwParamType(参数类型)新增取值：
    - ACS\_PAPAM\_EVENT\_CARD\_LINKAGE(事件及卡号联动参数)、ACS\_PAPAM\_CARD\_PASSWD\_CFG(密码开门使能参数)。
  - 门禁主机报警上传扩展：
    - 1) [NET\\_DVR\\_ACS\\_EVENT\\_INFO](#)(门禁主机事件信息)使用 2 个保留字节新增参数：byWhiteListNo(白名单单号)、byReportChannel(报告上传通道)。
    - 2) [NET\\_DVR\\_ACS\\_ALARM\\_INFO](#)(门禁主机事件信息)中 dwMinor 新增操作报警次类型：0x40c~0x40f。
  - 门禁主机能力集(ACS\_ABILITY)扩展：
    - 1) 新增节点：<AcsHostCfg>(门禁主机参数配置能力)、<EventLinkage>(事件及卡号联动配置能力)、<FingerPrint>(指纹参数配置能力)、<SMS>(手机短信能力)。
    - 2) <CardReaderVerifyTypePlan>(读卡器验证方式计划能力)中<verifyType>(验证方式)新增取值：swipecardorpasswd。
    - 3) <Door>(门参数能力)新增节点：<unlockPassword>(解锁密码长度)。
    - 4) <Card>(卡参数能力)新增节点：<onlyPasswdOpen>(密码开门能力)。
    - 5) <CardReaderCfg>(读卡器参数能力)中<cardReaderType>(读卡器类型)新增取值：DS-K182AMF/ACF,韦根或 485 不在线,DS-K1101M/MK,DS-K1101C/CK,DS-K1102M/MK/M-A,DS-K1102C/CK,DS-K1103M/MK,DS-K1103C/CK,DS-K1104M/MK,DS-K1104C/CK,DS-K1102S/SK/S-A,DS-K1102G/GK,DS-K1100S-B,DS-K1102EM/EMK,DS-K1102E/EK,DS-K1200EF,DS-K1200MF,DS-K1200CF,DS-K1300EF,DS-K1300MF,DS-K1300CF,DS-K1105E,DS-K1105M,DS-K1105C,DS-K182AMF,DS-K196AMF,DS-K194AMP。
    - 6) <clearAcsParam>(清空门禁参数能力)新增取值：,eventandCardLinkage,cardPasswdOpendoor。

## 3 函数调用顺序

### 3.1 SDK 接口调用主要流程

图 3.1 SDK 调用主要流程

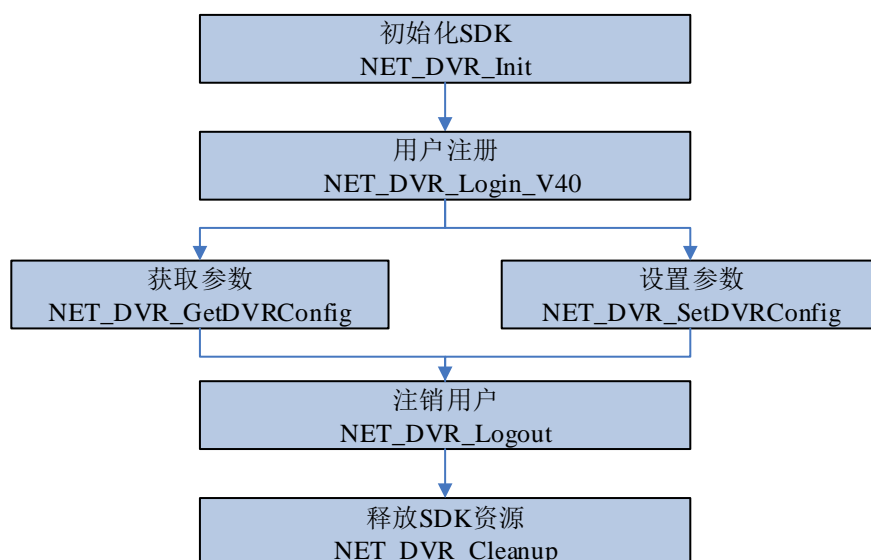


门禁主机主要包含以上几个功能模块，所有功能都需要先进行用户注册，[NET\\_DVR\\_Login\\_V40](#) 接口返回的用户 ID 号作为其他功能接口的参数。

- 参数配置：主要是门参数配置，读卡器参数配置，以及其他一些通用参数或门禁相关参数配置。[详见 3.2 门参数/读卡器参数配置流程。](#)
- 计划配置：主要包含门状态计划、读卡器验证方式计划、以及卡权限计划模板的配置。调用的接口为 [NET\\_DVR\\_GetDVRConfig](#) 和 [NET\\_DVR\\_SetDVRConfig](#)。如果已经通过配套客户端软件完成配置，则不需要调用这些接口再次配置。[详见 3.3 计划配置流程。](#)
- 布防获取门禁事件：通过对设备布防接收设备上传的门禁相关报警信息。[详见 3.4 布防获取门禁事件流程。](#)
- 获取和下发卡参数：获取和设置卡信息（如权限、有效期、密码、类型等）。[详见 3.5 获取和下发卡参数流程。](#)
- 远程开门：通过 [NET\\_DVR\\_ControlGateway](#) 远程控制开门。[详见 3.6 远程开门控制流程。](#)
- 获取设备能力集：通过 [NET\\_DVR\\_GetDeviceAbility](#) 接口门禁主机的能力信息。

## 3.2 门参数/读卡器参数配置流程

图 3.2 门参数/读卡器参数配置流程



- 建议每次在设置某类参数之前，先调用对应获取接口（[NET\\_DVR\\_GetDVRConfig](#)）得到完整的参数结构，修改需要更改的参数，作为设置接口中的输入参数，最后调用设置接口（[NET\\_DVR\\_SetDVRConfig](#)）进行设置，返回成功即设置成功。
- 门参数配置时对应接口及命令分别为：获取 [NET\\_DVR\\_GetDVRConfig](#)（NET\_DVR\_GET\_DOOR\_CFG），设置 [NET\\_DVR\\_SetDVRConfig](#)（NET\_DVR\_SET\_DOOR\_CFG）。

[调用实例代码](#)

- 读卡器参数配置时对应接口及命令分别为：获取 [NET\\_DVR\\_GetDVRConfig](#)（NET\_DVR\_GET\_CARD\_READER\_CFG\_V50），设置 [NET\\_DVR\\_SetDVRConfig](#)（NET\_DVR\_SET\_CARD\_READER\_CFG\_V50）。

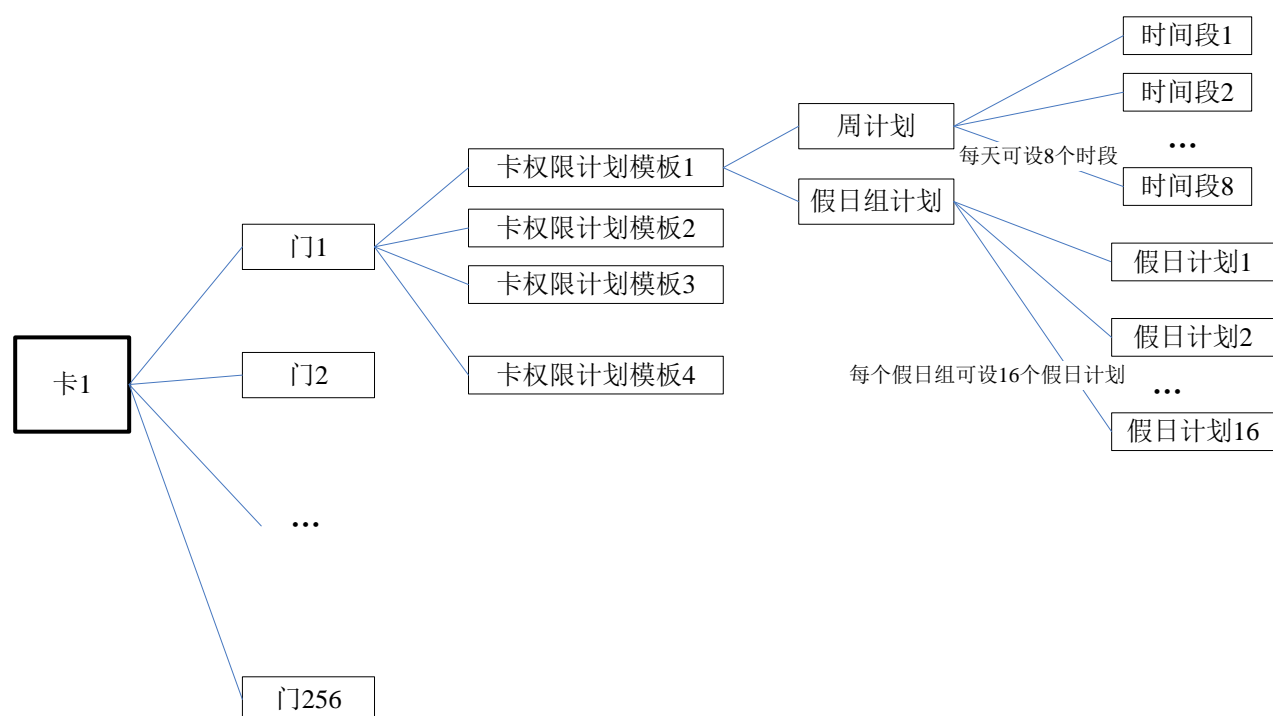
[调用实例代码](#)

- 配置其他参数时对应不同的命令和结构体，详见 [NET\\_DVR\\_GetDVRConfig](#)、[NET\\_DVR\\_SetDVRConfig](#) 接口下方表格。

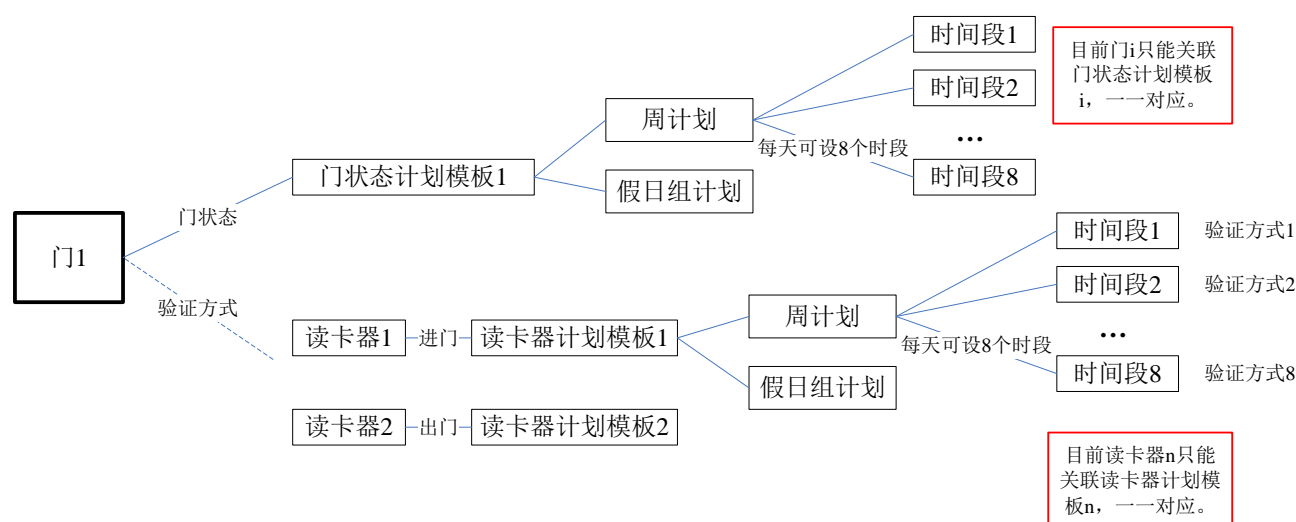


### 3.3 计划配置流程

说明一：门、读卡器、卡之间关联关系如下。



注：每张卡可根据需求对设备的每扇门配置卡权限计划模板，而且对应每个门最多可设置 4 个卡权限计划模板（不同模板间采用权限或的方式处理），其中需要卡权限周计划和卡权限假日组计划。



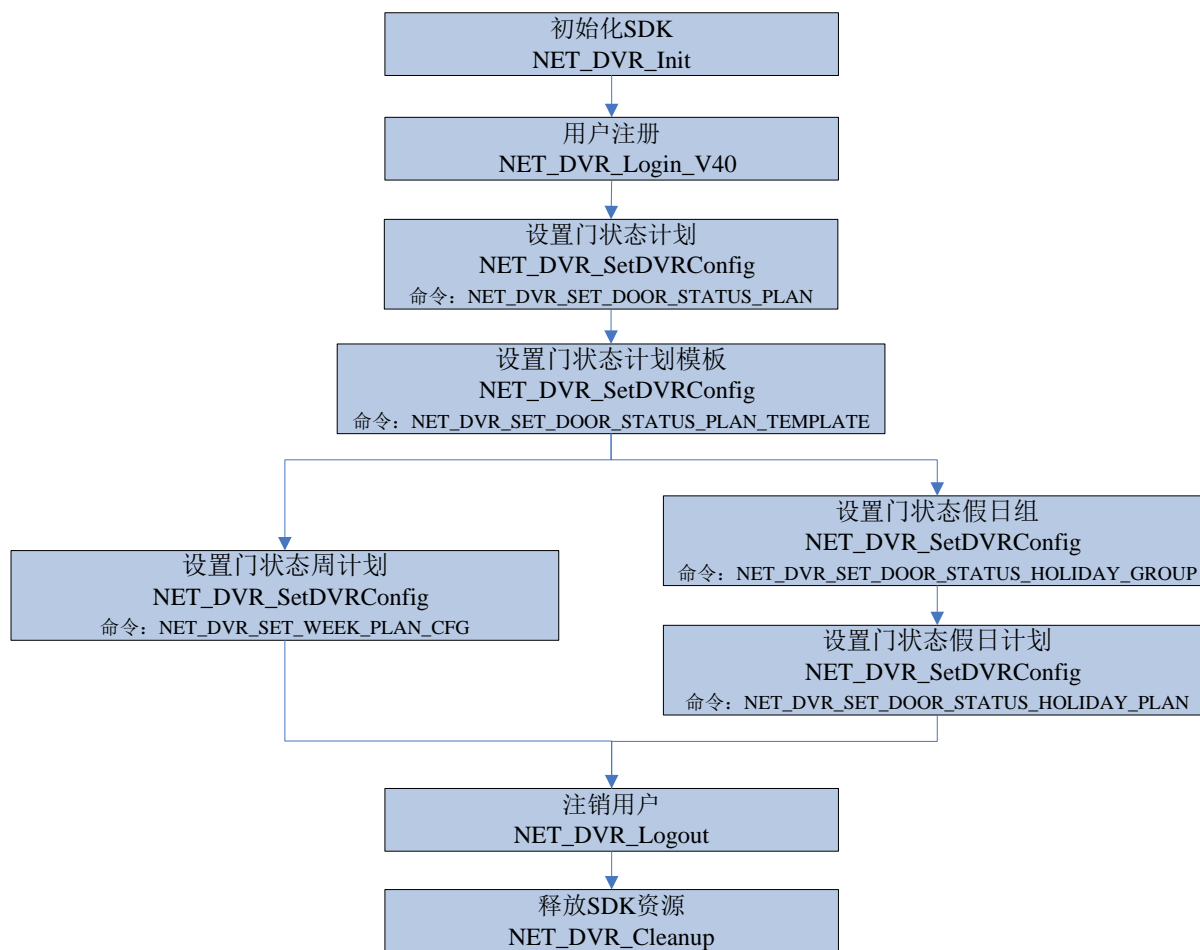
注：一个门只能设置一个门状态计划模板，其中需要门状态周计划和门状态假日组计划。一个门对应两个读卡器，门 1 对应读卡器 1（进门）、读卡器 2（出门），门 2 对应读卡器 3（进门）、读卡器 4（出门），以此类推。

门 i 只能关联门状态计划模板 i，一一对应；读卡器 n 只能关联读卡器计划模板 n，也是一一对应。

**说明二：**计划配置包含门状态计划配置、读卡器验证方式计划配置、卡权限计划模板配置，对应设置接口为 [NET\\_DVR\\_SetDVRConfig](#)，下文会列出详细流程图，但是建议在设置任一参数前先调用 [NET\\_DVR\\_GetDVRConfig](#) 获取一下该参数，然后修改需要设置的字段。

### 3.3.1 门状态计划配置流程

图 3.3.1 门状态计划配置流程

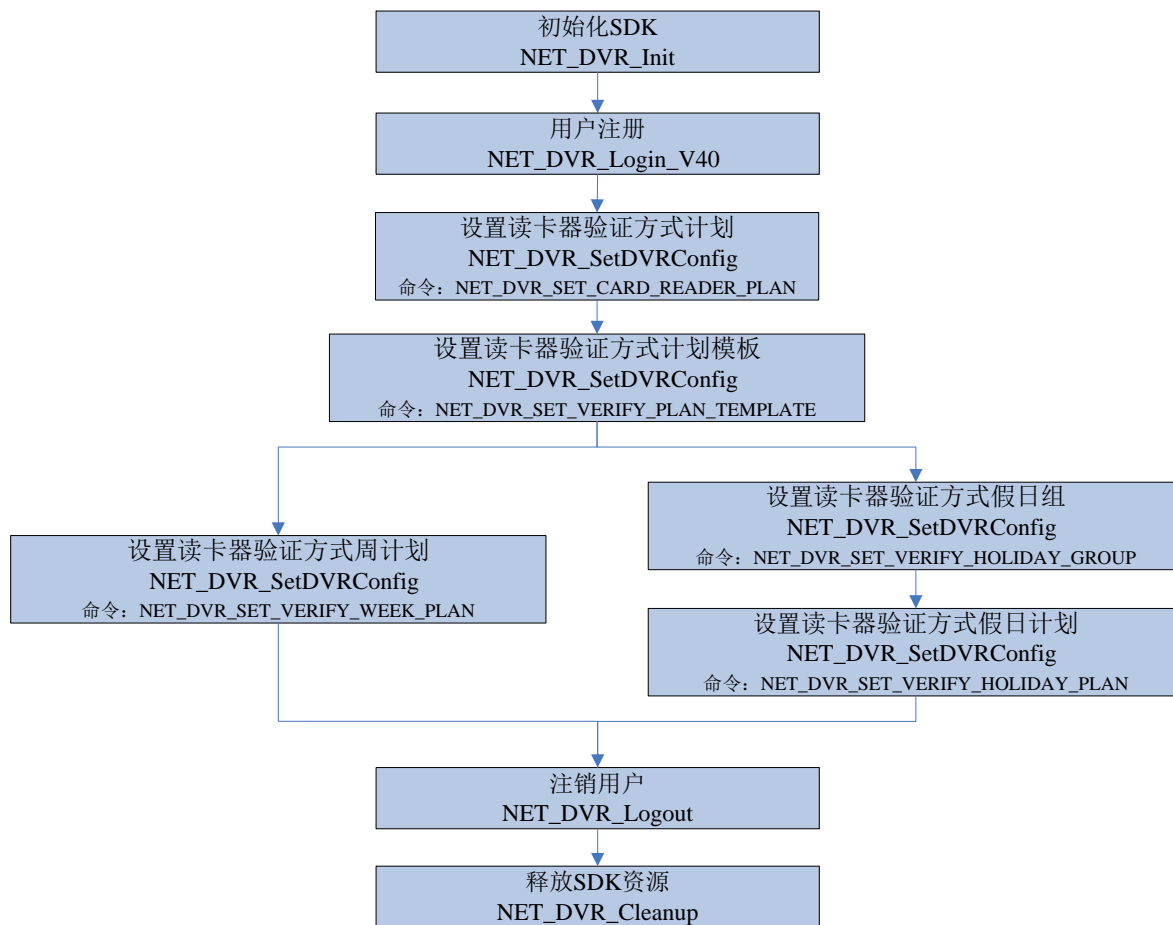


- 建议每次在设置某类参数之前，先调用对应获取接口（[NET\\_DVR\\_GetDVRConfig](#)）得到完整的参数结构，修改需要更改的参数，作为设置接口中的输入参数，最后调用设置接口（[NET\\_DVR\\_SetDVRConfig](#)）进行设置，返回成功即设置成功。
- 门状态计划：设置门状态计划（[NET\\_DVR\\_SET\\_DOOR\\_STATUS\\_PLAN](#)）<——设置门状态计划模板（[NET\\_DVR\\_SET\\_DOOR\\_STATUS\\_PLAN\\_TEMPLATE](#)）<——设置门状态周计划（[NET\\_DVR\\_SET\\_WEEK\\_PLAN\\_CFG](#)）、设置门状态假日组（[NET\\_DVR\\_SET\\_DOOR\\_STATUS\\_HOLIDAY\\_GROUP](#)）<——设置门状态假日计划（[NET\\_DVR\\_SET\\_DOOR\\_STATUS\\_HOLIDAY\\_PLAN](#)）。

[调用实例代码](#)

### 3.3.2 读卡器验证方式计划配置流程

图 3.3.2 读卡器验证方式计划配置流程



- 读卡器验证方式计划：设置读卡器验证方式计划（[NET\\_DVR\\_SET\\_CARD\\_READER\\_PLAN](#)）<——设置读卡器验证方式计划模板（[NET\\_DVR\\_SET\\_VERIFY\\_PLAN\\_TEMPLATE](#)）<——设置读卡器验证方式周计划（[NET\\_DVR\\_SET\\_VERIFY\\_WEEK\\_PLAN](#)）、设置读卡器验证方式假日组（[NET\\_DVR\\_SET\\_VERIFY\\_HOLIDAY\\_GROUP](#)）<——设置读卡器验证方式假日计划（[NET\\_DVR\\_SET\\_VERIFY\\_HOLIDAY\\_PLAN](#)）。

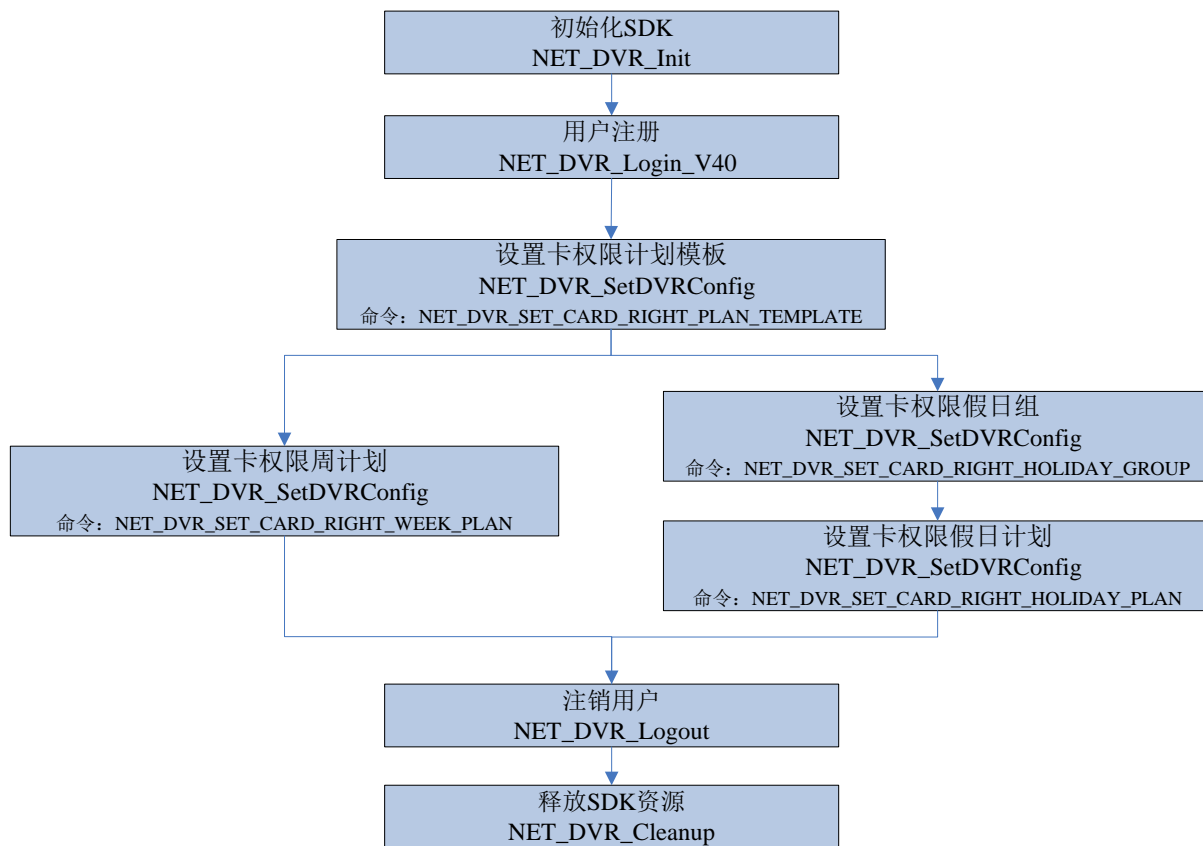
[调用实例代码](#)

**注：**门状态计划、读卡器验证计划、卡权限计划模板配置中都包含了 [NET\\_DVR\\_SINGLE\\_PLAN\\_SEGMENT](#) 结构体。该结构体中的门状态模式（byDoorStatus）仅供门状态计划配置使用，该结构体中的验证方式（byVerifyMode）仅供读卡器验证计划使用。

也可以通过门禁能力集中的<DoorStatusPlan><!--req,门状态计划能力>、<CardReaderVerifyTypePlan><!--req,读卡器验证方式计划能力>、<CardRightPlan><!--req,卡权限计划能力>子节点来判断这三个配置支持的字段。

### 3.3.3 卡权限计划模板配置流程

图 3.3.3 卡权限计划模板配置流程

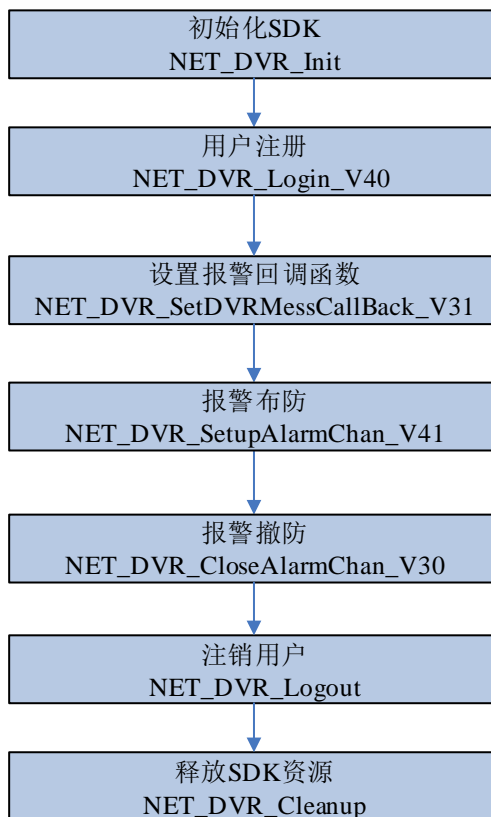


- 建议每次在设置某类参数之前，先调用对应获取接口（[NET\\_DVR\\_GetDVRConfig](#)）得到完整的参数结构，修改需要更改的参数，作为设置接口中的输入参数，最后调用设置接口（[NET\\_DVR\\_SetDVRConfig](#)）进行设置，返回成功即设置成功。
- 卡权限计划：设置卡权限计划模板（[NET\\_DVR\\_SET\\_CARD\\_RIGHT\\_PLAN\\_TEMPLATE](#)）<——设置卡权限周计划（[NET\\_DVR\\_SET\\_CARD\\_RIGHT\\_WEEK\\_PLAN](#)）、设置卡权限假日组（[NET\\_DVR\\_SET\\_CARD\\_RIGHT\\_HOLIDAY\\_GROUP](#)）<——设置卡权限假日计划（[NET\\_DVR\\_SET\\_CARD\\_RIGHT\\_HOLIDAY\\_PLAN](#)）
- 这里只是配置好了卡权限计划模板，发卡时关联配置的模板那么该卡就拥有了配置的权限。

[调用实例代码](#)

### 3.4 布防获取门禁事件流程

图 3.4 布防获取门禁事件流程



- “布防”是指 SDK 主动连接设备，并发起报警上传命令，设备发生刷卡等事件时会立即上传给 SDK。
- “布防”方式需要先进行用户注册（[NET\\_DVR\\_Login\\_V40](#)），接下来就是设置报警回调函数（[NET\\_DVR\\_SetDVRMessageCallBack\\_V31](#)），调用成功后再启用布防（[NET\\_DVR\\_SetupAlarmChan\\_V41](#)）。
- [NET\\_DVR\\_CloseAlarmChan\\_V30](#) 撤销报警上传通道，设备将不再上传门禁事件报警信息。
- 门禁主机支持的报警信息类型有：门禁主机报警信息（COMM\_ALARM\_ACS）、门禁身份证刷卡信息（COMM\_ID\_INFO\_ALARM）、门禁通行人数信息（COMM\_PASSNUM\_INFO\_ALARM）。

[调用实例代码](#)

### 3.5 获取和下发卡号功能流程

图 3.5.1 获取卡号参数

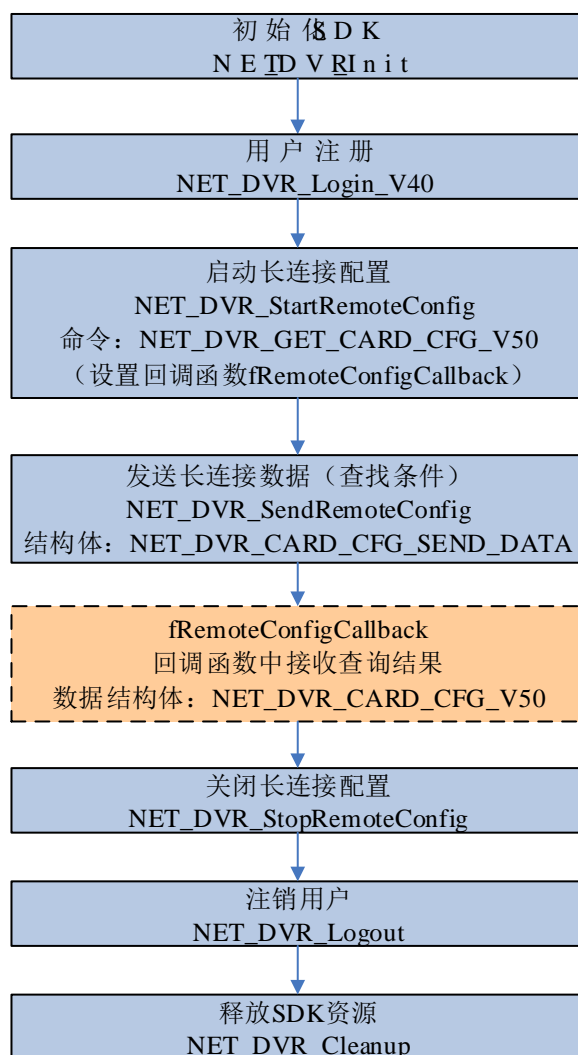
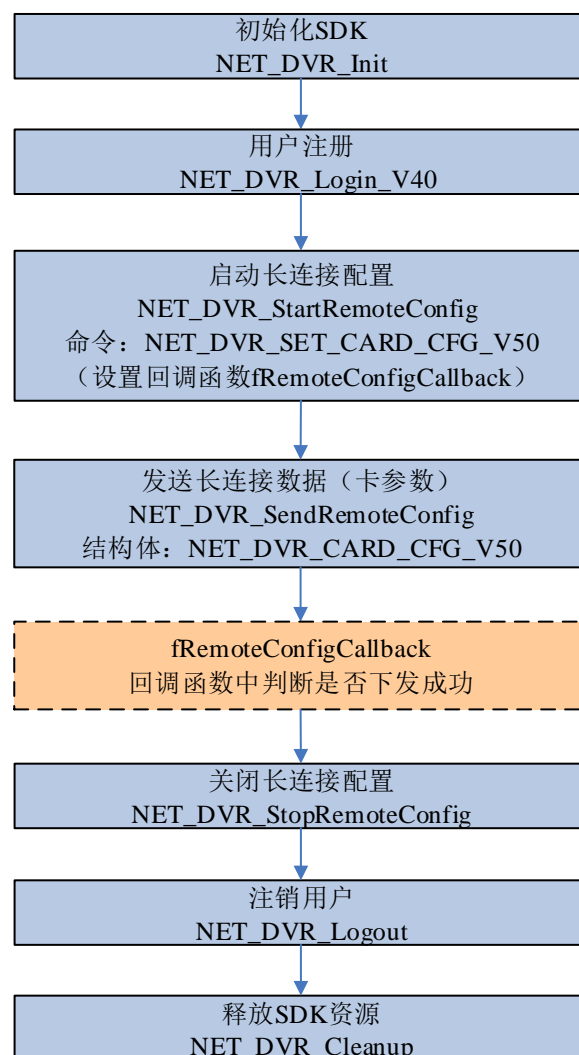


图 3.5.2 下发卡号



- NET\_DVR\_GET\_CARD\_CFG\_V50 获取卡参数时，在调用启动长连接配置后，还需要调用 NET\_DVR\_SendRemoteConfig 发送查找条件数据（pSendBuf 为查找条件，获取所有卡参数时不需要调用此发送接口），查找结果在 NET\_DVR\_StartRemoteConfig 设置的回调函数中返回。
- NET\_DVR\_SET\_CARD\_CFG\_V50 设置卡参数时，在调用启动长连接配置后，也需要调用 NET\_DVR\_SendRemoteConfig 向设备下发卡参数（pSendBuf 为下发的卡参数信息）。建立长连接的时候，NET\_DVR\_CARD\_CFG\_COND 里面 byCheckCardNo 参数建议赋值为 1，否则必须用户上层自己保证卡号的唯一性，即卡号不能重复，如果重复下发则卡参数会出现异常。
- 批量下发卡号实现方法：
  - NET\_DVR\_StartRemoteConfig 建立长连接时 NET\_DVR\_CARD\_CFG\_COND 里面的 dwCardNum 为批量发的卡号个数。
  - 建立一次连接后，调用 NET\_DVR\_SendRemoteConfig 单个下发卡参数，每次发送 pSendBuf 对应 NET\_DVR\_CARD\_CFG\_V50 单个结构体，每次调用 NET\_DVR\_SendRemoteConfig 后，要通过回调函数获取这次发送数据的返回值，下发完成之后再循环调用 NET\_DVR\_SendRemoteConfig 发送下一个。循环调用 dwCardNum 次 NET\_DVR\_SendRemoteConfig。

3) 全部下发完成之后调用 `NET_DVR_StopRemoteConfig` 断开长连接。

### 注意：

- 下发卡号的时候需要关联门权限和关联的计划模板，计划模板配置请参考 3.3.3 章节。
- 使用卡号关联用户信息参数获取/设置接口时要注意：由于卡参数 `V50` 接口中包含了 `byName` 字段，该字段即为卡号关联用户信息参数接口的 `sUsername`。在调用卡号关联用户信息接口前，首先获取门禁能力集，如果能力集中 `<Card>` 节点下包含了 `<name>` 子节点，说明卡参数 `V50` 接口支持用户名字段。则可不必要调用卡号关联用户信息参数接口获取用户名，直接通过卡参数 `V50` 接口便可获取相应卡对应的用户名。

[调用实例代码](#)

## 3.6 获取和下发人脸参数流程

图 3.3 获取人脸参数

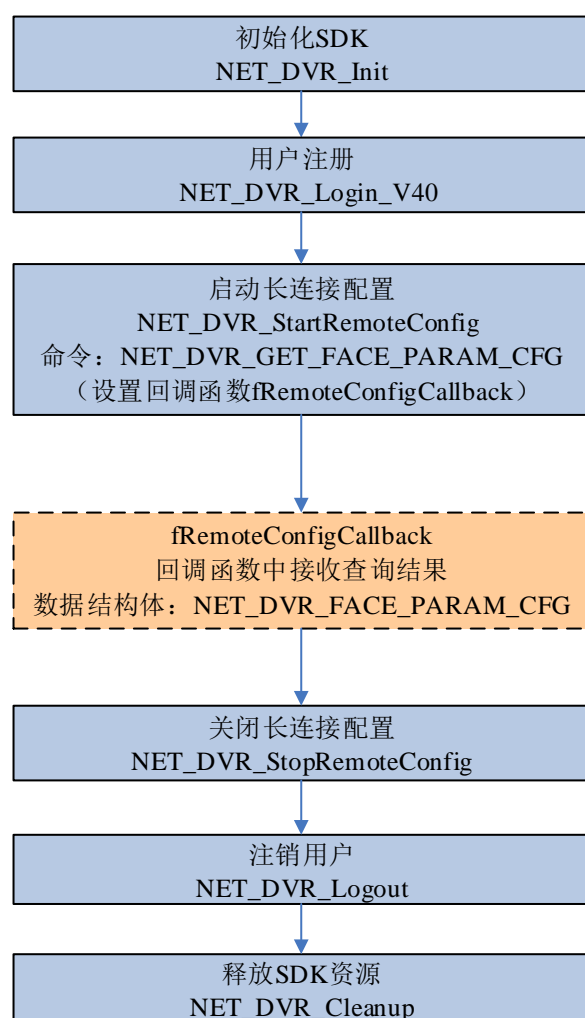
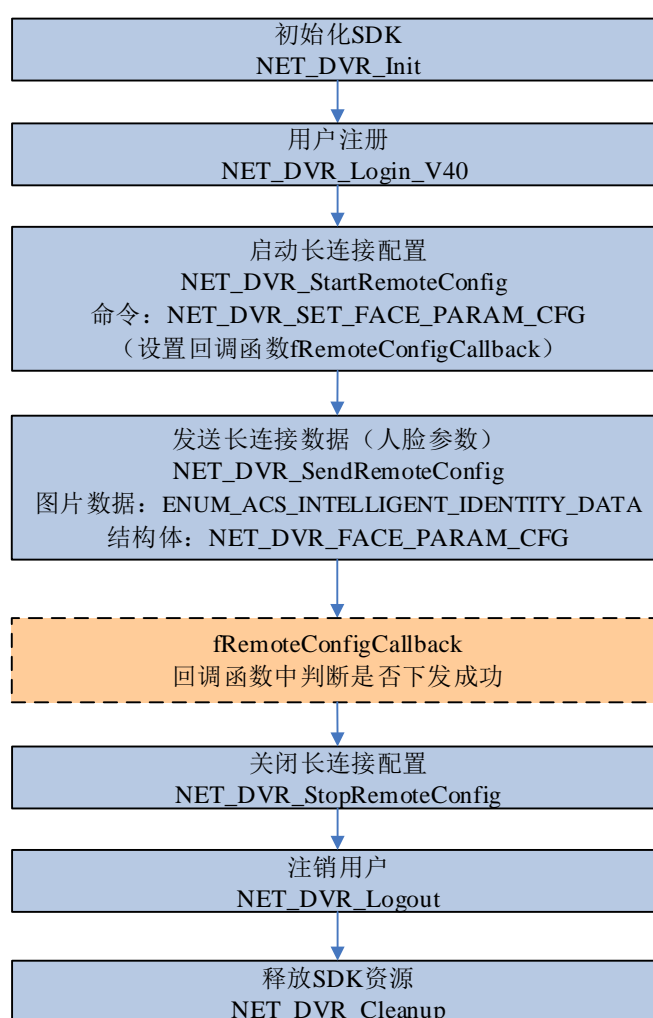


图 3.4 下发人脸图片



人脸图片是跟卡号关联的，需要先下发卡号，然后通过卡号下发关联的人脸图片，卡号获取和下发请参考 3.5 章节。

- `NET_DVR_GET_FACE_PARAM_CFG` 获取人脸参数时，在调用该接口启动长连接远程配置后，查找结果直接在 `NET_DVR_StartRemoteConfig` 设置的回调函数中返回。

- NET\_DVR\_SET\_FACE\_PARAM\_CFG 设置人脸参数时，在调用该接口启动长连接远程配置后，还需要调用 NET\_DVR\_SendRemoteConfig 向设备下发人脸参数信息。

### 3.7 获取和下发指纹参数流程

图 3.5 获取指纹参数

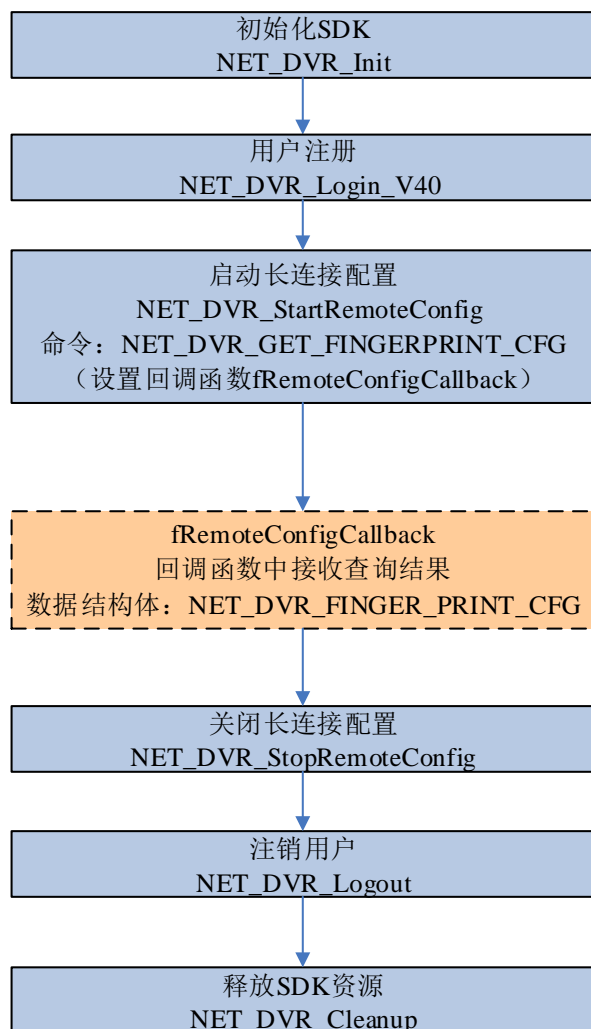
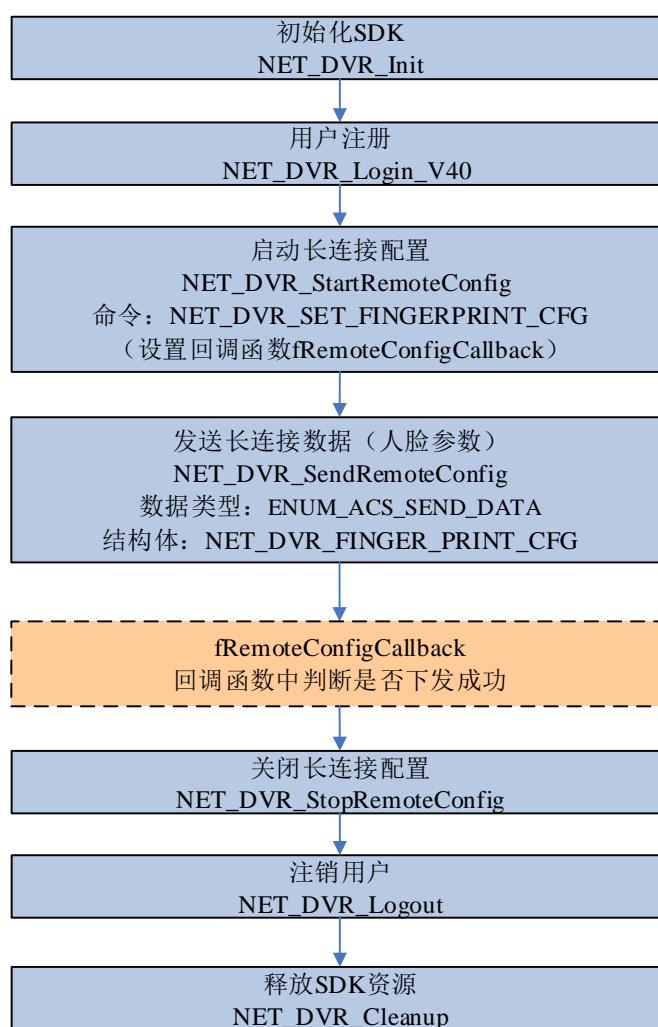


图 3.6 下发指纹数据



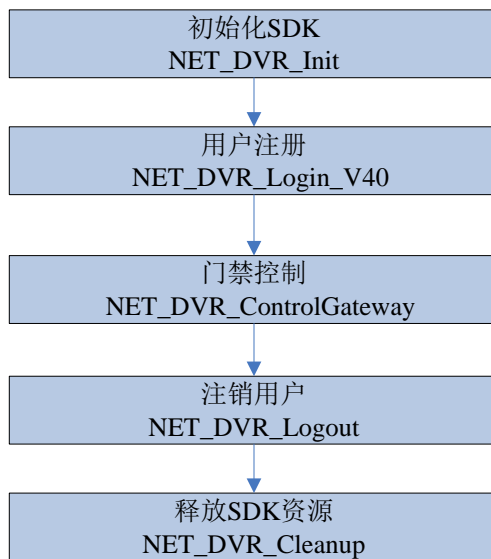
指纹是跟卡号关联的，需要先下发卡号，然后通过卡号下发关联的指纹数据，卡号获取和下发请参考 3.5 章节。

- NET\_DVR\_GET\_FINGERPRINT\_CFG 获取指纹参数时，调用该接口启动查询，查找结果在 NET\_DVR\_StartRemoteConfig 设置的回调函数中返回。
- NET\_DVR\_SET\_FINGERPRINT\_CFG 设置指纹参数时，在调用该接口启动长连接远程配置后，通过调用 NET\_DVR\_SendRemoteConfig 向设备下发指纹参数信息，在 NET\_DVR\_StartRemoteConfig 设置的回调函数中返回状态信息。



### 3.8 远程开门控制流程

图 3.6 门禁控制流程



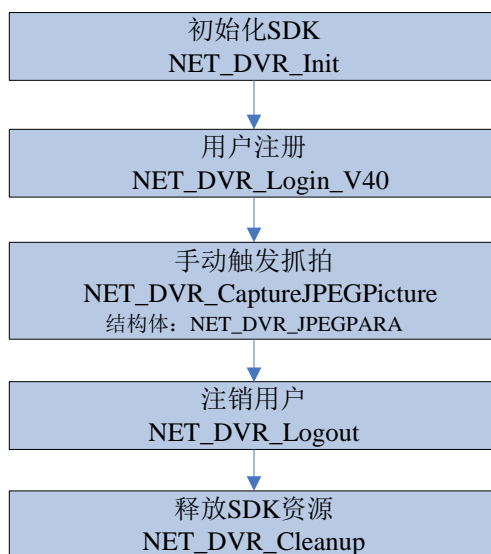
- 用户注册（[NET\\_DVR\\_Login\\_V40](#)）成功之后即可通过门禁控制接口 [NET\\_DVR\\_ControlGateway](#) 来控制门禁的开启和关闭。

**注：**在给设备配置了多重卡认证模式后，只有 [NET\\_DVR\\_GROUP\\_COMBINATION\\_INFO](#) 结构体中 dwGroupNo 字段设置成 0xffffffff 表示远程开门后，远程开门才能使用。否则远程开门这个接口则会操作失败。

[调用实例代码](#)

### 3.9 手动抓拍流程

图 3.7 手动抓拍流程



- 用户注册（[NET\\_DVR\\_Login\\_V40](#)）成功之后即可通过 JPEG 抓图接口 [NET\\_DVR\\_CaptureJPEGPicture](#) 实现手动抓拍图。

### 3.10 联动抓拍流程

图 3.8 联动抓拍流程



- 联动抓拍流程中：首先使用配置触发抓拍参数接口来配置相应的抓拍图片参数；再使用事件及卡号联动参数配置接口中的 `byCapturePic` 字段和 `uLinkageInfo` 字段关联对应的联动事件类型；然后使用门禁主机参数配置接口中的 `byUploadCapPic` 字段设置联动抓拍的图片是否上传。
- 这样配置好后，设备联动抓拍的图片便可以通过事件上传的方式传递给客户端了。就是说事先还需要先对设备进行布防（[NET\\_DVR\\_SetupAlarmChan\\_V41](#)）并设置回调函数（[NET\\_DVR\\_SetDVRMessageCallBack\\_V31](#)），通过回调获取联动抓拍上传的图片数据，回调的报警类型为 `COMM_ALARM_ACS`。

[调用实例代码](#)

## 4 函数调用实例

### 4.1 门参数配置示例代码

[相关模块流程图](#)

```
#include <stdio.h>
#include <iostream>
#include <afx.h>
#include "Windows.h"
#include "HCNetSDK.h"
using namespace std;

CString    m_csDoorName;
CString    m_csStressPasswd;
CString    m_csSuperPasswd;
CString    m_csUnlockPassword;

void main()
{
    //-----
    //初始化
    NET_DVR_Init();

    //设置连接超时时间与重连功能
    NET_DVR_SetConnectTime(2000, 1);
    NET_DVR_SetReconnect(10000, true);

    //-----
    //注册设备
    LONG lUserID;
    //登录参数，包括设备地址、登录用户、密码等
    NET_DVR_USER_LOGIN_INFO struLoginInfo = {0};
    struLoginInfo.bUseAsynLogin = 0; //同步登录方式
    strcpy(struLoginInfo.sDeviceAddress, "192.0.0.64"); //设备 IP 地址
    struLoginInfo.wPort = 8000; //设备服务端口
    strcpy(struLoginInfo.sUserName, "admin"); //登录用户名
    strcpy(struLoginInfo.sPassword, "abcd1234"); //登录密码

    //设备信息，输出参数
    NET_DVR_DEVICEINFO_V40 struDeviceInfoV40 = {0};

    lUserID = NET_DVR_Login_V40(&struLoginInfo, &struDeviceInfoV40);
    if (lUserID < 0)
```

```

{
    printf("Login failed, error code: %d\n", NET_DVR_GetLastError());
    NET_DVR_Cleanup();
    return;
}

//获取门 1 参数
DWORD dwReturnLen;
NET_DVR_DOOR_CFG struDoorCfg = {0};
BOOL bRet1 = NET_DVR_GetDVRConfig(IUserID, NET_DVR_GET_DOOR_CFG, 1, &struDoorCfg, \
    sizeof(NET_DVR_DOOR_CFG), &dwReturnLen);
if (!bRet1)
{
    printf("获取门参数失败, error:%d.\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

//-----
//设置门 1 参数
m_csDoorName = "东门";           //门名称
m_csStressPasswd = "11111111";   //胁迫密码
m_csSuperPasswd = "22222222";   //超级密码
m_csUnlockPassword = "33333333"; //解除码

struDoorCfg.dwSize = sizeof(NET_DVR_DOOR_CFG);
strncpy((char *)struDoorCfg.byDoorName, (LPCTSTR)m_csDoorName, DOOR_NAME_LEN);
struDoorCfg.byMagneticType = 1;    //门磁类型: 0- 常闭, 1- 常开, 根据实际设置
struDoorCfg.byOpenButtonType = 1; //开门按钮类型: 0- 常闭, 1- 常开, 根据实际设置
struDoorCfg.byOpenDuration = 10;  //开门持续时间, 取值范围: 1~255s
struDoorCfg.byDisabledOpenDuration = 20; //残疾人卡开门持续时间, 取值范围: 1~255s
struDoorCfg.byMagneticAlarmTimeout = 0; //门磁检测超时报警时间, 取值范围: 0~255s, 0 表示不报警
struDoorCfg.byEnableDoorLock = 1;  //是否启用闭门回锁: 0- 否, 1- 是
struDoorCfg.byEnableLeaderCard = 1; //是否启用首卡常开功能: 0- 否, 1- 是
struDoorCfg.byLeaderCardMode = 1; //首卡模式, 0-不启用首卡功能, 1-首卡常开模式, 2-首卡授权模式 (使用了此
//字段, 则 byEnableLeaderCard 无效)
struDoorCfg.byLockInputCheck = 1; //是否启用门锁输入检测: 0-不启用, 1-启用, 默认不启用)
struDoorCfg.byLockInputType = 0; //门锁输入类型: 0-常闭, 1-常开, 默认常闭)
struDoorCfg.dwLeaderCardOpenDuration = 60; //首卡常开持续时间, 取值范围: 1~1440, 单位: min (分钟)
strncpy((char *)struDoorCfg.byStressPassword, (LPCTSTR)m_csStressPasswd, STRESS_PASSWORD_LEN);
strncpy((char *)struDoorCfg.bySuperPassword, (LPCTSTR)m_csSuperPasswd, SUPER_PASSWORD_LEN);
strncpy((char *)struDoorCfg.byUnlockPassword, (LPCTSTR)m_csUnlockPassword, UNLOCK_PASSWORD_LEN);

BOOL bRet2 = NET_DVR_SetDVRConfig(IUserID, NET_DVR_SET_DOOR_CFG, 1, \

```

```
        &struDoorCfg, sizeof(NET_DVR_DOOR_CFG));

    if (!bRet2)
    {
        printf("设置门参数失败, error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(IUserID);
        NET_DVR_Cleanup();
        return;
    }

    //-----
    //退出
    Sleep(5000);

    //注销用户
    NET_DVR_Logout(IUserID);
    //释放 SDK 资源
    NET_DVR_Cleanup();
    return;
}
```

## 4.2 读卡器参数配置示例代码

[相关模块流程图](#)

```
#include <stdio.h>
#include <iostream>
#include <afx.h>
#include "Windows.h"
#include "HCNetSDK.h"
using namespace std;

void main()
{
    //-----
    //初始化
    NET_DVR_Init();

    //设置连接超时时间与重连功能
    NET_DVR_SetConnectTime(2000, 1);
    NET_DVR_SetReconnect(10000, true);

    //-----
    //注册设备
    LONG IUserID;
    //登录参数, 包括设备地址、登录用户、密码等
```

```
NET_DVR_USER_LOGIN_INFO struLoginInfo = {0};
struLoginInfo.bUseAsynLogin = 0; //同步登录方式
strcpy(struLoginInfo.sDeviceAddress, "192.0.0.64"); //设备 IP 地址
struLoginInfo.wPort = 8000; //设备服务端口
strcpy(struLoginInfo.sUserName, "admin"); //登录用户名
strcpy(struLoginInfo.sPassword, "abcd1234"); //登录密码

//设备信息, 输出参数
NET_DVR_DEVICEINFO_V40 struDeviceInfoV40 = {0};

UserID = NET_DVR_Login_V40(&struLoginInfo, &struDeviceInfoV40);
if (UserID < 0)
{
    printf("Login failed, error code: %d\n", NET_DVR_GetLastError());
    NET_DVR_Cleanup();
    return;
}

//获取读卡器 1 参数
DWORD dwReturnLen;
NET_DVR_CARD_READER_CFG_V50 struReaderCfg = {0};
BOOL bRet1 = NET_DVR_GetDVRConfig(UserID, NET_DVR_GET_CARD_READER_CFG_V50, 1, &struReaderCfg, \
    sizeof(struReaderCfg), &dwReturnLen);

if (!bRet1)
{
    printf("获取读卡器参数失败, error:%d.\n", NET_DVR_GetLastError());
    NET_DVR_Logout(UserID);
    NET_DVR_Cleanup();
    return;
}

//-----
//设置读卡器 1 参数, 根据获取值修改需要设置的字段
struReaderCfg.dwSize = sizeof(struReaderCfg);
struReaderCfg.byEnable = 1; //是否使能: 0- 不启用, 1- 启用
//struReaderCfg.byCardReaderType; //读卡器类型, 根据实际设置
//struReaderCfg.byOkLedPolarity = 0; //OK LED 极性: 0- 阴极, 1- 阳极, 根据实际设置
//struReaderCfg.byErrorLedPolarity = 0; //Error LED 极性: 0- 阴极, 1- 阳极, 根据实际设置
//struReaderCfg.byBuzzerPolarity = 0; //蜂鸣器极性: 0- 阴极, 1- 阳极, 根据实际设置
struReaderCfg.bySwipeInterval = 5; //重复刷卡间隔时间, 单位: 秒
struReaderCfg.byPressTimeout = 5; //按键超时时间, 单位: 秒, 取值范围: 1~255
struReaderCfg.byEnableFailAlarm = 1; //是否启用读卡失败超次报警: 0- 不启用, 1- 启用
struReaderCfg.byMaxReadCardFailNum = 8; //最大读卡失败次数, 取值范围: 1~10
struReaderCfg.byEnableTamperCheck = 1; //是否启用防拆检测: 0- 不启用, 1- 启用
struReaderCfg.byOfflineCheckTime = 30; //掉线检测时间, 单位: 秒, 取值范围: 0~255
```

```

        BOOL bRet2 = NET_DVR_SetDVRConfig(IUserID, NET_DVR_SET_CARD_READER_CFG_V50, 1, \
            &struReaderCfg, sizeof(struReaderCfg));

        if (!bRet2)
        {
            printf("设置读卡器参数失败, error:%d.\n", NET_DVR_GetLastError());
            NET_DVR_Logout(IUserID);
            NET_DVR_Cleanup();
            return;
        }
        //-----
        //退出
        Sleep(5000);

        //注销用户
        NET_DVR_Logout(IUserID);
        //释放 SDK 资源
        NET_DVR_Cleanup();
        return;
    }
}

```

## 4.3 门状态计划配置示例代码

[相关模块流程图](#)

```

#include <stdio.h>
#include <iostream>
#include <afx.h>
#include "Windows.h"
#include "HCNetSDK.h"
using namespace std;

void main()
{
    //-----
    //初始化
    NET_DVR_Init();

    //设置连接超时时间与重连功能
    NET_DVR_SetConnectTime(2000, 1);
    NET_DVR_SetReconnect(10000, true);

    //-----
    //注册设备
    LONG IUserID;
}

```

```

//登录参数, 包括设备地址、登录用户、密码等
NET_DVR_USER_LOGIN_INFO struLoginInfo = {0};
struLoginInfo.bUseAsynLogin = 0; //同步登录方式
strcpy(struLoginInfo.sDeviceAddress, "192.0.0.64"); //设备 IP 地址
struLoginInfo.wPort = 8000; //设备服务端口
strcpy(struLoginInfo.sUserName, "admin"); //登录用户名
strcpy(struLoginInfo.sPassword, "abcd1234"); //登录密码

//设备信息, 输出参数
NET_DVR_DEVICEINFO_V40 struDeviceInfoV40 = {0};

IUserID = NET_DVR_Login_V40(&struLoginInfo, &struDeviceInfoV40);
if (IUserID < 0)
{
    printf("Login failed, error code: %d\n", NET_DVR_GetLastError());
    NET_DVR_Cleanup();
    return;
}

//-----
//设置门状态计划, 门 1 关联门状态计划模板 1
NET_DVR_DOOR_STATUS_PLAN struDoorStatusPlan = {0};
struDoorStatusPlan.dwSize = sizeof(struDoorStatusPlan);
struDoorStatusPlan.dwTemplateNo = 1; //计划模板 1

BOOL bRet1 = NET_DVR_SetDVRConfig(IUserID, NET_DVR_SET_DOOR_STATUS_PLAN, 1, \
    &struDoorStatusPlan, sizeof(struDoorStatusPlan));
if (!bRet1)
{
    printf("设置门状态计划失败, error:%d.\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

//设置门状态划模板 1, 模板 1 关联周计划 1, 假日组 1
CString    m_csTemplateName = "门状态计划模板 1";
NET_DVR_PLAN_TEMPLATE struPlanTem = {0};
struPlanTem.dwSize = sizeof(struPlanTem);
struPlanTem.byEnable = 1; //是否使能: 0- 否, 1- 是
strncpy((char *)struPlanTem.byTemplateName, (LPCTSTR)m_csTemplateName, TEMPLATE_NAME_LEN);
struPlanTem.dwWeekPlanNo = 1; //周计划编号 1
struPlanTem.dwHolidayGroupNo[0] = 1; //假日组编号 1, 一个计划模板最多可关联 16 个假日组
//struPlanTem.dwHolidayGroupNo[1] = 2; //假日组编号 2

```



```

    BOOL bRet2 = NET_DVR_SetDVRConfig(IUserID, NET_DVR_SET_DOOR_STATUS_PLAN_TEMPLATE, 1, \
        &struPlanTem, sizeof(struPlanTem));
    if (!bRet2)
    {
        printf("设置门状态计划模板失败, error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(IUserID);
        NET_DVR_Cleanup();
        return;
    }

    //设置门状态周计划 1
    NET_DVR_WEEK_PLAN_CFG struWeekPlan = {0};
    struWeekPlan.dwSize = sizeof(struWeekPlan);
    struWeekPlan.byEnable = 1; //使能周计划

    NET_DVR_SINGLE_PLAN_SEGMENT struSinglePlanSegment = {0};
    LPNET_DVR_SINGLE_PLAN_SEGMENT lpPlanSegment = &struSinglePlanSegment;
    struSinglePlanSegment.byEnable = 1;
    struSinglePlanSegment.byDoorStatus = 3; //门状态模式: 0- 无效, 1- 休眠, 2- 常开状态, 3- 常闭状态
    struSinglePlanSegment.struTimeSegment.struBeginTime.byHour = 0; //开始时间
    struSinglePlanSegment.struTimeSegment.struBeginTime.byMinute = 0;
    struSinglePlanSegment.struTimeSegment.struBeginTime.bySecond = 0;

    struSinglePlanSegment.struTimeSegment.struEndTime.byHour = 23; //结束时间
    struSinglePlanSegment.struTimeSegment.struEndTime.byMinute = 59;
    struSinglePlanSegment.struTimeSegment.struEndTime.bySecond = 59;

    /*一周有 7 天, 每天最多可设置 8 个时间段, 每个时间段可设置门为不同的状态
    这里为了简便, 仅设置每天的 1 个时段, 从 0:00:00~23:59:59*/

    for (int iDate = 0; iDate < MAX_DAYS; iDate++)
    {
        memcpy(&struWeekPlan.struPlanCfg[iDate][0], lpPlanSegment, sizeof(struSinglePlanSegment));
    }

    BOOL bRet3 = NET_DVR_SetDVRConfig(IUserID, NET_DVR_SET_WEEK_PLAN_CFG, 1, \
        &struWeekPlan, sizeof(struWeekPlan));
    if (!bRet3)
    {
        printf("设置门状态周计划失败, error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(IUserID);
        NET_DVR_Cleanup();
        return;
    }

```

```

    }

    //设置门状态假日组
    CString      m_csGroupName = "门状态假日组 1";
    NET_DVR_HOLIDAY_GROUP_CFG struHolidayGroup1 = {0};
    struHolidayGroup1.dwSize = sizeof(struHolidayGroup1);
    struHolidayGroup1.byEnable = 1;
    strncpy((char *)struHolidayGroup1.byGroupName, (LPCTSTR)m_csGroupName, HOLIDAY_GROUP_NAME_LEN);
    struHolidayGroup1.dwHolidayPlanNo[0] = 1; //假日组 1 关联假日计划 1，一个假日组可关联 16 个假日计划

    BOOL bRet4 = NET_DVR_SetDVRConfig(IUserID, NET_DVR_SET_DOOR_STATUS_HOLIDAY_GROUP, 1, \
        &struHolidayGroup1, sizeof(struHolidayGroup1));
    if (!bRet4)
    {
        printf("设置门状态假日组失败，error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(IUserID);
        NET_DVR_Cleanup();
        return;
    }

    //设置门状态假日计划
    NET_DVR_HOLIDAY_PLAN_CFG struHolidayPlan = {0};
    struHolidayPlan.dwSize = sizeof(struHolidayPlan);
    struHolidayPlan.byEnable = 1;
    struHolidayPlan.struBeginDate.wYear = 2017; //假日开始日期
    struHolidayPlan.struBeginDate.byMonth = 10;
    struHolidayPlan.struBeginDate.byDay = 1;
    struHolidayPlan.struEndDate.wYear = 2017; //假日结束日期
    struHolidayPlan.struEndDate.byMonth = 10;
    struHolidayPlan.struEndDate.byDay = 7;
    //门状态假日计划复用周计划参数
    memcpy(struHolidayPlan.struPlanCfg, struWeekPlan.struPlanCfg,
        sizeof(NET_DVR_SINGLE_PLAN_SEGMENT)*MAX_DAYS*MAX_TIMESEGMENT_V30);

    BOOL bRet5 = NET_DVR_SetDVRConfig(IUserID, NET_DVR_SET_DOOR_STATUS_HOLIDAY_PLAN, 1, \
        &struHolidayPlan, sizeof(struHolidayPlan));
    if (!bRet5)
    {
        printf("设置门状态假日计划失败，error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(IUserID);
        NET_DVR_Cleanup();
        return;
    }

    //-----

```

```
//退出
Sleep(5000);

//注销用户
NET_DVR_Logout(IUserID);
//释放 SDK 资源
NET_DVR_Cleanup();
return;
}
```

## 4.4 读卡器验证方式计划配置示例代码

[相关模块流程图](#)

```
#include <stdio.h>
#include <iostream>
#include <afx.h>
#include "Windows.h"
#include "HCNetSDK.h"
using namespace std;

void main()
{
    //-----
    //初始化
    NET_DVR_Init();

    //设置连接超时时间与重连功能
    NET_DVR_SetConnectTime(2000, 1);
    NET_DVR_SetReconnect(10000, true);

    //-----
    //注册设备
    LONG IUserID;
    //登录参数，包括设备地址、登录用户、密码等
    NET_DVR_USER_LOGIN_INFO struLoginInfo = {0};
    struLoginInfo.bUseAsyncLogin = 0; //同步登录方式
    strcpy(struLoginInfo.sDeviceAddress, "192.0.0.64"); //设备 IP 地址
    struLoginInfo.wPort = 8000; //设备服务端口
    strcpy(struLoginInfo.sUserName, "admin"); //登录用户名
    strcpy(struLoginInfo.sPassword, "abcd1234"); //登录密码

    //设备信息，输出参数
    NET_DVR_DEVICEINFO_V40 struDeviceInfoV40 = {0};
```

```

IUserID = NET_DVR_Login_V40(&struLoginInfo, &struDeviceInfoV40);
if (IUserID < 0)
{
    printf("Login failed, error code: %d\n", NET_DVR_GetLastError());
    NET_DVR_Cleanup();
    return;
}

//-----
//设置读卡器验证方式计划，读卡器 1 关联读卡器计划模板 1
NET_DVR_CARD_READER_PLAN struCardReaderPlan = {0};
struCardReaderPlan.dwSize = sizeof(struCardReaderPlan);
struCardReaderPlan.dwTemplateNo = 1; //计划模板 1
BOOL bRet1 = NET_DVR_SetDVRConfig(IUserID, NET_DVR_SET_CARD_READER_PLAN, 1, \
    &struCardReaderPlan, sizeof(struCardReaderPlan));
if (!bRet1)
{
    printf("设置读卡器验证方式计划失败， error:%d.\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

//设置读卡器验证方式计划模板，模板 1 关联周计划 2，假日组 2
CString    m_csTemplateName = "读卡器验证方式计划模板 1";
NET_DVR_PLAN_TEMPLATE struPlanTem = {0};
struPlanTem.dwSize = sizeof(struPlanTem);
struPlanTem.byEnable = 1; //是否使能： 0- 否， 1- 是
strncpy((char *)struPlanTem.byTemplateName, (LPCTSTR)m_csTemplateName, TEMPLATE_NAME_LEN);
struPlanTem.dwWeekPlanNo = 2; //周计划编号 2
struPlanTem.dwHolidayGroupNo[0] = 2; //假日组编号 2， 一个计划模板最多可关联 16 个假日组

BOOL bRet2 = NET_DVR_SetDVRConfig(IUserID, NET_DVR_SET_VERIFY_PLAN_TEMPLATE, 1, \
    &struPlanTem, sizeof(struPlanTem));
if (!bRet2)
{
    printf("设置读卡器验证方式计划模板失败， error:%d.\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

//设置读卡器验证方式周计划 2
NET_DVR_WEEK_PLAN_CFG struWeekPlan2 = {0};

```

```

struWeekPlan2.dwSize = sizeof(struWeekPlan2);
struWeekPlan2.byEnable = 1; //使能周计划

NET_DVR_SINGLE_PLAN_SEGMENT struSinglePlanSegment = {0};
LPNET_DVR_SINGLE_PLAN_SEGMENT lpPlanSegment = &struSinglePlanSegment;
struSinglePlanSegment.byEnable = 1;
struSinglePlanSegment.byVerifyMode = 4; //验证方式: 0-无效, 1-休眠, 2-刷卡+密码, 3-刷卡, 4-刷卡或密码, 5-
指纹, 6-指纹+密码, 7-指纹或刷卡, 8-指纹+刷卡, 9-指纹+刷卡+密码
struSinglePlanSegment.struTimeSegment.struBeginTime.byHour = 0; //开始时间
struSinglePlanSegment.struTimeSegment.struBeginTime.byMinute = 0;
struSinglePlanSegment.struTimeSegment.struBeginTime.bySecond = 0;

struSinglePlanSegment.struTimeSegment.struEndTime.byHour = 23; //结束时间
struSinglePlanSegment.struTimeSegment.struEndTime.byMinute = 59;
struSinglePlanSegment.struTimeSegment.struEndTime.bySecond = 59;

/*一周有 7 天, 每天最多可设置 8 个时间段, 每个时间段可设置读卡器为不同的验证方式
这里为了简便, 仅设置每天的 1 个时段, 从 0:00:00~23:59:59*/

for (int iDate = 0; iDate < MAX_DAYS; iDate++)
{
    memcpy(&struWeekPlan2.struPlanCfg[iDate][0], lpPlanSegment, sizeof(struSinglePlanSegment));
}

BOOL bRet3 = NET_DVR_SetDVRConfig(IUserID, NET_DVR_SET_VERIFY_WEEK_PLAN, 2, \
    &struWeekPlan2, sizeof(struWeekPlan2));
if (!bRet3)
{
    printf("设置读卡器验证方式周计划失败, error:%d.\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

//设置读卡器验证方式假日组
CString    m_csGroupName = "假日组 2";
NET_DVR_HOLIDAY_GROUP_CFG struHolidayGroup2 = {0};
struHolidayGroup2.dwSize = sizeof(struHolidayGroup2);
struHolidayGroup2.byEnable = 1;
strncpy((char *)struHolidayGroup2.byGroupName, (LPCTSTR)m_csGroupName, HOLIDAY_GROUP_NAME_LEN);
struHolidayGroup2.dwHolidayPlanNo[0] = 2; //假日组 2 关联假日计划 2, 一个假日组可关联 16 个假日计划

BOOL bRet4 = NET_DVR_SetDVRConfig(IUserID, NET_DVR_SET_VERIFY_HOLIDAY_GROUP, 2, \
    &struHolidayGroup2, sizeof(struHolidayGroup2));

```

```

    if (!bRet4)
    {
        printf("设置读卡器验证方式假日组失败, error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(IUserID);
        NET_DVR_Cleanup();
        return;
    }

    //设置读卡器验证方式假日计划
    NET_DVR_HOLIDAY_PLAN_CFG struHolidayPlan2 = {0};
    struHolidayPlan2.dwSize = sizeof(struHolidayPlan2);
    struHolidayPlan2.byEnable = 1;
    struHolidayPlan2.struBeginDate.wYear = 2017; //假日开始日期
    struHolidayPlan2.struBeginDate.byMonth = 10;
    struHolidayPlan2.struBeginDate.byDay = 1;
    struHolidayPlan2.struEndDate.wYear = 2017; //假日结束日期
    struHolidayPlan2.struEndDate.byMonth = 10;
    struHolidayPlan2.struEndDate.byDay = 7;
    //读卡器验证方式假日计划复用周计划参数
    memcpy(struHolidayPlan2.struPlanCfg, struWeekPlan2.struPlanCfg,
sizeof(NET_DVR_SINGLE_PLAN_SEGMENT)*MAX_DAYS*MAX_TIMESEGMENT_V30);

    BOOL bRet5 = NET_DVR_SetDVRConfig(IUserID, NET_DVR_SET_VERIFY_HOLIDAY_PLAN, 2, \
        &struHolidayPlan2, sizeof(struHolidayPlan2));
    if (!bRet5)
    {
        printf("设置读卡器验证方式假日计划失败, error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(IUserID);
        NET_DVR_Cleanup();
        return;
    }

    //-----
    //退出
    Sleep(5000);

    //注销用户
    NET_DVR_Logout(IUserID);
    //释放 SDK 资源
    NET_DVR_Cleanup();
    return;
}

```

## 4.5 卡权限计划模板配置示例代码

[相关模块流程图](#)

```
#include <stdio.h>
#include <iostream>
#include <afx.h>
#include "Windows.h"
#include "HCNetSDK.h"
using namespace std;

void main()
{
    //-----
    //初始化
    NET_DVR_Init();

    //设置连接超时时间与重连功能
    NET_DVR_SetConnectTime(2000, 1);
    NET_DVR_SetReconnect(10000, true);

    //-----
    //注册设备
    LONG lUserID;
    //登录参数，包括设备地址、登录用户、密码等
    NET_DVR_USER_LOGIN_INFO struLoginInfo = {0};
    struLoginInfo.bUseAsynLogin = 0; //同步登录方式
    strcpy(struLoginInfo.sDeviceAddress, "192.0.0.64"); //设备 IP 地址
    struLoginInfo.wPort = 8000; //设备服务端口
    strcpy(struLoginInfo.sUserName, "admin"); //登录用户名
    strcpy(struLoginInfo.sPassword, "abcd1234"); //登录密码

    //设备信息，输出参数
    NET_DVR_DEVICEINFO_V40 struDeviceInfoV40 = {0};

    lUserID = NET_DVR_Login_V40(&struLoginInfo, &struDeviceInfoV40);
    if (lUserID < 0)
    {
        printf("Login failed, error code: %d\n", NET_DVR_GetLastError());
        NET_DVR_Cleanup();
        return;
    }
    //-----
    //设置卡权限计划模板，发卡时关联该模板即可
```

```
CString      m_csTemplateName = "卡权限计划模板 1";
NET_DVR_PLAN_TEMPLATE struPlanTem = {0};
struPlanTem.dwSize = sizeof(struPlanTem);
struPlanTem.byEnable = 1; //是否使能: 0- 否, 1- 是
strncpy((char *)struPlanTem.byTemplateName, (LPCTSTR)m_csTemplateName, TEMPLATE_NAME_LEN);
struPlanTem.dwWeekPlanNo = 1; //周计划编号 1
struPlanTem.dwHolidayGroupNo[0] = 1; //假日组编号 1, 一个计划模板最多可关联 16 个假日组
//struPlanTem.dwHolidayGroupNo[1] = 2; //假日组编号 2

BOOL bRet1 = NET_DVR_SetDVRConfig(IUserID, NET_DVR_SET_CARD_RIGHT_PLAN_TEMPLATE, 1, \
                                &struPlanTem, sizeof(struPlanTem));
if (!bRet1)
{
    printf("设置卡权限计划模板失败, error:%d.\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

//设置卡权限周计划 1
NET_DVR_WEEK_PLAN_CFG struWeekPlan = {0};
struWeekPlan.dwSize = sizeof(struWeekPlan);
struWeekPlan.byEnable = 1; //使能周计划

NET_DVR_SINGLE_PLAN_SEGMENT struSinglePlanSegment = {0};
LPNET_DVR_SINGLE_PLAN_SEGMENT lpPlanSegment = &struSinglePlanSegment;
struSinglePlanSegment.byEnable = 1;

struSinglePlanSegment.struTimeSegment.struBeginTime.byHour = 0; //开始时间
struSinglePlanSegment.struTimeSegment.struBeginTime.byMinute = 0;
struSinglePlanSegment.struTimeSegment.struBeginTime.bySecond = 0;

struSinglePlanSegment.struTimeSegment.struEndTime.byHour = 23; //结束时间
struSinglePlanSegment.struTimeSegment.struEndTime.byMinute = 59;
struSinglePlanSegment.struTimeSegment.struEndTime.bySecond = 59;

/*一周有 7 天, 每天最多可设置 8 个时间段。
这里为了简便, 仅设置每天的 1 个时段, 从 0:00:00~23:59:59*/

for (int iDate = 0; iDate < MAX_DAYS; iDate++)
{
    memcpy(&struWeekPlan.struPlanCfg[iDate][0], lpPlanSegment, sizeof(struSinglePlanSegment));
}
```



```

BOOL bRet2 = NET_DVR_SetDVRConfig(IUserID, NET_DVR_SET_CARD_RIGHT_WEEK_PLAN, 1, \
    &struWeekPlan, sizeof(struWeekPlan));
if (!bRet2)
{
    printf("设置卡权限周计划失败, error:%d.\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

```

//设置卡权限假日组

```

CString      m_csGroupName = "卡权限假日组 1";
NET_DVR_HOLIDAY_GROUP_CFG struHolidayGroup1 = {0};
struHolidayGroup1.dwSize = sizeof(struHolidayGroup1);
struHolidayGroup1.byEnable = 1;
strncpy((char *)struHolidayGroup1.byGroupName, (LPCTSTR)m_csGroupName, HOLIDAY_GROUP_NAME_LEN);
struHolidayGroup1.dwHolidayPlanNo[0] = 1; //假日组 1 关联假日计划 1, 一个假日组可关联 16 个假日计划

```

```

BOOL bRet3 = NET_DVR_SetDVRConfig(IUserID, NET_DVR_SET_CARD_RIGHT_HOLIDAY_GROUP, 1, \
    &struHolidayGroup1, sizeof(struHolidayGroup1));
if (!bRet3)
{
    printf("设置卡权限假日组失败, error:%d.\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

```

//设置卡权限假日计划

```

NET_DVR_HOLIDAY_PLAN_CFG struHolidayPlan = {0};
struHolidayPlan.dwSize = sizeof(struHolidayPlan);
struHolidayPlan.byEnable = 1;
struHolidayPlan.struBeginDate.wYear = 2017; //假日开始日期
struHolidayPlan.struBeginDate.byMonth = 10;
struHolidayPlan.struBeginDate.byDay = 1;
struHolidayPlan.struEndDate.wYear = 2017; //假日结束日期
struHolidayPlan.struEndDate.byMonth = 10;
struHolidayPlan.struEndDate.byDay = 7;

```

//卡权限假日计划复用周计划参数

```

memcpy(struHolidayPlan.struPlanCfg, struWeekPlan.struPlanCfg,
sizeof(NET_DVR_SINGLE_PLAN_SEGMENT)*MAX_DAYS*MAX_TIMESEGMENT_V30);

```

```

BOOL bRet4 = NET_DVR_SetDVRConfig(IUserID, NET_DVR_SET_CARD_RIGHT_HOLIDAY_PLAN, 1, \
    &struHolidayPlan, sizeof(struHolidayPlan));

```

```

    if (!bRet4)
    {
        printf("设置卡权限假日计划失败，error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(IUserID);
        NET_DVR_Cleanup();
        return;
    }
    //-----
    //退出
    Sleep(5000);

    //注销用户
    NET_DVR_Logout(IUserID);
    //释放 SDK 资源
    NET_DVR_Cleanup();
    return;
}

```

## 4.6 布防获取门禁事件示例代码

[相关模块流程图](#)

```

#include <stdio.h>
#include <iostream>
#include "Windows.h"
#include "HCNetSDK.h"
using namespace std;

BOOL CALLBACK MSesGCallback(LONG ICommand, NET_DVR_ALARMER *pAlarmer, char *pAlarmInfo, DWORD dwBufLen, void*
pUser)
{
    //回调函数中不可有耗时较长的操作，不能调用该 SDK（HCNetSDK.dll）本身的接口。
    //以下代码仅供参考，实际应用中不建议在回调函数中直接处理数据保存文件，
    //例如可以使用消息的方式(PostMessage)在消息响应函数里进行处理。

    switch (ICommand)
    {
        case COMM_ALARM_ACS://门禁主机报警信息
        {
            NET_DVR_ACS_ALARM_INFO struAcsAlarmInfo = {0};
            memcpy(&struAcsAlarmInfo, pAlarmInfo, sizeof(struAcsAlarmInfo));
            //按需处理报警信息结构体中的信息.....
            break;
        }
        case COMM_ID_INFO_ALARM://门禁身份证刷卡信息
        {

```

```

        NET_DVR_ID_CARD_INFO_ALARM struID_CardInfo = {0};
        memcpy(&struID_CardInfo, pAlarmInfo, sizeof(struID_CardInfo));
        //按需处理报警信息结构体中的信息.....
        break;
    }
case COMM_PASSNUM_INFO_ALARM://门禁通行人数信息
    {
        NET_DVR_PASSNUM_INFO_ALARM struPassnumInfo = {0};
        memcpy(&struPassnumInfo, pAlarmInfo, sizeof(struPassnumInfo));
        //按需处理报警信息结构体中的信息.....
        break;
    }
default:
    break;
}
return true;
}
void main()
{
    //-----
    //初始化
    NET_DVR_Init();

    //设置连接超时时间与重连功能
    NET_DVR_SetConnectTime(2000, 1);
    NET_DVR_SetReconnect(10000, true);

    //-----
    //注册设备
    LONG lUserID;
    //登录参数, 包括设备地址、登录用户、密码等
    NET_DVR_USER_LOGIN_INFO struLoginInfo = {0};
    struLoginInfo.bUseAsynLogin = 0; //同步登录方式
    strcpy(struLoginInfo.sDeviceAddress, "192.0.0.64"); //设备 IP 地址
    struLoginInfo.wPort = 8000; //设备服务端口
    strcpy(struLoginInfo.sUserName, "admin"); //登录用户名
    strcpy(struLoginInfo.sPassword, "abcd1234"); //登录密码

    //设备信息, 输出参数
    NET_DVR_DEVICEINFO_V40 struDeviceInfoV40 = {0};

    lUserID = NET_DVR_Login_V40(&struLoginInfo, &struDeviceInfoV40);
    if (lUserID < 0)
    {

```

```

        printf("Login failed, error code: %d\n", NET_DVR_GetLastError());
        NET_DVR_Cleanup();
        return;
    }

    //设置报警回调函数，刷卡等事件都会触发报警回调函数
    NET_DVR_SetDVRMessageCallBack_V31(MSesGCallback, NULL);
    //启用布防
    NET_DVR_SETUPALARM_PARAM struSetupParam={0};
    struSetupParam.dwSize=sizeof(NET_DVR_SETUPALARM_PARAM);

    LONG  IHandle = NET_DVR_SetupAlarmChan_V41(IUserID,&struSetupParam);
    if (IHandle < 0)
    {
        printf("NET_DVR_SetupAlarmChan_V41 error, %d\n", NET_DVR_GetLastError());
        NET_DVR_Logout(IUserID);
        NET_DVR_Cleanup();
        return;
    }

    //等待 60s，等待接收设备上传报警
    Sleep(60000);
    //撤销布防
    if (!NET_DVR_CloseAlarmChan_V30(IHandle))
    {
        printf("NET_DVR_CloseAlarmChan_V30 error, %d\n", NET_DVR_GetLastError());
        NET_DVR_Logout(IUserID);
        NET_DVR_Cleanup();
        return;
    }

    //注销用户
    NET_DVR_Logout(IUserID);
    //释放 SDK 资源
    NET_DVR_Cleanup();
    return;
}

```

## 4.7 获取和下发卡号相关联参数示例代码

[相关模块流程图](#)

```

#include <stdio.h>
#include <iostream>
#include <afx.h>
#include "Windows.h"

```

```

#include "HCNetSDK.h"
using namespace std;

LONG m_lSetCardCfgHandle;
LONG m_lGetCardCfgHandle;
CString m_csCardNo;
CString m_csCardPassword;
BOOL bGetCardCfgFinish = FALSE;
BOOL bSetCardCfgFinish = FALSE;

void CALLBACK g_fGetGatewayCardCallback(DWORD dwType, void* lpBuffer, DWORD dwBufLen, void* pUserData)
{
    //回调函数中不可有耗时较长的操作，不能调用该 SDK（HCNetSDK.dll）本身的接口。
    //以下代码仅供参考，实际应用中不建议在回调函数中直接处理数据，
    //例如可以使用消息的方式(PostMessage)在消息响应函数里进行处理。

    if (dwType == NET_SDK_CALLBACK_TYPE_DATA)//数据信息
    {
        LPNET_DVR_CARD_CFG_V50 lpCardCfg = new NET_DVR_CARD_CFG_V50;
        memcpy(lpCardCfg, lpBuffer, sizeof(*lpCardCfg)); //拷贝回调的卡参数
        //PostMessage(WM_MSG_ADD_CARDCFG_TOLIST, (WPARAM)lpCardCfg,0);

        char *pCardNo;
        BYTE byCardType;
        pCardNo = (char *)lpCardCfg->byCardNo;
        byCardType = lpCardCfg->byCardType;
        //其他处理.....
    }
    else if (dwType == NET_SDK_CALLBACK_TYPE_STATUS)//状态值
    {
        DWORD dwStatus = *(DWORD*)lpBuffer;
        if (dwStatus == NET_SDK_CALLBACK_STATUS_SUCCESS)
        {
            bGetCardCfgFinish = TRUE;//获取卡参数完成
            //PostMessage(WM_MSG_GETCARD_FINISH,0,0);
            //其他处理.....
        }
        else if ( dwStatus == NET_SDK_CALLBACK_STATUS_FAILED )
        {
            char szCardNumber[ACS_CARD_NO_LEN + 1] = "\0";
            DWORD dwErrCode = *(DWORD*)((char *)lpBuffer + 4); //错误码
            strncpy(szCardNumber,(char*)(lpBuffer) + 8,ACS_CARD_NO_LEN);//卡号
            printf("GetCard STATUS_FAILED, Error code %d, Card Number %s\n", dwErrCode,
szCardNumber);

```

```

        //其他处理.....
    }
}

void CALLBACK g_fSetGatewayCardCallback(DWORD dwType, void* lpBuffer, DWORD dwBufLen, void* pUserData)
{
    if (dwType != NET_SDK_CALLBACK_TYPE_STATUS)//下发卡时只会返回状态
    {
        return;
    }

    DWORD dwStatus = *(DWORD*)lpBuffer;//前 4 个字节为状态值

    if (dwStatus == NET_SDK_CALLBACK_STATUS_PROCESSING)//发送中
    {
        char szCardNumber[ACS_CARD_NO_LEN + 1] = "\0";
        strncpy(szCardNumber, (char*)(lpBuffer) + 4, ACS_CARD_NO_LEN);
        printf("SetCard PROCESSING, CardNo: %s\n", szCardNumber);
        //其他处理.....
    }
    else if (dwStatus == NET_SDK_CALLBACK_STATUS_FAILED)//发送失败
    {
        char szCardNumber[ACS_CARD_NO_LEN + 1] = "\0";
        DWORD dwErrCode = *((DWORD*)lpBuffer + 1);//错误码
        strncpy(szCardNumber, (char*)(lpBuffer) + 8, ACS_CARD_NO_LEN);//卡号
        printf("SetCard Err:%d, CardNo:%s\n", dwErrCode, szCardNumber);
        //其他处理.....
    }
    //下面两个关闭长连接
    else if (dwStatus == NET_SDK_CALLBACK_STATUS_SUCCESS)//发送成功
    {
        printf("SetCard SUCCESS!");
        bSetCardCfgFinish = TRUE;
        //其他处理.....
        //PostMessage(WM_MSG_SETCARD_FINISH, 0, 0);
    }
    else if (dwStatus == NET_SDK_CALLBACK_STATUS_EXCEPTION)//异常
    {
        bSetCardCfgFinish = TRUE;
        //其他处理.....
        //PostMessage(WM_MSG_SETCARD_FINISH, 0, 0);
    }
}

```

```
void main()
{
    //-----
    //初始化
    NET_DVR_Init();

    //设置连接超时时间与重连功能
    NET_DVR_SetConnectTime(2000, 1);
    NET_DVR_SetReconnect(10000, true);

    //-----
    //注册设备
    LONG lUserID;
    NET_DVR_DEVICEINFO_V30 struDeviceInfo;
    lUserID = NET_DVR_Login_V30("192.0.0.64", 8000, "admin", "12345", &struDeviceInfo);
    if (lUserID < 0)
    {
        printf("Login error, %d\n", NET_DVR_GetLastError());
        NET_DVR_Cleanup();
        return;
    }

    //获取卡参数-----
    NET_DVR_CARD_CFG_COND struCond = {0};
    struCond.dwSize = sizeof(struCond);
    struCond.dwCardNum = 1; //获取卡数量
    struCond.byCheckCardNo = 1;

    NET_DVR_CARD_CFG_SEND_DATA struSendData = {0};
    struSendData.dwSize = sizeof(struSendData);
    m_csCardNo = "12"; //卡号
    strncpy((char *)struSendData.byCardNo, (LPCTSTR)m_csCardNo, ACS_CARD_NO_LEN);

    //启动长连接
    m_lGetCardCfgHandle = NET_DVR_StartRemoteConfig(lUserID, NET_DVR_GET_CARD_CFG_V50, &struCond, \
        sizeof(struCond), g_fGetGatewayCardCallback, NULL);
    if (m_lGetCardCfgHandle == -1)
    {
        printf("NET_DVR_StartRemoteConfig fail, error: %d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(lUserID);
        NET_DVR_Cleanup();
        return;
    }
}
```

```

//发送长连接数据
if (! NET_DVR_SendRemoteConfig(m_IGetCardCfgHandle, ENUM_ACS_SEND_DATA, (char *)&struSendData,
sizeof(struSendData)))
{
    printf("NET_DVR_SendRemoteConfig fail, error:%d.\n", NET_DVR_GetLastError());
    NET_DVR_StopRemoteConfig(m_IGetCardCfgHandle);
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

//关闭长连接
Sleep(1000);
if (bGetCardCfgFinish)//若获取卡参数完成则关闭长连接
{
    NET_DVR_StopRemoteConfig(m_IGetCardCfgHandle);
}

//下发卡参数-----
NET_DVR_CARD_CFG_COND struCond_set = {0};
struCond_set.dwSize = sizeof(struCond_set);
struCond_set.dwCardNum = 1;//下发卡数量
struCond_set.byCheckCardNo = 1;
struCond_set.wLocalControllerID = 0;//就地控制器序号，表示往就地控制器下发离线卡参数，0 代表是门禁主机

//启动长连接
m_ISetCardCfgHandle = NET_DVR_StartRemoteConfig(IUserID,NET_DVR_SET_CARD_CFG_V50,&struCond_set,\
sizeof(struCond_set),g_fSetGatewayCardCallback,NULL);
if (m_ISetCardCfgHandle== -1)
{
    printf("NET_DVR_StartRemoteConfig fail, error:%d.\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

//设置卡参数
LPNET_DVR_CARD_CFG_V50 lpCardCfg = new NET_DVR_CARD_CFG_V50;
lpCardCfg->dwSize = sizeof(NET_DVR_CARD_CFG_V50);
lpCardCfg->dwModifyParamType = 0x000003FF;//需要修改的卡参数，按位表示，每位代表一种参数，值：0-不修改，
1-修改

m_csCardNo = "12";//卡号
strncpy((char *)lpCardCfg->byCardNo, (LPCTSTR)m_csCardNo, ACS_CARD_NO_LEN);
lpCardCfg->byCardValid = 1;//卡是否有效：0-无效，1-有效

```



```

lpCardCfg->byCardType = 1;//卡类型：1-普通卡（默认）
lpCardCfg->byLeaderCard = 0;//是否为首卡：1-是，0-否
lpCardCfg->byDoorRight[0] = 1;//byDoorRight[0]~byDoorRight[255]对应门 1-256，值：1-有权限，0-无权限
lpCardCfg->byDoorRight[1] = 1;//门 1、门 2 有权限

lpCardCfg->byBelongGroup[0] = 1;//byBelongGroup[0]~byBelongGroup[127]对应群组 1-128，值：1-属于，0-不属于
m_csCardPassword = "12345678";//卡密码
strncpy((char *)lpCardCfg->byCardPassword, (LPCTSTR)m_csCardPassword, CARD_PASSWORD_LEN);
//卡权限计划模板需先配置好
lpCardCfg->wCardRightPlan[0][0]=1;//该卡在门 1 具有卡权限计划模板 1 和 2 的权限
lpCardCfg->wCardRightPlan[0][1]=2;
lpCardCfg->wCardRightPlan[1][0]=3;//该卡在门 2 具有卡权限计划模板 3 和 4 的权限
lpCardCfg->wCardRightPlan[1][1]=4;

lpCardCfg->dwMaxSwipeTime = 0;//最大刷卡次数，0 无限次数
lpCardCfg->dwSwipeTime = 0;//已刷卡次数

lpCardCfg->struValid.byEnable = 1;//使能有效期
lpCardCfg->struValid.struBeginTime.wYear=2017;//开始时间：2017-01-01 00:00:00
lpCardCfg->struValid.struBeginTime.byMonth=1;
lpCardCfg->struValid.struBeginTime.byDay=1;
lpCardCfg->struValid.struBeginTime.byHour=0;
lpCardCfg->struValid.struBeginTime.byMinute=0;
lpCardCfg->struValid.struBeginTime.bySecond=0;

lpCardCfg->struValid.struEndTime.wYear=2018;//结束时间：2018-01-01 00:00:00
lpCardCfg->struValid.struEndTime.byMonth=1;
lpCardCfg->struValid.struEndTime.byDay=1;
lpCardCfg->struValid.struEndTime.byHour=0;
lpCardCfg->struValid.struEndTime.byMinute=0;
lpCardCfg->struValid.struEndTime.bySecond=0;

//发送长连接数据
if (!NET_DVR_SendRemoteConfig(m_ISetCardCfgHandle,ENUM_ACS_SEND_DATA, (char
*)lpCardCfg,sizeof(*lpCardCfg)))
{
    printf("NET_DVR_SendRemoteConfig fail, error:%d.\n", NET_DVR_GetLastError());
    NET_DVR_StopRemoteConfig(m_ISetCardCfgHandle);
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

//关闭长连接
Sleep(1000);

```

```

        if (bSetCardCfgFinish)//若发卡完成或回调状态为异常则关闭长连接
        {
            NET_DVR_StopRemoteConfig(m_lSetCardCfgHandle);
        }

        //-----
        //退出
        Sleep(5000);

        //注销用户
        NET_DVR_Logout(lUserID);
        //释放 SDK 资源
        NET_DVR_Cleanup();
        return;
    }

```

## 4.8 远程开门示例代码

[相关模块流程图](#)

```

#include <stdio.h>
#include <iostream>
#include <afx.h>
#include "Windows.h"
#include "HCNetSDK.h"
using namespace std;

void main()
{
    //-----
    //初始化
    NET_DVR_Init();

    //设置连接超时时间与重连功能
    NET_DVR_SetConnectTime(2000, 1);
    NET_DVR_SetReconnect(10000, true);

    //-----
    //注册设备
    LONG lUserID;
    //登录参数，包括设备地址、登录用户、密码等
    NET_DVR_USER_LOGIN_INFO struLoginInfo = {0};
    struLoginInfo.bUseAsyncLogin = 0; //同步登录方式

```

```
strcpy(struLoginInfo.sDeviceAddress, "192.0.0.64"); //设备 IP 地址
struLoginInfo.wPort = 8000; //设备服务端口
strcpy(struLoginInfo.sUserName, "admin"); //登录用户名
strcpy(struLoginInfo.sPassword, "abcd1234"); //登录密码

//设备信息, 输出参数
NET_DVR_DEVICEINFO_V40 struDeviceInfoV40 = {0};

UserID = NET_DVR_Login_V40(&struLoginInfo, &struDeviceInfoV40);
if (UserID < 0)
{
    printf("Login failed, error code: %d\n", NET_DVR_GetLastError());
    NET_DVR_Cleanup();
    return;
}

//开门, 以门 1 为例
BOOL bRet;
LONG lGatewayIndex = 1; //门禁序号, 从 1 开始, -1 表示对所有门进行操作
DWORD dwStaic = 1; //命令值: 0-关闭, 1-打开, 2-常开, 3-常关

bRet = NET_DVR_ControlGateway(UserID, lGatewayIndex, dwStaic);
if (!bRet)
{
    printf("NET_DVR_ControlGateway failed, error: %d\n", NET_DVR_GetLastError());
    NET_DVR_Logout(UserID);
    NET_DVR_Cleanup();
    return;
}

//-----
//退出
Sleep(5000);

//注销用户
NET_DVR_Logout(UserID);
//释放 SDK 资源
NET_DVR_Cleanup();
return;
}
```

## 4.9 联动抓拍示例代码

[相关模块流程图](#)

```
#include <stdio.h>
#include <iostream>
#include "Windows.h"
#include "HCNetSDK.h"
using namespace std;

BOOL CALLBACK MSesGCallback(LONG ICommand, NET_DVR_ALARMER *pAlarmer, char *pAlarmInfo, DWORD dwBufLen, void*
pUser)
{
    //回调函数中不可有耗时较长的操作，不能调用该 SDK（HCNetSDK.dll）本身的接口。
    //以下代码仅供参考，实际应用中不建议在回调函数中直接处理数据保存文件，
    //例如可以使用消息的方式(PostMessage)在消息响应函数里进行处理。
    switch (ICommand)
    {
        case COMM_ALARM_ACS://门禁主机报警信息
        {
            NET_DVR_ACS_ALARM_INFO struAcsAlarmInfo = {0};
            memcpy(&struAcsAlarmInfo, pAlarmInfo, sizeof(struAcsAlarmInfo));

            char szTime[50] = {0};//报警时间
            sprintf(szTime, "%4d-%2d-%2d %2d:%2d:%2d", struAcsAlarmInfo.struTime.dwYear,
struAcsAlarmInfo.struTime.dwMonth, struAcsAlarmInfo.struTime.dwDay, struAcsAlarmInfo.struTime.dwHour,
struAcsAlarmInfo.struTime.dwMinute, struAcsAlarmInfo.struTime.dwSecond);

            char szCardNo[50] = {0};//卡号
            sprintf(szCardNo, "CardNo:%s", (char *)struAcsAlarmInfo.struAcsEventInfo.byCardNo);
            BYTE byCardType = struAcsAlarmInfo.struAcsEventInfo.byCardType;//卡类型
            DWORD dwCardReaderNo = struAcsAlarmInfo.struAcsEventInfo.dwCardReaderNo;//读卡器编
号

            DWORD dwDoorNo = struAcsAlarmInfo.struAcsEventInfo.dwDoorNo;//门编号

            if (struAcsAlarmInfo.dwPicDataLen > 0 && struAcsAlarmInfo.pPicData != NULL)
            {
                char filename[128];
                FILE *fSnapPic=NULL;

                SYSTEMTIME t;
                GetLocalTime(&t);
                char chTime[128];

                sprintf(filename, "%4.4d%2.2d%2.2d%2.2d%2.2d%2.2d%3.3d", t.wYear, t.wMonth, t.wDay, t.wHour, t.wMinute, t.wSecond,
```

```

t.wMilliseconds);

                                //保存图片
                                fSnapPic=fopen(filename,"wb");
                                fwrite(struAcsAlarmInfo.pPicData,struAcsAlarmInfo.dwPicDataLen,1,fSnapPic);
                                fclose(fSnapPic);
                                }
                                //按需处理报警信息结构体中的其它信息.....
                                break;
                                }
default:
    break;
}
return true;
}
void main()
{
    //-----
    //初始化
    NET_DVR_Init();

    //设置连接超时时间与重连功能
    NET_DVR_SetConnectTime(2000, 1);
    NET_DVR_SetReconnect(10000, true);
    //-----
    //注册设备
    LONG IUserID;
    //登录参数，包括设备地址、登录用户、密码等
    NET_DVR_USER_LOGIN_INFO struLoginInfo = {0};
    struLoginInfo.bUseAsyncLogin = 0; //同步登录方式
    strcpy(struLoginInfo.sDeviceAddress, "192.168.1.64"); //设备 IP 地址
    struLoginInfo.wPort = 8000; //设备服务端口
    strcpy(struLoginInfo.sUserName, "admin"); //登录用户名
    strcpy(struLoginInfo.sPassword, "abcd1234"); //登录密码
    //设备信息，输出参数
    NET_DVR_DEVICEINFO_V40 struDeviceInfoV40 = {0};

    IUserID = NET_DVR_Login_V40(&struLoginInfo, &struDeviceInfoV40);
    if (IUserID < 0)
    {
        printf("Login failed, error code: %d\n", NET_DVR_GetLastError());
        NET_DVR_Cleanup();
        return;
    }
}

```

```

//设置报警回调函数，联动抓拍会触发报警回调
NET_DVR_SetDVRMessageCallBack_V31(MSesGCallback, NULL);

//启用布防
NET_DVR_SETUPALARM_PARAM struSetupParam={0};
struSetupParam.dwSize=sizeof(NET_DVR_SETUPALARM_PARAM);

LONG  IHandle = NET_DVR_SetupAlarmChan_V41(IUserID,&struSetupParam);
if (IHandle < 0)
{
    printf("NET_DVR_SetupAlarmChan_V41 error: %d\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

//-----
//配置触发抓拍参数
NET_DVR_SNAPCFG struSnapCfg = {0};
struSnapCfg.dwSize = sizeof(NET_DVR_SNAPCFG);
struSnapCfg.bySnapTimes = 2;//抓拍次数：0-不抓拍，非0-连拍次数，目前最大5次
struSnapCfg.wIntervalTime[0] = 1000;//连拍间隔时间，单位ms，取值范围[67,60000]
struSnapCfg.struJpegPara.wPicSize = 5;//图片分辨率：5-1280*720

if (!NET_DVR_ContinuousShoot(IUserID, &struSnapCfg))
{
    printf("NET_DVR_ContinuousShoot error: %d\n", NET_DVR_GetLastError());
    return;
}

//-----
//事件及卡号联动参数设置
NET_DVR_EVENT_CARD_LINKAGE_COND struEventCardLinkageCond = {0};
struEventCardLinkageCond.dwSize = sizeof(NET_DVR_EVENT_CARD_LINKAGE_COND);
struEventCardLinkageCond.dwEventID = 1;//事件ID，从1开始，设置不同的事件/卡号联动配置时依次递增即可

NET_DVR_EVENT_CARD_LINKAGE_CFG_V50 struEventCardLinkageCfgV50 = {0};
struEventCardLinkageCfgV50.dwSize = sizeof(NET_DVR_EVENT_CARD_LINKAGE_CFG_V50);
struEventCardLinkageCfgV50.byProMode = 0;//联动方式：0-事件，1-卡号
struEventCardLinkageCfgV50.byCapturePic = 1;//是否联动抓拍：0-不联动抓拍，1-联动抓拍

//事件源ID，0xffffffff表示联动全部，其他取值：当主类型为设备事件时无效；当主类型是为门事件时，为门编号；
//当主类型为读卡器事件时，为读卡器ID；当主类型为报警输入事件时，为防区报警输入ID或事件报警输入ID
struEventCardLinkageCfgV50.dwEventSourceID = 0xffffffff;

```

```

//联动事件主类型：0-设备事件，1-报警输入事件，2-门事件，3-读卡器事件
struEventCardLinkageCfgV50.uLinkageInfo.struEventLinkage.wMainEventType = 2;
//联动事件次类型：10-正常开门（门磁），这里以开门联动抓拍为例
struEventCardLinkageCfgV50.uLinkageInfo.struEventLinkage.wSubEventType = 10;

DWORD dwStatus = 0;
if (!NET_DVR_SetDeviceConfig(IUserID, NET_DVR_SET_EVENT_CARD_LINKAGE_CFG_V50,
1, &struEventCardLinkageCond, sizeof(struEventCardLinkageCond), &dwStatus, &struEventCardLinkageCfgV50, sizeof(struEventCardLinkageCfgV50)))
{
    printf("NET_DVR_SET_EVENT_CARD_LINKAGE_CFG_V50, error: %d\n", NET_DVR_GetLastError());
    return;
}

//-----
//门禁主机参数设置
NET_DVR_ACS_CFG struAcsCfg = {0};
struAcsCfg.dwSize = sizeof(NET_DVR_ACS_CFG);
struAcsCfg.byUploadCapPic = 1; //联动抓拍是否上传图片：0-不上传，1-上传

BOOL bRet = NET_DVR_SetDVRConfig(IUserID, NET_DVR_SET_ACS_CFG, 0, \
    &struAcsCfg, sizeof(struAcsCfg));
if (!bRet)
{
    printf("NET_DVR_SET_ACS_CFG, error: %d.\n", NET_DVR_GetLastError());
    return;
}

//-----
//等待 30s，等待接收设备联动抓拍上传图片
Sleep(30000);
//撤销布防
if (!NET_DVR_CloseAlarmChan_V30(IHandle))
{
    printf("NET_DVR_CloseAlarmChan_V30 error, %d\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

//注销用户
NET_DVR_Logout(IUserID);
//释放 SDK 资源
NET_DVR_Cleanup();
return;
}

```

## 5 功能接口介绍

### 5.1 通用接口介绍

功能	接口	支持的产品
初始化 SDK	<a href="#">NET_DVR_Init</a>	全部
释放 SDK 资源	<a href="#">NET_DVR_Cleanup</a>	全部
设置 SDK 网络连接超时时间和连接尝试次数	<a href="#">NET_DVR_SetConnectTime</a>	全部
设置 SDK 重连功能	<a href="#">NET_DVR_SetReconnect</a>	全部
设置接收超时时间	<a href="#">NET_DVR_SetRecvTimeOut</a>	全部
通过解析服务器，获取设备的动态 IP 地址	<a href="#">NET_DVR_GetDVRIPByResolveSvr_Ex</a>	全部
获取所有 IP，用于支持多网卡接口	<a href="#">NET_DVR_GetLocalIP</a>	全部
选择绑定 IP	<a href="#">NET_DVR_SetValidIP</a>	全部
注册接收异常、重连等消息的窗口句柄或回调函数	<a href="#">NET_DVR_SetExceptionCallBack_V30</a>	全部
获取 SDK 的版本号和 build 信息	<a href="#">NET_DVR_GetSDKBuildVersion</a>	全部
获取当前 SDK 的状态信息	<a href="#">NET_DVR_GetSDKState</a>	全部
获取当前 SDK 的功能信息	<a href="#">NET_DVR_GetSDKAbility</a>	全部
获取 SDK 本地参数	<a href="#">NET_DVR_GetSDKLocalCfg</a>	全部
设置 SDK 本地参数	<a href="#">NET_DVR_SetSDKLocalCfg</a>	全部
启用 SDK 写日志文件	<a href="#">NET_DVR_SetLogToFile</a>	全部
返回最后操作的错误码	<a href="#">NET_DVR_GetLastError</a>	全部
返回最后操作的错误码信息	<a href="#">NET_DVR_GetErrorMsg</a>	全部
注册（登录设备）	<a href="#">NET_DVR_Login_V40</a>	全部
注销登录	<a href="#">NET_DVR_Logout</a>	全部

### 5.2 门禁主机功能接口

门禁主机包括但不限于以下产品型号：DS-K2601(-G)、DS-K2602(-A/-G)、DS-K2604(-G)。

门禁主机除通用接口的功能外，还支持以下接口的功能，包括参数配置、远程控制、报警上传和设备维护等。

功能	接口	相关参数
获取设备能力集		
获取设备通用能力集	<a href="#">NET_DVR_GetDeviceAbility</a>	能力集类型： DEVICE_SOFTWARE_ABILITY、



		DEVICE_USER_ABILITY、 DEVICE_NETWORK_ABILITY、 DEVICE_NETAPP_ABILITY、 DEVICE_SERIAL_ABILITY、 DEVICE_ABILITY_INFO(其中的事件能力 EventAbility 和安全认证能力 SecurityAbility)
获取门禁主机报警能力集	<a href="#">NET_DVR_GetDeviceAbility</a>	能力集类型： DEVICE_ABILITY_INFO(报警主机能力集 AlarmHostAbility)
获取门禁能力集	<a href="#">NET_DVR_GetDeviceAbility</a>	能力集类型： ACS_ABILITY
<b>通用参数配置</b>		
获取设备参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令：NET_DVR_GET_DEVICECFG_V40 结构体： <a href="#">NET_DVR_DEVICECFG_V40</a>
设置设备参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令：NET_DVR_SET_DEVICECFG_V40 结构体： <a href="#">NET_DVR_DEVICECFG_V40</a>
获取时间参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令：NET_DVR_GET_TIMECFG 结构体： <a href="#">NET_DVR_TIME</a>
设置时间参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令：NET_DVR_SET_TIMECFG 结构体： <a href="#">NET_DVR_TIME</a>
获取时区和夏时制参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令：NET_DVR_GET_ZONEANDDST 结构体： <a href="#">NET_DVR_ZONEANDDST</a>
设置时区和夏时制参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令：NET_DVR_SET_ZONEANDDST 结构体： <a href="#">NET_DVR_ZONEANDDST</a>
获取网络参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令：NET_DVR_GET_NETCFG_V30 结构体： <a href="#">NET_DVR_NETCFG_V30</a>
设置网络参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令：NET_DVR_SET_NETCFG_V30 结构体： <a href="#">NET_DVR_NETCFG_V30</a>
获取异常参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令：NET_DVR_GET_EXCEPTIONCFG_V40 结构体： <a href="#">NET_DVR_EXCEPTION_V40</a>
设置异常参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令：NET_DVR_SET_EXCEPTIONCFG_V40 结构体： <a href="#">NET_DVR_EXCEPTION_V40</a>
获取网络应用(NTP)参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令：NET_DVR_GET_NTPCFG 结构体： <a href="#">NET_DVR_NTPPARA</a>
设置网络应用(NTP)参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令：NET_DVR_SET_NTPCFG 结构体： <a href="#">NET_DVR_NTPPARA</a>
获取安全认证配置	<a href="#">NET_DVR_GetDVRConfig</a>	命令：NET_DVR_GET_SECURITY_CFG 结构体： <a href="#">NET_DVR_SECURITY_CFG</a>
设置安全认证配置	<a href="#">NET_DVR_SetDVRConfig</a>	命令：NET_DVR_SET_SECURITY_CFG 结构体： <a href="#">NET_DVR_SECURITY_CFG</a>

获取短信相关参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_SMSRELATIVEPARA 结构体: <a href="#">NET_DVR_SMSRELATIVEPARAM</a> (V1.1 新增支持)
设置短信相关参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_SMSRELATIVEPARA 结构体: <a href="#">NET_DVR_SMSRELATIVEPARAM</a> (V1.1 新增支持)
获取上传中心网络参数配置	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_ALARMHOST_NETCFG 结构体: <a href="#">NET_DVR_ALARMHOST_NETCFG</a> (V1.1 新增支持)
设置上传中心网络参数配置	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_ALARMHOST_NETCFG 结构体: <a href="#">NET_DVR_ALARMHOST_NETCFG</a> (V1.1 新增支持)
获取无线网络参数配置	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_ALARMHOST_WIRELESS_NETWORK_CFG 结构体: <a href="#">NET_DVR_ALARMHOST_WIRELESS_NETWORK_CFG</a> (V1.1 新增支持)
设置无线网络参数配置	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_ALARMHOST_WIRELESS_NETWORK_CFG 结构体: <a href="#">NET_DVR_ALARMHOST_WIRELESS_NETWORK_CFG</a> (V1.1 新增支持)
获取数据上传方式参数	<a href="#">NET_DVR_GetDeviceConfig</a>	命令: NET_DVR_GET_ALARMHOST_REPORT_CENTER_V40 结构体: <a href="#">NET_DVR_ALARMHOST_REPORT_CENTER_CFG_V40</a> (V1.1 新增支持)
设置数据上传方式参数	<a href="#">NET_DVR_SetDeviceConfig</a>	命令: NET_DVR_SET_ALARMHOST_REPORT_CENTER_V40 结构体: <a href="#">NET_DVR_ALARMHOST_REPORT_CENTER_CFG_V40</a> (V1.1 新增支持)
获取多网卡配置参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_NETCFG_MULTI 结构体: <a href="#">NET_DVR_NETCFG_MULTI</a> (三层架构门禁新增)
设置多网卡配置参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_NETCFG_MULTI 结构体: <a href="#">NET_DVR_NETCFG_MULTI</a> (三层架构门禁新增)
<b>RS485/防区/触发器配置和控制</b>		
获取 RS485 参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_ALARM_RS485CFG

		结构体: <a href="#">NET_DVR_ALARM_RS485CFG</a>
设置 RS485 参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_ALARM_RS485CFG 结构体: <a href="#">NET_DVR_ALARM_RS485CFG</a>
获取防区参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_ALARMIN_PARAM 结构体: <a href="#">NET_DVR_ALARMIN_PARAM</a>
设置防区参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_ALARMIN_PARAM 结构体: <a href="#">NET_DVR_ALARMIN_PARAM</a>
对防区布防	<a href="#">NET_DVR_AlarmHostSetupAlarmChan</a>	
对防区撤防	<a href="#">NET_DVR_AlarmHostCloseAlarmChan</a>	
获取触发器参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_ALARMOUT_PARAM 结构体: <a href="#">NET_DVR_ALARMOUT_PARAM</a>
设置触发器	<a href="#">NET_DVR_SetAlarmHostOut</a>	
设置触发器参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_ALARMOUT_PARAM 结构体: <a href="#">NET_DVR_ALARMOUT_PARAM</a>
获取设备状态(防区、触发器等)	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_ALARMHOST_MAIN_STATUS 结构体: <a href="#">NET_DVR_ALARMHOST_MAIN_STATUS</a>
<b>用户配置</b>		
获取设备用户配置	<a href="#">NET_DVR_GetAlarmDeviceUser</a>	
设置设备用户配置	<a href="#">NET_DVR_SetAlarmDeviceUser</a>	
<b>门禁相关参数配置</b>		
获取周计划参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_WEEK_PLAN_CFG、 NET_DVR_GET_VERIFY_WEEK_PLAN、 NET_DVR_GET_CARD_RIGHT_WEEK_PLAN 结构体: <a href="#">NET_DVR_WEEK_PLAN_CFG</a>
设置周计划参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_WEEK_PLAN_CFG、 NET_DVR_SET_VERIFY_WEEK_PLAN、 NET_DVR_SET_CARD_RIGHT_WEEK_PLAN 结构体: <a href="#">NET_DVR_WEEK_PLAN_CFG</a>
获取假日计划参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_DOOR_STATUS_HOLIDAY_PLAN、 NET_DVR_GET_VERIFY_HOLIDAY_PLAN、 NET_DVR_GET_CARD_RIGHT_HOLIDAY_PLAN 结构体: <a href="#">NET_DVR_HOLIDAY_PLAN_CFG</a>
设置假日计划参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_DOOR_STATUS_HOLIDAY_PLAN、 NET_DVR_SET_VERIFY_HOLIDAY_PLAN、 NET_DVR_SET_CARD_RIGHT_HOLIDAY_PLAN 结构体: <a href="#">NET_DVR_HOLIDAY_PLAN_CFG</a>
获取假日组参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_DOOR_STATUS_HOLIDAY_GROUP、

		NET_DVR_GET_VERIFY_HOLIDAY_GROUP、 NET_DVR_GET_CARD_RIGHT_HOLIDAY_GROUP 结构体: <a href="#">NET_DVR_HOLIDAY_GROUP_CFG</a>
设置假日组参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_DOOR_STATUS_HOLIDAY_GROUP、 NET_DVR_SET_VERIFY_HOLIDAY_GROUP、 NET_DVR_SET_CARD_RIGHT_HOLIDAY_GROUP 结构体: <a href="#">NET_DVR_HOLIDAY_GROUP_CFG</a>
获取计划模板参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_DOOR_STATUS_PLAN_TEMPLATE、 NET_DVR_GET_VERIFY_PLAN_TEMPLATE NET_DVR_GET_CARD_RIGHT_PLAN_TEMPLATE 结构体: <a href="#">NET_DVR_PLAN_TEMPLATE</a>
设置计划模板参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_DOOR_STATUS_PLAN_TEMPLATE、 NET_DVR_SET_VERIFY_PLAN_TEMPLATE、 NET_DVR_SET_CARD_RIGHT_PLAN_TEMPLATE 结构体: <a href="#">NET_DVR_PLAN_TEMPLATE</a>
获取门参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_DOOR_CFG 结构体: <a href="#">NET_DVR_DOOR_CFG</a>
设置门参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_DOOR_CFG 结构体: <a href="#">NET_DVR_DOOR_CFG</a>
获取门状态计划参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_DOOR_STATUS_PLAN 结构体: <a href="#">NET_DVR_DOOR_STATUS_PLAN</a>
设置门状态计划参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_DOOR_STATUS_PLAN 结构体: <a href="#">NET_DVR_DOOR_STATUS_PLAN</a>
获取读卡器验证计划参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_CARD_READER_PLAN 结构体: <a href="#">NET_DVR_CARD_READER_PLAN</a>
设置读卡器验证计划参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_CARD_READER_PLAN 结构体: <a href="#">NET_DVR_CARD_READER_PLAN</a>
获取群组参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_GROUP_CFG 结构体: <a href="#">NET_DVR_GROUP_CFG</a>
设置群组参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_GROUP_CFG 结构体: <a href="#">NET_DVR_GROUP_CFG</a>
获取多重卡参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_MULTI_CARD_CFG 结构体: <a href="#">NET_DVR_MULTI_CARD_CFG</a>
设置多重卡参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_MULTI_CARD_CFG 结构体: <a href="#">NET_DVR_MULTI_CARD_CFG</a>
获取反潜回参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_SNEAK_CFG 结构体: <a href="#">NET_DVR_ANTI_SNEAK_CFG</a>
设置反潜回参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_SNEAK_CFG 结构体: <a href="#">NET_DVR_ANTI_SNEAK_CFG</a>

获取读卡器反潜回参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_CARD_READER_ANTI_SNEAK_CFG 结构体: <a href="#">NET_DVR_CARD_READER_ANTI_SNEAK_CFG</a>
设置读卡器反潜回参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_CARD_READER_ANTI_SNEAK_CFG 结构体: <a href="#">NET_DVR_CARD_READER_ANTI_SNEAK_CFG</a>
获取多门互锁参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_MULTI_DOOR_INTERLOCK_CFG 结构体: <a href="#">NET_DVR_MULTI_DOOR_INTERLOCK_CFG</a>
设置多门互锁参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_MULTI_DOOR_INTERLOCK_CFG 结构体: <a href="#">NET_DVR_MULTI_DOOR_INTERLOCK_CFG</a>
获取读卡器参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_CARD_READER_CFG 结构体: <a href="#">NET_DVR_CARD_READER_CFG</a>
设置读卡器参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_CARD_READER_CFG 结构体: <a href="#">NET_DVR_CARD_READER_CFG</a>
获取门禁主机工作状态	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_ACS_WORK_STATUS 结构体: <a href="#">NET_DVR_ACS_WORK_STATUS</a>
获取事件报警输入参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_CASE_SENSOR_CFG 结构体: <a href="#">NET_DVR_CASE_SENSOR_CFG</a>
设置事件报警输入参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_CASE_SENSOR_CFG 结构体: <a href="#">NET_DVR_CASE_SENSOR_CFG</a>
获取手机关联门权限参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_PHONE_DOOR_RIGHT_CFG 结构体: <a href="#">NET_DVR_PHONE_DOOR_RIGHT_CFG</a> (V1.1 新增)
设置手机关联门权限参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_PHONE_DOOR_RIGHT_CFG 结构体: <a href="#">NET_DVR_PHONE_DOOR_RIGHT_CFG</a> (V1.1 新增)
获取事件/卡号联动配置	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_EVENT_CARD_LINKAGE_CFG 结构体: <a href="#">NET_DVR_EVENT_CARD_LINKAGE_CFG</a> (V1.1 新增)
设置事件/卡号联动配置	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_EVENT_CARD_LINKAGE_CFG 结构体: <a href="#">NET_DVR_EVENT_CARD_LINKAGE_CFG</a> (V1.1 新增)
获取门禁主机参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_ACS_CFG 结构体: <a href="#">NET_DVR_ACS_CFG</a> (V1.1 新增)
设置门禁主机参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_ACS_CFG 结构体: <a href="#">NET_DVR_ACS_CFG</a> (V1.1 新增)
获取门禁主机串口外设参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_ACS_EXTERNAL_DEV_CFG 结构体: <a href="#">NET_DVR_ACS_EXTERNAL_DEV_CFG</a> (人员通道 V1.0 新增)
设置门禁主机串口外设参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_ACS_EXTERNAL_DEV_CFG

		结构体: <a href="#">NET_DVR_ACS_EXTERNAL_DEV_CFG</a> (人员通道 V1.0 新增)
获取人员通道参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_PERSONNEL_CHANNEL_CFG 结构体: <a href="#">NET_DVR_PERSONNEL_CHANNEL_CFG</a> (人员通道 V1.0 新增)
设置人员通道参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_PERSONNEL_CHANNEL_CFG 结构体: <a href="#">NET_DVR_PERSONNEL_CHANNEL_CFG</a> (人员通道 V1.0 新增)
获取人数统计参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_PERSON_STATISTICS_CFG 结构体: <a href="#">NET_DVR_PERSON_STATISTICS_CFG</a> (人员通道 V1.0 新增)
设置人数统计参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_PERSON_STATISTICS_CFG 结构体: <a href="#">NET_DVR_PERSON_STATISTICS_CFG</a> (人员通道 V1.0 新增)
获取屏幕字符串显示参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_ACS_SCREEN_DISPLAY_CFG 结构体: <a href="#">NET_DVR_ACS_SCREEN_DISPLAY_CFG</a> (人员通道 V1.0 新增)
设置屏幕字符串显示参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_ACS_SCREEN_DISPLAY_CFG 结构体: <a href="#">NET_DVR_ACS_SCREEN_DISPLAY_CFG</a> (人员通道 V1.0 新增)
获取人员通道闸门时间参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_GATE_TIME_CFG 结构体: <a href="#">NET_DVR_GATE_TIME_CFG</a> (人员通道 V1.0 新增)
设置人员通道闸门时间参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_GATE_TIME_CFG 结构体: <a href="#">NET_DVR_GATE_TIME_CFG</a> (人员通道 V1.0 新增)
获取人脸检测规则	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_VCA_GET_FACEDETECT_RULECFG_V41 结构体: <a href="#">NET_DVR_FACEDETECT_RULECFG_V41</a> (人员通道 V1.0 新增)
设置人脸检测规则	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_VCA_SET_FACEDETECT_RULECFG_V41 结构体: <a href="#">NET_DVR_FACEDETECT_RULECFG_V41</a> (人员通道 V1.0 新增)
下发平台认证结果	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_PLATFORM_VERIFY_CFG 结构体: <a href="#">NET_DVR_PLATFORM_VERIFY_CFG</a> (人员通道 V1.0 新增)
<b>卡参数配置</b>		
启动长连接远程配置	<a href="#">NET_DVR_StartRemoteConfig</a>	配置命令: NET_DVR_GET_CARD_CFG NET_DVR_SET_CARD_CFG
发送长连接数据	<a href="#">NET_DVR_SendRemoteConfig</a>	配置命令:

		NET_DVR_GET_CARD_CFG NET_DVR_SET_CARD_CFG
关闭长连接配置接口所创建的句柄，释放资源	<a href="#">NET_DVR_StopRemoteConfig</a>	
<b>指纹管理配置</b>		
启动长连接远程配置	<a href="#">NET_DVR_StartRemoteConfig</a>	配置命令： NET_DVR_GET_FINGERPRINT_CFG NET_DVR_SET_FINGERPRINT_CFG (V1.1 新增)
发送长连接数据	<a href="#">NET_DVR_SendRemoteConfig</a>	配置命令： NET_DVR_SET_FINGERPRINT_CFG (V1.1 新增)
关闭长连接配置接口所创建的句柄，释放资源	<a href="#">NET_DVR_StopRemoteConfig</a>	
删除指纹参数	<a href="#">NET_DVR_RemoteControl</a>	配置命令： NET_DVR_DEL_FINGERPRINT_CFG (V1.1 新增)
<b>卡密码开门使能配置</b>		
启动长连接远程配置	<a href="#">NET_DVR_StartRemoteConfig</a>	配置命令： NET_DVR_GET_CARD_PASSWD_CFG NET_DVR_SET_CARD_PASSWD_CFG (V1.1 新增)
发送长连接数据	<a href="#">NET_DVR_SendRemoteConfig</a>	配置命令： NET_DVR_SET_CARD_PASSWD_CFG (V1.1 新增)
关闭长连接配置接口所创建的句柄，释放资源	<a href="#">NET_DVR_StopRemoteConfig</a>	
<b>远程控制</b>		
清空门禁主机参数	<a href="#">NET_DVR_RemoteControl</a>	控制命令： NET_DVR_CLEAR_ACS_PARAM
<b>门禁控制</b>		
门禁控制	<a href="#">NET_DVR_ControlGateway</a>	
<b>客户端报警布防</b>		
注册回调函数，接收设备报警消息等	<a href="#">NET_DVR_SetDVRMessageCallBack_V31</a>	支持的报警信息类型有： 1.COMM_ALARM_ACS(门禁报警信息)， 对应结构： <a href="#">NET_DVR_ACS_ALARM_INFO</a> 2.COMM_ID_INFO_ALARM(身份证刷卡信息，人员通道 V1.0 新增)， 对应结构： <a href="#">NET_DVR_ID_CARD_INFO_ALARM</a> 3.COMM_PASSNUM_INFO_ALARM(通行人数信息，人

		员通道 V1.0 新增), 对应结构: <a href="#">NET_DVR_PASSNUM_INFO_ALARM</a>
建立报警上传通道, 获取报警等信息	<a href="#">NET_DVR_SetupAlarmChan_V41</a>	
撤销报警上传通道	<a href="#">NET_DVR_CloseAlarmChan_V30</a>	
<b>设备维护管理</b>		
设置远程升级时网络环境	<a href="#">NET_DVR_SetNetworkEnvironment</a>	
远程升级	<a href="#">NET_DVR_Upgrade_V40</a>	升级类型: ENUM_UPGRADE_DVR、ENUM_UPGRADE_ACS
获取远程升级的进度	<a href="#">NET_DVR_GetUpgradeProgress</a>	
获取远程升级的状态	<a href="#">NET_DVR_GetUpgradeState</a>	
关闭远程升级句柄, 释放资源	<a href="#">NET_DVR_CloseUpgradeHandle</a>	
恢复设备默认参数	<a href="#">NET_DVR_RestoreConfig</a>	
重启设备	<a href="#">NET_DVR_RebootDVR</a>	

## 5.3 指纹门禁一体机功能接口

指纹门禁一体机包括但不限于以下产品型号:

DS-K1T105C(-C)、DS-K1T105E(-C)、DS-K1T105M(-C)、DS-K1T200CF(-C)、DS-K1T200EF(-C)、DS-K1T200MF(-C)、DS-K1T300CF(-C)、DS-K1T300EF(-C)、DS-K1T300MF(-C)。

指纹门禁一体机除通用接口的功能外, 还支持以下接口的功能, 包括参数配置、远程控制、报警上传和设备维护等。

功能	接口	相关参数
获取设备能力集		
获取设备通用能力集	<a href="#">NET_DVR_GetDeviceAbility</a>	能力集类型: DEVICE_SOFTHARDWARE_ABILITY、 DEVICE_NETWORK_ABILITY、 DEVICE_NETAPP_ABILITY、 DEVICE_SERIAL_ABILITY、 DEVICE_JPEG_CAP_ABILITY
获取智能能力集	<a href="#">NET_DVR_GetDeviceAbility</a>	能力集类型: DEVICE_ABILITY_INFO(智能通道分析能力 VcaChanAbility)
获取门禁能力集	<a href="#">NET_DVR_GetDeviceAbility</a>	能力集类型: ACS_ABILITY
<b>通用参数配置</b>		
获取设备参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_DEVICECFG_V40



		结构体: <a href="#">NET_DVR_DEVICECFG_V40</a>
设置设备参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_DEVICECFG_V40 结构体: <a href="#">NET_DVR_DEVICECFG_V40</a>
获取时间参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_TIMECFG 结构体: <a href="#">NET_DVR_TIME</a>
设置时间参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_TIMECFG 结构体: <a href="#">NET_DVR_TIME</a>
获取时区和夏时制参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_ZONEANDDST 结构体: <a href="#">NET_DVR_ZONEANDDST</a>
设置时区和夏时制参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_ZONEANDDST 结构体: <a href="#">NET_DVR_ZONEANDDST</a>
获取网络参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_NETCFG_V30 结构体: <a href="#">NET_DVR_NETCFG_V30</a>
设置网络参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_NETCFG_V30 结构体: <a href="#">NET_DVR_NETCFG_V30</a>
获取 IP 监控设备无线参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_WIFI_CFG 结构体: <a href="#">NET_DVR_WIFI_CFG</a>
设置 IP 监控设备无线参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_WIFI_CFG 结构体: <a href="#">NET_DVR_WIFI_CFG</a>
<b>门禁相关参数配置</b>		
获取周计划参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_WEEK_PLAN_CFG、 NET_DVR_GET_VERIFY_WEEK_PLAN、 NET_DVR_GET_CARD_RIGHT_WEEK_PLAN 结构体: <a href="#">NET_DVR_WEEK_PLAN_CFG</a>
设置周计划参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_WEEK_PLAN_CFG、 NET_DVR_SET_VERIFY_WEEK_PLAN、 NET_DVR_SET_CARD_RIGHT_WEEK_PLAN 结构体: <a href="#">NET_DVR_WEEK_PLAN_CFG</a>
获取假日计划参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_DOOR_STATUS_HOLIDAY_PLAN、 NET_DVR_GET_VERIFY_HOLIDAY_PLAN、 NET_DVR_GET_CARD_RIGHT_HOLIDAY_PLAN 结构体: <a href="#">NET_DVR_HOLIDAY_PLAN_CFG</a>
设置假日计划参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_DOOR_STATUS_HOLIDAY_PLAN、 NET_DVR_SET_VERIFY_HOLIDAY_PLAN、 NET_DVR_SET_CARD_RIGHT_HOLIDAY_PLAN 结构体: <a href="#">NET_DVR_HOLIDAY_PLAN_CFG</a>
获取假日组参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_DOOR_STATUS_HOLIDAY_GROUP、 NET_DVR_GET_VERIFY_HOLIDAY_GROUP、 NET_DVR_GET_CARD_RIGHT_HOLIDAY_GROUP

		结构体: <a href="#">NET_DVR_HOLIDAY_GROUP_CFG</a>
设置假日组参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_DOOR_STATUS_HOLIDAY_GROUP、 NET_DVR_SET_VERIFY_HOLIDAY_GROUP、 NET_DVR_SET_CARD_RIGHT_HOLIDAY_GROUP 结构体: <a href="#">NET_DVR_HOLIDAY_GROUP_CFG</a>
获取计划模板参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_DOOR_STATUS_PLAN_TEMPLATE、 NET_DVR_GET_VERIFY_PLAN_TEMPLATE NET_DVR_GET_CARD_RIGHT_PLAN_TEMPLATE 结构体: <a href="#">NET_DVR_PLAN_TEMPLATE</a>
设置计划模板参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_DOOR_STATUS_PLAN_TEMPLATE、 NET_DVR_SET_VERIFY_PLAN_TEMPLATE、 NET_DVR_SET_CARD_RIGHT_PLAN_TEMPLATE 结构体: <a href="#">NET_DVR_PLAN_TEMPLATE</a>
获取门参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_DOOR_CFG 结构体: <a href="#">NET_DVR_DOOR_CFG</a>
设置门参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_DOOR_CFG 结构体: <a href="#">NET_DVR_DOOR_CFG</a>
获取门状态计划参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_DOOR_STATUS_PLAN 结构体: <a href="#">NET_DVR_DOOR_STATUS_PLAN</a>
设置门状态计划参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_DOOR_STATUS_PLAN 结构体: <a href="#">NET_DVR_DOOR_STATUS_PLAN</a>
获取读卡器验证计划参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_CARD_READER_PLAN 结构体: <a href="#">NET_DVR_CARD_READER_PLAN</a>
设置读卡器验证计划参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_CARD_READER_PLAN 结构体: <a href="#">NET_DVR_CARD_READER_PLAN</a>
获取反潜回参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_SNEAK_CFG 结构体: <a href="#">NET_DVR_ANTI_SNEAK_CFG</a>
设置反潜回参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_SNEAK_CFG 结构体: <a href="#">NET_DVR_ANTI_SNEAK_CFG</a>
获取读卡器反潜回参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_CARD_READER_ANTI_SNEAK_CFG 结构体: <a href="#">NET_DVR_CARD_READER_ANTI_SNEAK_CFG</a>
设置读卡器反潜回参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_CARD_READER_ANTI_SNEAK_CFG 结构体: <a href="#">NET_DVR_CARD_READER_ANTI_SNEAK_CFG</a>
获取读卡器参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_CARD_READER_CFG 结构体: <a href="#">NET_DVR_CARD_READER_CFG</a>
设置读卡器参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_CARD_READER_CFG 结构体: <a href="#">NET_DVR_CARD_READER_CFG</a>
获取门禁一体机工作状态	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_ACS_WORK_STATUS 结构体: <a href="#">NET_DVR_ACS_WORK_STATUS</a>

获取事件/卡号联动配置	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_EVENT_CARD_LINKAGE_CFG 结构体: <a href="#">NET_DVR_EVENT_CARD_LINKAGE_CFG</a> (V1.1 新增)
设置事件/卡号联动配置	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_EVENT_CARD_LINKAGE_CFG 结构体: <a href="#">NET_DVR_EVENT_CARD_LINKAGE_CFG</a> (V1.1 新增)
获取门禁一体机参数	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_DVR_GET_ACS_CFG 结构体: <a href="#">NET_DVR_ACS_CFG</a> (V1.1 新增)
设置门禁一体机参数	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_DVR_SET_ACS_CFG 结构体: <a href="#">NET_DVR_ACS_CFG</a> (V1.1 新增)
获取人脸检测规则	<a href="#">NET_DVR_GetDVRConfig</a>	命令: NET_VCA_GET_FACEDETECT_RULECFG_V41 结构体: <a href="#">NET_DVR_FACEDETECT_RULECFG_V41</a>
设置人脸检测规则	<a href="#">NET_DVR_SetDVRConfig</a>	命令: NET_VCA_SET_FACEDETECT_RULECFG_V41 结构体: <a href="#">NET_DVR_FACEDETECT_RULECFG_V41</a>
获取卡号关联用户信息参数	<a href="#">NET_DVR_GetDeviceConfig</a>	命令: NET_DVR_GET_CARD_USERINFO_CFG 结构体: <a href="#">NET_DVR_CARD_USER_INFO_CFG</a>
设置卡号关联用户信息参数	<a href="#">NET_DVR_SetDeviceConfig</a>	命令: NET_DVR_SET_CARD_USERINFO_CFG 结构体: <a href="#">NET_DVR_CARD_USER_INFO_CFG</a>
<b>卡参数配置</b>		
启动长连接远程配置	<a href="#">NET_DVR_StartRemoteConfig</a>	配置命令: NET_DVR_GET_CARD_CFG NET_DVR_SET_CARD_CFG
发送长连接数据	<a href="#">NET_DVR_SendRemoteConfig</a>	配置命令: NET_DVR_GET_CARD_CFG NET_DVR_SET_CARD_CFG
关闭长连接配置接口所创建的句柄, 释放资源	<a href="#">NET_DVR_StopRemoteConfig</a>	
<b>指纹管理配置</b>		
启动长连接远程配置	<a href="#">NET_DVR_StartRemoteConfig</a>	配置命令: NET_DVR_GET_FINGERPRINT_CFG NET_DVR_SET_FINGERPRINT_CFG
发送长连接数据	<a href="#">NET_DVR_SendRemoteConfig</a>	配置命令: NET_DVR_SET_FINGERPRINT_CFG (V1.1 新增)
关闭长连接配置接口所创建的句柄, 释放资源	<a href="#">NET_DVR_StopRemoteConfig</a>	
删除指纹参数	<a href="#">NET_DVR_RemoteControl</a>	配置命令: NET_DVR_DEL_FINGERPRINT_CFG

卡密码开门使能配置		
启动长连接远程配置	<a href="#">NET_DVR_StartRemoteConfig</a>	配置命令： NET_DVR_GET_CARD_PASSWD_CFG NET_DVR_SET_CARD_PASSWD_CFG (V1.1 新增)
发送长连接数据	<a href="#">NET_DVR_SendRemoteConfig</a>	配置命令： NET_DVR_SET_CARD_PASSWD_CFG
关闭长连接配置接口所创建的句柄，释放资源	<a href="#">NET_DVR_StopRemoteConfig</a>	
远程控制		
清空门禁一体机参数	<a href="#">NET_DVR_RemoteControl</a>	控制命令： NET_DVR_CLEAR_ACS_PARAM
门禁控制		
门禁控制	<a href="#">NET_DVR_ControlGateway</a>	
客户端报警布防		
注册回调函数，接收设备报警消息等	<a href="#">NET_DVR_SetDVRMessageCallBack_V31</a>	支持的报警信息类型有： COMM_ALARM_ACS (报警信息结构： <a href="#">NET_DVR_ACS_ALARM_INFO</a> )
建立报警上传通道，获取报警等信息	<a href="#">NET_DVR_SetupAlarmChan_V41</a>	
撤销报警上传通道	<a href="#">NET_DVR_CloseAlarmChan_V30</a>	
手动抓拍		
单帧数据捕获并保存成 JPEG 图	<a href="#">NET_DVR_CaptureJPEGPicture</a>	
单帧数据捕获并保存成 JPEG 存放在指定的内存空间中	<a href="#">NET_DVR_CaptureJPEGPicture_NEW</a>	
网络触发连拍	<a href="#">NET_DVR_ContinuousShoot</a>	
设备维护管理		
设置远程升级时网络环境	<a href="#">NET_DVR_SetNetworkEnvironment</a>	
远程升级	<a href="#">NET_DVR_Upgrade_V40</a>	升级类型：ENUM_UPGRADE_DVR、ENUM_UPGRADE_ACS
获取远程升级的进度	<a href="#">NET_DVR_GetUpgradeProgress</a>	
获取远程升级的状态	<a href="#">NET_DVR_GetUpgradeState</a>	
关闭远程升级句柄，释放资源	<a href="#">NET_DVR_CloseUpgradeHandle</a>	
恢复设备默认参数	<a href="#">NET_DVR_RestoreConfig</a>	
重启设备	<a href="#">NET_DVR_RebootDVR</a>	

## 6 函数说明

### 6.1 SDK 初始化

#### 6.1.1 初始化 SDK NET\_DVR\_Init

函 数： BOOL NET\_DVR\_Init()  
参 数： 无  
返回值： TRUE 表示成功，FALSE 表示失败。  
说 明： 调用设备网络 SDK 其他函数的前提。

[返回目录](#)

#### 6.1.2 释放 SDK 资源 NET\_DVR\_Cleanup

函 数： BOOL NET\_DVR\_Cleanup()  
参 数： 无  
返回值： TRUE 表示成功，FALSE 表示失败。  
说 明： 在结束之前最后调用。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

[返回目录](#)

### 6.2 SDK 本地功能

#### SDK 本地参数配置

#### 6.2.1 获取 SDK 本地参数 NET\_DVR\_GetSDKLocalCfg

函 数： BOOL NET\_DVR\_GetSDKLocalCfg(NET\_SDK\_LOCAL\_CFG\_TYPE enumType, void \*lpOutBuff)  
参 数： [in] enumType 配置类型，不同的取值对应不同的 SDK 参数，详见表 6.1  
[out] lpOutBuff 输出参数，不同的配置类型，输出参数对应不同的结构，详见表 6.1  
返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。  
说 明：

表 6.1 本地参数类型

enumType 宏定义	类型值	含义	lpOutBuff 对应结构体
NET_SDK_LOCAL_CFG_TYPE_TCP_PORT_BIND	0	本地 TCP 端口绑定配置	<a href="#">NET_DVR_LOCAL_TCP_PORT_BIND_CFG</a>

NET_SDK_LOCAL_CFG_TYPE_UDP_PORT_BIND	1	本地 UDP 端口绑定配置	<a href="#">NET_DVR_LOCAL_UDP_PORT_BIND_CFG</a>
NET_SDK_LOCAL_CFG_TYPE_MEM_POOL	2	内存池本地配置	<a href="#">NET_DVR_LOCAL_MEM_POOL_CFG</a>
NET_SDK_LOCAL_CFG_TYPE_MODULE_RECV_TIMEOUT	3	按模块配置超时时间	<a href="#">NET_DVR_LOCAL_MODULE_RECV_TIMEOUT_CFG</a>
NET_SDK_LOCAL_CFG_TYPE_ABILITY_PARSE	4	是否使用能力集解析库	<a href="#">NET_DVR_LOCAL_ABILITY_PARSE_CFG</a>
NET_SDK_LOCAL_CFG_TYPE_TALK_MODE	5	对讲模式配置	<a href="#">NET_DVR_LOCAL_TALK_MODE_CFG</a>
NET_SDK_LOCAL_CFG_TYPE_CHECK_DEV	10	心跳交互间隔时间配置	<a href="#">NET_DVR_LOCAL_CHECK_DEV</a>

[返回目录](#)

## 6.2.2 设置 SDK 本地参数 **NET\_DVR\_SetSDKLocalCfg**

函 数： BOOL NET\_DVR\_SetSDKLocalCfg(NET\_SDK\_LOCAL\_CFG\_TYPE enumType, void\* const lpInBuff)  
 参 数： [in] enumType 配置类型，不同的取值对应不同的 SDK 参数，详见表 6.2  
           [in] lpInBuff 输入参数，不同的配置类型，输出参数对应不同的结构，详见表 6.2  
 返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明：

表 6.2 本地参数类型

enumType 宏定义	类型值	含义	lpInBuff 对应结构体
NET_SDK_LOCAL_CFG_TYPE_TCP_PORT_BIND	0	本地 TCP 端口绑定配置	<a href="#">NET_DVR_LOCAL_TCP_PORT_BIND_CFG</a>
NET_SDK_LOCAL_CFG_TYPE_UDP_PORT_BIND	1	本地 UDP 端口绑定配置	<a href="#">NET_DVR_LOCAL_UDP_PORT_BIND_CFG</a>
NET_SDK_LOCAL_CFG_TYPE_MEM_POOL	2	内存池本地配置	<a href="#">NET_DVR_LOCAL_MEM_POOL_CFG</a>
NET_SDK_LOCAL_CFG_TYPE_MODULE_RECV_TIMEOUT	3	按模块配置超时时间	<a href="#">NET_DVR_LOCAL_MODULE_RECV_TIMEOUT_CFG</a>
NET_SDK_LOCAL_CFG_TYPE_ABILITY_PARSE	4	是否使用能力集解析库	<a href="#">NET_DVR_LOCAL_ABILITY_PARSE_CFG</a>
NET_SDK_LOCAL_CFG_TYPE_TALK_MODE	5	对讲模式配置	<a href="#">NET_DVR_LOCAL_TALK_MODE_CFG</a>
NET_SDK_LOCAL_CFG_TYPE_CHECK_DEV	10	心跳交互间隔时间配置	<a href="#">NET_DVR_LOCAL_CHECK_DEV</a>
NET_SDK_LOCAL_CFG_TYPE_CHAR_ENCODE	13	配置字符编码相关处理回调	<a href="#">NET_DVR_LOCAL_BYTE_ENCODE_CONVERT</a>

[返回目录](#)

## 连接和接收超时时间及重连设置

## 6.2.3 设置网络连接超时时间和连接尝试次数 **NET\_DVR\_SetConnectTime**

函 数： BOOL NET\_DVR\_SetConnectTime(DWORD dwWaitTime, DWORD dwTryTime)  
 参 数： [in] dwWaitTime 超时时间，单位毫秒，取值范围[300,75000]，实际最大超时时间因系统的 connect 超时时间而不同。  
           [in] dwTryTimes 连接尝试次数（保留）

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说明:** SDK 默认建立连接的超时时间为 3 秒。SDK4.0 及以后版本中当设置的超时时间超过或低于限制的值时接口不返回失败, 将取最接近的上下限限制值作为实际的超时时间。

[返回目录](#)

## 6.2.4 设置重连功能 **NET\_DVR\_SetReconnect**

函数: BOOL NET\_DVR\_SetReconnect (DWORD dwInterval, BOOL bEnableRecon)

参数: [in] dwInterval 重连间隔, 单位:毫秒  
[in] bEnableRecon 是否重连, 0-不重连, 1-重连, 参数默认值为 1

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说明:** 该接口可以同时控制预览、透明通道和布防的重连功能。不调用该接口时, SDK 默认启动预览、透明通道和布防的重连功能, 重连时间间隔为 5 秒。

[返回目录](#)

## 6.2.5 设置接收超时时间 **NET\_DVR\_SetRecvTimeOut**

函数: BOOL NET\_DVR\_SetRecvTimeOut(DWORD nRecvTimeOut)

参数: [in] nRecvTimeOut 接收超时时间, 单位毫秒, 默认为 5000, 最小为 3000 毫秒

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说明:** 该接口用于设置接收超时时间, 例如预览接收实时流数据、回放下载接收录像数据、报警接收报警信息等接收超时时间。

[返回目录](#)

## 多网卡绑定

## 6.2.6 获取所有 IP, 用于支持多网卡接口 **NET\_DVR\_GetLocalIP**

函数: BOOL NET\_DVR\_GetLocalIP(char strIP[16][16], DWORD \*pValidNum, BOOL \*pEnableBind)

参数: [out] strIP 存放 IP 的缓冲区, 不能为空  
[out] pValidNum 所有有效 IP 的数量  
[out] pEnableBind 是否绑定

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说明:** 该接口获取客户端本地多网卡的所有 IP 地址, 可以通过接口 NET\_DVR\_SetValidIP 选择要使用的 IP 地址。

[返回目录](#)

### 6.2.7 设置 IP 绑定 **NET\_DVR\_SetValidIP**

函 数: BOOL NET\_DVR\_SetValidIP(DWORD dwIPIndex, BOOL bEnableBind)

参 数: [in] dwIPIndex 选择使用的 IP 下标, 由 NET\_DVR\_GetLocalIP 获取  
[in] bEnableBind 是否绑定

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明:

[返回目录](#)

## SDK 版本、状态和能力

### 6.2.8 获取 SDK 的版本号和 build 信息 **NET\_DVR\_GetSDKBuildVersion**

函 数: DWORD NET\_DVR\_GetSDKBuildVersion()

参 数:

返回值: 获取 SDK 的版本号和 build 信息。

说 明: SDK 的版本号和 build 信息。2 个高字节表示版本号: 25~32 位表示主版本号, 17~24 位表示次版本号; 2 个低字节表示 build 信息。如 0x03000101: 表示版本号为 3.0, build 号是 0101。

[返回目录](#)

### 6.2.9 获取当前 SDK 的状态信息 **NET\_DVR\_GetSDKState**

函 数: BOOL NET\_DVR\_GetSDKState(LPNET\_DVR\_SDKSTATE pSDKState);

参 数: [out] pSDKState 状态信息结构, 请参见结构体: [NET\\_DVR\\_SDKSTATE](#)

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明:

[返回目录](#)

### 6.2.10 获取当前 SDK 的功能信息 **NET\_DVR\_GetSDKAbility**

函 数: BOOL NET\_DVR\_GetSDKAbility(LPNET\_DVR\_SDKABL pSDKAbI)

参 数: [out] pSDKAbI 功能信息, 请参见结构体: [NET\\_DVR\\_SDKABL](#)

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明:

[返回目录](#)



## SDK 启用写日志

### 6.2.11 启用写日志文件 **NET\_DVR\_SetLogToFile**

函 数: `BOOL NET_DVR_SetLogToFile(DWORD bLogEnable, char* strLogDir, BOOL bAutoDel)`

参 数: `[in] bLogEnable` 日志的等级（默认为 0）：  
 0-表示关闭日志，  
 1-表示只输出 ERROR 错误日志，  
 2-输出 ERROR 错误信息和 DEBUG 调试信息，  
 3-输出 ERROR 错误信息、DEBUG 调试信息和 INFO 普通信息等所有信息

`[in] strLogDir` 日志文件的路径，windows 默认值为"C:\\SdkLog\\"；linux 默认值"/home/sdklog/"

`[in] bAutoDel` 是否删除超出的文件数，默认值为 TRUE

返回值: TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明: 日志文件路径必须是绝对路径，且以"\"结尾，例如"C:\\SdkLog\\"，建议用户先手动创建文件。若未指定文件路径，则采用默认路径"C:\\SdkLog\\"。可多次调用该接口创建新的日志文件，同时最多支持创建 10 个文件，当设置了删除超出的文件时（即 bAutoDel 为 TRUE），那么将会自动删除超出的文件。更改目录时到下一次写文件时才会使用新的目录写文件。

[返回目录](#)

## 异常消息回调

### 6.2.12 注册接收异常、重连等消息的窗口句柄或回调函数

#### **NET\_DVR\_SetExceptionCallBack\_V30**

函 数: [Windows 系统下](#):  
`BOOL NET_DVR_SetExceptionCallBack_V30 (UINT nMessage, HWND hWnd, fExceptionCallBack cbExceptionCallBack, void* pUser)`

[Linux 系统下](#):  
`BOOL NET_DVR_SetExceptionCallBack_V30(UINT nMessage, void* hWnd, fExceptionCallBack cbExceptionCallBack, void* pUser)`

参 数: `[in] nMessage` 消息，Linux 下该参数保留  
`[in] hWnd` 接收异常消息的窗口句柄，Linux 下该参数保留  
`[in] cbExceptionCallBack` 接收异常消息的回调函数，回调当前异常的相关信息  
`[in] pUser` 用户数据

`typedef void(CALLBACK* fExceptionCallBack)(DWORD dwType, LONG lUserID, LONG lHandle, void* pUser)`

`[out] dwType` 异常或重连等消息的类型，详见表 6.3

[out]IUserID                    登录 ID  
[out]IHandle                   出现异常的相应类型的句柄  
[out]pUser                    用户数据

表 6.3 异常消息类型

dwType 宏定义	宏定义值	含义
EXCEPTION_EXCHANGE	0x8000	用户交互时异常（注册心跳超时，心跳间隔为 2 分钟）
EXCEPTION_AUDIOEXCHANGE	0x8001	语音对讲异常
EXCEPTION_ALARM	0x8002	报警异常
EXCEPTION_PREVIEW	0x8003	网络预览异常
EXCEPTION_SERIAL	0x8004	透明通道异常
EXCEPTION_RECONNECT	0x8005	预览时重连
EXCEPTION_ALARMRECONNECT	0x8006	报警时重连
EXCEPTION_SERIALRECONNECT	0x8007	透明通道重连
SERIAL_RECONNECTSUCCESS	0x8008	透明通道重连成功
EXCEPTION_PLAYBACK	0x8010	回放异常
EXCEPTION_DISKFMT	0x8011	硬盘格式化
PREVIEW_RECONNECTSUCCESS	0x8015	预览时重连成功
ALARM_RECONNECTSUCCESS	0x8016	报警时重连成功
RESUME_EXCHANGE	0x8017	用户交互恢复
NETWORK_FLOWTEST_EXCEPTION	0x8018	网络流量检测异常

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：** Windows 下该函数的 hWnd 和 cbExceptionCallBack 不能同时为 NULL，Linux 下 cbExceptionCallBack 不能设置为 NULL，否则将接收不到异常消息。

如果此结构是以回调方式反馈异常消息，那么应用程序中的异常回调函数实现如下，该函数中的参数 dwType 表示异常消息类型（见上表）；IHandle 表示发生异常的相应类型的句柄。

#### 示例代码：

```
//注册接收异常消息的回调函数
NET_DVR_SetExceptionCallBack_V30(WM_NULL, NULL, g_ExceptionCallBack, NULL);

//接收异常消息的回调函数的外部实现
void CALLBACK g_ExceptionCallBack(DWORD dwType, LONG IUserID, LONG IHandle, void *pUser)
{
    char tempbuf[256];
    ZeroMemory(tempbuf,256);
    switch(dwType)
    {
        case EXCEPTION_AUDIOEXCHANGE: //语音对讲时网络异常
            sprintf(tempbuf,"语音对讲时网络异常!!!");
            TRACE("%s",tempbuf);
    }
}
```

```

        //TODO: 关闭语音对讲
        break;

    case EXCEPTION_ALARM:                //报警上传时网络异常
        sprintf(tempbuf,"报警上传时网络异常!!!");
        TRACE("%s",tempbuf);
        //TODO: 关闭报警上传
        break;

    case EXCEPTION_PREVIEW:              //网络预览时异常
        sprintf(tempbuf,"网络预览时网络异常!!!");
        TRACE("%s",tempbuf);
        //TODO: 关闭网络预览
        break;

    case EXCEPTION_RECONNECT:             //预览时重连
        break;

    default:
        break;

}
}

```

[返回目录](#)

## 获取错误信息

### 6.2.13 返回最后操作的错误码 **NET\_DVR\_GetLastError**

函 数: DWORD NET\_DVR\_GetLastError()

参 数:

返回值: 返回最后操作的错误码。详见[错误码宏定义](#)

说 明: 返回值为错误码。错误码主要分为网络通讯库错误码、RTSP 通讯库错误码和软硬解库错误码。

[返回目录](#)

### 6.2.14 返回最后操作的错误码信息 **NET\_DVR\_GetErrorMsg**

函 数: char\* NET\_DVR\_GetErrorMsg(LONG \*pErrorNo)

参 数: [out] pErrorNo 错误码数值的指针

返回值: 返回值为错误码信息的指针。错误码主要分为网络通讯库错误码、RTSP 通讯库错误码和软硬解库错误码。详见[错误码宏定义](#)

说 明:

[返回目录](#)

## 6.3 用户注册

### 6.3.1 激活设备 **NET\_DVR\_ActivateDevice**

函 数： BOOL NET\_DVR\_ActivateDevice(char\* sDVRIP, WORD wDVRPort, LPNET\_DVR\_ACTIVATECFG lpActivateCfg)

参 数： [in]sDVRIP                      设备 IP 地址  
          [in]wDVRPort                设备端口，默认：8000  
          [in]lpActivateCfg           激活参数，包括激活使用的初始密码，详见结构体：  
    [NET\\_DVR\\_ACTIVATECFG](#)

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明： 出厂设备需要先激活，然后再使用激活使用的初始密码登录设备。

[返回目录](#)

### 6.3.2 通过解析服务器，获取设备的动态 IP 地址和端口号

#### **NET\_DVR\_GetDVRIPByResolveSvr\_EX**

函 数： BOOL NET\_DVR\_GetDVRIPByResolveSvr\_EX (char\* sServerIP, WORD wServerPort, BYTE\* sDVRName, WORD wDVRNameLen, BYTE\* sDVRSerialNumber, WORD wDVRSerialLen, char\* sGetIP, DWORD\* dwPort)

参 数： [in]sServerIP                      解析服务器的 IP 地址  
          [in]wServerPort                解析服务器的端口号，IP Server 解析服务器端口号为 7071，  
    HiDDNS 服务器的端口号为 80  
          [in]sDVRName                    设备名称  
          [in]wDVRNameLen                设备名称的长度  
          [in]sDVRSerialNumber          设备的序列号  
          [in]wDVRSerialLen              设备序列号的长度  
          [out]sGetIP                      获取到的设备 IP 地址指针  
          [out]dwPort                      获取到的设备端口号指针

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明： 该接口中的设备名称和设备序列号不能同时为空。通过设备域名或者序列号解析出设备当前 IP 地址和端口，然后调用 [NET\\_DVR\\_Login\\_V40](#) 登录设备。支持的解析服务器有 IPServer 和 hiDDNS。

[返回目录](#)

### 6.3.3 用户注册设备 **NET\_DVR\_Login\_V40**

函 数： LONG NET\_DVR\_Login\_V40(LPNET\_DVR\_USER\_LOGIN\_INFO pLoginInfo, LPNET\_DVR\_DEVICEINFO\_V40 lpDeviceInfo)

参 数： [in]pLoginInfo                      登录参数，包括设备地址、登录用户、密码等，详见结构体：

[NET\\_DVR\\_USER\\_LOGIN\\_INFO](#)

[out]IpDeviceInfo 设备信息(同步登录即 pLoginInfo 中 bUseAsynLogin 为 0 时有效) ,  
详见结构体: [NET\\_DVR\\_DEVICEINFO\\_V40](#)

返回值: 异步登录的状态、用户 ID 和设备信息通过 NET\_DVR\_USER\_LOGIN\_INFO 结构体中设置的回调函数(fLoginResultCallBack)返回。对于同步登录, 接口返回-1 表示登录失败, 其他值表示返回的用户 ID 值。用户 ID 具有唯一性, 后续对设备的操作都需要通过此 ID 实现。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说明:**

- pLoginInfo 中 bUseAsynLogin 为 0 时登录为同步模式, 接口返回成功即表示登录成功; pLoginInfo 中 bUseAsynLogin 为 1 时登录为异步模式, 登录是否成功在输入参数设置的回调函数中返回。
- 设备同时最多允许 128 个用户注册。
- SDK 支持 2048 个注册, 返回 UserID 的取值范围为 0~2047。

[返回目录](#)

### 6.3.4 用户注销 **NET\_DVR\_Logout**

函数: BOOL NET\_DVR\_Logout(LONG lUserID)

参数: [in] lUserID 用户 ID 号, NET\_DVR\_Login\_V40 的返回值

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说明:** 建议使用此接口实现注销功能。

[返回目录](#)

## 6.4 获取设备能力集

### 6.4.1 获取设备能力集 **NET\_DVR\_GetDeviceAbility**

函数: BOOL NET\_DVR\_GetDeviceAbility(LONG lUserID, DWORD dwAbilityType, char\* pInBuf, DWORD dwInLength, char\* pOutBuf, DWORD dwOutLength)

参数:

- [in] lUserID NET\_DVR\_Login\_V40 的返回值
- [in] dwAbilityType 能力类型, 具体定义见表 6.4
- [in] pInBuf 输入缓冲区指针, 详见表 6.5
- [in] dwInLength 输入缓冲区的长度
- [out] pOutBuf 输出缓冲区指针, 详见表 6.5
- [in] dwOutLength 接收数据的缓冲区的长度

表 6.4 设备能力集类型

宏定义	宏定义值	含义
ACS_ABILITY	0x801	门禁能力集
DEVICE_SOFTWARE_ABILITY	0x001	设备软硬件能力
DEVICE_USER_ABILITY	0x00c	设备用户管理参数能力

DEVICE_NETWORK_ABILITY	0x002	设备无线网络能力
DEVICE_NETAPP_ABILITY	0x00d	设备网络应用参数能力
DEVICE_SERIAL_ABILITY	0x010	设备 RS232 和 RS485 串口能力
DEVICE_ABILITY_INFO	0x011	设备通用能力类型，具体能力根据发送的能力节点来区分

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说明：** 接口中 pInBuf 参数的具体定义格式按照不同的设备规定有所不同，需要输入参数和输出参数的格式定义如表 6.5 所示。

表 6.5 设备能力集描述

能力类型宏定义	能力类型说明	pInBuf	pOutBuf
ACS_ABILITY	获取门禁能力集	门禁能力集获取输入的 XML 描述	门禁能力集 XML 描述(AcsAbility)
DEVICE_SOFTHARDWARE_ABILITY	获取设备软硬件能力	无	设备软硬件能力 XML 描述(BasicCapability)
DEVICE_USER_ABILITY	获取设备用户管理参数能力	用户管理参数能力获取输入 XML 描述	设备用户管理参数能力 XML 描述(UserAbility)
DEVICE_NETWORK_ABILITY	获取无线设备网络能力	无	设备无线网络能力 XML 描述(NetworkSetting)
DEVICE_NETAPP_ABILITY	获取网络应用参数能力	网络应用参数能力输入 XML 描述	设备网络应用参数能力 XML 描述(NetAppAbility)
DEVICE_SERIAL_ABILITY	获取 RS232 和 RS485 串口能力	串口能力获取输入 XML 描述	设备串口能力 XML 描述(SerialAbility)
DEVICE_ABILITY_INFO	获取报警事件处理能力	获取报警事件处理能力集 XML 描述	报警事件处理能力 XML 描述(EventAbility)
	获取安全认证配置能力集	获取安全认证配置能力集 XML 描述	安全认证配置能力 XML 描述(SecurityAbility)
	获取协议接入能力集	获取协议接入能力集的 XML 描述	设备协议接入能力 XML 描述(AccessProtocolAbility)
	获取报警主机 XML 能力集	报警主机 XML 能力集获取输入描述	报警主机能力集 XML 描述(AlarmHostAbility)

注：能力集 XML 描述详细内容请参见《设备网络 SDK 使用手册.chm》。

[返回目录](#)

## 6.4.2 获取门禁总能力集 NET\_DVR\_STDXMLConfig

函数： BOOL NET\_DVR\_STDXMLConfig(LONG IUserID, NET\_DVR\_XML\_CONFIG\_INPUT\* lpInputParam, NET\_DVR\_XML\_CONFIG\_OUTPUT\* lpOutputParam)

参数： [in] IUserID                      NET\_DVR\_Login\_V40 登录接口的返回值

[in] IpInputParam 输入参数，详见 [NET\\_DVR\\_XML\\_CONFIG\\_INPUT](#)

[out] IpOutputParam 输出参数，详见 [NET\\_DVR\\_XML\\_CONFIG\\_OUTPUT](#)

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：**

#### Remarks

该接口为 ISAPI 协议透传接口，不同功能对应输入参数和输出参数具体如下表所示。其中所涉及各个 XML 格式描述的详细内容请参见《设备网络 SDK 使用手册.chm》门禁主机部分对应接口。

功能描述	IpInputParam->IpRequestUrl	IpInputParam->IpInBuffer	IpOutputParam->IpOutBuffer
获取门禁总能力集	GET /ISAPI/AccessControl/capabilities	NULL	AccessControl (门禁总能力集 XML 格式)

[返回目录](#)

## 6.5 布防、撤防

### 设置报警等信息上传的回调函数

#### 6.5.1 注册报警回调函数 **NET\_DVR\_SetDVRMessageCallBack\_V31**

函 数： BOOL NET\_DVR\_SetDVRMessageCallBack\_V31(MSGCallBack\_V31 fMessageCallBack, void\* pUser)

参 数： [in]fMessageCallBack 报警信息回调函数

[in]pUser 用户数据

```
typedef BOOL(CALLBACK * MSGCallBack_V31)(LONG ICommand, NET_DVR_ALARMER *pAlarmer, char *pAlarmInfo, DWORD dwBufLen, void *pUser)
```

[out]ICommand 上传的消息类型，详见表 6.

[out]pAlarmer 报警设备信息，详见 [NET\\_DVR\\_ALARMER](#)

[out]pAlarmInfo 报警信息，详见表 6.

[out]dwBufLen 报警信息缓存大小

[out]pUser 用户数据

表 6.6 门禁报警信息类型

宏定义	宏定义值	含义
COMM_ALARM_ACS	0x5002	门禁主机报警信息
COMM_ID_INFO_ALARM	0x5200	门禁身份证刷卡信息
COMM_PASSNUM_INFO_ALARM	0x5201	门禁通行人数信息

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：** 该接口中回调函数的第一个参数（ICommand）和第三个参数（pAlarmInfo）是密切关联的，其关系如表 6.所示。



表 6.7 报警信息结构

消息类型 (ICommand)	上传内容	pAlarmInfo 对应的结构体
COMM_ALARM_ACS	门禁主机报警信息	<a href="#">NET_DVR_ACS_ALARM_INFO</a>
COMM_ID_INFO_ALARM	门禁身份证刷卡信息	<a href="#">NET_DVR_ID_CARD_INFO_ALARM</a>
COMM_PASSNUM_INFO_ALARM	门禁通行人数信息	<a href="#">NET_DVR_PASSNUM_INFO_ALARM</a>

[返回目录](#)

## 布防撤防

### 6.5.2 建立报警上传通道，获取报警等信息 **NET\_DVR\_SetupAlarmChan\_V41**

**函 数：** LONG NET\_DVR\_SetupAlarmChan\_V41(LONG IUserID, LPNET\_DVR\_SETUPALARM\_PARAM lpSetupParam)

**参 数：** [in]IUserID                      NET\_DVR\_Login\_V40 的返回值  
[in]lpSetupParam                      报警布防参数，请参见结构体：[NET\\_DVR\\_SETUPALARM\\_PARAM](#)

**返回值：** -1 表示失败，其他值作为 NET\_DVR\_CloseAlarmChan\_V30 函数的句柄参数。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：** 启动布防前，需要调用注册回调函数的接口（如 [NET\\_DVR\\_SetDVRMessageCallBack\\_V31](#)）才能获取到上传的报警等信息。

[返回目录](#)

### 6.5.3 撤销报警上传通道 **NET\_DVR\_CloseAlarmChan\_V30**

**函 数：** BOOL NET\_DVR\_CloseAlarmChan\_V30(LONG IAlarmHandle)

**参 数：** [in]IAlarmHandle                      NET\_DVR\_SetupAlarmChan\_V41 的返回值

**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：**

[返回目录](#)

## 6.6 远程参数配置

### 通用参数配置

#### 6.6.1 获取设备的配置信息 **NET\_DVR\_GetDVRConfig**

**函 数：** BOOL NET\_DVR\_GetDVRConfig(LONG IUserID, DWORD dwCommand, LONG IChannel, LPVOID lpOutBuffer, DWORD dwOutBufferSize, LPDWORD lpBytesReturned)

**参 数：** [in] IUserID                      用户 ID 号，NET\_DVR\_Login\_V40 的返回值  
[in] dwCommand                      设备配置命令，详见表 6.  
[in] IChannel                      通道号，如果命令不需要通道号，该参数无效，置为 0xFFFFFFFF 即可  
[out] lpOutBuffer                      接收数据的缓冲指针，详见表 6.  
[in] dwOutBufferSize                      接收数据的缓冲长度(以字节为单位)，不能为 0  
[out] lpBytesReturned                      实际收到的数据长度指针，不能为 NULL

**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通



过错误码判断出错原因。

**说明：** 不同的获取功能对应不同的结构体和命令号，如表 6.所示。

表 6.8 通用参数获取

dwCommand 宏定义	dwCommand 含义	通道号	lpOutBuffer 对应结构体	宏定义值
NET_DVR_GET_DEVICECFG_V40	获取设备参数(扩展)	无效	<a href="#">NET_DVR_DEVICECFG_V40</a>	1100
NET_DVR_GET_TIMECFG	获取时间参数	无效	<a href="#">NET_DVR_TIME</a>	118
NET_DVR_GET_EXCEPTIONCFG_V40	获取异常参数	组号，从 0 开始，每组 32 种异常	<a href="#">NET_DVR_EXCEPTION_V40</a>	6177
NET_DVR_GET_ZONEANDDST	获取时区和夏时制参数	无效	<a href="#">NET_DVR_ZONEANDDST</a>	128
NET_DVR_GET_NETCFG_V30	获取网络参数	无效	<a href="#">NET_DVR_NETCFG_V30</a>	1000
NET_DVR_GET_NETCFG_V50	获取网络参数(V50 双监听)	无效	<a href="#">NET_DVR_NETCFG_V50</a>	1015
NET_DVR_GET_NTPCFG	获取网络应用参数(NTP)	无效	<a href="#">NET_DVR_NTPPARA</a>	224
NET_DVR_GET_SECURITY_CFG	获取安全认证配置	无效	<a href="#">NET_DVR_SECURITY_CFG</a>	147
NET_DVR_GET_SMSRELATIVEPARA	获取短信相关参数	无效	<a href="#">NET_DVR_SMSRELATIVEPARAM</a>	1173
NET_DVR_GET_ALARMHOST_NETCFG	获取上传中心网络参数配置	无效	<a href="#">NET_DVR_ALARMHOST_NETCFG</a>	2007
NET_DVR_GET_ALARMHOST_WIRELESS_NETWORK_CFG	获取无线网络参数配置	无效	<a href="#">NET_DVR_ALARMHOST_WIRELESS_NETWORK_CFG</a>	2005
NET_DVR_GET_WIFI_CFG	获取 IP 监控设备无线参数(指纹门禁一体机)	无效	<a href="#">NET_DVR_WIFI_CFG</a>	307
NET_DVR_GET_NETCFG_MULTI	获取多网卡配置参数	无效	<a href="#">NET_DVR_NETCFG_MULTI</a>	1161

[返回目录](#)

## 6.6.2 设置设备的配置信息 **NET\_DVR\_SetDVRConfig**

**函数：** BOOL NET\_DVR\_SetDVRConfig(LONG IUserID, DWORD dwCommand, LONG IChannel, LPVOID lpInBuffer, DWORD dwInBufferSize)

**参数：**

- [in] IUserID 用户 ID 号，NET\_DVR\_Login\_V40 的返回值
- [in] dwCommand 设备配置命令，详见表 6.9
- [in] IChannel 通道号，如果命令不需要通道号，该参数无效，置为 0xFFFFFFFF 即可
- [in] lpInBuffer 输入数据的缓冲指针，详见表 6.9
- [in] dwInBufferSize 输入数据的缓冲长度(以字节为单位)

**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说明：** 不同的设置功能对应不同的结构体和命令号，如表 6.9 所示。

表 6.9 通用参数设置

dwCommand 宏定义	dwCommand 含义	通道号	lpInBuffer 对应结构体	宏定义值
NET_DVR_SET_DEVICECFG_V40	设置设备参数(扩展)	无效	<a href="#">NET_DVR_DEVICECFG_V40</a>	1101

NET_DVR_SET_TIMECFG	设置时间参数（校时）	无效	<a href="#">NET_DVR_TIME</a>	119
NET_DVR_SET_EXCEPTIONCFG_V40	设置异常参数	组号，从 0 开始，每组 32 种异常	<a href="#">NET_DVR_EXCEPTION_V40</a>	6178
NET_DVR_SET_ZONEANDDST	设置时区和夏时制参数	无效	<a href="#">NET_DVR_ZONEANDDST</a>	129
NET_DVR_SET_NETCFG_V30	设置网络参数	无效	<a href="#">NET_DVR_NETCFG_V30</a>	1001
NET_DVR_SET_NETCFG_V50	设置网络参数(V50 双监听)	无效	<a href="#">NET_DVR_NETCFG_V50</a>	1016
NET_DVR_SET_NTPCFG	设置网络应用参数(NTP)	无效	<a href="#">NET_DVR_NTPPARA</a>	225
NET_DVR_SET_SECURITY_CFG	设置安全认证配置	无效	<a href="#">NET_DVR_SECURITY_CFG</a>	148
NET_DVR_SET_SMSRELATIVEPARA	设置短信相关参数	无效	<a href="#">NET_DVR_SMSRELATIVEPARAM</a>	1174
NET_DVR_SET_ALARMHOST_NETCFG	设置上传中心网络参数配置	无效	<a href="#">NET_DVR_ALARMHOST_NETCFG</a>	2008
NET_DVR_SET_ALARMHOST_WIRELESS_NETWORK_CFG	设置无线网络参数	无效	<a href="#">NET_DVR_ALARMHOST_WIRELESS_NETWORK_CFG</a>	2006
NET_DVR_SET_WIFI_CFG	设置 IP 监控设备无线参数 (指纹门禁一体机)	无效	<a href="#">NET_DVR_WIFI_CFG</a>	306
NET_DVR_SET_NETCFG_MULTI	设置多网卡配置参数	无效	<a href="#">NET_DVR_NETCFG_MULTI</a>	1162

[返回目录](#)

## 门禁相关参数配置

### 6.6.3 获取设备的配置信息 **NET\_DVR\_GetDVRConfig**

函 数： BOOL NET\_DVR\_GetDVRConfig(LONG IUserID, DWORD dwCommand, LONG IChannel, LPVOID lpOutBuffer, DWORD dwOutBufferSize, LPDWORD lpBytesReturned)

参 数： [in] IUserID 用户 ID 号，NET\_DVR\_Login\_V40 的返回值  
[in] dwCommand 设备配置命令，详见表 6.6  
[in] IChannel 通道号，如果命令不需要通道号，该参数无效，置为 0xFFFFFFFF 即可  
[out] lpOutBuffer 接收数据的缓冲指针，详见表 6.6  
[in] dwOutBufferSize 接收数据的缓冲长度(以字节为单位)，不能为 0  
[out] lpBytesReturned 实际收到的数据长度指针，不能为 NULL

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明： 不同的获取功能对应不同的结构体和命令号，如表 6.6 所示。

表 6.6 门禁相关参数获取

dwCommand 宏定义	dwCommand 含义	通道号	lpOutBuffer 对应结构体	宏定义值
NET_DVR_GET_WEEK_PLAN_CFG	获取门状态周计划参数	周计划编号，从 1 开始	<a href="#">NET_DVR_WEEK_PLAN_CFG</a>	2100
NET_DVR_GET_VERIFY_WEEK_PLAN	获取读卡器验证方式周	周计划编号，从 1	<a href="#">NET_DVR_WEEK_PLAN_CFG</a>	2124

	计划参数	开始		
NET_DVR_GET_CARD_RIGHT_WEEK_PLAN	获取卡权限周计划参数	周计划编号, 从 1 开始	<a href="#">NET_DVR_WEEK_PLAN_CFG</a>	2126
NET_DVR_GET_DOOR_STATUS_HOLIDAY_PLAN	获取门状态假日计划参数	假日计划编号, 从 1 开始	<a href="#">NET_DVR_HOLIDAY_PLAN_CFG</a>	2102
NET_DVR_GET_VERIFY_HOLIDAY_PLAN	获取读卡器验证方式假日计划参数	假日计划编号, 从 1 开始	<a href="#">NET_DVR_HOLIDAY_PLAN_CFG</a>	2128
NET_DVR_GET_CARD_RIGHT_HOLIDAY_PLAN	获取卡权限假日计划参数	假日计划编号, 从 1 开始	<a href="#">NET_DVR_HOLIDAY_PLAN_CFG</a>	2130
NET_DVR_GET_DOOR_STATUS_HOLIDAY_GROUP	获取门状态假日组参数	假日组编号, 从 1 开始	<a href="#">NET_DVR_HOLIDAY_GROUP_CFG</a>	2104
NET_DVR_GET_VERIFY_HOLIDAY_GROUP	获取读卡器验证方式假日组参数	假日组编号, 从 1 开始	<a href="#">NET_DVR_HOLIDAY_GROUP_CFG</a>	2132
NET_DVR_GET_CARD_RIGHT_HOLIDAY_GROUP	获取卡权限假日组参数	假日组编号, 从 1 开始	<a href="#">NET_DVR_HOLIDAY_GROUP_CFG</a>	2134
NET_DVR_GET_DOOR_STATUS_PLAN_TEMPLATE	获取门状态计划模板参数	计划模板编号, 从 1 开始	<a href="#">NET_DVR_PLAN_TEMPLATE</a>	2106
NET_DVR_GET_VERIFY_PLAN_TEMPLATE	获取读卡器验证方式计划模板参数	计划模板编号, 从 1 开始	<a href="#">NET_DVR_PLAN_TEMPLATE</a>	2136
NET_DVR_GET_CARD_RIGHT_PLAN_TEMPLATE	获取卡权限计划模板参数	计划模板编号, 从 1 开始	<a href="#">NET_DVR_PLAN_TEMPLATE</a>	2138
NET_DVR_GET_DOOR_CFG	获取门参数	门编号, 从 1 开始	<a href="#">NET_DVR_DOOR_CFG</a>	2108
NET_DVR_GET_DOOR_STATUS_PLAN	获取门状态计划参数	门编号, 从 1 开始	<a href="#">NET_DVR_DOOR_STATUS_PLAN</a>	2110
NET_DVR_GET_CARD_READER_PLAN	获取读卡器验证计划参数	读卡器编号, 从 1 开始	<a href="#">NET_DVR_CARD_READER_PLAN</a>	2142
NET_DVR_GET_GROUP_CFG	获取群组参数	群组编号, 从 1 开始	<a href="#">NET_DVR_GROUP_CFG</a>	2112
NET_DVR_GET_MULTI_CARD_CFG	获取多重卡参数	门编号, 从 1 开始	<a href="#">NET_DVR_MULTI_CARD_CFG</a>	2114
NET_DVR_GET_SNEAK_CFG	获取反潜回参数	无效	<a href="#">NET_DVR_ANTI_SNEAK_CFG</a>	2119
NET_DVR_GET_CARD_READER_ANTI_SNEAK_CFG	获取读卡器反潜回参数	无效	<a href="#">NET_DVR_CARD_READER_ANTI_SNEAK_CFG</a>	2146
NET_DVR_GET_MULTI_DOOR_INTERLOCK_CFG	获取多门互锁参数	无效	<a href="#">NET_DVR_MULTI_DOOR_INTERLOCK_CFG</a>	2121
NET_DVR_GET_CARD_READER_CFG	获取读卡器参数	读卡器编号	<a href="#">NET_DVR_CARD_READER_CFG</a>	2140
NET_DVR_GET_CARD_READER_CFG_V50	获取读卡器参数(V50)	读卡器编号	<a href="#">NET_DVR_CARD_READER_CFG_V50</a>	2505
NET_DVR_GET_ACS_WORK_STATUS	获取门禁主机工作状态	无效	<a href="#">NET_DVR_ACS_WORK_STATUS</a>	2123
NET_DVR_GET_ACS_WORK_STATUS_V50	获取门禁主机工作状态(V50)	无效	<a href="#">NET_DVR_ACS_WORK_STATUS_V50</a>	2180

NET_DVR_GET_CASE_SENSOR_CFG	获取事件报警输入参数	事件报警输入编号，从 1 开始	<a href="#">NET_DVR_CASE_SENSOR_CFG</a>	2144
NET_DVR_GET_PHONE_DOOR_RIGHT_CFG	获取手机关联门权限参数	手机白名单编号，从 1 开始	<a href="#">NET_DVR_PHONE_DOOR_RIGHT_CFG</a>	2148
NET_DVR_GET_EVENT_CARD_LINKAGE_CFG	获取事件/卡号联动配置参数	联动事件编号，从 1 开始	<a href="#">NET_DVR_EVENT_CARD_LINKAGE_CFG</a>	2153
NET_DVR_GET_ACS_CFG	获取门禁主机参数	无效	<a href="#">NET_DVR_ACS_CFG</a>	2159
NET_DVR_GET_ACS_EXTERNAL_DEV_CFG	获取门禁主机串口外设参数	RS485 串口号，从 1 开始	<a href="#">NET_DVR_ACS_EXTERNAL_DEV_CFG</a>	2165
NET_DVR_GET_PERSONNEL_CHANNEL_CFG	获取人员通道参数	人员通道号，从 1 开始	<a href="#">NET_DVR_PERSONNEL_CHANNEL_CFG</a>	2167
NET_DVR_GET_PERSON_STATISTICS_CFG	获取人数统计参数	无效	<a href="#">NET_DVR_PERSON_STATISTICS_CFG</a>	2170
NET_DVR_GET_ACS_SCREEN_DISPLAY_CFG	获取屏幕字符串显示参数	屏幕编号，从 0 开始	<a href="#">NET_DVR_ACS_SCREEN_DISPLAY_CFG</a>	2172
NET_DVR_GET_GATE_TIME_CFG	获取人员通道闸门时间参数	无效	<a href="#">NET_DVR_GATE_TIME_CFG</a>	2174
NET_VCA_GET_FACEDETECT_RULECFG_V41	获取人脸检测规则	通道号	<a href="#">NET_DVR_FACEDETECT_RULECFG_V41</a>	5017

[返回目录](#)

## 6.6.4 设置设备的配置信息 **NET\_DVR\_SetDVRConfig**

函 数： BOOL NET\_DVR\_SetDVRConfig(LONG lUserID, DWORD dwCommand, LONG lChannel, LPVOID lpInBuffer, DWORD dwInBufferSize)

参 数： [in] lUserID 用户 ID 号，NET\_DVR\_Login\_V40 的返回值  
[in] dwCommand 设备配置命令，详见表 6.7  
[in] lChannel 通道号，如果命令不需要通道号，该参数无效，置为 0xFFFFFFFF 即可  
[in] lpInBuffer 输入数据的缓冲指针，详见表 6.7  
[in] dwInBufferSize 输入数据的缓冲长度(以字节为单位)

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明： 不同的获取功能对应不同的结构体和命令号，如表 6.7 所示。

表 6.7 门禁相关参数设置

dwCommand 宏定义	dwCommand 含义	通道号	lpOutBuffer 对应结构体	宏定义值
NET_DVR_SET_WEEK_PLAN_CFG	设置门状态周计划参数	周计划编号，从 1 开始	<a href="#">NET_DVR_WEEK_PLAN_CFG</a>	2101
NET_DVR_SET_VERIFY_WEEK_PLAN	设置读卡器验证方式周计划参数	周计划编号，从 1 开始	<a href="#">NET_DVR_WEEK_PLAN_CFG</a>	2125
NET_DVR_SET_CARD_RIGHT_WEEK_PLAN	设置卡权限周计划参数	周计划编号，从 1 开始	<a href="#">NET_DVR_WEEK_PLAN_CFG</a>	2127

NET_DVR_SET_DOOR_STATUS_HOLIDAY_PLAN	设置门状态假日计划参数	假日计划编号, 从 1 开始	<a href="#">NET_DVR_HOLIDAY_PLAN_CFG</a>	2103
NET_DVR_SET_VERIFY_HOLIDAY_PLAN	设置读卡器验证方式假日计划参数	假日计划编号, 从 1 开始	<a href="#">NET_DVR_HOLIDAY_PLAN_CFG</a>	2129
NET_DVR_SET_CARD_RIGHT_HOLIDAY_PLAN	设置卡权限假日计划参数	假日计划编号, 从 1 开始	<a href="#">NET_DVR_HOLIDAY_PLAN_CFG</a>	2131
NET_DVR_SET_DOOR_STATUS_HOLIDAY_GROUP	设置门状态假日组参数	假日组编号, 从 1 开始	<a href="#">NET_DVR_HOLIDAY_GROUP_CFG</a>	2105
NET_DVR_SET_VERIFY_HOLIDAY_GROUP	设置读卡器验证方式假日组参数	假日组编号, 从 1 开始	<a href="#">NET_DVR_HOLIDAY_GROUP_CFG</a>	2133
NET_DVR_SET_CARD_RIGHT_HOLIDAY_GROUP	设置卡权限假日组参数	假日组编号, 从 1 开始	<a href="#">NET_DVR_HOLIDAY_GROUP_CFG</a>	2135
NET_DVR_SET_DOOR_STATUS_PLAN_TEMPLATE	设置门状态计划模板参数	计划模板编号, 从 1 开始	<a href="#">NET_DVR_PLAN_TEMPLATE</a>	2107
NET_DVR_SET_VERIFY_PLAN_TEMPLATE	设置读卡器验证方式计划模板参数	计划模板编号, 从 1 开始	<a href="#">NET_DVR_PLAN_TEMPLATE</a>	2137
NET_DVR_SET_CARD_RIGHT_PLAN_TEMPLATE	设置卡权限计划模板参数	计划模板编号, 从 1 开始	<a href="#">NET_DVR_PLAN_TEMPLATE</a>	2139
NET_DVR_SET_DOOR_CFG	设置门参数	门编号, 从 1 开始	<a href="#">NET_DVR_DOOR_CFG</a>	2109
NET_DVR_SET_DOOR_STATUS_PLAN	设置门状态计划参数	门编号, 从 1 开始	<a href="#">NET_DVR_DOOR_STATUS_PLAN</a>	2111
NET_DVR_SET_CARD_READER_PLAN	设置读卡器验证计划参数	读卡器编号, 从 1 开始	<a href="#">NET_DVR_CARD_READER_PLAN</a>	2143
NET_DVR_SET_GROUP_CFG	设置群组参数	群组编号, 从 1 开始	<a href="#">NET_DVR_GROUP_CFG</a>	2113
NET_DVR_SET_MULTI_CARD_CFG	设置多重卡参数	门编号, 从 1 开始	<a href="#">NET_DVR_MULTI_CARD_CFG</a>	2115
NET_DVR_SET_SNEAK_CFG	设置反潜回参数	无效	<a href="#">NET_DVR_ANTI_SNEAK_CFG</a>	2120
NET_DVR_SET_CARD_READER_ANTI_SNEAK_CFG	设置读卡器反潜回参数	无效	<a href="#">NET_DVR_CARD_READER_ANTI_SNEAK_CFG</a>	2147
NET_DVR_SET_MULTI_DOOR_INTERLOCK_CFG	设置多门互锁参数	无效	<a href="#">NET_DVR_MULTI_DOOR_INTERLOCK_CFG</a>	2122
NET_DVR_SET_CARD_READER_CFG	设置读卡器参数	读卡器编号	<a href="#">NET_DVR_CARD_READER_CFG</a>	2141
NET_DVR_SET_CARD_READER_CFG_V50	设置读卡器参数(V50)	读卡器编号	<a href="#">NET_DVR_CARD_READER_CFG_V50</a>	2506
NET_DVR_SET_CASE_SENSOR_CFG	设置事件报警输入参数	事件报警输入编号, 从 1 开始	<a href="#">NET_DVR_CASE_SENSOR_CFG</a>	2145
NET_DVR_SET_PHONE_DOOR_RIGHT_CFG	设置手机关联门权限参数	手机白名单编号, 从 1 开始	<a href="#">NET_DVR_PHONE_DOOR_RIGHT_CFG</a>	2149
NET_DVR_SET_EVENT_CARD_LINKAGE_CFG	设置事件/卡号联动配置参数	联动事件编号, 从 1 开始	<a href="#">NET_DVR_EVENT_CARD_LINKAGE_CFG</a>	2154

NET_DVR_SET_ACS_CFG	设置门禁主机参数	无效	<a href="#">NET_DVR_ACS_CFG</a>	2160
NET_DVR_SET_ACS_EXTERNAL_DEV_CFG	设置门禁主机串口外设参数	RS485 串口号, 从 1 开始	<a href="#">NET_DVR_ACS_EXTERNAL_DEV_CFG</a>	2166
NET_DVR_SET_PERSONNEL_CHANNEL_CFG	设置人员通道参数	人员通道号, 从 1 开始	<a href="#">NET_DVR_PERSONNEL_CHANNEL_CFG</a>	2168
NET_DVR_SET_PLATFORM_VERIFY_CFG	下发平台认证结果	无效	<a href="#">NET_DVR_PLATFORM_VERIFY_CFG</a>	2169
NET_DVR_SET_PERSON_STATISTICS_CFG	设置人数统计参数	无效	<a href="#">NET_DVR_PERSON_STATISTICS_CFG</a>	2171
NET_DVR_SET_ACS_SCREEN_DISPLAY_CFG	设置屏幕字符串显示参数	屏幕编号, 从 0 开始	<a href="#">NET_DVR_ACS_SCREEN_DISPLAY_CFG</a>	2173
NET_DVR_SET_GATE_TIME_CFG	设置人员通道闸门时间参数	无效	<a href="#">NET_DVR_GATE_TIME_CFG</a>	2175
NET_VCA_SET_FACEDETECT_RULECFG_V41	设置人脸检测规则	通道号	<a href="#">NET_DVR_FACEDETECT_RULECFG_V41</a>	5018

[返回目录](#)

### 6.6.5 批量获取配置信息 **NET\_DVR\_GetDeviceConfig**

- 函 数:** BOOL NET\_DVR\_GetDeviceConfig(LONG IUserID, DWORD dwCommand, DWORD dwCount, LPVOID lpInBuffer, DWORD dwInBufferSize, LPVOID lpStatusList, LPVOID lpOutBuffer, DWORD dwOutBufferSize)
- 参 数:**
- [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V40 的返回值
  - [in] dwCommand 设备配置命令, 详见表 6.12
  - [in] dwCount 一次要获取配置参数的个数, 0 和 1 都表示 1 个信息, 2 表示 2 个信息, 最大 64 个
  - [in] lpInBuffer 配置条件缓冲区, 详见表 6.13
  - [in] dwInBufferSize 缓冲区长度
  - [out] lpStatusList 错误信息列表, 和要查询的配置一一对应, 例如 lpStatusList[2]就对应 lpInBuffer[2], 由用户分配内存, 每个错误信息为 4 个字节, 参数值: 0- 成功, 大于 0-失败
  - [out] lpOutBuffer 设备返回的参数内容 (详见表 6.13 和要查询的监控点一一对应。如果某个监控点对应的 lpStatusList 信息为大于 0 值, 对应 lpOutBuffer 的内容就是无效的)
  - [in] dwOutBufferSize 输出缓冲区大小
- 返回值:** TRUE 表示成功, 但不代表每一个配置都成功, 哪一个成功, 对应查看 lpStatusList[n]值; FALSE 表示全部失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。
- 说 明:** 该接口是带有发送数据的批量获取配置信息的通用接口。lpInBuffer 指定需要获取哪 dwCount 个信息, lpOutBuffer 保存获取得到的 dwCount 个相应配置信息。不同的获取功能对应不同的结构体和命令号, 如表 6.13 所示。

表 6.12 ITC 参数批量获取命令

dwCommand 宏定义	含义	宏定义值
NET_DVR_GET_ALARMHOST_REPORT_CENTER_V40	获取数据上传方式参数	2064
NET_DVR_GET_CARD_USERINFO_CFG	获取卡号关联用户信息参数 (dwCount 设为 1)	2163
NET_DVR_GET_EVENT_CARD_LINKAGE_CFG_V50	获取事件卡号联动配置参数(V50), dwCount 设为 1	2181
NET_DVR_GET_CARD_RIGHT_WEEK_PLAN_V50	获取卡权限周计划参数(V50), dwCount 设为 1	2304
NET_DVR_GET_CARD_RIGHT_HOLIDAY_PLAN_V50	获取卡权限假日计划参数(V50), dwCount 设为 1	2310
NET_DVR_GET_CARD_RIGHT_HOLIDAY_GROUP_V50	获取卡权限假日组参数(V50), dwCount 设为 1	2316
NET_DVR_GET_CARD_RIGHT_PLAN_TEMPLATE_V50	获取卡权限计划模板参数(V50), dwCount 设为 1	2322

表 6.13 批量获取 ITC 参数

dwCommand 宏定义	lpInBuffer 对应结构体	lpOutBuffer 对应结构体
NET_DVR_GET_ALARMHOST_REPORT_CENTER_V40	dwCount 个中心组序号, 每个序号为 4 个字节	dwCount 个 <a href="#">NET_DVR_ALARMHOST_REPORT_CENTER_CFG_V40</a>
NET_DVR_GET_CARD_USERINFO_CFG	dwCount 个 (dwCount 设为 1) <a href="#">NET_DVR_CARD_CFG_SEND_DATA</a>	<a href="#">NET_DVR_CARD_USER_INFO_CFG</a>
NET_DVR_GET_EVENT_CARD_LINKAGE_CFG_V50	dwCount 个 (dwCount 设为 1) <a href="#">NET_DVR_EVENT_CARD_LINKAGE_COND</a>	<a href="#">NET_DVR_EVENT_CARD_LINKAGE_CFG_V50</a>
NET_DVR_GET_CARD_RIGHT_WEEK_PLAN_V50	dwCount 个 (dwCount 设为 1) <a href="#">NET_DVR_WEEK_PLAN_COND</a>	<a href="#">NET_DVR_WEEK_PLAN_CFG</a>
NET_DVR_GET_CARD_RIGHT_HOLIDAY_PLAN_V50	dwCount 个 (dwCount 设为 1) <a href="#">NET_DVR_HOLIDAY_PLAN_COND</a>	<a href="#">NET_DVR_HOLIDAY_PLAN_CFG</a>
NET_DVR_GET_CARD_RIGHT_HOLIDAY_GROUP_V50	dwCount 个 (dwCount 设为 1) <a href="#">NET_DVR_HOLIDAY_GROUP_COND</a>	<a href="#">NET_DVR_HOLIDAY_GROUP_CFG</a>
NET_DVR_GET_CARD_RIGHT_PLAN_TEMPLATE_V50	dwCount 个 (dwCount 设为 1) <a href="#">NET_DVR_PLAN_TEMPLATE_COND</a>	<a href="#">NET_DVR_PLAN_TEMPLATE</a>

[返回目录](#)

### 6.6.6 批量设置配置信息 **NET\_DVR\_SetDeviceConfig**

函 数: BOOL NET\_DVR\_SetDeviceConfig(LONG lUserID, DWORD dwCommand, DWORD dwCount, LPVOID lpInBuffer, DWORD dwInBufferSize, LPVOID lpStatusList, LPVOID lpInParamBuffer, DWORD dwInParamBufferSize)

参 数: [in] lUserID                      用户 ID 号, NET\_DVR\_Login\_V40 的返回值  
[in] dwCommand                    设备配置命令, 详见表 6.14  
[in] dwCount                      一次要设置的配置参数个数, 0 和 1 都表示 1 个, 2 表示 2 个, 最



	大 64 个
[in] lpInBuffer	配置条件缓冲区，详见表 6.15
[in] dwInBufferSize	缓冲区长度
[out] lpStatusList	错误信息列表，和要设置的配置一一对应，例如 lpStatusList[2]就对应 lpInBuffer[2]，由用户分配内存，每个错误信息为 4 个字节，参数值：0- 成功，大于 0-失败
[in] lpInParamBuffer	需要设置给设备的参数内容（详见表 6.15），和 lpInBuffer 一一对应。如果某个配置对应的 lpStatusList 信息为大于 0 值，表示对应的 lpInBuffer 设置失败，为 0 则设置成功
[in] dwInParamBufferSize	设置内容缓冲区大小

返回值： TRUE 表示成功，但不代表每一个配置都成功，哪一个成功，对应查看 lpStatusList[n]值；FALSE 表示全部失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：** 该接口是带有发送数据的批量设置子设备配置信息的通用接口，lpInBuffer 指定需要设置哪 dwCount 个，lpInParamBuffer 是设置 dwCount 个配置的参数信息。不同的获取功能对应不同的结构体和命令号，如表 6.15 所示。

表 6.14 ITC 参数批量设置命令

dwCommand 宏定义	含义	宏定义值
NET_DVR_SET_ALARMHOST_REPORT_CENTER_V40	设置数据上传方式参数	2065
NET_DVR_SET_CARD_USERINFO_CFG	设置卡号关联用户信息参数（dwCount 设为 1）	2164
NET_DVR_SET_EVENT_CARD_LINKAGE_CFG_V50	设置事件卡号联动配置参数(V50)，dwCount 设为 1	2182
NET_DVR_SET_CARD_RIGHT_WEEK_PLAN_V50	设置卡权限周计划参数(V50)，dwCount 设为 1	2305
NET_DVR_SET_CARD_RIGHT_HOLIDAY_PLAN_V50	设置卡权限假日计划参数(V50)，dwCount 设为 1	2311
NET_DVR_SET_CARD_RIGHT_HOLIDAY_GROUP_V50	设置卡权限假日组参数(V50)，dwCount 设为 1	2317
NET_DVR_SET_CARD_RIGHT_PLAN_TEMPLATE_V50	设置卡权限计划模板参数(V50)，dwCount 设为 1	2323

表 6.15 批量设置 ITC 参数

dwCommand 宏定义	lpInBuffer 对应结构体	lpOutBuffer 对应结构体
NET_DVR_SET_ALARMHOST_REPORT_CENTER_V40	dwCount 个中心组序号，每个序号为 4 个字节	dwCount 个 <a href="#">NET_DVR_ALARMHOST_REPORT_CENTER_CFG_V40</a>
NET_DVR_SET_CARD_USERINFO_CFG	dwCount 个（dwCount 设为 1） <a href="#">NET_DVR_CARD_CFG_SEND_DATA</a>	<a href="#">NET_DVR_CARD_USER_INFO_CFG</a>
NET_DVR_SET_EVENT_CARD_LINKAGE_CFG_V50	dwCount 个（dwCount 设为 1） <a href="#">NET_DVR_EVENT_CARD_LINKAGE_COND</a>	<a href="#">NET_DVR_EVENT_CARD_LINKAGE_CFG_V50</a>
NET_DVR_SET_CARD_RIGHT_WEEK_PLAN_V50	dwCount 个（dwCount 设为 1） <a href="#">NET_DVR_WEEK_PLAN_COND</a>	<a href="#">NET_DVR_WEEK_PLAN_CFG</a>
NET_DVR_SET_CARD_RIGHT_HOLIDAY_PLAN_V50	dwCount 个（dwCount 设为 1） <a href="#">NET_DVR_HOLIDAY_PLAN_COND</a>	<a href="#">NET_DVR_HOLIDAY_PLAN_CFG</a>



NET_DVR_SET_CARD_RIGHT_HOLIDAY_GROUP_V 50	dwCount 个 (dwCount 设为 1) <a href="#">NET_DVR_HOLIDAY_GROUP_COND</a>	<a href="#">NET_DVR_HOLIDAY_GROUP_CFG</a>
NET_DVR_SET_CARD_RIGHT_PLAN_TEMPLATE_V5 0	dwCount 个 (dwCount 设为 1) <a href="#">NET_DVR_PLAN_TEMPLATE_COND</a>	<a href="#">NET_DVR_PLAN_TEMPLATE</a>

[返回目录](#)

## RS485/防区/触发器配置和控制

### 6.6.7 获取 **NET\_DVR\_GetDVRConfig**

函 数: BOOL NET\_DVR\_GetDVRConfig(LONG IUserID, DWORD dwCommand, LONG IChannel, LPVOID lpOutBuffer, DWORD dwOutBufferSize, LPDWORD lpBytesReturned)

参 数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V40 的返回值  
[in] dwCommand 设备配置命令, 详见表 6.16  
[in] IChannel 通道号, 如果命令不需要通道号, 该参数无效, 置为 0xFFFFFFFF 即可  
[out] lpOutBuffer 接收数据的缓冲指针, 详见表 6.16  
[in] dwOutBufferSize 接收数据的缓冲长度(以字节为单位), 不能为 0  
[out] lpBytesReturned 实际收到的数据长度指针, 不能为 NULL

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明: 不同的获取功能对应不同的结构体和命令号, 如表 6.16 所示。

表 6.8 获取参数

dwCommand 宏定义	dwCommand 含义	通道号	lpOutBuffer 对应结构体	宏定义值
NET_DVR_GET_ALARM_RS485CFG	获取报警主机 RS485 参数	485 通道号, 从 1 开始	<a href="#">NET_DVR_ALARM_RS485CFG</a>	1189
NET_DVR_GET_ALARMIN_PARAM	获取防区参数	防区号, 从 0 开始	<a href="#">NET_DVR_ALARMIN_PARAM</a>	1183
NET_DVR_GET_ALARMOUT_PARAM	获取触发器参数	触发器号, 从 0 开始	<a href="#">NET_DVR_ALARMOUT_PARAM</a>	1185
NET_DVR_GET_ALARMHOST_MAIN_STATUS	获取设备主要状态(防区、触发器等)	无效	<a href="#">NET_DVR_ALARMHOST_MAIN_STATU S</a>	1190

[返回目录](#)

### 6.6.8 设置 **NET\_DVR\_SetDVRConfig**

函 数: BOOL NET\_DVR\_SetDVRConfig(LONG IUserID, DWORD dwCommand, LONG IChannel, LPVOID lpInBuffer, DWORD dwInBufferSize)

参 数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V40 的返回值  
[in] dwCommand 设备配置命令, 详见表 6.17  
[in] IChannel 通道号, 如果命令不需要通道号, 该参数无效, 置为 0xFFFFFFFF

即可

[in] lpInBuffer                    输入数据的缓冲指针, 详见表 6.17

[in] dwInBufferSize                输入数据的缓冲长度(以字节为单位)

返回值:    TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说 明:**    不同的设置功能对应不同的结构体和命令号, 如表 6.17 所示。

表 6.17 通用参数设置

dwCommand 宏定义	dwCommand 含义	通道号	lpInBuffer 对应结构体	宏定义值
NET_DVR_SET_ALARM_RS485CFG	设置报警主机 RS485 参数	485 通道号, 从 1 开始	<a href="#">NET_DVR_ALARM_RS485CFG</a>	1188
NET_DVR_SET_ALARMIN_PARAM	设置防区参数	防区号, 从 0 开始	<a href="#">NET_DVR_ALARMIN_PARAM</a>	1182
NET_DVR_SET_ALARMOUT_PARAM	设置触发器参数	触发器号, 从 0 开始	<a href="#">NET_DVR_ALARMOUT_PARAM</a>	1184

[返回目录](#)

### 6.6.9 设置触发器 **NET\_DVR\_SetAlarmHostOut**

函 数:    BOOL NET\_DVR\_SetAlarmHostOut (LONG IUserID, LONG IAlarmOutPort, LONG IAlarmOutStatic)

参 数:    [in] IUserID                    NET\_DVR\_Login\_V40 的返回值

          [in] IAlarmOutPort            触发器号。初始触发器号从 0 开始, 0xffffffff 表示全部触发器。

          [in] IAlarmOutStatic          触发器状态: 0—停止输出, 1—输出

返回值:    TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说 明:**

[返回目录](#)

### 6.6.10 对防区布防 **NET\_DVR\_AlarmHostSetupAlarmChan**

函 数:    BOOL NET\_DVR\_AlarmHostSetupAlarmChan (LONG IUserID, NET\_DVR\_ALARMIN\_SETUP \* lpInter)

参 数:    [in] IUserID                    NET\_DVR\_Login\_V40 的返回值

          [in] lpInter                    防区参数, 请参见结构体: [NET\\_DVR\\_ALARMIN\\_SETUP](#)

返回值:    TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说 明:**

[返回目录](#)

### 6.6.11 对防区撤防 **NET\_DVR\_AlarmHostCloseAlarmChan**

函 数:    BOOL NET\_DVR\_AlarmHostCloseAlarmChan (LONG IUserID, NET\_DVR\_ALARMIN\_SETUP \* lpInter)

参 数:    [in] IUserID                    NET\_DVR\_Login\_V40 的返回值

[in] IpInter 防区参数，请参见结构体：[NET\\_DVR\\_ALARMIN\\_SETUP](#)

返回值：TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：**

[返回目录](#)

## 获取所有防区/触发器及外接模块

### 6.6.12 启动长连接远程配置 **NET\_DVR\_StartRemoteConfig**

函 数：LONG NET\_DVR\_StartRemoteConfig(LONG IUserID, DWORD dwCommand, LPVOID lpInBuffer, DWORD dwInBufferLen, fRemoteConfigCallback cbStateCallback, LPVOID pUserData)

参 数：[in] IUserID 用户 ID 号，NET\_DVR\_Login\_V40 返回值  
 [in] dwCommand 配置命令，详见表 6.18  
 [in] lpInBuffer 输入参数，具体内容跟配置命令相关，详见表 6.18  
 [in] dwInBufferLen 输入缓冲的大小  
 [in] cbStateCallback 状态回调函数  
 [in] pUserData 用户数据

typedef void(CALLBACK \*fRemoteConfigCallback)(DWORD dwType, void \*lpBuffer, DWORD dwBufLen, void \*pUserData)

[out] dwType 配置状态，具体取值参见表 6.19  
 [out] lpBuffer 存放数据的缓冲区指针，具体取值参见表 6.19  
 [out] dwBufLen 缓冲区大小  
 [out] pUserData 用户数据

返回值：-1 表示失败，其他值作为 NET\_DVR\_GetNextRemoteConfig、NET\_DVR\_StopRemoteConfig 接口的句柄。接口返回失败请用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：**不同的功能对应不同的命令号(dwCommand)，同时 lpInBuffer 对应不同的结构体，如表 6.18 所示。

表 6.18 长连接配置

dwCommand 宏定义	宏定义值	控制功能	lpInBuffer	回调函数
NET_DVR_GET_ALARMHOST_ZONE_LIST_IN_SUBSYSTEM	2034	获取指定子系统内的所有防区	<a href="#">NET_DVR_LIST_INFO</a>	NULL
NET_DVR_GET_ALARMHOST_TRIGGER_LIST	2035	获取所有触发器	NULL	NULL
NET_DVR_SEARCH_ALARMHOST_EXTERNAL_MODULE	2041	自动搜索外接模块	NULL	返回进度和状态
NET_DVR_REGISTER_ALARMHOST_EXTERNAL_MODULE	2042	自动注册外接模块	NULL	返回进度和状态

表 6.19 长连接回调参数取值

dwType	含义	lpBuffer 对应内容
NET_SDK_CALLBACK_TYPE_STATUS	状态值	4 字节状态值 typedef enum

		<pre> {     NET_SDK_CALLBACK_STATUS_SUCCESS = 1000, //成功     NET_SDK_CALLBACK_STATUS_PROCESSING, //处理中     NET_SDK_CALLBACK_STATUS_FAILED //失败 }NET_SDK_CALLBACK_STATUS_NORMAL;</pre>
NET_SDK_CALLBACK_TYPE_PROGRESS	进度值	lpBuffer 的值表示进度(DWORD)
NET_SDK_CALLBACK_TYPE_DATA	数据信息	lpBuffer 的值表示信息数据（这里无数据回调）

**Remarks**

调用该接口启动长连接远程配置后，还需要调用其他接口获取相关参数，如下表所示：

dwCommand 宏定义	含义	后续接口调用
NET_DVR_GET_ALARMHOST_ZONE_LIST_IN_SUBSYSTEM	获取指定子系统内的所有防区	<a href="#">NET_DVR_GetNextRemoteConfig</a>
NET_DVR_GET_ALARMHOST_TRIGGER_LIST	获取所有触发器	<a href="#">NET_DVR_GetNextRemoteConfig</a>
NET_DVR_SEARCH_ARMHOST_EXTERNAL_MODULE	自动搜索外接模块	<a href="#">NET_DVR_StopRemoteConfig</a>
NET_DVR_REGISTER_ALARMHOST_EXTERNAL_MODULE	自动注册外接模块	<a href="#">NET_DVR_StopRemoteConfig</a>

- 当命令 dwCommand 为 NET\_DVR\_SEARCH\_ARMHOST\_EXTERNAL\_MODULE 或 NET\_DVR\_REGISTER\_ALARMHOST\_EXTERNAL\_MODULE 时，调用该接口启动自动搜索或者自动注册，通过回调函数 fRemoteConfigCallback 获取进度和状态。自动搜索或者自动注册完成后调用 NET\_DVR\_StopRemoteConfig 释放资源。然后再调用该接口，使用命令 NET\_DVR\_GET\_ALARMHOST\_ZONE\_LIST\_IN\_SUBSYSTEM 和 NET\_DVR\_GET\_ALARMHOST\_TRIGGER\_LIST 分别获取“子系统内所有防区”和“所有触发器”信息。
- 自动搜索：搜索出当前可以通信的外接模块。自动注册：自动注册后原有的模块信息（包括外接触发器和外接防区）的地址信息都会改变。设备会根据自身的一套规则给模块重新分配地址。

[返回目录](#)

### 6.6.13 逐个获取查找到的结果信息 **NET\_DVR\_GetNextRemoteConfig**

函数：LONG NET\_DVR\_GetNextRemoteConfig (LONG IHandle, void\* lpOutBuff, DWORD dwOutBuffSize)

参数：[in] IHandle 查找句柄，NET\_DVR\_StartRemoteConfig 的返回值  
 [out] lpOutBuff 输出数据缓冲区，与 NET\_DVR\_StartRemoteConfig 的命令（dwCommand）有关，详见下方“Remarks”  
 [in] dwOutBuffSize 缓冲区长度

返回值：-1 表示失败。其他值表示当前的获取状态等信息，详见下表 6.20。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说明：此接口用于获取一条已查找到的信息，若要获取全部的已查找到的信息，需要循环调用此接口。

表 6.20

宏定义	宏定义值	含义
NET_SDK_GET_NEXT_STATUS_SUCCESS	1000	成功读取到数据，处理完本次数据后需要再次调用 NET_DVR_GetNextRemoteConfig 获取下一条数据

NET_SDK_GET_NETX_STATUS_NEED_WAIT	1001	需等待设备发送数据，继续调用 NET_DVR_GetNextRemoteConfig
NET_SDK_GET_NEXT_STATUS_FINISH	1002	数据全部取完，可调用 NET_DVR_StopRemoteConfig 结束长连接
NET_SDK_GET_NEXT_STATUS_FAILED	1003	出现异常，可调用 NET_DVR_StopRemoteConfig 结束长连接

### Remarks

调用 NET\_DVR\_StartRemoteConfig 时传入不同的命令号(dwCommand), lpOutBuff 对应不同内容，具体如下表：

dwCommand 宏定义	宏定义值	含义	lpInBuffer 对应结构体
NET_DVR_GET_ALARMHOST_ZONE_LIST_IN_SUBSYSTEM	2034	获取指定子系统内的所有防区	<a href="#">NET_DVR_ALARMIN_PARAM</a>
NET_DVR_GET_ALARMHOST_TRIGGER_LIST	2035	获取所有触发器	<a href="#">NET_DVR_ALARMOUT_PARAM</a>

[返回目录](#)

## 6.6.14 关闭长连接配置接口所创建的句柄，释放资源

### NET\_DVR\_StopRemoteConfig

函 数： BOOL NET\_DVR\_StopRemoteConfig(LONG IHandle)

参 数： [in] IHandle 句柄，NET\_DVR\_StartRemoteConfig 的返回值

返回值： TRUE 表示成功，FALSE 表示失败。获取错误码调用 [NET\\_DVR\\_GetLastError](#)。

说 明：

[返回目录](#)

## 用户配置

## 6.6.15 获取设备用户配置 NET\_DVR\_GetAlarmDeviceUser

函 数： BOOL NET\_DVR\_GetAlarmDeviceUser (LONG IUserID, LONG IUserIndex, NET\_DVR\_ALARM\_DEVICE\_USER \* lpDeviceUser)

参 数： [in] IUserID NET\_DVR\_Login\_V40 的返回值

[in] IUserIndex 报警主机设备用户索引

[out] lpDeviceUser 设备用户配置参数，详见：[NET\\_DVR\\_ALARM\\_DEVICE\\_USER](#)

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明： 报警主机设备用户，即网络用户，包括 admin 用户、管理员、普通操作员，最大个数和支持的权限可以通过能力集 [NET\\_DVR\\_GetDeviceAbility](#)（能力集类型：DEVICE\_USER\_ABILITY，对应节点 <AlarmhostPermission>）获取。

[返回目录](#)

### 6.6.16 设置设备用户配置 **NET\_DVR\_SetAlarmDeviceUser**

**函 数：** BOOL NET\_DVR\_SetAlarmDeviceUser (LONG IUserID, LONG IUserIndex, NET\_DVR\_ALARM\_DEVICE\_USER \* IpDeviceUser)

**参 数：** [in] IUserID                      NET\_DVR\_Login\_V40 的返回值  
           [in] IUserIndex              报警主机设备用户索引  
           [in] IpDeviceUser            设备用户配置参数，详见：[NET\\_DVR\\_ALARM\\_DEVICE\\_USER](#)

**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：** 报警主机设备用户，即网络用户，包括 admin 用户、管理员、普通操作员，最大个数和支持的权限可以通过能力集 [NET\\_DVR\\_GetDeviceAbility](#)（能力集类型：DEVICE\_USER\_ABILITY，对应节点 <AlarmhostPermission>）获取。

[返回目录](#)

## 长连接参数配置

### 6.6.17 启动长连接远程配置 **NET\_DVR\_StartRemoteConfig**

**函 数：** LONG NET\_DVR\_StartRemoteConfig(LONG IUserID, DWORD dwCommand, LPVOID lpInBuffer, DWORD dwInBufferLen, fRemoteConfigCallback cbStateCallback, LPVOID pUserData)

**参 数：** [in] IUserID                      用户 ID 号，NET\_DVR\_Login\_V40 返回值  
           [in] dwCommand              配置命令，详见表 6.18  
           [in] lpInBuffer              输入参数，具体内容跟配置命令相关，详见表 6.18  
           [in] dwInBufferLen          输入缓冲的大小  
           [in] cbStateCallback        状态回调函数  
           [in] pUserData              用户数据

**typedef void(CALLBACK \*fRemoteConfigCallback)(DWORD dwType, void \*lpBuffer, DWORD dwBufLen, void \*pUserData)**

[out] dwType                      配置状态，具体取值参见表 6.19  
       [out] lpBuffer                存放数据的缓冲区指针，具体取值参见表 6.19  
       [out] dwBufLen                缓冲区大小  
       [out] pUserData                用户数据

**返回值：** -1 表示失败，其他值作为 [NET\\_DVR\\_SendRemoteConfig](#) 等接口的句柄。接口返回失败请用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：** 不同的功能对应不同的命令号(dwCommand)，同时 lpInBuffer 对应不同的结构体，如表 6.18 所示。

表 6.18 长连接配置

dwCommand 宏定义	宏定义值	控制功能	lpInBuffer	回调函数
NET_DVR_GET_CARD_CFG	2116	获取卡参数	<a href="#">NET_DVR_CARD_CFG_COND</a>	返回状态、信息数据
NET_DVR_SET_CARD_CFG	2117	设置卡参数	<a href="#">NET_DVR_CARD_CFG_COND</a>	返回状态

NET_DVR_GET_CARD_CFG_V50	2178	获取卡参数（V50）	<a href="#">NET_DVR_CARD_CFG_COND</a>	返回状态、信息数据
NET_DVR_SET_CARD_CFG_V50	2179	设置卡参数（V50）	<a href="#">NET_DVR_CARD_CFG_COND</a>	返回状态
NET_DVR_GET_FINGERPRINT_CFG	2150	获取指纹参数	<a href="#">NET_DVR_FINGER_PRINT_INFO_COND</a>	返回状态、信息数据
NET_DVR_SET_FINGERPRINT_CFG	2151	设置指纹参数	<a href="#">NET_DVR_FINGER_PRINT_INFO_COND</a>	返回状态、信息数据
NET_DVR_GET_CARD_PASSWD_CFG	2161	获取卡密码开门使能参数	<a href="#">NET_DVR_CARD_CFG_COND</a>	返回状态、信息数据
NET_DVR_SET_CARD_PASSWD_CFG	2162	设置卡密码开门使能参数	<a href="#">NET_DVR_CARD_CFG_COND</a>	返回状态、信息数据
NET_DVR_CAPTURE_FINGERPRINT_INFO	2504	采集指纹信息（视频门禁一体机 V1.0 新增）	<a href="#">NET_DVR_CAPTURE_FINGERPRINT_COND</a>	返回状态、信息数据

表 6.19 长连接回调参数取值

dwType	含义	lpBuffer 对应内容
NET_SDK_CALLBACK_TYPE_STATUS	状态值	<p>前 4 个字节为状态值(dwStatus):</p> <p>dwStatus 为 NET_SDK_CALLBACK_STATUS_SUCCESS, 表示获取和配置成功并且结束;</p> <p>dwStatus 为 NET_SDK_CALLBACK_STATUS_PROCESSING, lpBuffer: 4 字节状态 + 32 字节卡号;</p> <p>dwStatus 为 NET_SDK_CALLBACK_STATUS_FAILED, lpBuffer: 4 字节状态 + 4 字节错误码 + 32 字节卡号;</p> <p>dwStatus 为 NET_SDK_CALLBACK_STATUS_EXCEPTION, 表示长连接配置异常</p> <p>dwStatus 为 NET_DVR_CALLBACK_STATUS_SEND_WAIT, 表示需要等待一段时间再发送</p>
NET_SDK_CALLBACK_TYPE_PROGRESS	进度值	lpBuffer 的值表示进度(DWORD)
NET_SDK_CALLBACK_TYPE_DATA	信息数据	<p>lpBuffer 的值表示信息数据</p> <p>NET_DVR_GET_CARD_CFG 时, 对应结构体: <a href="#">NET_DVR_CARD_CFG</a></p> <p>NET_DVR_GET_CARD_CFG_V50 时, 对应结构体: <a href="#">NET_DVR_CARD_CFG_V50</a></p> <p>NET_DVR_GET_FINGERPRINT_CFG 时, 对应结构体: <a href="#">NET_DVR_FINGER_PRINT_CFG</a></p> <p>NET_DVR_SET_FINGERPRINT_CFG 时, 对应结构体: <a href="#">NET_DVR_FINGER_PRINT_STATUS</a></p> <p>NET_DVR_GET_CARD_PASSWD_CFG 时, 对应结构体: <a href="#">NET_DVR_CARD_PASSWD_CFG</a></p> <p>NET_DVR_SET_CARD_PASSWD_CFG 时, 对应结构体: <a href="#">NET_DVR_CARD_PASSWD_STATUS</a></p> <p>NET_DVR_CAPTURE_FINGERPRINT_INFO 时, 对应结构体: <a href="#">NET_DVR_CAPTURE_FINGERPRINT_CFG</a></p>

**Remarks**

- NET\_DVR\_GET\_CARD\_CFG 获取卡参数时, 在调用该接口启动长连接远程配置后, 还需要调用 [NET\\_DVR\\_SendRemoteConfig](#) 发送查找条件数据(获取所有卡参数时不需要调用该发送接口), 查找结果在 [NET\\_DVR\\_StartRemoteConfig](#) 设置的回调函数中返回。NET\_DVR\_SET\_CARD\_CFG 设置卡参数时, 在调用该接口启动长连接远程配置后, 通过调用 [NET\\_DVR\\_SendRemoteConfig](#) 向设备下发卡参数信息。
- NET\_DVR\_GET\_FINGERPRINT\_CFG 获取指纹参数时, 调用该接口启动查询, 查找结果在 [NET\\_DVR\\_StartRemoteConfig](#) 设置的回调函数中返回。NET\_DVR\_SET\_FINGERPRINT\_CFG 设置指纹参数时, 在调用该接口启动长连接远程配置后, 通过调用 [NET\\_DVR\\_SendRemoteConfig](#) 向设备下发指纹参数信息,



在 [NET\\_DVR\\_StartRemoteConfig](#) 设置的回调函数中返回状态信息。

- NET\_DVR\_GET\_CARD\_PASSWD\_CFG 获取卡密码开门使能参数时，在调用该接口启动长连接远程配置后，还需要调用 [NET\\_DVR\\_SendRemoteConfig](#) 发送查找条件数据(获取所有卡信息时不需要调用该发送接口)，查找结果在 [NET\\_DVR\\_StartRemoteConfig](#) 设置的回调函数中返回。NET\_DVR\_SET\_CARD\_PASSWD\_CFG 设置卡密码开门使能参数时，在调用该接口启动长连接远程配置后，通过调用 [NET\\_DVR\\_SendRemoteConfig](#) 向设备下发卡密码开门使能信息，在 [NET\\_DVR\\_StartRemoteConfig](#) 设置的回调函数中返回状态信息。

[返回目录](#)

### 6.6.18 发送长连接数据 [NET\\_DVR\\_SendRemoteConfig](#)

函数： BOOL NET\_DVR\_SendRemoteConfig(LONG IHandle, DWORD dwDataType, char \*pSendBuf, DWORD dwBufSize)

参数： [in] IHandle 长连接句柄，NET\_DVR\_StartRemoteConfig 的返回值  
[in] dwDataType 数据类型，详见表 6.9  
[in] pSendBuf 保存发送数据的缓冲区，与 dwDataType 有关，详见表 6.9  
[in] dwBufSize 发送数据的长度

返回值： TRUE 表示成功，FALSE 表示失败。获取错误码调用 [NET\\_DVR\\_GetLastError](#)。

说明： 长连接接口 [NET\\_DVR\\_StartRemoteConfig](#) 中不同的命令(dwCommand)，对应不同的数据类型(dwDataType)和发送数据内容(pSendBuf 对应不同的结构体)，如表 6.9 所示。

表 6.9 长连接发送数据

dwCommand 宏定义	宏定义值	含义	
NET_DVR_GET_CARD_CFG	2116	获取卡参数	
dwDataType 宏定义	宏定义值	控制功能	pSendBuf 对应结构体
ENUM_ACS_SEND_DATA	0x3	门禁主机数据类型	<a href="#">NET_DVR_CARD_CFG_SEND_DATA</a>

dwCommand 宏定义	宏定义值	含义	
NET_DVR_SET_CARD_CFG	2117	设置卡参数	
dwDataType 宏定义	宏定义值	含义	pSendBuf 对应结构体
ENUM_ACS_SEND_DATA	0x3	门禁主机数据类型	<a href="#">NET_DVR_CARD_CFG</a>

dwCommand 宏定义	宏定义值	含义	
NET_DVR_GET_CARD_CFG_V50	2178	获取卡参数（V50）	
dwDataType 宏定义	宏定义值	含义	pSendBuf 对应结构体
ENUM_ACS_SEND_DATA	0x3	门禁主机数据类型	<a href="#">NET_DVR_CARD_CFG_SEND_DATA</a>

dwCommand 宏定义	宏定义值	含义	
NET_DVR_SET_CARD_CFG_V50	2179	设置卡参数（V50）	
dwDataType 宏定义	宏定义值	含义	pSendBuf 对应结构体
ENUM_ACS_SEND_DATA	0x3	门禁主机数据类型	<a href="#">NET_DVR_CARD_CFG_V50</a>



dwCommand 宏定义	宏定义值	含义	
NET_DVR_SET_FINGERPRINT_CFG	2151	设置指纹参数	
dwDataType 宏定义	宏定义值	含义	pSendBuf 对应结构体
ENUM_ACS_SEND_DATA	0x3	门禁主机数据类型	<a href="#">NET_DVR_FINGER_PRINT_CFG</a>

dwCommand 宏定义	宏定义值	含义	
NET_DVR_GET_CARD_PASSWD_CFG	2161	获取卡密码开门使能参数	
dwDataType 宏定义	宏定义值	含义	pSendBuf 对应结构体
ENUM_ACS_SEND_DATA	0x3	门禁主机数据类型	<a href="#">NET_DVR_CARD_CFG_SEND_DATA</a>

dwCommand 宏定义	宏定义值	含义	
NET_DVR_SET_CARD_PASSWD_CFG	2162	设置卡密码开门使能参数	
dwDataType 宏定义	宏定义值	含义	pSendBuf 对应结构体
ENUM_ACS_SEND_DATA	0x3	门禁主机数据类型	<a href="#">NET_DVR_CARD_PASSWD_CFG</a>

#### Remarks

- NET\_DVR\_GET\_CARD\_CFG 获取卡参数或者 NET\_DVR\_GET\_CARD\_PASSWD\_CFG 获取卡密码开门使能参数时，pSendBuf 为查找条件，查找到的卡参数信息或者卡密码开门使能信息在 NET\_DVR\_StartRemoteConfig 设置的回调函数中返回。
- NET\_DVR\_SET\_CARD\_CFG 设置卡参数时，pSendBuf 为下发的卡参数信息，必须保证卡号的唯一性，即卡号的整型值不能重复（即不能同时含有 1 和 01 两种卡号），否则将返回失败。

[返回目录](#)

## 6.6.19 关闭长连接配置接口所创建的句柄，释放资源

### NET\_DVR\_StopRemoteConfig

函 数： BOOL NET\_DVR\_StopRemoteConfig(LONG lHandle)

参 数： [in] lHandle 句柄，NET\_DVR\_StartRemoteConfig 的返回值

返回值： TRUE 表示成功，FALSE 表示失败。获取错误码调用 [NET\\_DVR\\_GetLastError](#)。

说 明：

[返回目录](#)

## 6.7 远程控制

### 6.7.1 远程控制 NET\_DVR\_RemoteControl

函 数： BOOL NET\_DVR\_RemoteControl(LONG lUserID, DWORD dwCommand, LPVOID lpInBuffer, DWORD dwInBufferSize)

**参 数:** [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V40 返回值  
[in] dwCommand 控制命令, 详见表 6.10  
[in] lpInBuffer 输入参数, 跟控制命令相关, 详见列表  
[in] dwInBufferSize 输入参数长度

**返回值:** TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说 明:** 不同的控制功能对应不同的命令号, 同时 lpInBuffer 对应不同的结构体, 如表 6.10 所示。

表 6.10 远程控制命令

dwCommand 宏定义	宏定义值	控制功能	lpInBuffer 对应结构体
NET_DVR_CLEAR_ACS_PARAM	2118	清空门禁主机参数	<a href="#">NET_DVR_ACS_PARAM_TYPE</a>
NET_DVR_DEL_FINGERPRINT_CFG	2152	删除指纹参数	<a href="#">NET_DVR_FINGER_PRINT_INFO_CTRL</a>

[返回目录](#)

## 6.8 门禁控制

### 6.8.1 门禁控制 **NET\_DVR\_ControlGateway**

**函 数:** BOOL NET\_DVR\_ControlGateway (LONG IUserID, LONG IGatewayIndex, DWORD dwStaic)

**参 数:** [in] IUserID NET\_DVR\_Login\_V40 的返回值  
[in] IGatewayIndex 门禁序号, 从 1 开始, -1 表示对所有门进行操作  
[in] dwStaic 命令值: 0- 关闭, 1- 打开, 2- 常开, 3- 常关

**返回值:** TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说 明:**

[返回目录](#)

## 6.9 就地控制器相关

### 6.9.1 就地控制器管理 **NET\_DVR\_STDXMLConfig**

**函 数:** BOOL NET\_DVR\_STDXMLConfig(LONG IUserID, NET\_DVR\_XML\_CONFIG\_INPUT\* lpInputParam, NET\_DVR\_XML\_CONFIG\_OUTPUT\* lpOutputParam)

**参 数:** [in] IUserID NET\_DVR\_Login\_V40 登录接口的返回值  
[in] lpInputParam 输入参数, 详见 [NET\\_DVR\\_XML\\_CONFIG\\_INPUT](#)  
[out] lpOutputParam 输出参数, 详见 [NET\\_DVR\\_XML\\_CONFIG\\_OUTPUT](#)

**返回值:** TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说 明:**

#### Remarks

该接口为 ISAPI 协议透传接口, 不同功能对应输入参数和输出参数具体如下表所示。其中所涉及各个 XML 格式描述的详细内容请参见《设备网络 SDK 使用手册.chm》门禁主机部分对应接口。

功能描述	IpInputParam->IpRequestUrl	IpInputParam->IpInBuffer	IpOutputParam->IpOutBuffer
添加就地控制器	PUT /ISAPI/AccessControl/localController/addDevice	LocalController (XML 格式)	NULL
获取就地控制器参数	GET /ISAPI/AccessControl/localController/localControllerID/<ID> (ID 为就地控制器编号)	NULL	LocalController (XML 格式)
设置就地控制器参数	PUT /ISAPI/AccessControl/localController/localControllerID/<ID> (ID 为就地控制器编号)	LocalController (XML 格式)	NULL
删除单个就地控制器	DELETE /ISAPI/AccessControl/localController/localControllerID/<ID> (ID 为就地控制器编号)	NULL	NULL
删除所有就地控制器	DELETE /ISAPI/AccessControl/localController	NULL	NULL
获取就地控制器能力	GET /ISAPI/AccessControl/localController/capabilities	NULL	LocalController (XML 格式)

[返回目录](#)

## 6.9.2 就地控制器控制 **NET\_DVR\_STDXMLConfig**

函 数： BOOL NET\_DVR\_STDXMLConfig(LONG IUserID, NET\_DVR\_XML\_CONFIG\_INPUT\* IpInputParam, NET\_DVR\_XML\_CONFIG\_OUTPUT\* IpOutputParam)

参 数： [in] IUserID                      NET\_DVR\_Login\_V40 登录接口的返回值  
[in] IpInputParam                      输入参数，详见 [NET\\_DVR\\_XML\\_CONFIG\\_INPUT](#)  
[out] IpOutputParam                      输出参数，详见 [NET\\_DVR\\_XML\\_CONFIG\\_OUTPUT](#)

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明：

### Remarks

该接口为 ISAPI 协议透传接口，不同功能对应输入参数和输出参数具体如下表所示。其中所涉及各个 XML 格式描述的详细内容请参见《设备网络 SDK 使用手册.chm》门禁主机部分对应接口。

功能描述	IpInputParam->IpRequestUrl	IpInputParam->IpInBuffer	IpOutputParam->IpOutBuffer
获取就地控制器控制能力	GET /ISAPI/AccessControl/localController/control/capabilities	NULL	LocalControllerControl (XML 格式)
就地控制器控制	PUT /ISAPI/AccessControl/localController/control/localControllerID/<ID> (ID 为就地控制器编号)	LocalController Control (XML 格式) 当命令为“恢复”时需将超时时间改为 20s	NULL

[返回目录](#)

### 6.9.3 就地控制器搜索及状态查询 **NET\_DVR\_StartRemoteConfig**

函 数: LONG NET\_DVR\_StartRemoteConfig(LONG IUserID, DWORD dwCommand, LPVOID lpInBuffer, DWORD dwInBufferLen, fRemoteConfigCallback cbStateCallback, LPVOID pUserData)

参 数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V40 返回值  
 [in] dwCommand 配置命令, 详见表 6.20  
 [in] lpInBuffer 输入参数, 具体内容跟配置命令相关, 详见表 6.20  
 [in] dwInBufferLen 输入缓冲的大小  
 [in] cbStateCallback 状态回调函数  
 [in] pUserData 用户数据

typedef void(CALLBACK \*fRemoteConfigCallback)(DWORD dwType, void \*lpBuffer, DWORD dwBufLen, void \*pUserData)

[out] dwType 配置状态, 具体取值参见表 6.21  
 [out] lpBuffer 存放数据的缓冲区指针, 具体取值参见表 6.21  
 [out] dwBufLen 缓冲区大小  
 [out] pUserData 用户数据

返回值: -1 表示失败, 其他值作为 [NET\\_DVR\\_SendRemoteConfig](#) 等接口的句柄。接口返回失败请用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明: 不同的功能对应不同命令号(dwCommand), 同时 lpInBuffer 对应不同的结构体, 如表 6.20 所示。

表 6.20 长连接配置

dwCommand 宏定义	宏定义值	控制功能	lpInBuffer	回调函数
NET_DVR_GET_LOCAL_CONTROLLER_STATUS	2176	获取就地控制器状态	<a href="#">NET_DVR_LOCAL_CONTROLLER_STATUS_COND</a>	返回状态、信息数据
NET_DVR_GET_ONLINE_LOCAL_CONTROLLER	2177	搜索在线就地控制器	NULL	返回状态

表 6.21 长连接回调参数取值

dwType	含义	lpBuffer 对应内容
NET_SDK_CALLBACK_TYPE_STATUS	状态值	前 4 个字节为状态值(dwStatus): dwStatus 为 NET_SDK_CALLBACK_STATUS_SUCCESS, 表示获取和配置成功并且结束; dwStatus 为 NET_SDK_CALLBACK_STATUS_PROCESSING, lpBuffer: 4 字节状态 + 32 字节卡号; dwStatus 为 NET_SDK_CALLBACK_STATUS_FAILED, lpBuffer: 4 字节状态 + 4 字节错误码 + 32 字节卡号; dwStatus 为 NET_SDK_CALLBACK_STATUS_EXCEPTION, 表示长连接配置异常 dwStatus 为 NET_DVR_CALLBACK_STATUS_SEND_WAIT, 表示需要等待一段时间再发送
NET_SDK_CALLBACK_TYPE_PROGRESS	进度值	lpBuffer 的值表示进度(DWORD)
NET_SDK_CALLBACK_TYPE_DATA	信息数据	lpBuffer 的值表示信息数据 NET_DVR_GET_LOCAL_CONTROLLER_STATUS 时, 对应结构体: <a href="#">NET_DVR_LOCAL_CONTROLLER_STATUS</a> NET_DVR_GET_ONLINE_LOCAL_CONTROLLER 时, 对应结构体: <a href="#">NET_DVR_ONLINE_LOCAL_CONTROLLER_CFG</a>

### Remarks

- NET\_DVR\_GET\_LOCAL\_CONTROLLER\_STATUS、NET\_DVR\_GET\_ONLINE\_LOCAL\_CONTROLLER 获取就地控制器状态和搜索在线就地控制器，调用该接口启动查询，查找结果在 NET\_DVR\_StartRemoteConfig 设置的回调函数中返回。

[返回目录](#)

## 6.10 USB 设备管理

### 6.10.1 USB 设备管理 NET\_DVR\_STDXMLConfig

函 数： BOOL NET\_DVR\_STDXMLConfig(LONG IUserID, NET\_DVR\_XML\_CONFIG\_INPUT\* IpInputParam, NET\_DVR\_XML\_CONFIG\_OUTPUT\* IpOutputParam)

参 数： [in] IUserID                                NET\_DVR\_Login\_V40 登录接口的返回值  
[in] IpInputParam                            输入参数，详见 [NET\\_DVR\\_XML\\_CONFIG\\_INPUT](#)  
[out] IpOutputParam                        输出参数，详见 [NET\\_DVR\\_XML\\_CONFIG\\_OUTPUT](#)

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

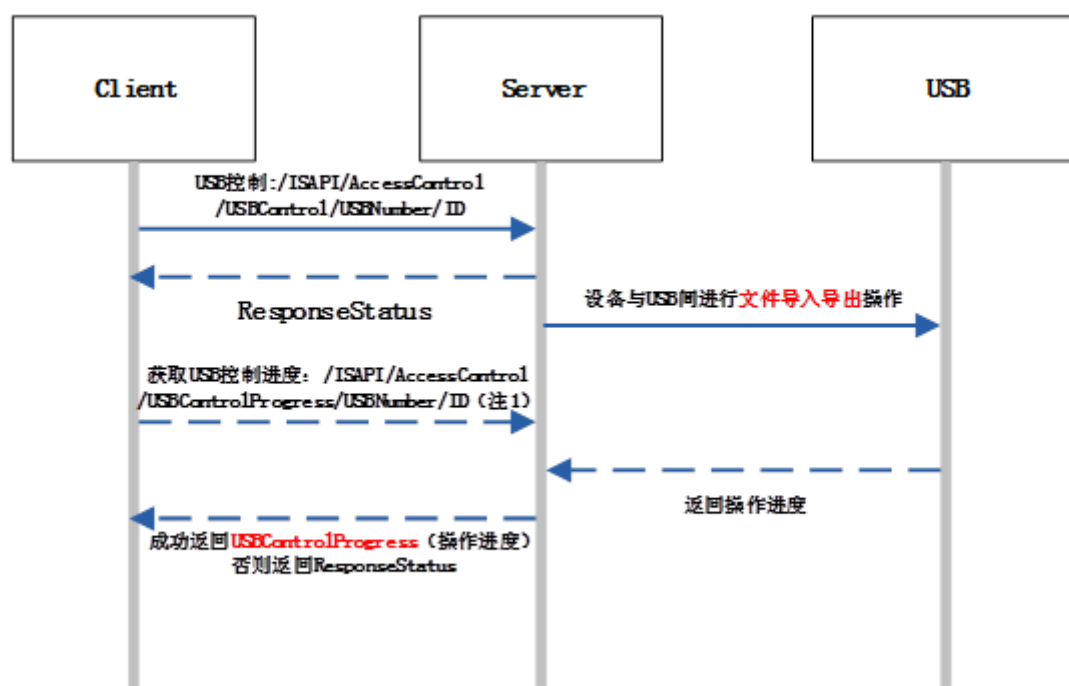
说 明：

### Remarks

该接口为 ISAPI 协议透传接口，不同功能对应输入参数和输出参数具体如下表所示。其中所涉及各个 XML 格式描述的详细内容请参见《设备网络 SDK 使用手册.chm》门禁主机部分对应接口。

功能描述	IpInputParam->IpRequestUrl	IpInputParam->IpInBuffer	IpOutputParam->IpOutBuffer
获取 USB 设备状态	GET /ISAPI/AccessControl/USBStatus/USBNumber/<ID> (ID 为 USB 口编号)	NULL	USBStatus (XML 格式)
获取 USB 设备状态能力	GET /ISAPI/AccessControl/USBStatus/capabilities	NULL	USBStatus (XML 格式)
USB 控制	PUT /ISAPI/AccessControl/USBControl/USBNumber/<ID> (ID 为 USB 口编号)	USBControl (XML 格式)	NULL
获取 USB 控制能力	GET /ISAPI/AccessControl/USBControl/capabilities	NULL	USBControl (XML 格式)
获取 USB 控制进度	GET /ISAPI/AccessControl/USBControlProgress/USBNumber/<ID> (ID 为 USB 口编号)	NULL	USBControlProgress (XML 格式)
获取 USB 控制进度能力	GET /ISAPI/AccessControl/USBControlProgress/capabilities	NULL	USBControlProgress (XML 格式)

USB 控制功能时序图：



- USB 控制接口返回成功只表示接口调用成功，还需调用获取 USB 控制进度命令当获取进度值达到 100 时表示控制完成，可每隔 3 秒调用一次获取控制进度命令查看当前进度值；若中途因不可抗拒因素导致操作失败，则设备返回错误码 29（设备操作失败）。
- 对同一台设备，同时只能进行一种控制命令，否则返回错误码 28（设备资源不足）。

[返回目录](#)

## 6.11 可视对讲呼叫功能

### 6.11.1 启动长连接远程配置 **NET\_DVR\_StartRemoteConfig**

函 数: LONG NET\_DVR\_StartRemoteConfig(LONG lUserID, DWORD dwCommand, LPVOID lpInBuffer, DWORD dwInBufferLen, fRemoteConfigCallback cbStateCallback, LPVOID pUserData)

参 数: [in] lUserID 用户 ID 号，NET\_DVR\_Login\_V40 返回值  
 [in] dwCommand 配置命令，详见表 6.22  
 [in] lpInBuffer 输入参数，具体内容跟配置命令相关，详见表 6.22  
 [in] dwInBufferLen 输入缓冲的大小  
 [in] cbStateCallback 状态回调函数  
 [in] pUserData 用户数据

```
typedef void(CALLBACK *fRemoteConfigCallback)(DWORD dwType, void *lpBuffer, DWORD dwBufLen, void *pUserData)
```

[out] dwType 回调数据类型，具体取值参见表 6.23  
 [out] lpBuffer 存放数据的缓冲区指针，具体取值参见表 6.23  
 [out] dwBufLen 缓冲区大小  
 [out] pUserData 用户数据

返回值: -1 表示失败，其他值作为 [NET\\_DVR\\_SendRemoteConfig](#) 等接口的句柄。接口返回失败请用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明: 不同的功能对应不同命令号(dwCommand)，同时 lpInBuffer 对应不同的结构体，如表 6.22 所示。

表 6.22 长连接配置

dwCommand 宏定义	宏定义值	控制功能	lpInBuffer	回调函数
NET_DVR_VIDEO_CALL_SIGNAL_PROCESS	16032	可视对讲信令处理	<a href="#">NET_DVR_VIDEO_CALL_COND</a>	返回信息数据

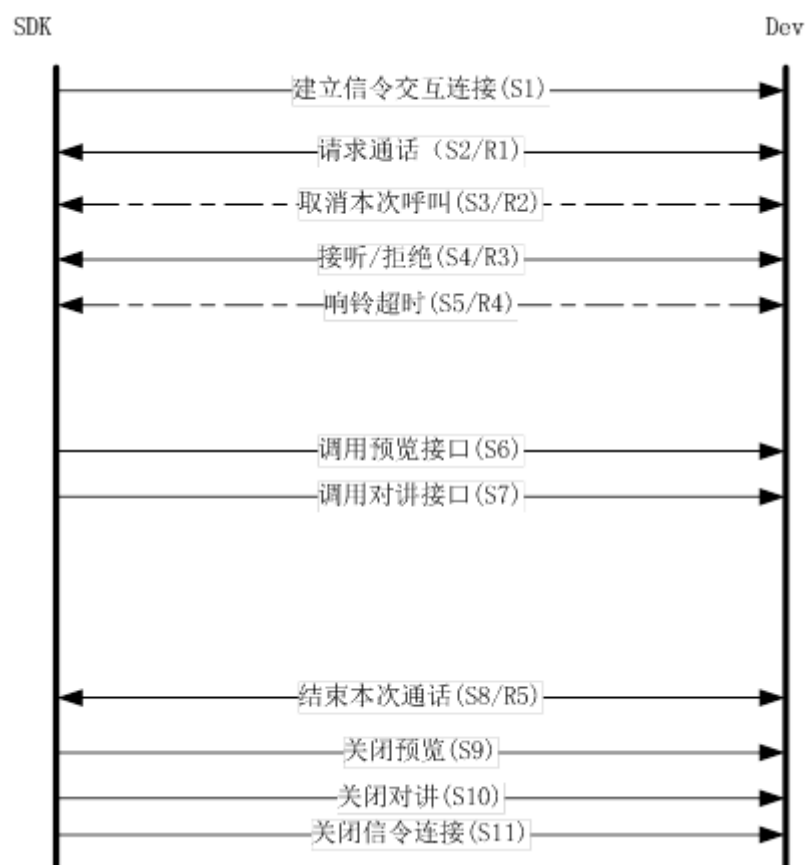
表 6.19 长连接回调参数取值

dwType	含义	lpBuffer 对应内容
NET_SDK_CALLBACK_TYPE_STATUS	状态值	typedef enum { NET_SDK_CALLBACK_STATUS_SUCCESS = 1000, //成功 NET_SDK_CALLBACK_STATUS_PROCESSING, //处理中 NET_SDK_CALLBACK_STATUS_FAILED //失败 }NET_SDK_CALLBACK_STATUS_NORMAL;
NET_SDK_CALLBACK_TYPE_PROGRESS	进度值	lpBuffer 的值表示进度(DWORD)
NET_SDK_CALLBACK_TYPE_DATA	信息数据	lpBuffer 的值表示信息数据 dwCommand 为 NET_DVR_VIDEO_CALL_SIGNAL_PROCESS 时对应结构体： <a href="#">NET_DVR_VIDEO_CALL_PARAM</a>

**Remarks**

- 该长连接配置接口结合报警、预览及对讲接口，可以完成可视通话的功能，具体流程机制如下所示：
- 客户端主动发起对讲：
  - 1)当客户端主动发起连接请求后，客户端如果需要建立可视通话，则需要发送通话请求信令（S2），设备侧接收到该请求后，如果无人接听的情况下，30s 后设备端会返回响铃超时（R4），此时应用层直接关闭信令链接（S11）即可。
  - 2)在客户端发送 S2 请求后，设备未返回 R3 及 R4 之前，客户端可以直接发送 S3，取消本次呼叫，然后直接调用 S7 即可。在客户端发送 S2 请求且设备做了应答后，则无法取消本次呼叫。
  - 3)在客户端发送 S2 请求后，如果设备侧如果拒绝接听 R3，则客户端直接关闭连接即可。
  - 4)在客户端发送 S2 请求后，如果设备接听，则客户端后续打开预览（不播放声音）、对讲即可进行视频通话（S6、S7）；如果中间任何一方想关闭连接，需要先要发送结束本次通话（S8/R5），后续关闭预览、对讲及信令连接即可。
- 设备侧主动发起对讲：
  - 1)当客户端主动发起连接请求后，设备段如果需要建立可视通话，则需要发送通话请求信令（R1），客户端接收到该请求后，如果无人接听的情况下，30s 后会返回响铃超时（S5），此时应用层直接关闭信令链接（S11）即可。
  - 2)在设备端发送 R1 请求后，客户端未返回 S4 及 S5 之前，设备端可以直接发送 R2，取消本次呼叫，然后直接调用 S7 即可。在设备端发送 R1 请求且客户端做了应答后，则无法取消本次呼叫。
  - 3)在设备端发送 R1 请求后，如果客户如果拒绝接听 R3，则客户端直接关闭连接即可。
  - 4)在设备侧发送 R1 请求后，如果客户端接听，则客户端后续打开预览（不播放声音）、对讲即可进行视频通话（S6、S7）；如果中间任何一方想关闭连接，需要先要发送结束本次通话（S8/R5），后续关闭预览、对讲及信令连接即可。





[返回目录](#)

### 6.11.2 发送长连接数据 **NET\_DVR\_SendRemoteConfig**

函 数: `BOOL NET_DVR_SendRemoteConfig(LONG IHandle, DWORD dwDataType, char *pSendBuf, DWORD dwBufSize)`

参 数:

- [in] IHandle 长连接句柄, `NET_DVR_StartRemoteConfig` 的返回值
- [in] dwDataType 数据类型, 可视对讲时为 0
- [in] pSendBuf 保存发送数据的缓冲区, 为一个 [NET\\_DVR\\_VIDEO\\_CALL\\_PARAM](#) 结构体
- [in] dwBufSize 发送数据的长度

返回值: TRUE 表示成功, FALSE 表示失败。获取错误码调用 [NET\\_DVR\\_GetLastError](#)。

说 明:

[返回目录](#)

### 6.11.3 关闭长连接配置接口所创建的句柄, 释放资源

#### **NET\_DVR\_StopRemoteConfig**

函 数: `BOOL NET_DVR_StopRemoteConfig(LONG IHandle)`

参 数:

- [in] IHandle 句柄, `NET_DVR_StartRemoteConfig` 的返回值

返回值: TRUE 表示成功, FALSE 表示失败。获取错误码调用 [NET\\_DVR\\_GetLastError](#)。

说 明:

[返回目录](#)



## 6.12 考勤相关功能

### 6.12.1 考勤相关配置 **NET\_DVR\_STDXMLConfig**

函 数： `BOOL NET_DVR_STDXMLConfig(LONG IUserID, NET_DVR_XML_CONFIG_INPUT* IpInputParam, NET_DVR_XML_CONFIG_OUTPUT* IpOutputParam)`

参 数： [in] IUserID                      NET\_DVR\_Login\_V40 登录接口的返回值  
          [in] IpInputParam              输入参数，详见 [NET\\_DVR\\_XML\\_CONFIG\\_INPUT](#)  
          [out] IpOutputParam            输出参数，详见 [NET\\_DVR\\_XML\\_CONFIG\\_OUTPUT](#)

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明：

#### Remarks

该接口为 ISAPI 协议透传接口，不同功能对应输入参数和输出参数具体如下表所示。其中所涉及各个 XML 格式描述的详细内容请参见《设备网络 SDK 使用手册.chm》门禁主机部分对应接口。

功能描述	IpInputParam->IpRequestUrl	IpInputParam->IpInBuffer	IpOutputParam->IpOutBuffer
部门参数配置能力	GET /ISAPI/AccessControl/DepartmentParam/capabilities	NULL	DepartmentParam（部门参数配置能力 XML 格式）
获取部门参数	GET /ISAPI/AccessControl/DepartmentParam/DepartmentNo/<ID> （ID 为部门编号）	NULL	DepartmentParam（部门参数 XML 格式）
设置部门参数	PUT /ISAPI/AccessControl/DepartmentParam/DepartmentNo/<ID> （ID 为部门编号）	DepartmentParam（部门参数 XML 格式）	NULL
排班计划配置能力	GET /ISAPI/AccessControl/SchedulePlan/capabilities	NULL	SchedulePlan（排班计划配置能力 XML 格式）
获取排班计划	GET /ISAPI/AccessControl/SchedulePlan/SchedulePlanNo/<ID>（ID 为排班计划编号）	NULL	SchedulePlan（排班计划 XML 格式）
设置排班计划	PUT /ISAPI/AccessControl/SchedulePlan/SchedulePlanNo/<ID>（ID 为排班计划编号）	SchedulePlan（排班计划 XML 格式）	NULL
考勤规则配置能力	GET /ISAPI/AccessControl/AttendanceRule/capabilities	NULL	AttendanceRule（考勤规则配置能力 XML 格式）
获取考勤规则	GET /ISAPI/AccessControl/AttendanceRule/AttendanceRuleNo/<ID> （ID 为考勤规则编号）	NULL	AttendanceRule（考勤规则 XML 格式）
设置考勤规则	PUT /ISAPI/AccessControl/AttendanceRule/AttendanceRuleNo/<ID> （ID 为考勤规则编号）	AttendanceRule（考勤规则 XML 格式）	NULL

普通班参数配置能力	GET /ISAPI/AccessControl/OrdinaryClass/capabilities	NULL	OrdinaryClass（普通班参数配置能力 XML 格式）
获取普通班参数	GET /ISAPI/AccessControl/OrdinaryClass/OrdinaryClassNo/<ID>（ID 为班次编号）	NULL	OrdinaryClass（普通班参数 XML 格式）
设置普通班参数	PUT /ISAPI/AccessControl/OrdinaryClass/OrdinaryClassNo/<ID>（ID 为班次编号）	OrdinaryClass（普通班参数 XML 格式）	NULL
工时班参数配置能力	GET /ISAPI/AccessControl/WorkingClass/capabilities	NULL	WorkingClass（工时班参数配置能力 XML 格式）
获取工时班参数	GET /ISAPI/AccessControl/WorkingClass/WorkingClassNo/<ID>（ID 为班次编号）	NULL	WorkingClass（工时班参数 XML 格式）
设置工时班参数	PUT /ISAPI/AccessControl/WorkingClass/WorkingClassNo/<ID>（ID 为班次编号）	WorkingClass（工时班参数 XML 格式）	NULL
考勤假日组配置能力	GET /ISAPI/AccessControl/AttendanceHolidayGroup/capabilities	NULL	AttendanceHolidayGroup（考勤假日组配置能力 XML 格式）
获取考勤假日组配置	GET /ISAPI/AccessControl/AttendanceHolidayGroup/HolidayGroupNo/<ID>（ID 为假日组编号）	NULL	AttendanceHolidayGroup（考勤假日组配置 XML 格式）
设置考勤假日组配置	PUT /ISAPI/AccessControl/AttendanceHolidayGroup/HolidayGroupNo/<ID>（ID 为假日组编号）	AttendanceHolidayGroup（考勤假日组配置 XML 格式）	NULL
考勤假日计划配置能力	GET /ISAPI/AccessControl/AttendanceHolidayPlan/capabilities	NULL	AttendanceHolidayPlan（考勤假日计划配置能力 XML 格式）
获取考勤假日计划配置	GET /ISAPI/AccessControl/AttendanceHolidayPlan/HolidayPlanNo/<ID>（ID 为假日计划编号）	NULL	AttendanceHolidayPlan（考勤假日计划配置 XML 格式）
设置考勤假日计划配置	PUT /ISAPI/AccessControl/AttendanceHolidayPlan/HolidayPlanNo/<ID>（ID 为假日计划编号）	AttendanceHolidayPlan（考勤假日计划配置 XML 格式）	NULL
考勤有效 ID 获取能力	GET /ISAPI/AccessControl/AttendanceEffectiveID/capabilities	NULL	AttendanceEffectiveID（考勤有效 ID 获取能力 XML 格式）
获取考勤有效 ID	GET /ISAPI/AccessControl/AttendanceEffectiveID	NULL	AttendanceEffectiveID（考勤有效 ID XML 格式）

[返回目录](#)

## 6.13 手动抓拍

### 6.13.1 单帧数据捕获并保存成 JPEG 图片 **NET\_DVR\_CaptureJPEGPicture**

函 数: BOOL NET\_DVR\_CaptureJPEGPicture(LONG IUserID, LONG IChannel, LPNET\_DVR\_JPEGPARA lpJpegPara, char \*sPicFileName)

参 数: [in]IUserID NET\_DVR\_Login\_V40 的返回值  
 [in]IChannel 通道号  
 [in]lpJpegPara JPEG 图像参数, 详见 [NET\\_DVR\\_JPEGPARA](#)  
 [in]sPicFileName 保存 JPEG 图的文件路径, 路径长度和操作系统有关, sdk 不做限制, windows 默认路径长度小于等于 256 字节 (包括文件名在内)。

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明: 该接口用于设备的单帧数据捕获, 并保存成 JPEG 图片。

#### Remarks

- JPEG 抓图功能或者抓图分辨率需要设备支持, 如果不支持接口返回失败, 错误号 23 或者 29。
- 对接网络摄像机、门禁主机等设备, 设备是否支持 JPEG 抓图功能或者支持的参数能力, 可以通过设备能力集进行判断, 对应设备 JPEG 抓图能力集(JpegCaptureAbility), 相关接口: NET\_DVR\_GetDeviceAbility, 能力集类型: DEVICE\_JPEG\_CAP\_ABILITY, 节点: <ManualCapture>。

[返回目录](#)

### 6.13.2 单帧数据捕获并保存成 JPEG 存放在指定的内存空间中

#### **NET\_DVR\_CaptureJPEGPicture\_NEW**

函 数: BOOL NET\_DVR\_CaptureJPEGPicture\_NEW(LONG IUserID, LONG IChannel, LPNET\_DVR\_JPEGPARA lpJpegPara, char \*sJpegPicBuffer, DWORD dwPicSize, LPDWORD lpSizeReturned)

参 数: [in]IUserID NET\_DVR\_Login\_V40 的返回值  
 [in]IChannel 通道号  
 [in]lpJpegPara JPEG 图像参数, 详见 [NET\\_DVR\\_JPEGPARA](#)  
 [in]sJpegPicBuffer 保存 JPEG 数据的缓冲区  
 [in]dwPicSize 输入缓冲区大小, 不能小于图片数据的大小  
 [out]lpSizeReturned 返回图片数据的大小

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明: 该接口用于设备的单帧数据捕获, 并保存成 JPEG 图片到指定内存空间。

[返回目录](#)

### 6.13.3 网络触发抓拍 **NET\_DVR\_ContinuousShoot**

函 数: BOOL NET\_DVR\_ContinuousShoot(LONG IUserID, LPNET\_DVR\_SNAPCFG lpInter)

参 数: [in] IUserID NET\_DVR\_Login\_V40 的返回值  
 [in] lpInter 网络触发抓拍配置, 请参见 [NET\\_DVR\\_SNAPCFG](#)

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通

过错误码判断出错原因。

**说明：** 使用网络触发连拍接收图片，需要先对设备进行布防（[NET\\_DVR\\_SetupAlarmChan V41](#)）并设置回调函数（[NET\\_DVR\\_SetDVRMessageCallBack V31](#)），通过回调获取抓拍的图片数据。

#### Remarks

- 对于门禁主机，设备是否支持网络触发连拍或者支持的参数能力，可以通过设备能力集进行判断，对应门禁主机能力集(AcsAbility)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：ACS\_ABILITY，节点：<ContinuousShootCfg>。

[返回目录](#)

## 6.14 底图功能

### 6.14.1 底图上传 **NET\_DVR\_PicUpload**

**函数：** LONG NET\_DVR\_PicUpload (LONG IUserID, char const\* sFileName, LPNET\_DVR\_PICTURECFG lpPictureCfg)

**参数：** [in] IUserID                      NET\_DVR\_Login\_V40 的返回值  
[in] sFileName                      上传图片的图片路径（包括文件名）  
[in] lpPictureCfg                      图片参数，详见 [NET\\_DVR\\_PICTURECFG](#)

**返回值：** -1 表示失败，其他值作为 NET\_DVR\_GetPicUploadState 函数的参数。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说明：**

#### Remarks

- 通过该接口可以上传图片文件作为显示屏的底图。上传之后的图片如果需要删除，可以调用 NET\_DVR\_RemoteControl(命令：NET\_DVR\_DELETE\_PICTURE)实现。

[返回目录](#)

### 6.14.2 获取图片上传的进度 **NET\_DVR\_GetPicUploadProgress**

**函数：** LONG NET\_DVR\_GetPicUploadProgress (LONG IUploadHandle )

**参数：** [in] IUploadHandle                      图片上传句柄，NET\_DVR\_PicUpload 的返回值

**返回值：** -1 表示失败，0~100 表示图片上传进度。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说明：**

[返回目录](#)

### 6.14.3 获取远程上传的状态 **NET\_DVR\_GetPicUploadState**

**函数：** LONG NET\_DVR\_GetPicUploadState (LONG IUploadHandle )

**参数：** [in] IUploadHandle                      图片上传句柄，NET\_DVR\_PicUpload 的返回值

**返回值：** -1 表示失败，1- 完成，2- 正在上传，3- 上传失败，4- 未知错误。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说明：**

[返回目录](#)

#### 6.14.4 关闭上传句柄，释放资源 **NET\_DVR\_CloseUploadHandle**

**函 数：** BOOL NET\_DVR\_CloseUploadHandle (LONG IUploadHandle )  
**参 数：** [in] IUploadHandle 图片上传句柄，NET\_DVR\_PicUpload 的返回值  
**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。  
**说 明：**

[返回目录](#)

#### 6.14.5 远程控制 **NET\_DVR\_RemoteControl**

**函 数：** BOOL NET\_DVR\_RemoteControl (LONG IUserID, DWORD dwCommand, LPVOID lpInBuffer, DWORD dwInBufferSize)  
**参 数：** [in] IUserID NET\_DVR\_Login\_V40 的返回值  
[in] dwCommand 控制命令，详见下表  
[in] lpInBuffer 输入参数，具体内容跟控制命令相关，详见列表  
dwInBufferSize 输入参数长度  
**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。  
**说 明：**

##### Remarks

不同的控制功能对应不同的命令号(dwCommand)，同时 lpInBuffer 对应不同的结构体，如下表所示：

dwCommand 宏定义	宏定义值	控制功能	lpInBuffer 对应结构体
NET_DVR_DELETE_PICTURE	9126	删除底图	4 字节底图序号

[返回目录](#)

### 6.15 设备维护管理

#### 在线状态检测

##### 6.15.1 设备在线状态检测 **NET\_DVR\_RemoteControl**

**函 数：** BOOL NET\_DVR\_RemoteControl(LONG IUserID, DWORD dwCommand, LPVOID lpInBuffer, DWORD dwInBufferSize)  
**参 数：** [in]IUserID 用户 ID 号，NET\_DVR\_Login\_V40 的返回值  
[in]dwCommand 控制命令，详见表 6.11  
[in]lpInBuffer 输入参数，具体内容跟控制命令相关，详见表 6.11  
[in]dwInBufferSize 输入参数长度  
**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。  
**说 明：** 该控制命令用于手动检测设备是否在线，接口返回 TRUE 表示在线，FALSE 表示与设备通信失败或者返回错误状态。设备在线状态自动巡检功能通过 [NET\\_DVR\\_SetSDKLocalCfg](#)（配置类型：NET\_SDK\_LOCAL\_CFG\_TYPE\_CHECK\_DEV）进行配置。

表 6.11 远程控制命令

dwCommand 宏定义	宏定义值	控制功能	lpInBuffer 对应结构体
NET_DVR_CHECK_USER_STATUS	20005	检测设备是否在线	NULL

[返回目录](#)

## 远程升级

### 6.15.2 设置远程升级时网络环境 **NET\_DVR\_SetNetworkEnvironment**

函 数: `BOOL NET_DVR_SetNetworkEnvironment(DWORD dwEnvironmentLevel)`

参 数: `[in] dwEnvironmentLevel` 网络环境级别

```
enum{
    LOCAL_AREA_NETWORK = 0, //局域网环境
    WIDE_AREA_NETWORK    //广域网环境
}
```

返回值: `TRUE` 表示成功, `FALSE` 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说 明:** 接口中的网络环境级别参数分为两类,  
[LOCAL\\_AREA\\_NETWORK](#) 表示局域网环境(网络环境好, 通讯流畅);  
[WIDE\\_AREA\\_NETWORK](#) 表示广域网环境(网络环境差, 易阻塞)。  
 在调用远程升级接口之前, 可以通过此接口适应不同的升级环境。

[返回目录](#)

### 6.15.3 远程升级 **NET\_DVR\_Upgrade**

函 数: `LONG NET_DVR_Upgrade(LONG lUserID, char *sFileName)`

参 数: `[in] lUserID` `NET_DVR_Login_V40` 的返回值  
`[in] sFileName` 升级的文件路径(包括文件名)。路径长度和操作系统有关, `sdk` 不做限制, `windows` 默认路径长度小于等于 256 字节(包括文件名在内)。

返回值: `-1` 表示失败, 其他值作为 `NET_DVR_GetUpgradeState` 等函数的参数。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说 明:**

[返回目录](#)

### 6.15.4 获取远程升级的进度 **NET\_DVR\_GetUpgradeProgress**

函 数: `int NET_DVR_GetUpgradeProgress(LONG lUpgradeHandle)`

参 数: `[in] lUpgradeHandle` `NET_DVR_Upgrade` 的返回值

返回值: `-1` 表示失败, `0~100` 表示升级进度。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说 明:**

[返回目录](#)

### 6.15.5 获取远程升级的状态 **NET\_DVR\_GetUpgradeState**

函 数: int NET\_DVR\_GetUpgradeState(LONG IUpgradeHandle)  
 参 数: [in] IUpgradeHandle NET\_DVR\_Upgrade 的返回值  
 返回值: -1 表示失败, 其他值定义: 1- 升级成功; 2- 正在升级; 3- 升级失败; 4- 网络断开, 状态未知; 5- 升级文件语言版本不匹配。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明:

[返回目录](#)

### 6.15.6 关闭远程升级句柄, 释放资源 **NET\_DVR\_CloseUpgradeHandle**

函 数: BOOL NET\_DVR\_CloseUpgradeHandle(LONG IUpgradeHandle)  
 参 数: [in] IUpgradeHandle NET\_DVR\_Upgrade 的返回值  
 返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明:

[返回目录](#)

## 恢复设备默认参数

### 6.15.7 恢复设备默认参数 **NET\_DVR\_RestoreConfig**

函 数: BOOL NET\_DVR\_RestoreConfig(LONG IUserID)  
 参 数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V40 的返回值  
 返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明:

[返回目录](#)

### 6.15.8 完全恢复出厂默认参数 **NET\_DVR\_RemoteControl**

函 数: BOOL NET\_DVR\_RemoteControl(LONG IUserID, DWORD dwCommand, LPVOID lpInBuffer, DWORD dwInBufferSize)  
 参 数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V40 的返回值  
       [in] dwCommand 控制命令, 详见表 6.12  
       [in] lpInBuffer 输入参数, 跟控制命令相关, 详见列表  
       [in] dwInBufferSize 输入参数长度  
 返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明: 不同的控制功能对应不同的命令号, 同时 lpInBuffer 对应不同的结构体, 如表 6.12 所示。



表 6.12 远程控制命令

dwCommand 宏定义	宏定义值	控制功能	lpInBuffer 对应结构体
NET_DVR_COMPLETE_RESTORE_CTRL	3420	设置完全恢复出厂值	<a href="#">NET_DVR_COMPLETE_RESTORE_INFO</a>

[返回目录](#)

## 导入/导出配置文件

### 6.15.9 导出配置文件 **NET\_DVR\_GetConfigFile\_V30**

**函 数：** BOOL NET\_DVR\_GetConfigFile\_V30(LONG IUserID, char \*sOutBuffer, DWORD dwOutSize, DWORD \*pReturnSize)

**参 数：** [in] IUserID 用户 ID 号，NET\_DVR\_Login\_V40 的返回值  
 [out] sOutBuffer 存放配置参数的缓冲区  
 [in] dwOutSize 缓冲区大小  
 [out] pReturnSize 实际获得的缓冲区大小

**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：** 当 sOutBuffer = NULL、dwOutSize = 0 且 pReturnSize != NULL 时用于获取参数配置文件的所需的缓冲区长度；  
 当 sOutBuffer != NULL 且 dwOutSize != 0 时用于获取参数配置文件的所需的缓冲区内容。

[返回目录](#)

### 6.15.10 导出配置文件 **NET\_DVR\_GetConfigFile**

**函 数：** BOOL NET\_DVR\_GetConfigFile(LONG IUserID, char \*sFileName)

**参 数：** [in] IUserID 用户 ID 号，NET\_DVR\_Login\_V40 的返回值  
 [in] sFileName 存放保存配置文件的文件路径（二进制文件）

**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：**

[返回目录](#)

### 6.15.11 导入配置文件 **NET\_DVR\_SetConfigFile\_EX**

**函 数：** BOOL NET\_DVR\_SetConfigFile\_EX(LONG IUserID, char \*sInBuffer, DWORD dwInSize)

**参 数：** [in] IUserID 用户 ID 号，NET\_DVR\_Login\_V40 的返回值  
 [in] sInBuffer 存放配置参数的缓冲区  
 [in] dwInSize 缓冲区大小

**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：**

[返回目录](#)



### 6.15.12 导入配置文件 **NET\_DVR\_SetConfigFile**

函 数: BOOL NET\_DVR\_SetConfigFile(LONG IUserID, char \*sFileName)

参 数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V40 的返回值  
[in] sFileName 存放保存配置文件的文件路径 (二进制文件)

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明:

[返回目录](#)

## 远程重启

### 6.15.13 重启设备 **NET\_DVR\_RebootDVR**

函 数: BOOL NET\_DVR\_RebootDVR(LONG IUserID)

参 数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V40 的返回值

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明:

[返回目录](#)

## 7 错误代码及说明

### 7.1 网络通讯库错误码

错误名称	错误值	说明
NET_DVR_NOERROR	0	没有错误。
NET_DVR_PASSWORD_ERROR	1	用户名密码错误。注册时输入的用户名或者密码错误。
NET_DVR_NOENOUGHPRI	2	权限不足。该注册用户没有权限执行当前对设备的操作，可以与远程用户参数配置做对比。
NET_DVR_NOINIT	3	SDK 未初始化。
NET_DVR_CHANNEL_ERROR	4	通道号错误。设备没有对应的通道号。
NET_DVR_OVER_MAXLINK	5	连接到设备的用户个数超过最大。
NET_DVR_VERSIONNOMATCH	6	版本不匹配。SDK 和设备的版本不匹配。
NET_DVR_NETWORK_FAIL_CONNECT	7	连接设备失败。设备不在线或网络原因引起的连接超时等。
NET_DVR_NETWORK_SEND_ERROR	8	向设备发送失败。
NET_DVR_NETWORK_RECV_ERROR	9	从设备接收数据失败。
NET_DVR_NETWORK_RECV_TIMEOUT	10	从设备接收数据超时。
NET_DVR_NETWORK_ERRORDATA	11	传送的数据有误。发送给设备或者从设备接收到的数据错误，如远程参数配置时输入设备不支持的值。
NET_DVR_ORDER_ERROR	12	调用次序错误。
NET_DVR_OPERNOPERMIT	13	无此权限。
NET_DVR_COMMANDTIMEOUT	14	设备命令执行超时。
NET_DVR_ERRORSERIALPORT	15	串口号错误。指定的设备串口号不存在。
NET_DVR_ERRORALARMPORT	16	报警端口错误。指定的设备报警输出端口不存在。
NET_DVR_PARAMETER_ERROR	17	参数错误。SDK 接口中给入的输入或输出参数为空。
NET_DVR_CHAN_EXCEPTION	18	设备通道处于错误状态
NET_DVR_NODISK	19	设备无硬盘。当设备无硬盘时，对设备的录像文件、硬盘配置等操作失败。
NET_DVR_ERRORDISKNUM	20	硬盘号错误。当对设备进行硬盘管理操作时，指定的硬盘号不存在时返回该错误。
NET_DVR_DISK_FULL	21	设备硬盘满。
NET_DVR_DISK_ERROR	22	设备硬盘出错
NET_DVR_NOSUPPORT	23	设备不支持。
NET_DVR_BUSY	24	设备忙。
NET_DVR_MODIFY_FAIL	25	设备修改不成功。
NET_DVR_PASSWORD_FORMAT_ERROR	26	密码输入格式不正确
NET_DVR_DISK_FORMATING	27	硬盘正在格式化，不能启动操作。
NET_DVR_DVRNORESOURCE	28	设备资源不足。
NET_DVR_DVROPRATEFAILED	29	设备操作失败。
NET_DVR_OPENHOSTSOUND_FAIL	30	语音对讲、语音广播操作中采集本地音频或打开音频输出失败。
NET_DVR_DVRVOICEOPENED	31	设备语音对讲被占用。

NET_DVR_TIMEINPUTERROR	32	时间输入不正确。
NET_DVR_NOSPECFILE	33	回放时设备没有指定的文件。
NET_DVR_CREATEFILE_ERROR	34	创建文件出错。本地录像、保存图片、获取配置文件和远程下载录像时创建文件失败。
NET_DVR_FILEOPENFAIL	35	打开文件出错。设置配置文件、设备升级、上传审讯文件时打开文件失败。
NET_DVR_OPERNOTFINISH	36	上次的操作还没有完成
NET_DVR_GETPLAYTIMEFAIL	37	获取当前播放的时间出错。
NET_DVR_PLAYFAIL	38	播放出错。
NET_DVR_FILEFORMAT_ERROR	39	文件格式不正确。
NET_DVR_DIR_ERROR	40	路径错误
NET_DVR_ALLOC_RESOURCE_ERROR	41	SDK 资源分配错误。
NET_DVR_AUDIO_MODE_ERROR	42	声卡模式错误。当前打开声音播放模式与实际设置的模式不符出错。
NET_DVR_NOENOUGH_BUF	43	缓冲区太小。接收设备数据的缓冲区或存放图片缓冲区不足。
NET_DVR_CREATESOCKET_ERROR	44	创建 SOCKET 出错。
NET_DVR_SETSOCKET_ERROR	45	设置 SOCKET 出错。
NET_DVR_MAX_NUM	46	个数达到最大。分配的注册连接数、预览连接数超过 SDK 支持的最大数。
NET_DVR_USERNOTEXIST	47	用户不存在。注册的用户 ID 已注销或不可用。
NET_DVR_WRITEFLASHERROR	48	写 FLASH 出错。设备升级时写 FLASH 失败。
NET_DVR_UPGRADEFAIL	49	设备升级失败。网络或升级文件语言不匹配等原因升级失败。
NET_DVR_CARDHAVEINIT	50	解码卡已经初始化过。
NET_DVR_PLAYERFAILED	51	调用播放库中某个函数失败。
NET_DVR_MAX_USERNUM	52	登录设备的用户数达到最大。
NET_DVR_GETLOCALIPANDMACFAIL	53	获得本地 PC 的 IP 地址或物理地址失败。
NET_DVR_NOENCODEING	54	设备该通道没有启动编码。
NET_DVR_IPMISMATCH	55	IP 地址不匹配。
NET_DVR_MACMISMATCH	56	MAC 地址不匹配。
NET_DVR_UPGRADELANGMISMATCH	57	升级文件语言不匹配。
NET_DVR_MAX_PLAYERPORT	58	播放器路数达到最大。
NET_DVR_NOSPACEBACKUP	59	备份设备中没有足够空间进行备份。
NET_DVR_NODEVICEBACKUP	60	没有找到指定的备份设备。
NET_DVR_PICTURE_BITS_ERROR	61	图像素位数不符，限 24 色。
NET_DVR_PICTURE_DIMENSION_ERROR	62	图片高*宽超限，限 128*256。
NET_DVR_PICTURE_SIZ_ERROR	63	图片大小超限，限 100K。
NET_DVR_LOADPLAYERSDKFAILED	64	载入当前目录下 Player Sdk 出错。
NET_DVR_LOADPLAYERSDKPROC_ERROR	65	找不到 Player Sdk 中某个函数入口。
NET_DVR_LOADDSSDKFAILED	66	载入当前目录下 DsSdk 出错。
NET_DVR_LOADDSSDKPROC_ERROR	67	找不到 DsSdk 中某个函数入口。
NET_DVR_DSSDK_ERROR	68	调用硬解码库 DsSdk 中某个函数失败。

NET_DVR_VOICEMONOPOLIZE	69	声卡被独占。
NET_DVR_JOINMULTICASTFAILED	70	加入多播组失败。
NET_DVR_CREATEDIR_ERROR	71	建立日志文件目录失败。
NET_DVR_BINDSOCKET_ERROR	72	绑定套接字失败。
NET_DVR_SOCKETCLOSE_ERROR	73	socket 连接中断，此错误通常是由于连接中断或目的地不可达。
NET_DVR_USERID_ISUSING	74	注销时用户 ID 正在进行某操作。
NET_DVR_SOCKETLISTEN_ERROR	75	监听失败。
NET_DVR_PROGRAM_EXCEPTION	76	程序异常。
NET_DVR_WRITEFILE_FAILED	77	写文件失败。本地录像、远程下载录像、下载图片等操作时写文件失败。
NET_DVR_FORMAT_READONLY	78	禁止格式化只读硬盘。
NET_DVR_WITHSAMEUSERNAME	79	远程用户配置结构中存在相同的用户名。
NET_DVR_DEVICEYPE_ERROR	80	导入参数时设备型号不匹配。
NET_DVR_LANGUAGE_ERROR	81	导入参数时语言不匹配。
NET_DVR_PARAVERSION_ERROR	82	导入参数时软件版本不匹配。
NET_DVR_IPCHAN_NOTALIVE	83	预览时外接 IP 通道不在线。
NET_DVR_RTSP_SDK_ERROR	84	加载标准协议通讯库 StreamTransClient 失败。
NET_DVR_CONVERT_SDK_ERROR	85	加载转封装库失败。
NET_DVR_IPC_COUNT_OVERFLOW	86	超出最大的 IP 接入通道数。
NET_DVR_MAX_ADD_NUM	87	添加录像标签或者其他操作超出最多支持的个数。
NET_DVR_PARAMMODE_ERROR	88	图像增强仪，参数模式错误（用于硬件设置时，客户端进行软件设置时错误值）。
NET_DVR_CODESPITTER_OFFLINE	89	码分器不在线。
NET_DVR_BACKUP_COPYING	90	设备正在备份。
NET_DVR_CHAN_NOTSUPPORT	91	通道不支持该操作。
NET_DVR_CALLINEINVALID	92	高度线位置太集中或长度线不够倾斜。
NET_DVR_CALCANCELCONFLICT	93	取消标定冲突，如果设置了规则及全局的实际大小尺寸过滤。
NET_DVR_CALPOINTOUTRANGE	94	标定点超出范围。
NET_DVR_FILTERRECTINVALID	95	尺寸过滤器不符合要求。
NET_DVR_DDNS_DEVOFFLINE	96	设备没有注册到 ddns 上。
NET_DVR_DDNS_INTER_ERROR	97	DDNS 服务器内部错误。
NET_DVR_ALIAS_DUPLICATE	150	别名重复（EasyDDNS 的配置）
NET_DVR_DEV_NET_OVERFLOW	800	网络流量超过设备能力上限
NET_DVR_STATUS_RECORDFILE_WRITING_NOT_LOCK	801	录像文件在录像，无法被锁定
NET_DVR_STATUS_CANT_FORMAT_LITTLE_DISK	802	由于硬盘太小无法格式化
门禁主机错误码		
NET_ERR_TIME_OVERLAP	1900	时间段重叠
NET_ERR_HOLIDAY_PLAN_OVERLAP	1901	假日计划重叠
NET_ERR_CARDNO_NOT_SORT	1902	卡号未排序
NET_ERR_CARDNO_NOT_EXIST	1903	卡号不存在
NET_ERR_ILLEGAL_CARDNO	1904	卡号错误
NET_ERR_ZONE_ALARM	1905	防区处于布防状态(参数修改不允许)

NET_ERR_ZONE_OPERATION_NOT_SUPPORT	1906	防区不支持该操作
NET_ERR_INTERLOCK_ANTI_CONFLICT	1907	多门互锁和反潜回同时配置错误
NET_ERR_DEVICE_CARD_FULL	1908	卡已满（卡达到 10W 后返回）
NET_ERR_HOLIDAY_GROUP_DOWNLOAD	1909	假日组下载失败
NET_ERR_LOCAL_CONTROL_OFF	1910	就地控制器离线
NET_ERR_LOCAL_CONTROL_DISADD	1911	就地控制器未添加
NET_ERR_LOCAL_CONTROL_HASADD	1912	就地控制器已添加
NET_ERR_LOCAL_CONTROL_DOORNO_CONFLICT	1913	与已添加的就地控制器门编号冲突
NET_ERR_LOCAL_CONTROL_COMMUNICATION_FAIL	1914	就地控制器通信失败
NET_ERR_OPERAND_INEXISTENCE	1915	操作对象不存在（对门、报警输出、报警输入相关操作，当对象未添加时返回）
NET_ERR_LOCAL_CONTROL_OVER_LIMIT	1916	就地控制器超出设备最大能力（主控对就地数量有限制）
NET_ERR_DOOR_OVER_LIMIT	1917	门超出设备最大能力
NET_ERR_ALARM_OVER_LIMIT	1918	报警输入输出超出设备最大能力
NET_ERR_LOCAL_CONTROL_ADDRESS_INCONFORMITY_TY PE	1919	就地控制器地址与类型不符
NET_ERR_NOT_SUPPORT_ONE_MORE_CARD	1920	不支持一人多卡
NET_ERR_DELETE_NO_EXISTENCE_FACE	1921	删除的人脸不存在

## 7.2 RTSP 通讯库错误码

错误名称	错误值	说明
NET_DVR_RTSP_GETPORTFAILED	407	获取 RTSP 端口错误
NET_DVR_RTSP_DESCRIBESENDTIMEOUT	411	RTSP DESCRIBE 发送超时
NET_DVR_RTSP_DESCRIBESENDERERROR	412	RTSP DESCRIBE 发送失败
NET_DVR_RTSP_DESCRIBERECVTIMEOUT	413	RTSP DESCRIBE 接收超时
NET_DVR_RTSP_DESCRIBERECDATALOST	414	RTSP DESCRIBE 接收数据错误
NET_DVR_RTSP_DESCRIBERECEVERROR	415	RTSP DESCRIBE 接收失败
NET_DVR_RTSP_DESCRIBESERVERERR	416	RTSP DESCRIBE 服务器返回 401,501 等错误
NET_DVR_RTSP_SETUPSENDTIMEOUT	421	RTSP SETUP 发送超时
NET_DVR_RTSP_SETUPSENDERERROR	422	RTSP SETUP 发送错误
NET_DVR_RTSP_SETUPRECVTIMEOUT	423	RTSP SETUP 接收超时
NET_DVR_RTSP_SETUPRECVDATALOST	424	RTSP SETUP 接收数据错误
NET_DVR_RTSP_SETUPRECEVERROR	425	RTSP SETUP 接收失败
NET_DVR_RTSP_OVER_MAX_CHAN	426	设备超过最大连接数
NET_DVR_RTSP_PLAYSENDTIMEOUT	431	RTSP PLAY 发送超时
NET_DVR_RTSP_PLAYSENDERERROR	432	RTSP PLAY 发送错误
NET_DVR_RTSP_PLAYRECVTIMEOUT	433	RTSP PLAY 接收超时
NET_DVR_RTSP_PLAYRECVDATALOST	434	RTSP PLAY 接收数据错误
NET_DVR_RTSP_PLAYRECEVERROR	435	RTSP PLAY 接收失败

NET_DVR_RTSP_PLAYSERVERERR	436	RTSP PLAY 设备返回错误状态
NET_DVR_RTSP_TEARDOWNSENDTIMEOUT	441	RTSP TEARDOWN 发送超时
NET_DVR_RTSP_TEARDOWNSENDERERROR	442	RTSP TEARDOWN 发送错误
NET_DVR_RTSP_TEARDOWNRECVTIMEOUT	443	RTSP TEARDOWN 接收超时
NET_DVR_RTSP_TEARDOWNRECVDATALOST	444	RTSP TEARDOWN 接收数据错误
NET_DVR_RTSP_TEARDOWNRECVERERROR	445	RTSP TEARDOWN 接收失败
NET_DVR_RTSP_TEARDOWNSEVERERR	446	RTSP TEARDOWN 设备返回错误状态

## 7.3 软解码库错误码

错误名称	错误值	说明
NET_PLAYM4_NOERROR	500	没有错误
NET_PLAYM4_PARA_OVER	501	输入参数非法
NET_PLAYM4_ORDER_ERROR	502	调用顺序不对
NET_PLAYM4_TIMER_ERROR	503	多媒体时钟设置失败
NET_PLAYM4_DEC_VIDEO_ERROR	504	视频解码失败
NET_PLAYM4_DEC_AUDIO_ERROR	505	音频解码失败
NET_PLAYM4_ALLOC_MEMORY_ERROR	506	分配内存失败
NET_PLAYM4_OPEN_FILE_ERROR	507	文件操作失败
NET_PLAYM4_CREATE_OBJ_ERROR	508	创建线程事件等失败
NET_PLAYM4_CREATE_DDRAW_ERROR	509	创建 directDraw 失败
NET_PLAYM4_CREATE_OFFSCREEN_ERROR	510	创建后端缓存失败
NET_PLAYM4_BUF_OVER	511	缓冲区满，输入流失败
NET_PLAYM4_CREATE_SOUND_ERROR	512	创建音频设备失败
NET_PLAYM4_SET_VOLUME_ERROR	513	设置音量失败
NET_PLAYM4_SUPPORT_FILE_ONLY	514	只能在播放文件时才能使用此接口
NET_PLAYM4_SUPPORT_STREAM_ONLY	515	只能在播放流时才能使用此接口
NET_PLAYM4_SYS_NOT_SUPPORT	516	系统不支持，解码器只能工作在 Pentium 3 以上
NET_PLAYM4_FILEHEADER_UNKNOWN	517	没有文件头
NET_PLAYM4_VERSION_INCORRECT	518	解码器和编码器版本不对应
NET_PLAYM4_INIT_DECODER_ERROR	519	初始化解码器失败
NET_PLAYM4_CHECK_FILE_ERROR	520	文件太短或码流无法识别
NET_PLAYM4_INIT_TIMER_ERROR	521	初始化多媒体时钟失败
NET_PLAYM4_BLT_ERROR	522	位拷贝失败
NET_PLAYM4_UPDATE_ERROR	523	显示 overlay 失败
NET_PLAYM4_OPEN_FILE_ERROR_MULTI	524	打开混合流文件失败
NET_PLAYM4_OPEN_FILE_ERROR_VIDEO	525	打开视频流文件失败
NET_PLAYM4_JPEG_COMPRESS_ERROR	526	JPEG 压缩错误

---

NET_PLAYM4_EXTRACT_NOT_SUPPORT	527	不支持该文件版本.
NET_PLAYM4_EXTRACT_DATA_ERROR	528	提取文件数据失败

---

## 8 附录:结构体

### 8.1 宏定义

宏定义	宏定义值	含义
MAX_NAMELEN	16	设备本地登录名长度
MAX_RIGHT	32	设备支持的权限（1-12 表示本地权限，13-32 表示远程权限）
NAME_LEN	32	用户名长度
PASSWD_LEN	16	密码长度
SERIALNO_LEN	48	序列号长度
MACADDR_LEN	6	MAC 地址长度
MAX_LINK	6	最大视频流连接数
MAX_ANALOG_CHANNUM	32	最大 32 个模拟通道
MAX_ANALOG_ALARMOUT	32	最大 32 路模拟报警输出
MAX_ANALOG_ALARMIN	32	最大 32 路模拟报警输入
MAX_IP_CHANNEL	32	允许加入的最多 IP 通道数
MAX_IP_ALARMIN	128	允许加入的最多报警输入数
MAX_IP_ALARMOUT	64	允许加入的最多报警输出数
MAX_CHANNUM_V30	64	(MAX_ANALOG_CHANNUM + MAX_IP_CHANNEL)
MAX_ALARMOUT_V30	96	(MAX_ANALOG_ALARMOUT + MAX_IP_ALARMOUT)
MAX_ALARMIN_V30	160	(MAX_ANALOG_ALARMIN + MAX_IP_ALARMIN)
MAX_PRESET_V30	256	支持的云台预置点数
MAX_TRACK_V30	256	支持的云台轨迹数
MAX_CRUISE_V30	256	支持的云台巡航数
MAX_SERIAL_PORT	8	支持 232 串口数
MAX_FORTIFY_NUM	10	最大布防个数
MAX_DRIVECHAN_NUM	16	最大车道数
MAX_COIL_NUM	3	最大线圈个数
MAX_ETHERNET	2	设备可配以太网网络
MAX_DOMAIN_NAME	64	最大域名长度
MAX_EXCEPTIONNUM_V30	32	设备的最大异常处理数
MAX_SHELTERNUM	4	V3.0 以下版本支持的设备的最大遮挡区域数
MAX_STRINGNUM_V30	8	最大 OSD 字符行数
PATHNAME_LEN	128	路径长度
PICNAME_MAXITEM	15	最大命名项数目
MAX_TIMESEGMENT_V30	8	最大时间段数



MAX_VIDEOOUT_V30	4	视频输出数
MAX_VGA_V30	4	最大可接 VGA 数
MAX_MATRIXOUT	16	最大模拟矩阵输出个数
MAX_DAYS	7	每周的天数
PHONENUMBER_LEN	32	PPPoE 拨号号码最大长度
MAX_DISKNUM_V30	33	最大硬盘数
MAX_NFS_DISK	8	最大 NFS 硬盘数
MAX_LICENSE_LEN	16	车牌号最大长度
MAX_SIGNALLIGHT_NUM	6	最大信号灯个数
MAX_LANERECT_NUM	5	最大车牌识别区域数
MAX_INTERVAL_NUM	4	最大时间间隔个数
MAX_CHJC_NUM	3	最大车辆省份简称字符个数
MAX_VL_NUM	5	最大虚拟线圈个数
MAX_IOSPEED_GROUP_NUM	4	IO 测速组个数
MAX_IOOUT_NUM	4	最大 IO 输出口个数
MAX_IOIN_NUMEX	10	最大 IO 输入口个数(扩展)
MAX_IOIN_NUM	8	最大 IO 输入口个数
MAX_ITC_LANE_NUM	6	最大车道个数
MAX_LIGHT_NUM	6	最大交通灯数
MAX_LANEAREA_NUM	2	单车道最大区域个数
MAX_USERNUM_V30	32	设备的最大用户数
ITC_MAX_POLYGON_POINT_NUM	20	检测区域最多支持 20 个点的多边形
MAX_ITC_SERIALCHECK_NUM	8	串口校验类型个数
MAX_VIDEO_DETECT_LIGHT_NUM	12	视频检测交通信号灯检测区域最大个数
MAX_CABINET_COUNT	8	最大机柜个数
ITS_MAX_DEVICE_NUM	32	最大设备个数
MAX_CUSTOMDIR_LEN	32	自定义目录长度
MAX_ICR_NUM	8	抓拍机红外滤光片预置点数
MAX_ITC_EXCEPTIONOUT	32	抓拍机最大报警输出个数

## 8.2 NET\_DVR\_ACS\_ALARM\_INFO:门禁主机报警信息结构体

```

struct{
    DWORD                dwSize;
    DWORD                dwMajor;
    DWORD                dwMinor;
    NET_DVR_TIME         struTime;
    BYTE                 sNetUser[MAX_NAMELEN];

```

```

NET_DVR_IPADDR      struRemoteHostAddr ;
NET_DVR_ACS_EVENT_INFO struAcsEventInfo;
DWORD               dwPicDataLen;
char                *pPicData;
BYTE                byRes[24];
}NET_DVR_ACS_ALARM_INFO,*LPNET_DVR_ACS_ALARM_INFO;

```

### Members

*dwSize*

结构体大小

*dwMajor*

报警主类型，具体定义见“Remarks”说明

*dwMinor*

报警次类型，次类型含义根据主类型不同而不同，具体定义见“Remarks”说明

*struTime*

报警时间

*sNetUser*

网络操作的用户名

*struRemoteHostAddr*

远程主机地址

*struAcsEventInfo*

报警信息详细参数

*dwPicDataLen*

图片数据大小，不为 0 是表示后面带数据

*pPicData*

图片数据缓冲区

*byRes*

保留，置为 0

### Remarks

- 门禁主机报警信息获取只能通过 [NET\\_DVR\\_SetDVRMessageCallBack\\_V31](#) 设置报警回调函数，回调函数有返回值，需要返回 TRUE 告知设备已经接收数据，设备才会上传下一条信息。

报警主类型定义如下所示：

宏定义	宏定义值	含义
MAJOR_ALARM	0x1	报警
MAJOR_EXCEPTION	0x2	异常
MAJOR_OPERATION	0x3	操作
MAJOR_EVENT	0x5	事件

根据不同的主类型的次类型定义如下所示：

主类型宏定义	宏定义值	含义
MAJOR_ALARM	0x1	报警
次类型宏定义	宏定义值	含义
MINOR_ALARM_IN_SHORT_CIRCUIT	0x400	防区短路报警

MINOR_ALARMIN_BROKEN_CIRCUIT	0x401	防区断路报警
MINOR_ALARMIN_EXCEPTION	0x402	防区异常报警
MINOR_ALARMIN_RESUME	0x403	防区报警恢复
MINOR_HOST_DESMANTLE_ALARM	0x404	防区防拆报警
MINOR_HOST_DESMANTLE_RESUME	0x405	防区防拆恢复
MINOR_CARD_READER_DESMANTLE_ALARM	0x406	读卡器防拆报警
MINOR_CARD_READER_DESMANTLE_RESUME	0x407	读卡器防拆恢复
MINOR_CASE_SENSOR_ALARM	0x408	事件输入报警
MINOR_CASE_SENSOR_RESUME	0x409	事件输入恢复
MINOR_STRESS_ALARM	0x40a	胁迫报警
MINOR_OFFLINE_EVENT_NEARLY_FULL	0x40b	离线事件满 90%报警
MINOR_CARD_MAX_AUTHENTICATE_FAIL	0x40c	卡号认证失败超次报警
MINOR_SD_CARD_FULL	0x40d	SD 卡存储满报警
MINOR_LINKAGE_CAPTURE_PICTURE	0x40e	联动抓拍事件报警
MINOR_SECURITY_MODULE_DESMANTLE_ALARM	0x40f	门控安全模块防拆报警
MINOR_SECURITY_MODULE_DESMANTLE_RESUME	0x410	门控安全模块防拆恢复
MINOR_POS_START_ALARM	0x411	POS 开启
MINOR_POS_END_ALARM	0x412	POS 结束
MINOR_FIRE_IMPORT_SHORT_CIRCUIT	0x415	消防输入短路报警
MINOR_FIRE_IMPORT_BROKEN_CIRCUIT	0x416	消防输入断路报警
MINOR_FIRE_IMPORT_RESUME	0x417	消防输入恢复

主类型宏定义	宏定义值	含义
MAJOR_EXCEPTION	0x2	异常
次类型宏定义	宏定义值	含义
MINOR_NET_BROKEN	0x27	网络断开
MINOR_RS485_DEVICE_ABNORMAL	0x3a	RS485 连接状态异常
MINOR_RS485_DEVICE_REVERT	0x3b	RS485 连接状态异常恢复
MINOR_DEV_POWER_ON	0x400	设备上电启动
MINOR_DEV_POWER_OFF	0x401	设备掉电关闭
MINOR_WATCH_DOG_RESET	0x402	看门狗复位
MINOR_LOW_BATTERY	0x403	蓄电池电压低
MINOR_BATTERY_RESUME	0x404	蓄电池电压恢复正常
MINOR_AC_OFF	0x405	交流电断电

MINOR_AC_RESUME	0x406	交流电恢复
MINOR_NET_RESUME	0x407	网络恢复
MINOR_FLASH_ABNORMAL	0x408	FLASH 读写异常
MINOR_CARD_READER_OFFLINE	0x409	读卡器掉线
MINOR_CARD_READER_RESUME	0x40a	读卡器掉线恢复
MINOR_INDICATOR_LIGHT_OFF	0x40b	指示灯关闭
MINOR_INDICATOR_LIGHT_RESUME	0x40c	指示灯恢复
MINOR_CHANNEL_CONTROLLER_OFF	0x40d	通道控制器掉线
MINOR_CHANNEL_CONTROLLER_RESUME	0x40e	通道控制器恢复
MINOR_SECURITY_MODULE_OFF	0x40f	门控安全模块掉线
MINOR_SECURITY_MODULE_RESUME	0x410	门控安全模块掉线恢复
MINOR_LOCAL_CONTROL_NET_BROKEN	0x413	就地控制器网络断开
MINOR_LOCAL_CONTROL_NET_RSUME	0x414	就地控制器网络恢复
MINOR_MASTER_RS485_LOOPNODE_BROKEN	0x415	主控 RS485 环路节点断开
MINOR_MASTER_RS485_LOOPNODE_RESUME	0x416	主控 RS485 环路节点恢复
MINOR_LOCAL_CONTROL_OFFLINE	0x417	就地控制器掉线
MINOR_LOCAL_CONTROL_RESUME	0x418	就地控制器掉线恢复
MINOR_LOCAL_DOWNSIDE_RS485_LOOPNODE_BROKEN	0x419	就地下行 RS485 环路断开
MINOR_LOCAL_DOWNSIDE_RS485_LOOPNODE_RESUME	0x41a	就地下行 RS485 环路恢复

主类型宏定义	宏定义值	含义
MAJOR_OPERATION	0x3	操作
次类型宏定义	宏定义值	含义
MINOR_LOCAL_UPGRADE	0x5a	本地升级
MINOR_REMOTE_LOGIN	0x70	远程登录
MINOR_REMOTE_LOGOUT	0x71	远程注销登陆
MINOR_REMOTE_ARM	0x79	远程布防
MINOR_REMOTE_DISARM	0x7a	远程撤防
MINOR_REMOTE_REBOOT	0x7b	远程重启
MINOR_REMOTE_UPGRADE	0x7e	远程升级
MINOR_REMOTE_CFGFILE_OUTPUT	0x86	远程导出配置文件
MINOR_REMOTE_CFGFILE_INTPUT	0x87	远程导入配置文件
MINOR_REMOTE_ALARMOUT_OPEN_MAN	0xd6	远程手动开启报警输出
MINOR_REMOTE_ALARMOUT_CLOSE_MAN	0xd7	远程手动关闭报警输出

MINOR_REMOTE_OPEN_DOOR	0x400	远程开门
MINOR_REMOTE_CLOSE_DOOR	0x401	远程关门
MINOR_REMOTE_ALWAYS_OPEN	0x402	远程常开
MINOR_REMOTE_ALWAYS_CLOSE	0x403	远程常关
MINOR_REMOTE_CHECK_TIME	0x404	远程手动校时
MINOR_NTP_CHECK_TIME	0x405	NTP 自动校时
MINOR_REMOTE_CLEAR_CARD	0x406	远程清空卡号
MINOR_REMOTE_RESTORE_CFG	0x407	远程恢复默认参数
MINOR_ALARMIN_ARM	0x408	防区布防
MINOR_ALARMIN_DISARM	0x409	防区撤防
MINOR_LOCAL_RESTORE_CFG	0x40a	本地恢复默认参数
MINOR_REMOTE_CAPTURE_PIC	0x40b	远程抓拍
MINOR_MOD_NET_REPORT_CFG	0x40c	修改网络中心参数配置
MINOR_MOD_GPRS_REPORT_PARAM	0x40d	修改 GPRS 中心参数配置
MINOR_MOD_REPORT_GROUP_PARAM	0x40e	修改中心组参数配置
MINOR_UNLOCK_PASSWORD_OPEN_DOOR	0x40f	解除码输入
MINOR_AUTO_RENUMBER	0x410	自动重新编号
MINOR_AUTO_COMPLEMENT_NUMBER	0x411	自动补充编号
MINOR_NORMAL_CFGFILE_INPUT	0x412	导入普通配置文件
MINOR_NORMAL_CFGFILE_OUTTPUT	0x413	导出普通配置文件
MINOR_CARD_RIGHT_INPUT	0x414	导入卡权限参数
MINOR_CARD_RIGHT_OUTTPUT	0x415	导出卡权限参数
MINOR_LOCAL_USB_UPGRADE	0x416	本地 U 盘升级

主类型宏定义	宏定义值	含义
MAJOR_EVENT	0x5	事件
次类型宏定义	宏定义值	含义
MINOR_LEGAL_CARD_PASS	0x01	合法卡认证通过
MINOR_CARD_AND_PSW_PASS	0x02	刷卡加密码认证通过
MINOR_CARD_AND_PSW_FAIL	0x03	刷卡加密码认证失败
MINOR_CARD_AND_PSW_TIMEOUT	0x04	刷卡加密码认证超时
MINOR_CARD_AND_PSW_OVER_TIME	0x05	刷卡加密码超次
MINOR_CARD_NO_RIGHT	0x06	未分配权限
MINOR_CARD_INVALID_PERIOD	0x07	无效时段
MINOR_CARD_OUT_OF_DATE	0x08	卡号过期

MINOR_INVALID_CARD	0x09	无此卡号
MINOR_ANTI_SNEAK_FAIL	0x0a	反潜回认证失败
MINOR_INTERLOCK_DOOR_NOT_CLOSE	0x0b	互锁门未关闭
MINOR_NOT_BELONG_MULTI_GROUP	0x0c	卡不属于多重认证群组
MINOR_INVALID_MULTI_VERIFY_PERIOD	0x0d	卡不在多重认证时间段内
MINOR_MULTI_VERIFY_SUPER_RIGHT_FAIL	0x0e	多重认证模式超级权限认证失败
MINOR_MULTI_VERIFY_REMOTE_RIGHT_FAIL	0x0f	多重认证模式远程认证失败
MINOR_MULTI_VERIFY_SUCCESS	0x10	多重认证成功
MINOR_LEADER_CARD_OPEN_BEGIN	0x11	首卡开门开始
MINOR_LEADER_CARD_OPEN_END	0x12	首卡开门结束
MINOR_ALWAYS_OPEN_BEGIN	0x13	常开状态开始
MINOR_ALWAYS_OPEN_END	0x14	常开状态结束
MINOR_LOCK_OPEN	0x15	门锁打开
MINOR_LOCK_CLOSE	0x16	门锁关闭
MINOR_DOOR_BUTTON_PRESS	0x17	开门按钮打开
MINOR_DOOR_BUTTON_RELEASE	0x18	开门按钮放开
MINOR_DOOR_OPEN_NORMAL	0x19	正常开门（门磁）
MINOR_DOOR_CLOSE_NORMAL	0x1a	正常关门（门磁）
MINOR_DOOR_OPEN_ABNORMAL	0x1b	门异常打开（门磁）
MINOR_DOOR_OPEN_TIMEOUT	0x1c	门打开超时（门磁）
MINOR_ALARMOUT_ON	0x1d	报警输出打开
MINOR_ALARMOUT_OFF	0x1e	报警输出关闭
MINOR_ALWAYS_CLOSE_BEGIN	0x1f	常关状态开始
MINOR_ALWAYS_CLOSE_END	0x20	常关状态结束
MINOR_MULTI_VERIFY_NEED_REMOTE_OPEN	0x21	多重多重认证需要远程开门
MINOR_MULTI_VERIFY_SUPERPASSWD_VERIFY_SUCCESS	0x22	多重认证超级密码认证成功事件
MINOR_MULTI_VERIFY_REPEAT_VERIFY	0x23	多重认证重复认证事件
MINOR_MULTI_VERIFY_TIMEOUT	0x24	多重认证重复认证事件
MINOR_DOORBELL_RINGING	0x25	门铃响
MINOR_FINGERPRINT_COMPARE_PASS	0x26	指纹比对通过
MINOR_FINGERPRINT_COMPARE_FAIL	0x27	指纹比对失败
MINOR_CARD_FINGERPRINT_VERIFY_PASS	0x28	刷卡加指纹认证通过
MINOR_CARD_FINGERPRINT_VERIFY_FAIL	0x29	刷卡加指纹认证失败
MINOR_CARD_FINGERPRINT_VERIFY_TIMEOUT	0x2a	刷卡加指纹认证超时

MINOR_CARD_FINGERPRINT_PASSWD_VERIFY_PASS	0x2b	刷卡加指纹加密码认证通过
MINOR_CARD_FINGERPRINT_PASSWD_VERIFY_FAIL	0x2c	刷卡加指纹加密码认证失败
MINOR_CARD_FINGERPRINT_PASSWD_VERIFY_TIMEOUT	0x2d	刷卡加指纹加密码认证超时
MINOR_FINGERPRINT_PASSWD_VERIFY_PASS	0x2e	指纹加密码认证通过
MINOR_FINGERPRINT_PASSWD_VERIFY_FAIL	0x2f	指纹加密码认证失败
MINOR_FINGERPRINT_PASSWD_VERIFY_TIMEOUT	0x30	指纹加密码认证超时
MINOR_FINGERPRINT_INEXISTENCE	0x31	指纹不存在
MINOR_CARD_PLATFORM_VERIFY	0x32	刷卡平台认证
MINOR_CALL_CENTER	0x33	呼叫中心事件
MINOR_FIRE_RELAY_TURN_ON_DOOR_ALWAYS_OPEN	0x34	消防继电器导通触发门常开
MINOR_FIRE_RELAY_RECOVER_DOOR_RECOVER_NORMAL	0x35	消防继电器恢复门恢复正常
MINOR_FIRSTCARD_AUTHORIZE_BEGIN	0x51	首卡授权开始
MINOR_FIRSTCARD_AUTHORIZE_END	0x52	首卡授权结束
MINOR_DOORLOCK_INPUT_SHORT_CIRCUIT	0x53	门锁输入短路报警
MINOR_DOORLOCK_INPUT_BROKEN_CIRCUIT	0x54	门锁输入断路报警
MINOR_DOORLOCK_INPUT_EXCEPTION	0x55	门锁输入异常报警
MINOR_DOORCONTACT_INPUT_SHORT_CIRCUIT	0x56	门磁输入短路报警
MINOR_DOORCONTACT_INPUT_BROKEN_CIRCUIT	0x57	门磁输入断路报警
MINOR_DOORCONTACT_INPUT_EXCEPTION	0x58	门磁输入异常报警
MINOR_OPENBUTTON_INPUT_SHORT_CIRCUIT	0x59	开门按钮输入短路报警
MINOR_OPENBUTTON_INPUT_BROKEN_CIRCUIT	0x5a	开门按钮输入断路报警
MINOR_OPENBUTTON_INPUT_EXCEPTION	0x5b	开门按钮输入异常报警
MINOR_DOORLOCK_OPEN_EXCEPTION	0x5c	门锁异常打开
MINOR_DOORLOCK_OPEN_TIMEOUT	0x5d	门锁打开超时
MINOR_FIRSTCARD_OPEN_WITHOUT_AUTHORIZE	0x5e	首卡未授权开门失败

### 8.3 NET\_DVR\_ACS\_CFG: 门禁主机参数配置结构体

```

struct{
    DWORD        dwSize;
    BYTE         byRS485Backup;
    BYTE         byShowCapPic;
    BYTE         byShowCardNo;
    BYTE         byShowUserInfo;

```

```

BYTE        byOverlayUserInfo;
BYTE        byVoicePrompt;
BYTE        byUploadCapPic;
BYTE        bySaveCapPic;
BYTE        byInputCardNo;
BYTE        byRes[503];
}NET_DVR_ACS_CFG, *LPNET_DVR_ACS_CFG;

```

### Members

*dwSize*

结构体大小

*byRS485Backup*

是否启用下行 RS485 通信备份功能：0- 不启用，1- 启用

*byShowCapPic*

是否显示抓拍图片：0- 不显示，1- 显示

*byShowCardNo*

是否显示卡号：0- 不显示，1- 显示

*byShowUserInfo*

是否显示用户信息：0- 不显示，1- 显示

*byOverlayUserInfo*

是否叠加用户信息：0- 不叠加，1- 叠加

*byVoicePrompt*

是否启用语音提示：0- 不启用，1- 启用

*byUploadCapPic*

联动抓拍是否上传图片：0- 不上传，1- 上传

*bySaveCapPic*

是否保存抓拍图片：0- 不保存，1- 保存

*byInputCardNo*

是否是否允许按键输入卡号：0- 不允许，1- 允许

*byRes*

保留，置为 0

### Remarks

- 设备是否支持门禁主机参数配置或者支持的参数能力，可以通过设备能力集进行判断，对应门禁主机能力集(**AcsAbility**)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：**ACS\_ABILITY**，节点：**<AcsHostCfg>**。

## 8.4 NET\_DVR\_ACS\_EVENT\_INFO:门禁主机事件信息

```

struct{
    DWORD        dwSize;
    BYTE        byCardNo[ACS_CARD_NO_LEN];
    BYTE        byCardType;
    BYTE        byWhiteListNo;
    BYTE        byReportChannel;
    BYTE        byCardReaderKind;
    DWORD        dwCardReaderNo;
    DWORD        dwDoorNo;
}

```



```

DWORD    dwVerifyNo;
DWORD    dwAlarmInNo;
DWORD    dwAlarmOutNo;
DWORD    dwCaseSensorNo;
DWORD    dwRs485No;
DWORD    dwMultiCardGroupNo;
WORD     wAccessChannel;
BYTE     byDeviceNo;
BYTE     byDistractControlNo;
DWORD    dwEmployeeNo;
WORD     wLocalControllerID;
BYTE     byInternetAccess;
BYTE     byType;
BYTE     byRes[20];
}NET_DVR_ACS_EVENT_INFO,*LPNET_DVR_ACS_EVENT_INFO;

```

### Members

*dwSize*

结构体大小

*byCardNo*

卡号

*byCardType*

卡类型：1- 普通卡，2- 残疾人卡，3- 黑名单卡，4- 巡更卡，5- 胁迫卡，6- 超级卡，7- 来宾卡，8- 解除卡，为 0 表示无效

*byWhiteListNo*

白名单单号，取值范围：1~8，0 表示无效

*byReportChannel*

报告上传通道：1- 布防上传，2- 中心组 1 上传，3- 中心组 2 上传，0 表示无效

*byCardReaderKind*

读卡器类型：0- 无效，1- IC 读卡器，2- 身份证读卡器，3- 二维码读卡器，4- 指纹头

*dwCardReaderNo*

读卡器编号，为 0 表示无效

*dwDoorNo*

门编号，为 0 表示无效

*dwVerifyNo*

多重卡认证序号，为 0 表示无效

*dwAlarmInNo*

报警输入号，为 0 表示无效

*dwAlarmOutNo*

报警输出号，为 0 表示无效

*dwCaseSensorNo*

事件报警输入编号

*dwRs485No*

RS485 通道号，为 0 表示无效

*dwMultiCardGroupNo*

群组编号

*wAccessChannel*

人员通道号

*byDeviceNo*

设备编号，为 0 无效

*byDistractControlNo*

分控器编号，为 0 无效

*dwEmployeeNo*

工号，为 0 无效

*wLocalControllerID*

就地控制器编号，0-门禁主机，1-64 代表就地控制器

*byInternetAccess*

网口 ID: (1-上行网口 1,2-上行网口 2,3-下行网口 1)

*byType*

防区类型，0-即时防区，1-24 小时防区，2-延时防区，3-内部防区，4-钥匙防区，5-火警防区，6-周界防区，7-24 小时无声防区，8-24 小时辅助防区，9-24 小时震动防区，10-门禁紧急开门防区，11-门禁紧急关门防区，0xff-无

*byRes*

保留，置为 0

## 8.5 NET\_DVR\_ACS\_EXTERNAL\_DEV\_CFG:门禁主机串口外设参数配置结构体

```
struct{
    DWORD        dwSize;
    BYTE          byIDCardUpMode;
    BYTE          byRes1;
    BYTE          byCardVerifyMode;
    BYTE          byACSDevType;
    BYTE          byDoorMode;
    BYTE          byRes2;
    WORD          wDevDetailType;
    BYTE          byRes[300];
}NET_DVR_ACS_EXTERNAL_DEV_CFG,*LPNET_DVR_ACS_EXTERNAL_DEV_CFG;
```

### Members

*dwSize*

结构体大小

*byIDCardUpMode*

身份证信息上报模式：0- 上传 18 位身份证号，1- 上传全部信息

*byRes1*

保留，置为 0

*byCardVerifyMode*

刷卡认证模式：0- 远程中心认证，1- 客户端平台认证

*byACSDevType*

设备型号：1- 身份证读卡器，2- IC 读卡器，3- 二维码读卡器，4- 指纹读卡器，5- 字符屏+二维码读卡器，6- 收卡器，7- 字符屏，8- 指纹头，9- 语音模块

*byDoorMode*

门出入类型：0- 进门，1- 出门

*byRes2*

保留，置为 0

*wDevDetailType*

外设详细设备型号

当 *byACSDevType* 为 1 时：1- iDR210，2- IDM10

当 *byACSDevType* 为 7 时：0- DC48270RS043\_01T，1- DC80480B070\_03T

*byRes*

保留，置为 0

### Remarks

- 设备是否支持串口外设参数配置或者支持的参数能力，可以通过设备能力集进行判断，对应门禁主机能力集(*AcsAbility*)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：ACS\_ABILITY，节点：*<ExternalDevCfg>*。

## 8.6 NET\_DVR\_ACS\_PARAM\_TYPE:门禁主机参数结构体

struct{

DWORD dwSize;

DWORD dwParamType;

WORD wLocalControllerID;

BYTE byRes[30];

}NET\_DVR\_ACS\_PARAM\_TYPE,\*LPNET\_DVR\_ACS\_PARAM\_TYPE;

### Members

*dwSize*

结构体大小

*dwParamType*

参数类型，按位表示，每位代表一种参数，值：0- 不处理，1- 处理

宏定义	宏定义值	含义
ACS_PARAM_DOOR_STATUS_WEEK_PLAN	0x00000001	门状态周计划参数
ACS_PARAM_VERIFY_WEEK_PALN	0x00000002	读卡器周计划参数
ACS_PARAM_CARD_RIGHT_WEEK_PLAN	0x00000004	卡权限周计划参数
ACS_PARAM_DOOR_STATUS_HOLIDAY_PLAN	0x00000008	门状态假日计划参数
ACS_PARAM_VERIFY_HOLIDAY_PALN	0x00000010	读卡器假日计划参数
ACS_PARAM_CARD_RIGHT_HOLIDAY_PLAN	0x00000020	卡权限假日计划参数
ACS_PARAM_DOOR_STATUS_HOLIDAY_GROUP	0x00000040	门状态假日组参数
ACS_PARAM_VERIFY_HOLIDAY_GROUP	0x00000080	读卡器验证方式假日组参数
ACS_PARAM_CARD_RIGHT_HOLIDAY_GROUP	0x00000100	卡权限假日组参数
ACS_PARAM_DOOR_STATUS_PLAN_TEMPLATE	0x00000200	门状态计划模板参数
ACS_PARAM_VERIFY_PALN_TEMPLATE	0x00000400	读卡器验证方式计划模板参数

ACS_PARAM_CARD_RIGHT_PALN_TEMPLATE	0x00000800	卡权限计划模板参数
ACS_PARAM_CARD	0x00001000	卡参数
ACS_PARAM_GROUP	0x00002000	群组参数
ACS_PARAM_ANTI_SNEAK_CFG	0x00004000	反潜回参数
ACS_PAPAM_EVENT_CARD_LINKAGE	0x00008000	事件及卡号联动参数
ACS_PAPAM_CARD_PASSWD_CFG	0x00010000	密码开门使能参数
ACS_PARAM_PERSON_STATISTICS_CFG	0x00020000	人数统计参数

*wLocalControllerID*

就地控制器序号[1,255],0 代表门禁主机

*byRes*

保留，置为 0

#### Remarks

- 清空门禁主机参数控制能力，对应门禁主机能力集（接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：ACS\_ABILITY）中节点<clearAcsParam>。

## 8.7 NET\_DVR\_ACS\_SCREEN\_DISPLAY\_CFG: 屏幕字符串显示参数结构体

struct{

DWORD dwSize;  
 DWORD dwFontSize;  
 DWORD dwRowSpacing;  
 DWORD dwColumnSpacing;  
 DWORD dwFirstRowPosition;  
 BYTE byDegree;  
 BYTE byScreenType;  
 BYTE byRes[306];

}NET\_DVR\_ACS\_SCREEN\_DISPLAY\_CFG, \*LPNET\_DVR\_ACS\_SCREEN\_DISPLAY\_CFG;

#### Members

*dwSize*

结构体大小

*dwFontSize*

字体大小，取值范围：[1,10]

*dwRowSpacing*

行间距，单位：像素点

*dwColumnSpacing*

列间距，单位：像素点

*dwFirstRowPosition*

起始行位置在屏幕哪一个分块：0- 0，1- 1/8，2- 2/8，3- 3/8，4- 4/8，5- 5/8，6- 6/8，7- 7/8

*byDegree*

字符显示方向角度：0- 0 度（正常），1- 90 度（侧着）

*byScreenType*

屏幕类型：0- DC48270RS043\_01T，1- DC80480B070\_03T

*byRes*

保留，置为 0

#### Remarks

- 设备是否支持屏幕字符串显示配置或者支持的参数能力，可以通过设备能力集进行判断，对应门禁主机能力集(**AcsAbility**)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：ACS\_ABILITY，节点：<ScreenDisplayCfg>。

## 8.8 NET\_DVR\_ACS\_WORK\_STATUS:门禁主机工作状态结构体

```
struct{
    DWORD        dwSize;
    BYTE          byDoorLockStatus[MAX_DOOR_NUM];
    BYTE          byDoorStatus[MAX_DOOR_NUM];
    BYTE          byMagneticStatus[MAX_DOOR_NUM];
    BYTE          byCaseStatus[MAX_CASE_SENSOR_NUM];
    WORD          wBatteryVoltage;
    BYTE          byBatteryLowVoltage;
    BYTE          byPowerSupplyStatus;
    BYTE          byMultiDoorInterlockStatus;
    BYTE          byAntiSneakStatus;
    BYTE          byHostAntiDismantleStatus;
    BYTE          byIndicatorLightStatus;
    BYTE          byCardReaderOnlineStatus[MAX_CARD_READER_NUM];
    BYTE          byCardReaderAntiDismantleStatus[MAX_CARD_READER_NUM];
    BYTE          byCardReaderVerifyMode[MAX_CARD_READER_NUM];
    BYTE          bySetupAlarmStatus[MAX_ALARMHOST_ALARMIN_NUM];
    BYTE          byAlarmInStatus[MAX_ALARMHOST_ALARMIN_NUM];
    BYTE          byAlarmOutStatus[MAX_ALARMHOST_ALARMOUT_NUM];
    DWORD        dwCardNum;
    BYTE          byRes2[32];
}NET_DVR_ACS_WORK_STATUS,*LPNET_DVR_ACS_WORK_STATUS;
```

#### Members

*dwSize*

结构体大小

*byDoorLockStatus*

门锁状态：0- 关，1- 开

*byDoorStatus*

门状态：1- 休眠，2- 常开状态，3- 常闭状态，4- 普通状态

*byMagneticStatus*

门磁状态：0- 闭合，1- 开启

*byCaseStatus*

事件报警输入状态：0- 无输入，1- 有输入

*wBatteryVoltage*

蓄电池电压值，实际值乘 10，单位：伏特

*byBatteryLowVoltage*

蓄电池是否处于低压状态：0- 否，1- 是

*byPowerSupplyStatus*

设备供电状态：1- 交流电供电，2- 蓄电池供电

*byMultiDoorInterlockStatus*

多门互锁状态：0- 关闭，1- 开启

*byAntiSneakStatus*

反潜回状态：0-关闭，1-开启

*byHostAntiDismantleStatus*

主机防拆状态：0- 关闭，1- 开启

*byIndicatorLightStatus*

指示灯状态

*byCardReaderOnlineStatus*

读卡器在线状态：0- 不在线，1- 在线

*byCardReaderAntiDismantleStatus*

读卡器防拆状态：0- 关闭，1- 开启

*byCardReaderVerifyMode*

读卡器当前验证方式：0- 无效，1- 休眠，2- 刷卡+密码，3- 刷卡，4- 刷卡或密码，5- 指纹，6- 指纹加密码，7- 指纹或刷卡，8- 指纹加刷卡，9- 指纹加刷卡加密码

*bySetupAlarmStatus*

报警输入口布防状态：0- 对应报警输入口处于撤防状态，1- 对应报警输入口处于布防状态

*byAlarmInStatus*

报警输入口报警状态：0- 对应报警输入口当前无报警，1- 对应报警输入口当前有报警

*byAlarmOutStatus*

报警输出口状态：0- 对应报警输出口无报警，1- 对应报警输出口有报警

*dwCardNum*

已添加的卡数量

*byRes2*

保留，置为 0

## 8.9 NET\_DVR\_ACS\_WORK\_STATUS\_V50:门禁主机工作状态结构体

```
struct{
    DWORD        dwSize;
    BYTE          byDoorLockStatus[MAX_DOOR_NUM_256];
    BYTE          byDoorStatus[MAX_DOOR_NUM_256];
    BYTE          byMagneticStatus[MAX_DOOR_NUM_256];
    BYTE          byCaseStatus[MAX_CASE_SENSOR_NUM];
    WORD          wBatteryVoltage;
    BYTE          byBatteryLowVoltage;
    BYTE          byPowerSupplyStatus;
    BYTE          byMultiDoorInterlockStatus;
    BYTE          byAntiSneakStatus;
    BYTE          byHostAntiDismantleStatus;
```

```

BYTE        byIndicatorLightStatus;
BYTE        byCardReaderOnlineStatus[MAX_CARD_READER_NUM_512];
BYTE        byCardReaderAntiDismantleStatus[MAX_CARD_READER_NUM_512];
BYTE        byCardReaderVerifyMode[MAX_CARD_READER_NUM_512];
BYTE        bySetupAlarmStatus[MAX_ALARMHOST_ALARMIN_NUM];
BYTE        byAlarmInStatus[MAX_ALARMHOST_ALARMIN_NUM];
BYTE        byAlarmOutStatus[MAX_ALARMHOST_ALARMOUT_NUM];
DWORD       dwCardNum;
BYTE        byFireAlarmStatus;
BYTE        byRes2[123];

```

```
}NET_DVR_ACS_WORK_STATUS,*LPNET_DVR_ACS_WORK_STATUS;
```

## Members

*dwSize*

结构体大小

*byDoorLockStatus*

锁状态, 0-正常关, 1-正常开, 2-短路报警, 3-断路报警, 4-异常报警

*byDoorStatus*

门状态: 1- 休眠, 2- 常开状态, 3- 常闭状态, 4- 普通状态

*byMagneticStatus*

门磁状态, 0-正常关, 1-正常开, 2-短路报警, 3-断路报警, 4-异常报警

*byCaseStatus*

事件报警输入状态: 0- 无输入, 1- 有输入

*wBatteryVoltage*

蓄电池电压值, 实际值乘 10, 单位: 伏特

*byBatteryLowVoltage*

蓄电池是否处于低压状态: 0- 否, 1- 是

*byPowerSupplyStatus*

设备供电状态: 1- 交流电供电, 2- 蓄电池供电

*byMultiDoorInterlockStatus*

多门互锁状态: 0- 关闭, 1- 开启

*byAntiSneakStatus*

反潜回状态: 0-关闭, 1-开启

*byHostAntiDismantleStatus*

主机防拆状态: 0- 关闭, 1- 开启

*byIndicatorLightStatus*

指示灯状态, 0-掉线, 1-在线

*byCardReaderOnlineStatus*

读卡器在线状态: 0- 不在线, 1- 在线

*byCardReaderAntiDismantleStatus*

读卡器防拆状态: 0- 关闭, 1- 开启

*byCardReaderVerifyMode*

读卡器当前验证方式: 0- 无效, 1- 休眠, 2- 刷卡+密码, 3- 刷卡, 4- 刷卡或密码, 5- 指纹, 6- 指纹加密码, 7- 指纹或刷卡, 8- 指纹加刷卡, 9- 指纹加刷卡加密码

*bySetupAlarmStatus*

报警输入口布防状态：0- 对应报警输入口处于撤防状态，1- 对应报警输入口处于布防状态

*byAlarmInStatus*

按字节表示报警输入口状态：0- 对应报警输入口当前无报警，1- 对应报警输入口当前有报警

*byAlarmOutStatus*

按字节表示报警输出口状态：0- 对应报警输出口无报警，1- 对应报警输出口有报警

*dwCardNum*

已添加的卡数量

*byFireAlarmStatus*

消防报警状态显示：0-正常、1-短路报警、2-断开报警

*byRes2*

保留，置为 0

## 8.10 NET\_DVR\_ACTIVATECFG:设备激活参数

```
struct{
    DWORD    dwSize;
    BYTE      sPassword[PASSWD_LEN];
    BYTE      byRes[108];
}NET_DVR_ACTIVATECFG,*LPNET_DVR_ACTIVATECFG;
```

### Members

*dwSize*

结构体大小

*sPassword*

初始密码，密码等级弱或者以上

*byRes*

保留，置为 0

### Remarks

- 出厂设备需要先激活，然后再使用激活使用的初始密码登录设备。
- 将密码输入分为数字(0~9)、小写字母(a~z)、大写字母(A~Z)、特殊符号 (:\除外) 4 类，等级分为 4 个等级，如下所示：
  - 1) 等级 0 (风险密码)：密码长度小于 8 位，或者只包含 4 类字符中的任意一类，或者密码与用户名一样，或者密码是用户名的倒写。例如：12345、abcdef。
  - 2) 等级 1 (弱密码)：包含两类字符，且组合为 (数字+小写字母) 或 (数字+大写字母)，且长度大于等于 8 位。例如：abc12345、123ABCDEF
  - 3) 等级 2 (中密码)：包含两类字符，且组合不能为 (数字+小写字母) 和 (数字+大写字母)，且长度大于等于 8 位。例如：12345\*\*\*++、ABCDabcd。
  - 4) 等级 3 (强密码)：包含三类字符及以上，且长度大于等于 8 位。例如：Abc12345、abc12345++。

## 8.11 NET\_DVR\_ALARMER:报警设备信息

```
struct{
    BYTE      byUserIDValid;
    BYTE      bySerialValid;
    BYTE      byVersionValid;
    BYTE      byDeviceNameValid;
    BYTE      byMacAddrValid;
```



```

BYTE    byLinkPortValid;
BYTE    byDeviceIPValid;
BYTE    bySocketIPValid;
LONG    lUserID;
BYTE    sSerialNumber[SERIALNO\_LEN];
DWORD   dwDeviceVersion;
char    sDeviceName[NAME\_LEN];
BYTE    byMacAddr[MACADDR\_LEN];
WORD    wLinkPort;
char    sDeviceIP[128];
char    sSocketIP[128];
BYTE    byIpProtocol;
BYTE    byRes2[11];
}NET_DVR_ALARMER,*LPNET_DVR_ALARMER;

```

## Members

### *byUserIDValid*

userid 是否有效：0—无效；1—有效

### *bySerialValid*

序列号是否有效：0—无效；1—有效

### *byVersionValid*

版本号是否有效：0—无效；1—有效

### *byDeviceNameValid*

设备名字是否有效：0—无效；1—有效

### *byMacAddrValid*

MAC 地址是否有效：0—无效；1—有效

### *byLinkPortValid*

Login 端口是否有效：0—无效；1—有效

### *byDeviceIPValid*

设备 IP 是否有效：0—无效；1—有效

### *bySocketIPValid*

Socket IP 是否有效：0-无效；1-有效

### *lUserID*

NET\_DVR\_Login 或 NET\_DVR\_Login\_V40 的返回值，布防时有效

### *sSerialNumber*

序列号

### *dwDeviceVersion*

版本信息：V3.0 以上版本支持的设备最高 8 位为主版本号，次高 8 位为次版本号，低 16 位为修复版本号；V3.0 以下版本支持的设备高 16 位表示主版本，低 16 位表示次版本

### *sDeviceName*

设备名称

### *byMacAddr*

MAC 地址

### *wLinkPort*

设备通讯端口

*sDeviceIP*

设备 IP 地址

*sSocketIP*

报警主动上传时的 Socket IP 地址

*byIpProtocol*

IP 协议: 0- IPV4, 1- IPV6

*byRes2*

保留, 置为 0

## 8.12 NET\_DVR\_ALARM\_DEVICE\_USER:报警主机设备用户配置结构体

```
struct{
    DWORD                dwSize;
    BYTE                 sUserName[NAME_LEN];
    BYTE                 sPassword[PASSWD_LEN];
    NET_DVR_IPADDR       struUserIP;
    BYTE                 byMACAddr[MACADDR_LEN];
    BYTE                 byUserType;
    BYTE                 byAlarmOnRight;
    BYTE                 byAlarmOffRight;
    BYTE                 byBypassRight;
    BYTE                 byOtherRight[MAX_RIGHT];
    BYTE                 byNetPreviewRight[MAX_ALARMHOST_VIDEO_CHAN];
    BYTE                 byNetRecordRight[MAX_ALARMHOST_VIDEO_CHAN];
    BYTE                 byNetPlaybackRight[MAX_ALARMHOST_VIDEO_CHAN];
    BYTE                 byNetPTZRight[MAX_ALARMHOST_VIDEO_CHAN];
    BYTE                 byRes2[168];
}NET_DVR_ALARM_DEVICE_USER,*LPNET_DVR_ALARM_DEVICE_USER;
```

### Members

*dwSize*

结构体大小

*sUserName*

用户名

*sPassword*

密码

*struUserIP*

用户 IP 地址(为 0 时表示允许任何地址)

*byMACAddr*

物理地址

*byUserType*

用户类型: 0- 普通用户, 1- 管理员用户

*byAlarmOnRight*

布防权限

*byAlarmOffRight*

撤防权限

*byBypassRight*

旁路权限

*byOtherRight*

其他权限，参数取值为 1 表示使能：

*byOtherRight*[0]: 日志权限*byOtherRight*[1]: 重启关机*byOtherRight*[2]: 参数设置权限*byOtherRight*[3]: 参数获取权限*byOtherRight*[4]: 恢复默认参数权限*byOtherRight*[5]: 警号输出权限*byOtherRight*[6]: PTZ 控制权限*byOtherRight*[7]: 远程升级权限*byOtherRight*[8]: 报警输出控制*byOtherRight*[9]: 串口控制*byOtherRight*[10]: 门禁控制*byOtherRight*[11]: 语音对讲*byOtherRight*[12]: 远程控制本地输出*byOtherRight*[13]: 硬盘配置*byOtherRight*[14]: 格式化硬盘*byOtherRight*[15]: 模拟量控制*byNetPreviewRight*远程可以预览的通道，按位表示各通道（*bit*0: 通道 1，*bit*1: 通道 2，依次类推）：1- 有权限，0- 无权限*byNetRecordRight*远程可以录像的通道，按位表示各通道（*bit*0: 通道 1，*bit*1: 通道 2，依次类推）：1- 有权限，0- 无权限*byNetPlaybackRight*远程可以回放的通道，按位表示各通道（*bit*0: 通道 1，*bit*1: 通道 2，依次类推）：1- 有权限，0- 无权限*byNetPTZRight*远程可以云台控制的通道，按位表示各通道（*bit*0: 通道 1，*bit*1: 通道 2，依次类推）：1- 有权限，0- 无权限*byRes2*

保留

## 8.13 NET\_DVR\_ALARMIN\_PARAM:防区参数结构体

struct{

DWORD	<i>dwSize</i> ;
BYTE	<i>byName</i> [ <i>NAME_LEN</i> ];
WORD	<i>wDetectorType</i> ;
BYTE	<i>byType</i> ;
BYTE	<i>byUploadAlarmRecoveryReport</i> ;
DWORD	<i>dwParam</i> ;
<a href="#">NET_DVR_SCHEDTIME</a>	<i>struAlarmTime</i> [ <i>MAX_DAYS</i> ][ <i>MAX_TIMESEGMENT</i> ];

```

BYTE                byAssociateAlarmOut[MAX_ALARMHOST_ALARMOUT_NUM];
BYTE                byAssociateSirenOut[8];
BYTE                bySensitivityParam;
BYTE                byArrayBypass;
BYTE                byJointSubSystem;
BYTE                byModuleStatus;
WORD                wModuleAddress;
BYTE                byModuleChan;
BYTE                byModuleType;
WORD                wZoneIndex;
WORD                wInDelay;
WORD                wOutDelay;
BYTE                byAlarmType;
BYTE                byZoneResistor;
float               fZoneResistorManual;
BYTE                byRes2[32];
}NET_DVR_ALARMIN_PARAM, *LPNET_DVR_ALARMIN_PARAM;

```

### Members

*dwSize*

结构体大小

*byName*

防区名称

*wDetectorType*

防区探测器类型，具体定义如下：

```

enum _DETECTOR_TYPE_{
    PANIC_BUTTON = 0,
    MAGNETIC_CONTACT,
    SMOKE_DETECTOR,
    ACTIVE_INFRARED_DETECTOR,
    PASSIVE_INFRARED_DETECTOR,
    GLASS_BREAK_DETECTOR,
    VIBRATION_DETECTOR,
    DUAL_TECHNOLOGY_PIR_DETECTOR,
    TRIPLE_TECHNOLOGY_PIR_DETECTOR,
    HUMIDITY_DETECTOR,
    TEMPERATURE_DETECTOR,
    COMBUSTIBLE_GAS_DETECTOR,
    DYNAMIC_SWITCH,
    CONTROL_SWITCH,
    OTHER_DETECTOR = 0xffff
}DETECTOR_TYPE

```

*PANIC\_BUTTON*

紧急开关

<i>MAGNETIC_CONTACT</i>	门磁开关
<i>SMOKE_DETECTOR</i>	烟感探测器
<i>ACTIVE_INFRARED_DETECTOR</i>	主动红外探测器
<i>PASSIVE_INFRARED_DETECTOR</i>	被动红外探测器
<i>GLASS_BREAK_DETECTOR</i>	玻璃破碎探测器
<i>VIBRATION_DETECTOR</i>	震动探测器
<i>DUAL_TECHNOLOGY_PIR_DETECTOR</i>	双鉴移动探测器
<i>TRIPLE_TECHNOLOGY_PIR_DETECTOR</i>	三技术探测器
<i>HUMIDITY_DETECTOR</i>	湿度探测器
<i>TEMPERATURE_DETECTOR</i>	温感探测器
<i>COMBUSTIBLE_GAS_DETECTOR</i>	可燃气体探测器
<i>DYNAMIC_SWITCH</i>	随动开关
<i>CONTROL_SWITCH</i>	控制开关
<i>OTHER_DETECTOR</i>	其他探测器

**byType**

防区报警类型，0- 即时防区，1- 24 小时有声防区，2- 延时防区，3- 内部防区，4- 钥匙布撤防防区，5- 火警防区，6- 周界防区，7- 24 小时无声防区，8- 24 小时辅助防区，9- 24 小时震动防区，10-门禁紧急开门防区，11-门禁紧急关门防区，0xff- 无

**byUploadAlarmRecoveryReport**

是否上传防区报警恢复报告：0-不上传，1-上传

**dwParam**

防区参数，延时防区延时多长时间。通过能力集 `NET_DVR_ALARMHOST_ABILITY` 中的 `bySupportAlarmInDelay` 字段来判断是否通过该参数设置。

`bySupportAlarmInDelay` 为 1 时，表示客户端应该使用 `dwParam` 来设置延时时间，动环监控主机和自助行报警主机使用这种配置延时方式；

如果 `bySupportAlarmInDelay` 为 0 时，表示通过 `NET_DVR_ALARMSUBSYSTEMPARAM` 中的 `wEnterDelay`、`wExitDelay` 来设置延时时间。

**struAlarmTime**

布防时间时间段

*byAssociateAlarmOut*

防区关联报警输出

*byAssociateSirenOut*

关联警号输出，数组 0 表示警号 1，数组 1 表示警号 2，以此类推。值为 1 表示输出，0 表示不输出

*bySensitivityParam*

防区灵敏度参数：0- 10ms，1- 250ms，2- 500ms，3- 750ms

*byArrayBypass*

是否加入旁路组：0- 不支持组旁路，1- 支持组旁路

*byJointSubSystem*

防区所属的子系统号，该参数只能获取

*byModuleStatus*

外接防区模块状态：1- 在线，2- 离线，该参数只能获取

*wModuleAddress*

模块地址，扩展模块从 0~255，0xFFFF 表示无效，该参数只能获取

*byModuleChan*

模块通道号，从 1 开始，最大值根据模块类型来决定，0xFF 表示无效，该参数只能获取

*byModuleType*

模块类型：1- 本地防区，2- 单防区，3- 双防区，4- 8 防区，5- 8 路模拟量防区，6- 单防区触发器，7-1 门就地控制器，8-2 门就地控制器，9-4 门就地控制器

*wZoneIndex*

防区号，该参数只能获取

*wInDelay*

进入延时，取值范围：0~255 秒

*wOutDelay*

退出延时，取值范围：0~255 秒

*byAlarmType*

报警器类型：0- 无效，1- 常开，2- 常闭

*byZoneResistor*

防区电阻，单位：千欧，取值：0- 无效，1- 2.2，2- 3.3，3- 4.7，4- 5.6，5- 8.2，0xff- 自定义

*fZoneResistorManual*

防区手动电阻，取值范围：1.0~10.0，精确到小数点后一位，单位：千欧，byZoneResistor 为 0xff 时有效

*byRes2*

保留

#### Remarks

- 如果能力集 NET\_DVR\_ALARMHOST\_ABILITY 中的 wExpandAlarmInNum 大于 0，则表示支持防区编号功能，此时，该结构体的参数 byJointSubSystem、byModuleStatus、wModuleAddress、byModuleChan、byModuleType 以及 wZoomZoneIndex 有效，支持“获取子系统内防区”、“自动搜索外接模块”和“自动注册外接模块”功能。
- 下列情况下防区参数不能被修改：(1)防区处于布防状态，(2)防区所在子系统处于布防状态下，(3)主机处于编程模式，(4)主机处于步测模式。

## 8.14 NET\_DVR\_ALARMIN\_SETUP:布防参数结构体

```
struct{
```

```

    BYTE    byAssociateAlarmIn[MAX\_ALARMHOST\_ALARMIN\_NUM];
    BYTE    byRes[100];
}NET_DVR_ALARMIN_SETUP, *LPNET_DVR_ALARMIN_SETUP;

```

### Members

#### *byAssociateAlarmIn*

防区通道，数组下标 0 对应防区 1，依次类推，例如：byAssociateAlarmIn[i]=1 表示对防区 i+1 进行布防

#### *byRes*

保留

## 8.15 NET\_DVR\_ALARMOUT\_PARAM:触发器参数结构体

```

struct{
    DWORD    dwSize;
    BYTE     byName[NAME\_LEN];
    WORD     wDelay;
    WORD     wTriggerIndex;
    BYTE     byAssociateAlarmIn[MAX\_ALARMHOST\_ALARMIN\_NUM];
    BYTE     byModuleType;
    BYTE     byModuleStatus;
    WORD     wModuleAddress;
    BYTE     byModuleChan;
    BYTE     byWorkMode;
    BYTE     byAlarmOutMode;
    BYTE     byTimeOn;
    BYTE     byTimeOff;
    BYTE     byRes2[51];
}NET_DVR_ALARMOUT_PARAM, *LPNET_DVR_ALARMOUT_PARAM;

```

### Members

#### *dwSize*

结构体大小

#### *byName*

触发器名称

#### *wDelay*

输出延迟，单位：秒。其取值范围如下：

网络报警主机、总线式网络报警主机、视频报警主机：0~5999，0 表示有报警时候输出而无报警时候关闭

动环报警主机：0~65535，0 表示一直输出

自助行报警主机：0~5999，0 表示一直输出

#### *wTriggerIndex*

触发器号

#### *byAssociateAlarmIn*

触发器跟随的防区通道（多个防区同时触发一个警号输出），byAssociateAlarmIn[0]表示防区 1，byAssociateAlarmIn[1]表示防区 2，依次类推，值：0- 不跟随，1- 跟随

#### *byModuleType*

外接触发器模块类型：1- 本地触发器，2- 4路触发器，3- 8路触发器，4- 单防区触发器，5- 32路触发器，6-1 门就地控制器、7-2 门就地控制器、8-4 门就地控制器

*byModuleStatus*

外接触发器模块状态：1- 在线，2- 离线

*wModuleAddress*

外接触发器模块地址，扩展模块从 1~253，0xFFFF 表示无效

*byModuleChan*

外接触发器模块通道号，从 1 开始，最大值根据模块类型来决定，0xFF 表示无效

*byWorkMode*

工作模式：1- 联动，2- 随动

*byAlarmOutMode*

输出模式：1- 非脉冲模式，2- 脉冲模式

*byTimeOn*

开时间，取值范围：1~60，单位：s

*byTimeOff*

关时间，取值范围：1~60，单位：s

*byRes2*

保留

#### Remarks

- 本地触发器的 *wTriggerIndex*、*byModuleType*、*byModuleStatus*、*wModuleAddress* 和 *byModuleChan* 参数不支持修改，只能获取。

## 8.16 NET\_DVR\_ALARM\_RS485CFG:报警主机 RS485 参数结构体

```
struct{
    DWORD    dwSize;
    BYTE     sDeviceName[NAME_LEN];
    WORD     wDeviceType;
    WORD     wDeviceProtocol;
    DWORD    dwBaudRate;
    BYTE     byDataBit;
    BYTE     byStopBit;
    BYTE     byParity;
    BYTE     byFlowcontrol;
    BYTE     byDuplex;
    BYTE     byWorkMode;
    BYTE     byChannel;
    BYTE     byRes[37];
}NET_DVR_ALARM_RS485CFG, *LPNET_DVR_ALARM_RS485CFG;
```

#### Members

*dwSize*

结构体大小

*sDeviceName*

前端设备名称

*wDeviceType*



前端设备类型

*wDeviceProtocol*

前端设备协议，通过获取协议列表 `NET_DVR_GetDeviceProtoList` 获取

*dwBaudRate*

波特率(bps)，0-50，1-75，2-110，3-150，4-300，5-600，6-1200，7-2400，8-4800，9-9600，10-19200，11-38400，12-57600，13-76800，14-115.2k

*byDataBit*

数据有几位：0-5 位，1-6 位，2-7 位，3-8 位

*byStopBit*

停止位：0-1 位，1-2 位

*byParity*

是否校验：0-无校验，1-奇校验，2-偶校验

*byFlowcontrol*

是否流控：0-无，1-软流控,2-硬流控

*byDuplex*

0- 半双工，1- 全双工

*byWorkMode*

工作模式：0- 控制台，1- 透明通道，2- 梯控，3- 读卡器，0xfe- 自定义(保留)，0xff- 禁用

*byChannel*

RS485 通道号

*byRes*

保留

#### Remarks

- RS485 参数能力，对应 RS232 和 RS485 串口能力集（接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：`DEVICE_SERIAL_ABILITY`）中节点<RS485>。

## 8.17 NET\_DVR\_ALARMHOST\_MAIN\_STATUS:报警主机主要状态信息结

### 构体

struct{

DWORD dwSize;

BYTE bySetupAlarmStatus[[MAX\\_ALARMHOST\\_ALARMIN\\_NUM](#)];

BYTE byAlarmInStatus[[MAX\\_ALARMHOST\\_ALARMIN\\_NUM](#)];

BYTE byAlarmOutStatus[[MAX\\_ALARMHOST\\_ALARMOUT\\_NUM](#)];

BYTE byBypassStatus[[MAX\\_ALARMHOST\\_ALARMIN\\_NUM](#)];

BYTE bySubSystemGuardStatus[[MAX\\_ALARMHOST\\_SUBSYSTEM](#)];

BYTE byAlarmInFaultStatus[[MAX\\_ALARMHOST\\_ALARMIN\\_NUM](#)];

BYTE byRes[56];

}NET\_DVR\_ALARMHOST\_MAIN\_STATUS, \*LPNET\_DVR\_ALARMHOST\_MAIN\_STATUS;

#### Members

*dwSize*

结构体大小

*bySetupAlarmStatus*

防区布防状态(最大支持 512 个防区查询)。数组下标 0 对应防区 1，数组下标 1 对应防区 2，依次类

推，数组的值：0- 对应防区处于撤防状态，1- 对应防区处于布防状态。

#### *byAlarmInStatus*

防区报警状态(最大支持 512 个防区查询)。数组下标 0 对应防区 1，数组下标 1 对应防区 2，依次类推，数组的值：0- 对应防区当前无报警，1- 对应防区当前有报警。

#### *byAlarmOutStatus*

触发器状态(最大支持 512 个触发器查询)。数组下标 0 对应触发器 1，数组下标 1 对应触发器 2，依次类推，数组的值：0- 对应触发器无报警，1- 对应触发器有报警

#### *byBypassStatus*

旁路状态，数组下标 0 对应防区 1，数组下标 1 对应防区 2，依次类推，数组的值：0- 表示防区没有旁路，1- 表示防区旁路

#### *bySubSystemGuardStatus*

子系统布防状态，数组下标 0 对应子系统 1，数组下标 1 对应子系统 2，依次类推，数组的值：0 - 对应子系统处于撤防状态，1- 对应子系统处于布防状态

#### *byAlarmInFaultStatus*

防区故障状态，数组下标 0 对应防区 1，数组下标 1 对应防区 2，依次类推，数组的值：0-对应防区处于正常状态，1-对应防区处于故障状态

#### *byRes*

保留

## 8.18 NET\_DVR\_ALARMHOST\_NETCFG:报警主机中心网络参数结构体

```
struct{
    DWORD                dwSize;
    NET\_DVR\_ALARMHOST\_NETPARAM struNetCenter[MAX_CENTERNUM];
    BYTE                 byRes1[32];
}NET_DVR_ALARMHOST_NETCFG, *LPNET_DVR_ALARMHOST_NETCFG;
```

### Members

#### *dwSize*

结构体大小

#### *struNetCenter*

报警中心参数，报警主机最多 4 个报警中心，门禁主机只有 1 个报警中心

#### *byRes1*

保留

### Remarks

- 设备是否支持报警中心网络参数配置或者支持的参数能力，可以通过设备能力集进行判断，对应报警主机能力集 (AlarmHostAbility)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：DEVICE\_ABILITY\_INFO，节点：<netModuleNo>和<NetworkConfig>。

## 8.19 NET\_DVR\_ALARMHOST\_NETPARAM:上传中心网络参数配置结构体

```
struct{
    DWORD                dwSize;
    NET\_DVR\_IPADDR       struIP;
    WORD                 wPort;
```

```

    BYTE                byAddressType;
    BYTE                byRes1;
    BYTE                byDomainName[MAX_DOMAIN_NAME];
    BYTE                byReportProtocol;
    BYTE                byDevID[ACCOUNTNUM_LEN_32];
    BYTE                byRes2[7];
}NET_DVR_ALARMHOST_NETPARAM, *LPNET_DVR_ALARMHOST_NETPARAM;

```

### Members

*dwSize*

结构体大小

*strulP*

上传中心的 IP 地址, byAddressType 为 0 或者 1 时有效

*wPort*

上传中心的端口号

*byAddressType*

地址类型: 0- 无意义, 1- ipv4/ipv6 地址, 2- 域名

*byRes1*

保留, 置为 0

*byDomainName*

域名, byAddressType 为 2 时有效

*byReportProtocol*

报告协议: 1- private, 2- NAL2300, 3- ehome(门禁主机只支持 ehome)

*byDevID*

设备 ID, 对于报警主机, 协议为 NAL2300 时有效, 有效长度为 9; 对于门禁主机, 有效长度为 32  
有效字符: 0~9、a~f、A~F

*byRes2*

保留, 置为 0

## 8.20 NET\_DVR\_ALARMHOST\_REPORT\_CENTER\_CFG\_V40:报告上传参数

### 配置结构体

```

struct{
    DWORD    dwSize;
    BYTE     byValid;
    BYTE     byDataType;
    BYTE     byRes[2];
    BYTE     byChanAlarmMode[MAX_CHAN_NUM];
    BYTE     byDealFailCenter[MAX_CENTERGROUP_NUM];
    BYTE     byZoneReport[MAX_ALARMHOST_ALARMIN_NUM];
    BYTE     byNonZoneReport[MAX_EVENT_NUM];
    BYTE     byRes2[256];
}NET_DVR_ALARMHOST_REPORT_CENTER_CFG_V40, *LPNET_DVR_ALARMHOST_REPORT_CENTER_CFG_V40;

```

### Members

*dwSize*

结构体大小

*byValid*

是否启用：0- 禁用，1- 启用

*byDataType*

上传数据类型：1-所有报警数据，2-所有非报警数据，3-所有数据，4-防区报警报告，5-非防区报警报告

*byRes*

保留，置为 0

*byChanAlarmMode*

中心组报警通道：1- T1，2- T2，3- N1，4-N2，5-G1，6-G2。(如果设备支持 3G，G1、G2 表示 3G 模块，如果不支持，表示 GPRS 模块，一款设备中 3G 模块和 GPRS 模块只会出现一种)

*byDealFailCenter*

向指定中心组发送失败报告，用数组下标表示是哪个中心组：0- 不选择，1- 选择。

*byDealFailCenter*[0]==1 表示数据上传到中心组 1，*byDealFailCenter*[1]==1 表示数据上传到中心组 2，依次类推

*byZoneReport*

防区报警报告：0-不上传，1-上传。*byZoneReport*[0]==1 表示上传防区 1 报警报告，*byZoneReport*[1]==1 表示上传防区 2 报警报告，依次类推

*byNonZoneReport*

非防区报警报告，0-不上传，1-上传。每一个数组表示一种事件类型，如下所示：

*byNonZoneReport*[0]- 软防区报告

*byNonZoneReport*[1]- 系统状态报告

*byNonZoneReport*[2]- 取消报告

*byNonZoneReport*[3]- 测试报告

*byNonZoneReport*[4]- 布防报告

*byNonZoneReport*[5]- 撤防报告

*byNonZoneReport*[6]- 挟持报告

*byNonZoneReport*[7]- 报警恢复报告

*byNonZoneReport*[8]- 旁路报告

*byNonZoneReport*[9]- 旁路恢复报告

*byRes2*

保留，置为 0

#### Remarks

- 对于门禁主机，*byDealFailCenter*、*byZoneReport*、*dwNonZoneReport* 这些参数无效。
- 设备是否支持报告上传参数配置或者支持的参数能力，可以通过设备能力集进行判断，对应报警主机能力集(**AlarmHostAbility**)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：DEVICE\_ABILITY\_INFO，节点：<ReportModeConfig>。

## 8.21 NET\_DVR\_ALARMHOST\_WIRELESS\_NETWORK\_CFG:无线网络模块

### 参数配置结构体

struct{

DWORD

dwSize;

[NET\\_DVR\\_ALARMHOST\\_NETPARAM](#)

struNetCenter[*MAX\_CENTERNUM*];

```

BYTE                byAPNName[APN\_NAME\_LEN];
BYTE                byAPNUserName[APN\_USERNAME\_LEN];
BYTE                byAPNPassWord[APN\_USERPASSWORD\_LEN];
BYTE                byReconnTime;
BYTE                byOverTime;
BYTE                byDetectLinkTime;
BYTE                byRes1;
BYTE                bySIMNum[NAME\_LEN];
NET\_DVR\_IPADDR      struSIMIP;
BYTE                byRes2[64];
}NET_DVR_ALARMHOST_WIRELESS_NETWORK_CFG, *LPNET_DVR_ALARMHOST_WIRELESS_NETWORK_CFG;

```

## Members

*dwSize*

结构体大小

*struNetCenter*

中心 GPRS 网络相关参数的配置，每位数组表示一个中心（门禁主机目前只支持 1 个中心）

*byAPNName*

APN 名称，GPRS 参数配置时有效，3G 配置时无效

*byAPNUserName*

APN 账号，GPRS 参数配置时有效，3G 配置时无效

*byAPNPassWord*

APN 密码，GPRS 参数配置时有效，3G 配置时无效

*byReconnTime*

重连时间，连接失效后启用重连的时间，10 秒为单位，取值范围：1~30，GPRS 参数配置时有效，3G 配置时无效

*byOverTime*

超时时间，超过 OverTime 时间没有收到有效数据则重连，取值范围：1~254，单位：30 秒，GPRS 参数配置时有效，3G 配置时无效

*byDetectLinkTime*

探测链路是否还保持，取值范围：1~30，单位：10s，GPRS 参数配置时有效，3G 配置时无效

*byRes1*

保留，置为 0

*bySIMNum*

SIM 卡号（手机号）

*struSIMIP*

登陆网络后网络给分配的 IP 地址，只能获取

*byRes2*

保留，置为 0

## Remarks

- 对于门禁主机，byAPNUserName、byAPNPassWord、byReconnTime、byOverTime、byDetectLinkTime 这些参数无效。
- 设备是否支持无线网络模块参数配置或者支持的参数能力，可以通过设备能力集进行判断，对应报警主机能力集 (AlarmHostAbility)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：DEVICE\_ABILITY\_INFO，节点：<WirelessNetworkConfig>。

## 8.22 NET\_DVR\_ALARMINFO\_V30:上传的报警信息

```
struct{
    DWORD    dwAlarmType;
    DWORD    dwAlarmInputNumber;
    BYTE     byAlarmOutputNumber[MAX_ALARMOUT_V30];
    BYTE     byAlarmRelateChannel[MAX_CHANNUM_V30];
    BYTE     byChannel[MAX_CHANNUM_V30];
    BYTE     byDiskNumber[MAX_DISKNUM_V30];
}NET_DVR_ALARMINFO_V30,*LPNET_DVR_ALARMINFO_V30;
```

### Members

#### *dwAlarmType*

报警类型：0-信号量报警，1-硬盘满，2-信号丢失，3-移动侦测，4-硬盘未格式化，5-读写硬盘出错，6-遮挡报警，7-制式不匹配，8-非法访问，9-视频信号异常，10-录像/抓图异常，11-智能场景变化，12-阵列异常，13-前端/录像分辨率不匹配，17-闪光灯异常

#### *dwAlarmInputNumber*

报警输入端口

#### *byAlarmOutputNumber*

触发的报警输出端口，值为 1 表示该报警端口输出，如 *byAlarmOutputNumber*[0]=1 表示触发第 1 个报警输出端口输出，*byAlarmOutputNumber*[1]=1 表示触发第 2 个报警输出端口，依次类推。

#### *byAlarmRelateChannel*

触发的录像通道，值为 1 表示该通道录像，如 *byAlarmRelateChannel*[0]=1 表示触发第 1 个通道录像

#### *byChannel*

发生报警的通道。当报警类型为 2、3、6、9、10、11、13、17 时有效，如 *byChannel*[0]=1 表示第 1 个通道报警

#### *byDiskNumber*

发生报警的硬盘。当报警类型为 1、4、5 时有效，*byDiskNumber*[0]=1 表示 1 号硬盘异常

### Remarks

抓拍机支持类型：1-硬盘满、5-读写硬盘出错、8-非法访问，这里的硬盘即指 SD 卡。

## 8.23 NET\_DVR\_ANTI\_SNEAK\_CFG:反潜回配置结构体

```
struct{
    DWORD    dwSize;
    BYTE     byEnable;
    BYTE     byRes1[3];
    DWORD    dwStartCardReaderNo;
    BYTE     byRes2[64];
}NET_DVR_ANTI_SNEAK_CFG,*LPNET_DVR_ANTI_SNEAK_CFG;
```

### Members

#### *dwSize*

结构体大小

#### *byEnable*

使能反潜回功能：0- 不使能，1- 使能

#### *byRes1*

保留，置为 0

*dwStartCardReaderNo*

反潜回起始读卡器编号

*byRes2*

保留，置为 0

#### Remarks

互锁和反潜回功能不能同时生效。

## 8.24 NET\_DVR\_BASEMAP\_CFG:原图参数结构体

```
struct{
    BYTE        byScreenIndex;
    BYTE        byMapNum;
    BYTE        res[2];
    WORD        wSourWidth;
    WORD        wSourHeight;
}NET_DVR_BASEMAP_CFG,*LPNET_DVR_BASEMAP_CFG;
```

#### Members

*byScreenIndex*

屏幕的序号

*byMapNum*

被分割成了多少块

*res*

保留，置为 0

*wSourWidth*

原图片的宽度

*wSourHeight*

原图片的高度

## 8.25 NET\_DVR\_CALIBRATE\_TIME:时间校时参数

```
struct{
    DWORD        dwSize;
    NET\_DVR\_TIME struTime;
    WORD        wMilliSec;
    BYTE        byRes[14];
}NET_DVR_CALIBRATE_TIME,*LPNET_DVR_CALIBRATE_TIME;
```

#### Members

*dwSize*

结构体大小

*struTime*

时间参数，年月日时分秒

*wMilliSec*

毫秒

*byRes*

保留，置为 0

## 8.26 NET\_DVR\_CAPTURE\_FINGERPRINT\_COND:采集指纹信息条件结构

### 体

```
struct{
    DWORD    dwSize;
    BYTE     byFingerPrintPicType;
    BYTE     byFingerNo;
    BYTE     byRes[126];
} NET_DVR_CAPTURE_FINGERPRINT_COND,*LPNET_DVR_CAPTURE_FINGERPRINT_COND;
```

#### Members

*dwSize*

结构体大小

*byFingerPrintPicType*

图片类型：0-无意义

*byFingerNo*

手指编号，范围 1-10

*byRes*

保留，置为 0

## 8.27 NET\_DVR\_CAPTURE\_FINGERPRINT\_CFG:指纹信息结构体

```
struct{
    DWORD    dwSize;
    DWORD    dwFingerPrintDataSize;
    BYTE     byFingerData[MAX_FINGER_PRINT_LEN];
    DWORD    dwFingerPrintPicSize;
    char*    pFingerPrintPicBuffer
    BYTE     byFingerNo;
    BYTE     byFingerPrintQuality;
    BYTE     byRes[62];
} NET_DVR_CAPTURE_FINGERPRINT_CFG,*LPNET_DVR_CAPTURE_FINGERPRINT_CFG;
```

#### Members

*dwSize*

结构体大小

*dwFingerPrintDataSize*

指纹数据大小

*byFingerData*

指纹数据内容

*dwFingerPrintPicSize*

指纹图片大小，等于 0 时，代表无指纹图片数据

*pFingerPrintPicBuffer*

指纹图片缓存

*byFingerNo*

手指编号，范围 1-10



*byFingerPrintQuality*

指纹质量，范围 1-100

*byRes*

保留，置为 0

## 8.28 NET\_DVR\_CARD\_CFG:卡参数配置结构体

```
struct{
    DWORD                dwSize;
    DWORD                dwModifyParamType;
    BYTE                 byCardNo[ACS_CARD_NO_LEN];
    BYTE                 byCardValid;
    BYTE                 byCardType;
    BYTE                 byLeaderCard;
    BYTE                 byRes1;
    DWORD                dwDoorRight;
    NET_DVR_VALID_PERIOD_CFG struValid;
    DWORD                dwBelongGroup;
    BYTE                 byCardPassword[CARD_PASSWORD_LEN];
    BYTE                 byCardRightPlan[MAX_DOOR_NUM][MAX_CARD_RIGHT_PLAN_NUM];
    DWORD                dwMaxSwipeTime;
    DWORD                dwSwipeTime;
    WORD                 wRoomNumber;
    WORD                 wFloorNumber;
    BYTE                 byRes2[20];
}NET_DVR_CARD_CFG,*LPNET_DVR_CARD_CFG;
```

### Members

*dwSize*

结构体大小

*dwModifyParamType*

需要修改的卡参数（设置卡参数时有效），按位表示，每位代表一种参数，值：0- 不修改，1- 需要修改

宏定义	宏定义值	含义
CARD_PARAM_CARD_VALID	0x00000001	卡是否有效参数
CARD_PARAM_VALID	0x00000002	有效期参数
CARD_PARAM_CARD_TYPE	0x00000004	卡类型参数
CARD_PARAM_DOOR_RIGHT	0x00000008	门权限参数
CARD_PARAM_LEADER_CARD	0x00000010	首卡参数
CARD_PARAM_SWIPE_NUM	0x00000020	最大刷卡次数参数
CARD_PARAM_GROUP	0x00000040	所属群组参数
CARD_PARAM_PASSWORD	0x00000080	卡密码参数
CARD_PARAM_RIGHT_PLAN	0x00000100	卡权限计划参数

CARD_PARAM_SWIPED_NUM	0x00000200	已刷卡次数
-----------------------	------------	-------

**byCardNo**

卡号，特殊卡号定义如下：

0xFFFFFFFFFFFFFFFFF: 非法卡号

0xFFFFFFFFFFFFFFFFE: 胁迫码

0xFFFFFFFFFFFFFFFFD: 超级码

0xFFFFFFFFFFFFFFFFC~0xFFFFFFFFFFFFFFFF1: 预留的特殊卡

0xFFFFFFFFFFFFFFFF0: 最大合法卡号

**byCardValid**

卡是否有效：0- 无效，1- 有效（用于删除卡，设置时置为 0 进行删除，获取时此字段始终为 1）

**byCardType**

卡类型：1- 普通卡（默认），2- 残疾人卡，3- 黑名单卡，4- 巡更卡，5- 胁迫卡，6- 超级卡，7- 来宾卡，8- 解除卡

**byLeaderCard**

是否为首卡：1- 是，0- 否

**byRes1**

保留，置为 0

**dwDoorRight**

门权限，按位表示，从低位到高位表示对门 1~N 是否有权限，值：0- 无权限，1- 有权限

**struValid**

有效期参数

**dwBelongGroup**

所属群组，按位表示，从低位到高位表示是否从属群组 1~N，值：0- 不属于，1- 属于

**byCardPassword**

卡密码

**byCardRightPlan**

卡权限计划，取值为计划模板编号，同个门不同计划模板采用权限或的方式处理

**dwMaxSwipeTime**

最大刷卡次数，0 为无次数限制

**dwSwipeTime**

已刷卡次数

**wRoomNumber**

房间号

**wFloorNumber**

层号

**byRes2**

保留，置为 0

**Remarks**

- 卡参数能力，对应门禁主机能力集（接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：ACS\_ABILITY）中节点<Card>。

## 8.29 NET\_DVR\_CARD\_CFG\_V50:卡参数配置结构体

```
struct{
```

```

DWORD          dwSize;
DWORD          dwModifyParamType;
BYTE           byCardNo[ACS_CARD_NO_LEN];
BYTE           byCardValid;
BYTE           byCardType;
BYTE           byLeaderCard;
BYTE           byRes1;
BYTE           byDoorRight[MAX_DOOR_NUM_256];
NET_DVR_VALID_PERIOD_CFG struValid;
BYTE           byBelongGroup[MAX_GROUP_NUM_128];
BYTE           byCardPassword[CARD_PASSWORD_LEN];
WORD           wCardRightPlan[MAX_DOOR_NUM_256][MAX_CARD_RIGHT_PLAN_NUM];
DWORD          dwMaxSwipeTime;
DWORD          dwSwipeTime;
WORD           wRoomNumber;
SHORT          wFloorNumber;
DWORD          dwEmployeeNo;
BYTE           byName[NAME_LEN];
WORD           wDepartmentNo;
WORD           wSchedulePlanNo;
BYTE           bySchedulePlanType;
BYTE           byRes2[119];
}NET_DVR_CARD_CFG_V50,*LPNET_DVR_CARD_CFG_V50;

```

## Members

*dwSize*

结构体大小

*dwModifyParamType*

需要修改的卡参数（设置卡参数时有效），按位表示，每位代表一种参数，值：0- 不修改，1- 需要修改

宏定义	宏定义值	含义
CARD_PARAM_CARD_VALID	0x00000001	卡是否有效参数
CARD_PARAM_VALID	0x00000002	有效期参数
CARD_PARAM_CARD_TYPE	0x00000004	卡类型参数
CARD_PARAM_DOOR_RIGHT	0x00000008	门权限参数
CARD_PARAM_LEADER_CARD	0x00000010	首卡参数
CARD_PARAM_SWIPE_NUM	0x00000020	最大刷卡次数参数
CARD_PARAM_GROUP	0x00000040	所属群组参数
CARD_PARAM_PASSWORD	0x00000080	卡密码参数
CARD_PARAM_RIGHT_PLAN	0x00000100	卡权限计划参数
CARD_PARAM_SWIPED_NUM	0x00000200	已刷卡次数

**byCardNo**

卡号，特殊卡号定义如下：

0xFFFFFFFFFFFFFFFFF：非法卡号

0xFFFFFFFFFFFFFFFFE：胁迫码

0xFFFFFFFFFFFFFFFFD：超级码

0xFFFFFFFFFFFFFFFFC~0xFFFFFFFFFFFFFFFF1：预留的特殊卡

0xFFFFFFFFFFFFFFFF0：最大合法卡号

**byCardValid**

卡是否有效：0- 无效，1- 有效（用于删除卡，设置时置为 0 进行删除，获取时此字段始终为 1）

**byCardType**

卡类型：1- 普通卡（默认），2- 残疾人卡，3- 黑名单卡，4- 巡更卡，5- 胁迫卡，6- 超级卡，7- 来宾卡，8- 解除卡

**byLeaderCard**

是否为首卡：1- 是，0- 否

**byRes1**

保留，置为 0

**byDoorRight**

门权限，按字节表示，1-为有权限，0-为无权限，从低位到高位依次表示对门 1-N 是否有权限

**struValid**

有效期参数

**byBelongGroup**

所属群组，按字节表示，1-属于，0-不属于，从低位到高位表示是否从属群组 1~N

**byCardPassword**

卡密码

**wCardRightPlan**

卡权限计划，取值为计划模板编号，同个门不同计划模板采用权限或的方式处理

**dwMaxSwipeTime**

最大刷卡次数，0 为无次数限制

**dwSwipeTime**

已刷卡次数

**wRoomNumber**

房间号

**wFloorNumber**

层号

**dwEmployeeNo**

工号

**byname**

姓名

**wDepartmentNo**

部门编号

**wSchedulePlanNo**

排班计划编号

**bySchedulePlanType**

排班计划类型：0-无意义、1-个人、2-部门

*byRes2*

保留，置为 0

#### Remarks

- 卡参数能力，对应门禁主机能力集（接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：ACS\_ABILITY）中节点<Card>。
- 结构体中对于其他门禁设备：工号、姓名、部门编号、排班计划编号、排班计划类型不是必须项，但对于考勤一体机为必填项。

## 8.30 NET\_DVR\_CARD\_CFG\_COND:卡参数配置条件结构体

```
struct{
    DWORD    dwSize;
    DWORD    dwCardNum;
    BYTE     byCheckCardNo;
    BYTE     byRes1[3];
    WORD     wLocalControllerID;
    BYTE     byRes[26];
}NET_DVR_CARD_CFG_COND,*LPNET_DVR_CARD_CFG_COND;
```

#### Members

*dwSize*

结构体大小

*dwCardNum*

卡号

*byCheckCardNo*

设备是否进行卡号校验：0- 不校验，1- 校验

*byRes1*

保留，置为 0

*wLocalControllerID*

就地控制器序号，表示往就地控制器下发离线卡参数，0 代表是门禁主机

*byRes*

保留，置为 0

#### Remarks

- 设置卡参数（下发卡参数）时，如果将 *byCheckCardNo* 置为 0，那么设备将不校验应用层下发的卡号信息，直接写入本地存储，可以一定程度提高卡号下发的速度，但是需要上层应用自己保证卡号信息不重复（整型值不能重复，比如，不能同时含有 1 和 01 这两种卡号）。

## 8.31 NET\_DVR\_CARD\_CFG\_SEND\_DATA:获取卡参数的发送数据

```
struct{
    DWORD    dwSize;
    BYTE     byCardNo[ACS_CARD_NO_LEN];
    BYTE     byRes[16];
}NET_DVR_CARD_CFG_SEND_DATA,*LPNET_DVR_CARD_CFG_SEND_DATA;
```

#### Members

*dwSize*

结构体大小

*byCardNo*

卡号

*byRes*

保留，置为 0

### 8.32 NET\_DVR\_CARD\_PASSWD\_CFG:卡密码开门使能配置结构体

```
struct{
    DWORD    dwSize;
    BYTE     byCardNo[ACS_CARD_NO_LEN];
    BYTE     byCardPassword[CARD_PASSWORD_LEN];
    DWORD    dwErrorCode;
    BYTE     byCardValid;
    BYTE     byRes2[23];
}NET_DVR_CARD_PASSWD_CFG, *LPNET_DVR_CARD_PASSWD_CFG;
```

#### Members

*dwSize*

结构体大小

*byCardNo*

卡号

*byCardPassword*

卡密码

*dwErrorCode*

获取卡密码开门使能配置返回的错误码，0 表示配置成功，其他值表示失败的错误码

*byCardValid*

卡是否有效（用于删除卡，设置时置为 0 进行删除，获取时此字段始终为 1）：0- 无效，1- 有效

*byRes2*

保留，置为 0

#### Remarks

- 设备是否支持卡密码开门使能配置，可以通过设备能力集进行判断，对应门禁主机能力集(**AcsAbility**)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：ACS\_ABILITY，节点：<Card>中的<onlyPasswdOpen>。

### 8.33 NET\_DVR\_CARD\_PASSWD\_STATUS:卡密码开门使能配置状态结构

体

```
struct{
    DWORD    dwSize;
    BYTE     byCardNo[ACS_CARD_NO_LEN];
    DWORD    dwErrorCode;
    BYTE     byRes2[24];
}NET_DVR_CARD_PASSWD_STATUS, *LPNET_DVR_CARD_PASSWD_STATUS;
```

#### Members

*dwSize*

结构体大小

*byCardNo*

卡号

*dwErrorCode*

发送卡密码开门使能配置返回的错误码，0 表示配置成功，其他值表示失败的错误码

*byRes2*

保留，置为 0

#### Remarks

- 设备是否支持卡密码开门使能配置，可以通过设备能力集进行判断，对应门禁主机能力集(**AcsAbility**)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：ACS\_ABILITY，节点：<Card>中的<onlyPasswdOpen>。

## 8.34 NET\_DVR\_CARD\_READER\_ANTI\_SNEAK\_CFG:读卡器反潜回配置结

### 构体

```
struct{
    DWORD    dwSize;
    BYTE     byEnable;
    BYTE     byRes1[3];
    DWORD    dwFollowUpCardReader[MAX\_SNEAK\_PATH\_NODE];
    BYTE     byRes2[32];
}NET_DVR_ANTI_SNEAK_CFG,*LPNET_DVR_ANTI_SNEAK_CFG;
```

#### Members

*dwSize*

结构体大小

*byEnable*

是否加入反潜回路路径：0- 不加入，1- 加入

*byRes1*

保留，置为 0

*dwFollowUpCardReader*

后续读卡器编号

*byRes2*

保留，置为 0

## 8.35 NET\_DVR\_CARD\_READER\_CFG:读卡器参数配置结构体

```
struct{
    DWORD    dwSize;
    BYTE     byEnable;
    BYTE     byCardReaderType;
    BYTE     byOkLedPolarity;
    BYTE     byErrorLedPolarity;
    BYTE     byBuzzerPolarity;
    BYTE     bySwipeInterval;
    BYTE     byPressTimeout;
    BYTE     byEnableFailAlarm;
    BYTE     byMaxReadCardFailNum;
```

```

BYTE    byEnableTamperCheck;
BYTE    byOfflineCheckTime;
BYTE    byFingerPrintCheckLevel;
BYTE    byUseLocalController;
BYTE    byRes1;
WORD    wLocalControllerID;
WORD    wLocalControllerReaderID;
WORD    wCardReaderChannel;
BYTE    byRes[16];
}NET_DVR_CARD_READER_CFG,*LPNET_DVR_CARD_READER_CFG;

```

## Members

### *dwSize*

结构体大小

### *byEnable*

是否使能：0- 不启用，1- 启用

### *byCardReaderType*

读卡器类型：1- DS-K110XM/MK/C/CK, 2- DS-K192AM/AMP, 3- DS-K192BM/BMP, 4- DS-K182AM/AMP, 5- DS-K182BM/BMP, 6- DS-K182AMF/ACF, 7- 韦根或 485 不在线, 8- DS-K1101M/MK, 9- DS-K1101C/CK, 10- DS-K1102M/MK/M-A, 11- DS-K1102C/CK, 12- DS-K1103M/MK, 13- DS-K1103C/CK, 14- DS-K1104M/MK, 15- DS-K1104C/CK, 16- DS-K1102S/SK/S-A, 17- DS-K1102G/GK, 18- DS-K1100S-B, 19- DS-K1102EM/EMK, 20- DS-K1102E/EK, 21- DS-K1200EF, 22- DS-K1200MF, 23- DS-K1200CF, 24- DS-K1300EF, 25- DS-K1300MF, 26- DS-K1300CF, 27- DS-K1105E, 28- DS-K1105M, 29- DS-K1105C, 30- DS-K182AMF, 31- DS-K196AMF, 32- DS-K194AMP, 33- DS-K1T200EF/EF-C/MF-MF-C/CF/CF-C, 34- DS-K1T300EF/EF-C/MF-MF-C/CF/CF-C, 35-DS-K1T105E/E-C/M/M-C/C/C-C, 36-DS-K1T803F/MF/SF/EF, 37-DS-K1A801F/MF/SF/EF

### *byOkLedPolarity*

OK LED 极性：0- 阴极，1- 阳极

### *byErrorLedPolarity*

Error LED 极性：0- 阴极，1- 阳极

### *byBuzzerPolarity*

蜂鸣器极性：0- 阴极，1- 阳极

### *bySwipeInterval*

重复刷卡间隔时间，单位：秒

### *byPressTimeout*

按键超时时间，单位：秒，取值范围：1~255

### *byEnableFailAlarm*

是否启用读卡失败超次报警：0- 不启用，1- 启用

### *byMaxReadCardFailNum*

最大读卡失败次数，取值范围：1~10

### *byEnableTamperCheck*

是否启用防拆检测：0- 不启用，1- 启用

### *byOfflineCheckTime*

掉线检测时间，单位：秒，取值范围：0~255

### *byFingerPrintCheckLevel*



指纹识别等级: 1- 1/10 误认率, 2- 1/100 误认率, 3- 1/1000 误认率, 4- 1/10000 误认率, 5- 1/100000 误认率, 6- 1/1000000 误认率, 7- 1/10000000 误认率, 8- 1/100000000 误认率, 9- 3/100 误认率, 10- 3/1000 误认率, 11- 3/10000 误认率, 12- 3/100000 误认率, 13- 3/1000000 误认率, 14- 3/10000000 误认率, 15- 3/100000000 误认率, 16- Automatic Normal, 17- Automatic Secure, 18- Automatic More Secure

*byUseLocalController*

只读, 是否连接在就地控制器上, 0-否, 1-是

*byRes1*

保留, 置为 0

*wLocalControllerID*

只读, 就地控制器序号, *byUseLocalController*=1 时有效, 1-255, 0 代表未注册

*wLocalControllerReaderID*

只读, 就地控制器的读卡器 ID, *byUseLocalController*=1 时有效, 0 代表未注册

*wCardReaderChannel*

只读, 读卡器通信通道号, *byUseLocalController*=1 时有效, 0 韦根或离线, 1-RS485A, 2-RS485B

*byRes*

保留, 置为 0

### Remarks

设备是否支持读卡器参数配置或者支持的参数能力, 可以通过设备能力集进行判断, 对应门禁主机能力集(**AcsAbility**), 相关接口: [NET\\_DVR\\_GetDeviceAbility](#), 能力集类型: ACS\_ABILITY, 节点: <CardReaderCfg>。

## 8.36 NET\_DVR\_CARD\_READER\_CFG\_V50:读卡器参数配置结构体(V50)

struct{

```

    DWORD    dwSize;
    BYTE      byEnable;
    BYTE      byCardReaderType;
    BYTE      byOkLedPolarity;
    BYTE      byErrorLedPolarity;
    BYTE      byBuzzerPolarity;
    BYTE      bySwipeInterval;
    BYTE      byPressTimeout;
    BYTE      byEnableFailAlarm;
    BYTE      byMaxReadCardFailNum;
    BYTE      byEnableTamperCheck;
    BYTE      byOfflineCheckTime;
    BYTE      byFingerPrintCheckLevel;
    BYTE      byUseLocalController;
    BYTE      byRes1;
    WORD      wLocalControllerID;
    WORD      wLocalControllerReaderID;
    WORD      wCardReaderChannel;
    BYTE      byFingerPrintImageQuality;
    BYTE      byFingerPrintContrastTimeOut;
    BYTE      byFingerPrintRecognizeInterval;
    BYTE      byFingerPrintMatchFastMode;
```

```

BYTE    byFingerPrintModuleSensitive;
BYTE    byFingerPrintModuleLightCondition;
BYTE    byFaceMatchThresholdN;
BYTE    byFaceQuality;
BYTE    byFaceRecognizeTimeOut;
BYTE    byFaceRecognizeInterval;
WORD    wCardReaderFunction;
BYTE    byCardReaderDescription[CARD\_READER\_DESCRIPTION];
WORD    wFacelImageSensitometry;
BYTE    byLivingBodyDetect;
BYTE    byFaceMatchThreshold1;
BYTE    byRes[256];
} NET_DVR_CARD_READER_CFG_V50,*LPNET_DVR_CARD_READER_CFG_V50;

```

## Members

### *dwSize*

结构体大小

### *byEnable*

是否使能：0- 不启用，1- 启用

### *byCardReaderType*

读卡器类型：1- DS-K110XM/MK/C/CK, 2- DS-K192AM/AMP, 3- DS-K192BM/BMP, 4- DS-K182AM/AMP, 5- DS-K182BM/BMP, 6- DS-K182AMF/ACF, 7- 韦根或 485 不在线, 8- DS-K1101M/MK, 9- DS-K1101C/CK, 10- DS-K1102M/MK/M-A, 11- DS-K1102C/CK, 12- DS-K1103M/MK, 13- DS-K1103C/CK, 14- DS-K1104M/MK, 15- DS-K1104C/CK, 16- DS-K1102S/SK/S-A, 17- DS-K1102G/GK, 18- DS-K1100S-B, 19- DS-K1102EM/EMK, 20- DS-K1102E/EK, 21- DS-K1200EF, 22- DS-K1200MF, 23- DS-K1200CF, 24- DS-K1300EF, 25- DS-K1300MF, 26- DS-K1300CF, 27- DS-K1105E, 28- DS-K1105M, 29- DS-K1105C, 30- DS-K182AMF, 31- DS-K196AMF, 32- DS-K194AMP, 33- DS-K1T200EF/EF-C/MF-MF-C/CF/CF-C, 34- DS-K1T300EF/EF-C/MF-MF-C/CF/CF-C, 35-DS-K1T105E/E-C/M/M-C/C/C-C, 36-DS-K1T803F/MF/SF/EF, 37-DS-K1A801F/MF/SF/EF

### *byOkLedPolarity*

OK LED 极性：0- 阴极，1- 阳极

### *byErrorLedPolarity*

Error LED 极性：0- 阴极，1- 阳极

### *byBuzzerPolarity*

蜂鸣器极性：0- 阴极，1- 阳极

### *bySwipeInterval*

重复刷卡间隔时间，单位：秒

### *byPressTimeout*

按键超时时间，单位：秒，取值范围：1~255

### *byEnableFailAlarm*

是否启用读卡失败超次报警：0- 不启用，1- 启用

### *byMaxReadCardFailNum*

最大读卡失败次数，取值范围：1~10

### *byEnableTamperCheck*

是否启用防拆检测：0- 不启用，1- 启用

***byOfflineCheckTime***

掉线检测时间，单位：秒，取值范围：0~255

***byFingerPrintCheckLevel***

指纹识别等级：1- 1/10 误认率，2- 1/100 误认率，3- 1/1000 误认率，4- 1/10000 误认率，5- 1/100000 误认率，6- 1/1000000 误认率，7- 1/10000000 误认率，8- 1/100000000 误认率，9- 3/100 误认率，10- 3/1000 误认率，11- 3/10000 误认率，12- 3/100000 误认率，13- 3/1000000 误认率，14- 3/10000000 误认率，15- 3/100000000 误认率，16- Automatic Normal，17- Automatic Secure，18- Automatic More Secure

***byUseLocalController***

只读，是否连接在就地控制器上，0-否，1-是

***byRes1***

保留，置为 0

***wLocalControllerID***

只读，就地控制器序号，byUseLocalController=1 时有效，1-255,0 代表未注册

***wLocalControllerReaderID***

只读，就地控制器的读卡器 ID，byUseLocalController=1 时有效，0 代表未注册

***wCardReaderChannel***

只读，读卡器通信通道号，byUseLocalController=1 时有效，0 韦根或离线，1-RS485A,2-RS485B

***byFingerPrintImageQuality***

指纹图像质量，0-无效，1-低质量(V1)，2-中等质量(V1)，3-高质量(V1)，4-最高质量(V1)，5-低质量(V2)，6-中等质量(V2)，7-高质量(V2)，8-最高质量(V2)

***byFingerPrintContrastTimeOut***

指纹对比超时时间，0-无效，范围 1-20 代表:1s-20s，0xff-无限大

***byFingerPrintRecognizeInterval***

指纹连续识别间隔，0-无效，范围 1-10 代表:1s-10s，0xff-无延迟

***byFingerPrintMatchFastMode***

指纹匹配快速模式，0-无效，范围 1-5 代表:快速模式 1-快速模式 5，0xff-自动

***byFingerPrintModuleSensitive***

指纹模组灵敏度，0-无效，范围 1-8 代表：灵敏度级别 1-灵敏度级别 8

***byFingerPrintModuleLightCondition***

指纹模组光线条件，0-无效，1-室外，2-室内

***byFaceMatchThresholdN***

人脸比对阈值，范围 0-100

***byFaceQuality***

人脸质量，范围 0-100

***byFaceRecognizeTimeOut***

人脸识别超时时间，范围 1-20 代表：1s-20s，0xff-无限大

***byFaceRecognizeInterval***

人脸连续识别间隔，0-无效，范围 1-10 代表：1s-10s，0xff-无延迟

***wCardReaderFunction***

只读，读卡器种类，按位表示：第 1 位-指纹，第二位-人脸，第三位-指静脉

***byCardReaderDescription***

读卡器描述

***wFaceImageSensitometry***

只读，人脸图像曝光度，范围 0-65535

*byLivingBodyDetect*

真人检测, 0-无效, 1-不启用, 2-启用

*byFaceMatchThreshold1*

人脸 1:1 匹配阈值, 范围 0-100

*byRes*

保留, 置为 0

#### Remarks

设备是否支持读卡器参数配置或者支持的参数能力, 可以通过设备能力集进行判断, 对应门禁主机能力集(**AcsAbility**), 相关接口: [NET\\_DVR\\_GetDeviceAbility](#), 能力集类型: ACS\_ABILITY, 节点: <CardReaderCfg>。

## 8.37 NET\_DVR\_CARD\_READER\_PLAN:读卡器验证计划配置结构体

```
struct{
    DWORD        dwSize;
    DWORD        dwTemplateNo;
    BYTE         byRes[64];
}NET_DVR_CARD_READER_PLAN,*LPNET_DVR_CARD_READER_PLAN;
```

#### Members

*dwSize*

结构体大小

*dwTemplateNo*

计划模板编号, 为 0 表示取消关联、恢复默认状态 (刷卡开门)

*byRes*

保留, 置为 0

## 8.38 NET\_DVR\_CARD\_USER\_INFO\_CFG:卡号关联用户信息配置结构体

```
struct{
    DWORD        dwSize;
    BYTE         sUsername[NAME_LEN];
    BYTE         byRes2[256];
}NET_DVR_CARD_USER_INFO_CFG,*LPNET_DVR_CARD_USER_INFO_CFG;
```

#### Members

*dwSize*

结构体大小

*sUsername*

用户名

*byRes2*

保留, 置为 0

#### Remarks

- 设备是否支持卡号关联用户配置或者支持的参数能力, 可以通过设备能力集进行判断, 对应门禁主机能力集(**AcsAbility**), 相关接口: [NET\\_DVR\\_GetDeviceAbility](#), 能力集类型: ACS\_ABILITY, 节点: <RealteUserInfo>。

## 8.39 NET\_DVR\_CASE\_SENSOR\_CFG:事件报警输入参数配置结构体

```
struct{
    DWORD        dwSize;
    BYTE          byHostBuzzer;
    BYTE          byRes1[3];
    BYTE          byCardReaderBuzzer[MAX_CARD_READER_NUM];
    BYTE          byAssociateAlarmOut[MAX_ALARMHOST_ALARMOUT_NUM];
    BYTE          byDoorOpen[MAX_DOOR_NUM];
    BYTE          byDoorClose[MAX_DOOR_NUM];
    BYTE          byRes2[64];
}NET_DVR_CASE_SENSOR_CFG,*LPNET_DVR_CASE_SENSOR_CFG;
```

### Members

*dwSize*

结构体大小

*byHostBuzzer*

是否触发主机蜂鸣器：0- 不触发，1- 触发

*byRes1*

保留，置为 0

*byCardReaderBuzzer*

触发读卡器蜂鸣器，下标表示读卡器编号，取值：1- 触发，0- 不触发，例如：byCardReaderBuzzer[i]=1 表示触发第(i+1)个读卡器蜂鸣器

*byAssociateAlarmOut*

关联触发的报警输出，下标表示报警输出编号，取值：1-关联，0-不关联，例如：byAssociateAlarmOut[i]=1 表示关联触发第(i+1)个报警输出口

*byDoorOpen*

关联门打开，下标表示门编号，取值：0-不关联，1-关联（即打开），例如：byDoorOpen[i]=1 表示对门(i+1)关联门打开

*byDoorClose*

关联门关闭，下标表示门编号，取值：0-不关联，1-关联（即关闭），例如：byDoorOpen[i]=1 表示对门(i+1)关联门关闭

*byRes2*

保留，置为 0

### Remarks

一个事件报警输入不能同时关联一扇门的开启和关闭。

## 8.40 NET\_DVR\_COMPLETE\_RESTORE\_INFO:完全恢复出厂值控制参数

```
struct{
    DWORD        dwSize;
    DWORD        dwChannel;
    BYTE          byRes[64];
}NET_DVR_COMPLETE_RESTORE_INFO,*LPNET_DVR_COMPLETE_RESTORE_INFO;
```

### Members

*dwSize*

结构体大小

*dwChannel*

设备通道号，目前该参数无效

*byRes*

保留，置为 0

**Remarks**

远程控制设备完全恢复出厂默认值，包括设备 IP 地址会恢复成 192.0.0.64。只有 admin 管理员账号才支持该功能。

完全恢复之后设备会恢复为未激活状态，调用 [NET\\_DVR\\_ActivateDevice](#) 可以激活设备。

**8.41 NET\_DVR\_CURTRIGGERMODE: 当前触发模式**

```
struct{
    DWORD    dwSize;
    DWORD    dwTriggerType;
    BYTE     byRes[24];
}NET_DVR_CURTRIGGERMODE,*LPNET_DVR_CURTRIGGERMODE;
```

**Members***dwSize*

结构体大小

*dwTriggerType*触发类型，详见 [ITC\\_TRIGGERMODE\\_TYPE](#)*byRes*

保留

**8.42 NET\_DVR\_DATE: 日期信息结构体**

```
struct{
    WORD     wYear;
    BYTE     byMonth;
    BYTE     byDay;
}NET_DVR_DATE,*LPNET_DVR_DATE;
```

**Members***wYear*

年

*byMonth*

月

*byDay*

日

**8.43 NET\_DVR\_DDNSPARA\_V30: 网络应用参数（DDNS）**

```
struct{
```

```

BYTE    byEnableDDNS;
BYTE    byHostIndex;
BYTE    byRes1[2];
struct{
    BYTE    sUserName[NAME_LEN];
    BYTE    sPassword[PASSWD_LEN];
    BYTE    sDomainName[MAX_DOMAIN_NAME];
    BYTE    sServerName[MAX_DOMAIN_NAME];
    WORD    wDDNSPort;
    BYTE    byRes[10];
}struDDNS[MAX_DDNS_NUMS];
BYTE    byRes2[16];
}NET_DVR_DDNSPARA_V30,*LPNET_DVR_DDNSPARA_V30;

```

### Members

*byEnableDDNS*

是否使能：0- 否，1- 是

*byHostIndex*

0- Private DNS，1- Dyndns，2- PeanutHull(花生壳)，3- NO-IP，4- hiDDNS

*byRes1*

保留，置为 0

*sUsername*

DDNS 账号用户名

*sPassword*

DDNS 账号密码

*sDomainName*

域名

*sServerName*

DDNS 对应的服务器地址，可以是 IP 地址或域名

*wDDNSPort*

DDNS 端口

*byRes*

保留，置为 0

*byRes2*

保留，置为 0

## 8.44 NET\_DVR\_DEL\_FINGER\_PRINT\_MODE: 指纹删除处理方式联合体

```

union{
    BYTE                                uLen[588];
    NET\_DVR\_FINGER\_PRINT\_BYCARD        struByCard;
    NET\_DVR\_FINGER\_PRINT\_BYREADER      struByReader;
}NET_DVR_DEL_FINGER_PRINT_MODE,*LPNET_DVR_DEL_FINGER_PRINT_MODE;

```

### Members

*uLen*

联合体大小, 588 字节

*struByCard*

按卡号方式删除的处理方式

*struByReader*

按读卡器删除时的处理方式

## 8.45 NET\_DVR\_DECODERCFG\_V30:云台解码器(RS485)参数

```
struct{
    DWORD      dwSize;
    DWORD      dwBaudRate;
    BYTE       byDataBit;
    BYTE       byStopBit;
    BYTE       byParity;
    BYTE       byFlowcontrol;
    WORD       wDecoderType;
    WORD       wDecoderAddress;
    BYTE       bySetPreset[MAX_PRESET_V30];
    BYTE       bySetCruise[MAX_CRUISE_V30];
    BYTE       bySetTrack[MAX_TRACK_V30];
}NET_DVR_DECODERCFG_V30, *LPNET_DVR_DECODERCFG_V30;
```

### Members

*dwSize*

结构体大小

*dwBaudRate*

波特率(bps), 0-50, 1-75, 2-110, 3-150, 4-300, 5-600, 6-1200, 7-2400, 8-4800, 9-9600, 10-19200, 11-38400, 12-57600, 13-76800, 14-115.2k

*byDataBit*

数据有几位: 0-5 位, 1-6 位, 2-7 位, 3-8 位

*byStopBit*

停止位: 0-1 位, 1-2 位

*byParity*

是否校验: 0-无校验, 1-奇校验, 2-偶校验

*byFlowcontrol*

是否流控: 0-无, 1-软流控, 2-硬流控

*wDecoderType*

解码器类型, 通过 [NET\\_DVR\\_GetPTZProtocol](#) 获取, 该值对应于结构 [NET\\_DVR\\_PTZ\\_PROTOCOL](#) 中的

*dwType*

*wDecoderAddress*

解码器地址: [0,255]

*bySetPreset*

预置点是否设置: 0-没有设置, 1-设置

*bySetCruise*

巡航是否设置: 0-没有设置, 1-设置



*bySetTrack*

轨迹是否设置：0-没有设置，1-设置

**Remarks**

在早前的设备中规定了一系列云台协议，但在后期的设备仅保留一部分常用的协议，所以在配置解码器类型时必须调用 [NET\\_DVR\\_GetPTZProtocol](#) 获取当前设备支持的云台协议。

## 8.46 NET\_DVR\_DEVICECFG\_V40:设备参数

```
struct{
    DWORD    dwSize;
    BYTE      sDVRName[NAME_LEN];
    DWORD    dwDVRID;
    DWORD    dwRecycleRecord;
    BYTE      sSerialNumber[SERIALNO_LEN];
    DWORD    dwSoftwareVersion;
    DWORD    dwSoftwareBuildDate;
    DWORD    dwDSPSoftwareVersion;
    DWORD    dwDSPSoftwareBuildDate;
    DWORD    dwPanelVersion;
    DWORD    dwHardwareVersion;
    BYTE      byAlarmInPortNum;
    BYTE      byAlarmOutPortNum;
    BYTE      byRS232Num;
    BYTE      byRS485Num;
    BYTE      byNetworkPortNum;
    BYTE      byDiskCtrlNum;
    BYTE      byDiskNum;
    BYTE      byDVRType;
    BYTE      byChanNum;
    BYTE      byStartChan;
    BYTE      byDecordChans;
    BYTE      byVGANum;
    BYTE      byUSBNum;
    BYTE      byAuxoutNum;
    BYTE      byAudioNum;
    BYTE      byIPChanNum;
    BYTE      byZeroChanNum;
    BYTE      bySupport;
    BYTE      byEsataUseage;
    BYTE      byIPCPlug;
    BYTE      byStorageMode;
    BYTE      bySupport1;
    WORD     wDevType;
    BYTE      byDevTypeName[24];
}
```

```

    BYTE        byRes2[16];
}NET_DVR_DEVICECFG_V40,*LPNET_DVR_DEVICECFG_V40;

```

### Members

*dwSize*

结构体大小

*sDVRName*

设备名称

*dwDVRID*

设备 ID 号，用于遥控器，v1.4 的设备号范围为(0-99), v1.5 及以上版本的设备号为(0-255)

*dwRecycleRecord*

是否循环录像：0—不是；1—是

### 以下参数不可更改

*sSerialNumber*

设备序列号

*dwSoftwareVersion*

软件版本号，V3.0 以上版本支持的设备最高 8 位为主版本号，次高 8 位为次版本号，低 16 位为修复版本号；V3.0 以下版本支持的设备高 16 位表示主版本，低 16 位表示次版本

*dwSoftwareBuildDate*

软件生成日期，0xYYYYMMDD

*dwDSPSoftwareVersion*

DSP 软件版本，高 16 位是主版本，低 16 位是次版本

*dwDSPSoftwareBuildDate*

DSP 软件生成日期，0xYYYYMMDD

*dwPanelVersion*

前面板版本，高 16 位是主版本，低 16 位是次版本

*dwHardwareVersion*

硬件版本，高 16 位是主版本，低 16 位是次版本

*byAlarmInPortNum*

设备报警输入个数

*byAlarmOutPortNum*

设备报警输出个数

*byRS232Num*

设备 232 串口个数

*byRS485Num*

设备 485 串口个数

*byNetworkPortNum*

网络口个数

*byDiskCtrlNum*

硬盘控制器个数

*byDiskNum*

硬盘个数

*byDVRType*

设备类型

*byChanNum*

设备模拟通道个数

*byStartChan*

起始通道号

*byDecordChans*

设备解码路数

*byVGANum*

VGA 口的个数

*byUSBNum*

USB 口的个数

*byAuxoutNum*

辅口的个数

*byAudioNum*

语音口的个数

*byIPChanNum*

最大数字通道

*byZeroChanNum*

零通道编码个数

*bySupport*

能力，位与结果为 0 表示不支持，1 表示支持

*bySupport* & 0x1，表示是否支持智能搜索

*bySupport* & 0x2，表示是否支持备份

*bySupport* & 0x4，表示是否支持压缩参数能力获取

*bySupport* & 0x8，表示是否支持双网卡

*bySupport* & 0x10，表示支持远程 SADP

*bySupport* & 0x20，表示支持 Raid 卡功能

*bySupport* & 0x40，表示支持 IPSAN 搜索

*bySupport* & 0x80，表示支持 rtp over rtsp

*byEsataUseage*

Esata 的默认用途，0-默认备份，1-默认录像

*byIPCPlug*

0-不支持即插即用，1-支持即插即用

*byStorageMode*

0-盘组模式，1-磁盘配额

*bySupport1*

能力集扩充，位与结果为 0 表示不支持，1 表示支持

*bySupport1* & 0x1，表示是否支持 snmp v30

*bySupport1* & 0x2，支持区分回放和下载

*wDevType*

设备型号

*byDevTypeName*

设备型号名称

*byRes2*

保留，置为 0

Remarks

如果 byDVRType 是 0，则接口中解析 wDevType 作为设备型号，设备端同时将设备型号的名称传过来。

如果 byDVRType 不是 0，则接口中将不解析 wDevType 及 byDevTypeName，使用已有的设备型号及名称对 byDVRType、wDevType、byDevTypeName 进行填充，其中 byDVRType=wDevType。

建议开发时使用 wDevType、byDevTypeName，而不要使用 byDVRType，sdk 内部兼容。

byDVRType 和 wDevType 取值定义如下所示：

宏定义	宏定义值	设备类型
DS_K260X	850	门禁主机

## 8.47 NET\_DVR\_DEVICEINFO\_V30:设备参数

```
struct{
    BYTE    sSerialNumber[SERIALNO_LEN];
    BYTE    byAlarmInPortNum;
    BYTE    byAlarmOutPortNum;
    BYTE    byDiskNum;
    BYTE    byDVRType;
    BYTE    byChanNum;
    BYTE    byStartChan;
    BYTE    byAudioChanNum;
    BYTE    byIPChanNum;
    BYTE    byZeroChanNum;
    BYTE    byMainProto;
    BYTE    bySubProto;
    BYTE    bySupport;
    BYTE    bySupport1;
    BYTE    bySupport2;
    WORD    wDevType;
    BYTE    bySupport3;
    BYTE    byMultiStreamProto;
    BYTE    byStartDChan;
    BYTE    byStartDTalkChan;
    BYTE    byHighDChanNum;
    BYTE    byRes2[11];
}NET_DVR_DEVICEINFO_V30,*LPNET_DVR_DEVICEINFO_V30;
```

### Members

*sSerialNumber*

序列号

*byAlarmInPortNum*

报警输入个数

*byAlarmOutPortNum*

报警输出个数

*byDiskNum*

硬盘个数

*byDVRType*

设备类型，详见下文列表

*byChanNum*

设备模拟通道个数，数字（IP）通道最大个数通过 NET\_DVR\_GetDVRConfig（配置命令 NET\_DVR\_GET\_IPPARACFG\_V40）获取（dwDChanNum）。

*byStartChan*

模拟通道的起始通道号，目前设备模拟通道号从 1 开始，数字通道的起始通道号通过 NET\_DVR\_GetDVRConfig（配置命令 NET\_DVR\_GET\_IPPARACFG\_V40）获取（dwStartDChan）。

*byAudioChanNum*

设备语音通道数

*byIPChanNum*

设备最大数字通道个数，低 8 位，高 8 位见 byHighDChanNum。可以根据 IP 通道个数来判断是否调用 NET\_DVR\_GetDVRConfig（配置命令 NET\_DVR\_GET\_IPPARACFG\_V40）获取模拟和数字通道相关参数（NET\_DVR\_IPPARACFG\_V40）。

*byZeroChanNum*

零通道编码个数

*byMainProto*

主码流传输协议类型：0-private，1-rtsp

*bySubProto*

子码流传输协议类型：0-private，1-rtsp

*bySupport*

能力，位与结果为 0 表示不支持，1 表示支持

bySupport & 0x1，表示是否支持智能搜索

bySupport & 0x2，表示是否支持备份

bySupport & 0x4，表示是否支持压缩参数能力获取

bySupport & 0x8，表示是否支持双网卡

bySupport & 0x10，表示支持远程 SADP

bySupport & 0x20，表示支持 Raid 卡功能

bySupport & 0x40，表示支持 IPSAN 目录查找

bySupport & 0x80，表示支持 rtp over rtsp

*bySupport1*

能力集扩充，位与结果为 0 表示不支持，1 表示支持

bySupport1 & 0x1，表示是否支持 snmp v30

bySupport1 & 0x2，表示是否支持区分回放和下载

bySupport1 & 0x4，表示是否支持布防优先级

bySupport1 & 0x8，表示智能设备是否支持布防时间段扩展

bySupport1 & 0x10，表示是否支持多磁盘数（超过 33 个）

bySupport1 & 0x20，表示是否支持 rtsp over http

bySupport1 & 0x40，表示是否支持延时预览

bySupport1 & 0x80，表示是否支持车牌新报警信息

*bySupport2*

能力集扩充，位与结果为 0 表示不支持，1 表示支持

bySupport2 & 0x1，表示解码器是否支持通过 URL 取流解码

**wDevType**

设备型号，详见下文列表

**bySupport3**

能力集扩展，位与结果为 0 表示不支持，1 表示支持

bySupport3 & 0x1，表示是否支持多码流

**byMultiStreamProto**

多码流是否支持 rtsp 协议取流，按位表示，0-不支持，1-支持：bit0- 码流 3，bit1- 码流 4，依次类推

**byStartDChan**

起始数字通道号，0 表示无效

**byStartDTalkChan**

起始数字对讲通道号，区别于模拟对讲通道号，0 表示无效

**byHighDChanNum**

数字通道个数，高 8 位

**byRes2**

保留，置为 0

**Remarks**

如果 byDVRType 是 0，则接口中解析 wDevType 作为设备型号；如果 byDVRType 非 0，则接口中 byDVRType 和 wDevType 值相等，都是 byDVRType。推荐使用 wDevType 作为设备类型。

byDVRType 和 wDevType 取值定义如下所示：

宏定义	宏定义值	设备类型
DS_K260X	850	门禁主机

## 8.48 NET\_DVR\_DEVICEINFO\_V40:设备参数

```
struct{
    NET_DVR_DEVICEINFO_V30    struDeviceV30;
    BYTE                      bySupportLock;
    BYTE                      byRetryLoginTime;
    BYTE                      byPasswordLevel;
    BYTE                      byRes1;
    DWORD                    dwSurplusLockTime;
    BYTE                      byCharEncodeType;
    BYTE                      byRes2[255];
}NET_DVR_DEVICEINFO_V40,*LPNET_DVR_DEVICEINFO_V40;
```

**Members****struDeviceV30**

设备参数

**bySupportLock**

设备是否支持锁定功能，bySupportLock 为 1 时，dwSurplusLockTime 和 byRetryLoginTime 有效

**byRetryLoginTime**

剩余可尝试登陆的次数，用户名、密码错误时，此参数有效

**byPasswordLevel**

密码安全等级：0- 无效，1- 默认密码，2- 有效密码，3- 风险较高的密码，当管理员用户的密码为出厂默认密码（12345）或者风险较高的密码时，建议上层客户端提示用户更改密码

**byRes1**

保留，置为 0

**dwSurplusLockTime**

剩余时间，单位：秒，用户锁定时此参数有效。在锁定期间，用户尝试登陆，不管用户名密码输入对错，设备锁定剩余时间重新恢复到 30 分钟

**byCharEncodeType**

字符编码类型（SDK 所有接口返回的字符串编码类型，透传接口除外）：0- 无字符编码信息(老设备)，1- GB2312(简体中文)，2- GBK，3- BIG5(繁体中文)，4- Shift\_JIS(日文)，5- EUC-KR(韩文)，6- UTF-8，7- ISO8859-1，8- ISO8859-2，9- ISO8859-3，...，依次类推，21- ISO8859-15(西欧)

**byRes2**

保留，置为 0

**Remarks**

将密码输入分为数字(0~9)、小写字母(a~z)、大写字母(A~Z)、特殊符号（:\|除外）4 类，等级分为 4 个等级，如下所示：

- 等级 0（风险密码）：密码长度小于 8 位，或者只包含 4 类字符中的任意一类，或者密码与用户名一样，或者密码是用户名的倒写。例如：12345、abcdef。
- 等级 1（弱密码）：包含两类字符，且组合为（数字+小写字母）或（数字+大写字母），且长度大于等于 8 位。例如：abc12345、123ABCDEF。
- 等级 2（中密码）：包含两类字符，且组合不能为（数字+小写字母）和（数字+大写字母），且长度大于等于 8 位。例如：12345\*\*\*++、ABCDabcd。
- 等级 3（强密码）：包含三类字符及以上，且长度大于等于 8 位。例如：Abc12345、abc12345++。

## 8.49 NET\_DVR\_DOOR\_CFG:门参数配置结构体

```
struct{
    DWORD    dwSize;
    BYTE     byDoorName[DOOR_NAME_LEN];
    BYTE     byMagneticType;
    BYTE     byOpenButtonType;
    BYTE     byOpenDuration;
    BYTE     byDisabledOpenDuration;
    BYTE     byMagneticAlarmTimeout;
    BYTE     byEnableDoorLock;
    BYTE     byEnableLeaderCard;
    BYTE     byLeaderCardMode;
    DWORD    dwLeaderCardOpenDuration;
    BYTE     byStressPassword[STRESS_PASSWORD_LEN];
    BYTE     bySuperPassword[SUPER_PASSWORD_LEN];
    BYTE     byUnlockPassword[UNLOCK_PASSWORD_LEN];
    BYTE     byUseLocalController;
    BYTE     byRes1;
```

```

WORD    wLocalControllerID;
WORD    wLocalControllerDoorNumber;
WORD    wLocalControllerStatus;
BYTE    byLockInputCheck;
BYTE    byLockInputType;
BYTE    byDoorTerminalMode;
BYTE    byOpenButton;
BYTE    byRes2[44];
}NET_DVR_DOOR_CFG,*LPNET_DVR_DOOR_CFG;

```

## Members

*dwSize*

结构体大小

*byDoorName*

门名称

*byMagneticType*

门磁类型：0- 常闭，1- 常开

*byOpenButtonType*

开门按钮类型：0- 常闭，1- 常开

*byOpenDuration*

开门持续时间，取值范围：1~255s

*byDisabledOpenDuration*

残疾人卡开门持续时间，取值范围：1~255s

*byMagneticAlarmTimeout*

门磁检测超时报警时间，取值范围：0~255s，0 表示不报警

*byEnableDoorLock*

是否启用闭门回锁：0- 否，1- 是

*byEnableLeaderCard*

是否启用首卡常开功能：0- 否，1- 是

*byLeaderCardMode*

首卡模式，0-不启用首卡功能，1-首卡常开模式，2-首卡授权模式（使用了此字段，则 byEnableLeaderCard 无效）

*dwLeaderCardOpenDuration*

首卡常开持续时间，取值范围：1~1440，单位：min（分钟）

*byStressPassword*

胁迫密码

*bySuperPassword*

超级密码

*byUnlockPassword*

解除码，解锁密码

*byUseLocalController*

只读，是否连接在就地控制器上，0-否，1-是

*byRes1*

保留，置为 0

*wLocalControllerID*



只读，就地控制器序号，byUseLocalController=1 时有效，1-64,0 代表未注册

*wLocalControllerDoorNumber*

只读，就地控制器的门编号，byUseLocalController=1 时有效，1-4,0 代表未注册

*wLocalControllerStatus*

只读，byUseLocalController=1 时有效，就地控制器在线状态：0-离线，1-网络在线，2-环路 1 上的 RS485 串口 1，3-环路 1 上的 RS485 串口 2，4-环路 2 上的 RS485 串口 1，5-环路 2 上的 RS485 串口 2，6-环路 3 上的 RS485 串口 1，7-环路 3 上的 RS485 串口 2，8-环路 4 上的 RS485 串口 1，9-环路 4 上的 RS485 串口 2（只读）

*byLockInputCheck*

是否启用门锁输入检测(1 字节，0 不启用，1 启用，默认不启用)

*byLockInputType*

门锁输入类型(1 字节，0 常闭，1 常开，默认常闭)

*byDoorTerminalMode*

门相关端子工作模式(1 字节，0 防剪防短，1 普通，默认防剪防短)

*byOpenButton*

是否启用开门按钮(0 是，1 否，默认是)

*byRes2*

保留，置为 0

#### Remarks

门参数能力，对应门禁主机能力集（接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：ACS\_ABILITY）中节点<Door>。

## 8.50 NET\_DVR\_DOOR\_STATUS\_PLAN:门状态计划配置结构体

```
struct{
    DWORD        dwSize;
    DWORD        dwTemplateNo;
    BYTE         byRes[64];
}NET_DVR_DOOR_STATUS_PLAN,*LPNET_DVR_DOOR_STATUS_PLAN;
```

#### Members

*dwSize*

结构体大小

*dwTemplateNo*

计划模板编号，为 0 表示取消关联、恢复默认状态（普通状态）

*byRes*

保留，置为 0

## 8.51 NET\_DVR\_ETHERNET\_MULTI:单个网卡配置信息

```
struct{
    NET\_DVR\_IPADDR    struDVRIP;
    NET\_DVR\_IPADDR    struDVRIPMask;
```

```

DWORD          dwNetInterface;
BYTE           byCardType;
BYTE           byRes1;
WORD           wMTU;
BYTE           byMACAddr[MACADDR\_LEN];
BYTE           byRes2[2];
BYTE           byUseDhcp;
BYTE           byRes3[3];
NET\_DVR\_IPADDR struGatewayIpAddr;
NET\_DVR\_IPADDR struDnsServer1IpAddr;
NET\_DVR\_IPADDR struDnsServer2IpAddr;
}NET_DVR_ETHERNET_MULTI, *LPNET_DVR_ETHERNET_MULTI;

```

### Members

#### *struDVRIP*

设备 IP 地址

#### *struDVRIPMask*

设备 IP 地址掩码

#### *dwNetInterface*

网络接口：1-10MBase-T；2-10MBase-T 全双工；3-100MBase-TX；4-100M 全双工；5-10M/100M/1000M 自适应

#### *byCardType*

网卡类型：0-普通网卡，1-内网网卡，2-外网网卡

#### *byRes1*

保留

#### *wMTU*

MTU 设置，默认 1500

#### *byMACAddr*

设备物理地址，只读

#### *byRes2*

保留

#### *byUseDhcp*

是否启用 DHCP

#### *byRes3*

保留

#### *struGatewayIpAddr*

网关地址

#### *struDnsServer1IpAddr*

域名服务器 1 的 IP 地址

#### *struDnsServer2IpAddr*

域名服务器 2 的 IP 地址

### Remarks

MTU 的设置范围为 500-9676，若 MTU 设置过小客户端将无法注册到设备，并且客户端预览、回放、配置参数也会失败。

## 8.52 NET\_DVR\_ETHERNET\_V30:以太网配置

```
struct{
    NET_DVR_IPADDR    struDVRIP;
    NET_DVR_IPADDR    struDVRIPMask;
    DWORD             dwNetInterface;
    WORD              wDVRPort;
    WORD              wMTU;
    BYTE              byMACAddr[MACADDR_LEN];
    BYTE              byRes[2];
}NET_DVR_ETHERNET_V30, *LPNET_DVR_ETHERNET_V30;
```

### Members

*struDVRIP*

设备 IP 地址

*struDVRIPMask*

设备 IP 地址掩码

*dwNetInterface*

网络接口：1-10MBase-T；2-10MBase-T 全双工；3-100MBase-TX；4-100M 全双工；5-10M/100M/1000M 自适应；6-1000M 全双工

*wDVRPort*

设备端口号

*wMTU*

MTU 设置，默认 1500

*byMACAddr*

设备物理地址

*byRes*

保留

### Remarks

MTU 的设置范围为 500-9676，若 MTU 设置过小客户端将无法注册到设备，并且客户端预览、回放、配置参数也会失败。

## 8.53 NET\_DVR\_EVENT\_CARD\_LINKAGE\_CFG:事件/卡号联动配置结构体

```
struct{
    DWORD             dwSize;
    BYTE              byProMode;
    BYTE              byRes1[3];
    DWORD             dwEventSourceID;
    NET_DVR_EVENT_CARD_LINKAGE_UNION uLinkageInfo;
    BYTE              byAlarmout[MAX_ALARMHOST_ALARMOUT_NUM];
    BYTE              byRes2[32];
    BYTE              byOpenDoor[MAX_DOOR_NUM_256];
    BYTE              byCloseDoor[MAX_DOOR_NUM_256];
}
```

```

BYTE                byNormalOpen[MAX\_DOOR\_NUM\_256];
BYTE                byNormalClose[MAX\_DOOR\_NUM\_256];
BYTE                byMainDevBuzzer;
BYTE                byCapturePic;
BYTE                byRecordVideo;
BYTE                byRes3[29];
BYTE                byReaderBuzzer[MAX\_CARD\_READER\_NUM\_512];
BYTE                byRes[128];
}NET_DVR_EVENT_CARD_LINKAGE_CFG, *LPNET_DVR_EVENT_CARD_LINKAGE_CFG;

```

## Members

*dwSize*

结构体大小

*byProMode*

联动方式：0- 事件，1- 卡号

*byRes1*

保留，置为 0

*dwEventSourceID*

事件源 ID，0xffffffff 表示联动全部，其他取值：当主类型为设备事件时无效；当主类型是为门事件时，为门编号；当主类型为读卡器事件时，为读卡器 ID；当主类型为报警输入事件时，为防区报警输入 ID 或事件报警输入 ID

*uLinkageInfo*

联动方式参数

*byAlarmout*

关联的报警输出号，按数组表示，数组下标表示报警输出口序号，数组值：0- 不关联，1- 关联

*byRes2*

保留，置为 0

*byOpenDoor*

是否联动开门，按数组表示，数组下标表示门编号，数组值：0- 不联动，1- 联动

*byCloseDoor*

是否联动关门，按数组表示，数组下标表示门编号，数组值：0- 不联动，1- 联动

*byNormalOpen*

是否联动常开，按数组表示，数组下标表示门编号，数组值：0- 不联动，1- 联动

*byNormalClose*

是否联动常关，按数组表示，数组下标表示门编号，数组值：0- 不联动，1- 联动

*byMainDevBuzzer*

是否联动主机蜂鸣器：0- 不联动，1- 联动输出

*byCapturePic*

是否联动抓拍：0- 不联动抓拍，1- 联动抓拍

*byRecordVideo*

是否联动录像，0-不联动录像，1-联动录像（该字段暂未使用）

*byRes3*

保留，置为 0

*byReaderBuzzer*

是否联动读卡器蜂鸣器，按数组表示，数组下标表示读卡器编号，数组值：0-不联动，1-联动

*byRes*

保留，置为 0

#### Remarks

设备是否支持事件/卡号联动配置或者支持的参数能力，可以通过设备能力集进行判断，对应门禁主机能力集(**AcsAbility**)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：ACS\_ABILITY，节点：**<EventLinkage>**。

## 8.54 NET\_DVR\_EVENT\_CARD\_LINKAGE\_CFG\_V50:事件/卡号联动配置结构体

```
struct{
    DWORD                                dwSize;
    BYTE                                byProMode;
    BYTE                                byRes1[3];
    DWORD                                dwEventSourceID;
    NET\_DVR\_EVETN\_CARD\_LINKAGE\_UNION    uLinkageInfo;
    BYTE                                byAlarmout[MAX\_ALARMHOST\_ALARMOUT\_NUM];
    BYTE                                byRes2[32];
    BYTE                                byOpenDoor[MAX\_DOOR\_NUM\_256];
    BYTE                                byCloseDoor[MAX\_DOOR\_NUM\_256];
    BYTE                                byNormalOpen[MAX\_DOOR\_NUM\_256];
    BYTE                                byNormalClose[MAX\_DOOR\_NUM\_256];
    BYTE                                byMainDevBuzzer;
    BYTE                                byCapturePic;
    BYTE                                byRecordVideo;
    BYTE                                byRes3[29];
    BYTE                                byReaderBuzzer[MAX\_CARD\_READER\_NUM\_512];
    BYTE                                byAlarmOutClose[MAX\_ALARMHOST\_ALARMOUT\_NUM];
    BYTE                                byAlarmInSetup[MAX\_ALARMHOST\_ALARMIN\_NUM];
    BYTE                                byAlarmInClose[MAX\_ALARMHOST\_ALARMIN\_NUM];
    BYTE                                byRes[500];
} NET_DVR_EVENT_CARD_LINKAGE_CFG_V50, *LPNET_DVR_EVENT_CARD_LINKAGE_CFG_V50;
```

#### Members

*dwSize*

结构体大小

*byProMode*

联动方式：0- 事件，1- 卡号

*byRes1*

保留，置为 0

*dwEventSourceID*

事件源 ID，0xffffffff 表示联动全部，其他取值：当主类型为设备事件时无效；当主类型是为门事件时，为门编号；当主类型为读卡器事件时，为读卡器 ID；当主类型为报警输入事件时，为防区报警输入 ID 或事件报警输入 ID

***uLinkageInfo***

联动方式参数

***byAlarmout***

关联的报警输出号，按数组表示，数组下标表示报警输出口序号，数组值：0- 不关联，1- 关联

***byRes2***

保留，置为 0

***byOpenDoor***

是否联动开门，按数组表示，数组下标表示门编号，数组值：0- 不联动，1- 联动

***byCloseDoor***

是否联动关门，按数组表示，数组下标表示门编号，数组值：0- 不联动，1- 联动

***byNormalOpen***

是否联动常开，按数组表示，数组下标表示门编号，数组值：0- 不联动，1- 联动

***byNormalClose***

是否联动常关，按数组表示，数组下标表示门编号，数组值：0- 不联动，1- 联动

***byMainDevBuzzer***

是否联动主机蜂鸣器：0- 不联动，1- 联动输出

***byCapturePic***

是否联动抓拍：0- 不联动抓拍，1- 联动抓拍

***byRecordVideo***

是否联动录像，0-不联动录像，1-联动录像

***byRes3***

保留，置为 0

***byReaderBuzzer***

是否联动读卡器蜂鸣器，按数组表示，数组下标表示读卡器编号，数组值：0-不联动，1-联动

***byAlarmOutClose***

关联报警输出关闭，按字节表示，为 0 表示不关联，为 1 表示关联

***byAlarmInSetup***

关联防区布防，按字节表示，为 0 表示不关联，为 1 表示关联

***byAlarmInClose***

关联防区撤防，按字节表示，为 0 表示不关联，为 1 表示关联

***byRes***

保留，置为 0

**Remarks**

设备是否支持事件/卡号联动配置或者支持的参数能力，可以通过设备能力集进行判断，对应门禁主机能力集(**AcsAbility**)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：ACS\_ABILITY，节点：**<EventLinkage>**。

## 8.55 NET\_DVR\_EVENT\_CARD\_LINKAGE\_COND:事件/卡号联动配置条件结构体

```
struct{
    DWORD    dwSize;
    DWORD    dwEventID;
```

```

    WORD    wLocalControllerID;
    BYTE    byRes[106];
} NET_DVR_EVENT_CARD_LINKAGE_COND, *LPNET_DVR_EVENT_CARD_LINKAGE_COND;

```

### Members

*dwSize*

结构体大小

*dwEventID*

事件 ID，从 1 开始，设置不同的事件/卡号联动配置时依次递增即可

*wLocalControllerID*

就地控制器序号[1,64]，0 表示门禁主机（目前设备只支持门禁主机）

*byRes*

保留，置为 0

## 8.56 NET\_DVR\_EVETN\_CARD\_LINKAGE\_UNION: 事件/卡号联动方式联合体

```

union{
    BYTE                                byCardNo[ACS_CARD_NO_LEN];
    NET_DVR_EVENT_LINKAGE_INFO        struEventLinkage;
}NET_DVR_EVETN_CARD_LINKAGE_UNION,*LPNET_DVR_EVETN_CARD_LINKAGE_UNION;

```

### Members

*byCardNo*

卡号，byProMode 为 1 时有效

*struEventLinkage*

事件联动参数，byProMode 为 0 时有效

## 8.57 NET\_DVR\_EVENT\_LINKAGE\_INFO: 事件联动参数结构体

```

struct{
    WORD    wMainEventType;
    WORD    wSubEventType;
    BYTE    byRes[28];
}NET_DVR_EVENT_LINKAGE_INFO,*LPNET_DVR_EVENT_LINKAGE_INFO;

```

### Members

*wMainEventType*

事件主类型：0- 设备事件，1- 报警输入事件，2- 门事件，3- 读卡器事件

*wSubEventType*

事件次类型，不同的主类型对应不同的次类型，详见“Remarks”说明

*byRes*

保留，置为 0

### Remarks

不同的事件主类型对应不同的事件次类型，具体类型及含义请见《设备网络 SDK 使用手册.chm》该结构体页面。

## 8.58 NET\_DVR\_EXCEPTION\_V30:异常参数

```
struct{
    DWORD dwSize;
    NET_DVR_HANDLEEXCEPTION_V30 struExceptionHandleType[MAX_EXCEPTIONNUM_V30];
}NET_DVR_EXCEPTION_V30,*LPNET_DVR_EXCEPTION_V30;
```

### Members

*dwSize*

结构体大小

*struExceptionHandleType*

异常信息处理方式:

数组 0—硬盘满

数组 1—硬盘出错

数组 2—网线断

数组 3—IP 地址冲突

数组 4—非法访问

数组 5—输入/输出视频制式不匹配

数组 6—视频信号异常

数组 7—录像异常

## 8.59 NET\_DVR\_EXCEPTION\_V40:异常参数配置（扩展）结构体

```
struct{
    DWORD dwSize;
    DWORD dwMaxGroupNum;
    NET_DVR_HANDLEEXCEPTION_V41 struExceptionHandle[MAX_EXCEPTIONNUM_V30];
    BYTE byRes[128];
}NET_DVR_EXCEPTION_V40,*LPNET_DVR_EXCEPTION_V40;
```

### Members

*dwSize*

结构体大小

*dwMaxGroupNum*

设备支持的异常类型最大组数（只读）

*struExceptionHandle*

异常信息处理方式，数组的每个元素都表示一种异常类型：

数组 0—硬盘满

数组 1—硬盘出错

数组 2—网线断

数组 3—IP 地址冲突

数组 4—非法访问

数组 5—输入/输出视频制式不匹配

数组 6—视频信号异常

数组 7—录像异常



数组 8-阵列异常  
 数组 9-前端/录像分辨率不匹配异常  
 数组 10-行车超速（车载专用）  
 数组 11-热备异常（N+1 使用）  
 数组 12-温度异常  
 数组 13-子系统异常  
 数组 14-风扇异常  
 数组 15-POE 供电异常

*byRes*

保留，置为 0

## 8.60 NET\_DVR\_FACE\_PIPCFG:画中画参数结构体

```
struct{
    BYTE    byEnable;
    BYTE    byBackChannel;
    BYTE    byPosition;
    BYTE    byPIPDiv;
    BYTE    byRes[4];
}NET_DVR_FACE_PIPCFG,*LPNET_DVR_FACE_PIPCFG;
```

### Members

*byEnable*

是否开启画中画：0-否，1-是

*byBackChannel*

背景通道号（面板通道）

*byPosition*

叠加位置：0-左上，1-左下，2-右上，3-右下

*byPIPDiv*

分屏系数(人脸画面:面板画面)：0-1:4，1-1:9，2-1:16

*byRes*

保留，置为 0

## 8.61 NET\_DVR\_FACEDETECT\_RULECFG\_V41:人脸检测规则结构体

```
struct{
    DWORD    dwSize;
    BYTE    byEnable;
    BYTE    byEventType;
    BYTE    byUpLastAlarm;
    BYTE    byUpFacePic;
    BYTE    byRuleName[NAME_LEN];
    NET_VCA_POLYGON    struVcaPolygon;
    BYTE    byPicProType;
```

```

BYTE                bySensitivity;
DWORD               wDuration;
NET\_DVR\_JPEGPARA   struPictureParam;
NET\_VCA\_SIZE\_FILTER struSizeFilter;
NET\_DVR\_SCHEDTIME struAlarmTime[MAX\_DAYS][MAX\_TIMESEGMENT\_V30];
NET\_DVR\_HANDLEEXCEPTION\_V30 struHandleType;
BYTE                byRelRecordChan[MAX\_CHANNUM\_V30];
BYTE                byPicRecordEnable;
BYTE                byRes1;
WORD                wAlarmDelay;
NET\_DVR\_FACE\_PIPCFG struFacePIP;
WORD                wRelSnapChan[MAX\_REL\_SNAPCHAN\_NUM];
BYTE                byRes[22];
}NET_DVR_FACEDTECT_RULECFG_V41,*LPNET_DVR_FACEDTECT_RULECFG_V41;

```

## Members

*dwSize*

结构体大小

*byEnable*

是否启用人脸检测规则

*byEventType*

警戒事件类型， 0-异常人脸; 1-正常人脸;2-异常人脸&正常人脸

*byUpLastAlarm*

是否先上传最近一次的报警

*byUpFacePic*

是否上传人脸子图： 0- 否， 1- 是

*byRuleName*

规则名称

*struVcaPolygon*

人脸检测规则区域

*byPicProType*

报警时图片处理方式： 0-不处理， 非 0-上传

*bySensitivity*

规则灵敏度

*wDuration*

触发人脸报警时间阈值

*struPictureParam*

图片规格结构

*struSizeFilter*

尺寸过滤器

*struAlarmTime*

布防时间

*struHandleType*

处理方式

*byRelRecordChan*

报警触发的录像通道，为 1 表示触发该通道。按位表示通道，例如：byRelRecordChan[0]==1 表示触发通道 1 录像，byRelRecordChan[1]==1 表示触发通道 2 录像，依次类推

*byPicRecordEnable*

是否启用图片存储：0-不启用，1-启用

*byRes1*

保留，置为 0

*wAlarmDelay*

智能报警延时：0-5s，1-10s，2-30s，3-60s，4-120s，5-300s，6-600s

*struFacePIP*

画中画参数

*wRelSnapChan*

关联抓拍通道(当主通道报警时，同时会上传关联通道的抓拍图片)：0 表示不关联，其他值为关联通道号

*byRes*

保留，置为 0

#### Remarks

- 设备是否支持人脸检测智能功能或者支持的参数能力，可以通过设备能力集进行判断，对应智能通道分析能力集(**VcaChanAbility**)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：DEVICE\_ABILITY\_INFO，节点：<Face>中<FaceDetect>。

## 8.62 NET\_DVR\_FINGER\_PRINT\_BYCARD:按卡号方式删除指纹的处理方式结构体

```
struct{
    BYTE    byCardNo[ACS_CARD_NO_LEN];
    BYTE    byEnableCardReader[MAX_CARD_READER_NUM_512];
    BYTE    byFingerPrintID[MAX_FINGER_PRINT_NUM];
    BYTE    byRes1[34];
}NET_DVR_FINGER_PRINT_BYCARD, *LPNET_DVR_FINGER_PRINT_BYCARD;
```

#### Members

*byCardNo*

指纹关联的卡号

*byEnableCardReader*

指纹的读卡器是否有效，数组下标表示读卡器序号，数组值：0- 无效，1- 有效

*byFingerPrintID*

需要控制的指纹，数组下标表示指纹编号，数组值：0- 不删除，1- 删除

*byRes1*

保留，置为 0

## 8.63 NET\_DVR\_FINGER\_PRINT\_BYREADER:按按读卡器方式删除指纹的 处理方式结构体

```
struct{
    DWORD          dwCardReaderNo;
    BYTE           byClearAllCard;
    BYTE           byRes1[3];
    BYTE           byCardNo[ACS_CARD_NO_LEN];
    BYTE           byRes[100];
}NET_DVR_FINGER_PRINT_BYREADER, *LPNET_DVR_FINGER_PRINT_BYREADER;
```

### Members

*dwCardReaderNo*

读卡器编号

*byClearAllCard*

是否删除所有卡的指纹信息：0- 按卡号删除指纹信息，1- 删除所有卡的指纹信息

*byRes1*

保留，置为 0

*byCardNo*

指纹关联的卡号，byClearAllCard 为 0 时有效

*byRes*

保留，置为 0

## 8.64 NET\_DVR\_FINGER\_PRINT\_CFG:指纹参数配置结构体

```
struct{
    DWORD          dwSize;
    BYTE           byCardNo[ACS_CARD_NO_LEN];
    DWORD          dwFingerPrintLen;
    BYTE           byEnableCardReader[MAX_CARD_READER_NUM_512];
    BYTE           byFingerPrintID;
    BYTE           byFingerType;
    BYTE           byRes1[30];
    BYTE           byFingerData[MAX_FINGER_PRINT_LEN];
    BYTE           byRes[64];
}NET_DVR_FINGER_PRINT_CFG, *LPNET_DVR_FINGER_PRINT_CFG;
```

### Members

*dwSize*

结构体大小

*byCardNo*

指纹关联的卡号

*dwFingerPrintLen*

指纹数据长度

**byEnableCardReader**

需要下发指纹的读卡器，数组下标表示读卡器序号，数组值：0- 不下发，1- 下发

**byFingerPrintID**

指纹编号，有效值范围为 1~10

**byFingerType**

指纹类型：0- 普通指纹，1- 胁迫指纹

**byRes1**

保留，置为 0

**byFingerData**

指纹数据内容

**byRes**

保留，置为 0

**Remarks**

- 设备是否支持指纹参数配置或者支持的参数能力，可以通过设备能力集进行判断，对应门禁主机能力集(**AcsAbility**)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：ACS\_ABILITY，节点：<FingerPrint>。

## 8.65 NET\_DVR\_FINGER\_PRINT\_INFO\_COND: 指纹参数配置条件结构体

```
struct{
    DWORD          dwSize;
    BYTE           byCardNo[ACS_CARD_NO_LEN];
    BYTE           byEnableCardReader[MAX_CARD_READER_NUM_512];
    DWORD          dwFingerPrintNum;
    BYTE           byFingerPrintID;
    BYTE           byCallbackMode;
    BYTE           byRes1[26];
}NET_DVR_FINGER_PRINT_INFO_COND, *LPNET_DVR_FINGER_PRINT_INFO_COND;
```

**Members****dwSize**

结构体大小

**byCardNo**

指纹关联的卡号

**byEnableCardReader**

指纹的读卡器是否有效，数组下标表示读卡器序号，数组值：0- 无效，1- 有效

**dwFingerPrintNum**

设置或获指纹数量，获取时置为 0xffffffff 表示获取所有指纹信息

**byFingerPrintID**

指纹编号，有效值范围为 1~10，获取时置为 0xff 表示该卡所有指纹

**byCallbackMode**

设备回调方式：0- 已向所有读卡器下发完成，1- 在时间段内只下发了部分也返回

**byRes1**

保留，置为 0

**Remarks**

- 设备是否支持指纹参数配置或者支持的参数能力，可以通过设备能力集进行判断，对应门禁主机能力

集(**AcsAbility**), 相关接口: [NET\\_DVR\\_GetDeviceAbility](#), 能力集类型: ACS\_ABILITY, 节点: <FingerPrint>。

## 8.66 NET\_DVR\_FINGER\_PRINT\_INFO\_CTRL: 指纹删除控制参数结构体

```
struct{
    DWORD                dwSize;
    BYTE                 byMode;
    BYTE                 byRes1[3];
    NET\_DVR\_DEL\_FINGER\_PRINT\_MODE    struProcessMode;
    BYTE                 byRes[64];
}NET_DVR_FINGER_PRINT_INFO_CTRL, *LPNET_DVR_FINGER_PRINT_INFO_CTRL;
```

### Members

*dwSize*

结构体大小

*byMode*

删除方式: 0- 按卡号方式删除, 1- 按读卡器删除

*byRes1*

保留, 置为 0

*struProcessMode*

处理方式

*byRes*

保留, 置为 0

### Remarks

- 设备是否支持指纹参数配置或者支持的参数能力, 可以通过设备能力集进行判断, 对应门禁主机能力集(**AcsAbility**), 相关接口: [NET\\_DVR\\_GetDeviceAbility](#), 能力集类型: ACS\_ABILITY, 节点: <FingerPrint>。

## 8.67 NET\_DVR\_FINGER\_PRINT\_STATUS: 指纹状态参数结构体

```
struct{
    DWORD    dwSize;
    BYTE     byCardNo[ACS\_CARD\_NO\_LEN];
    BYTE     byCardReaderRecvStatus[MAX\_CARD\_READER\_NUM\_512];
    BYTE     byFingerPrintID;
    BYTE     byFingerType;
    BYTE     byTotalStatus;
    BYTE     byRes1;
    BYTE     byErrorMsg[ERROR\_MSG\_LEN];
    DWORD    dwCardReaderNo;
    BYTE     byRes[24];
}NET_DVR_FINGER_PRINT_STATUS, *LPNET_DVR_FINGER_PRINT_STATUS;
```

### Members

*dwSize*

结构体大小

*byCardNo*

指纹关联的卡号

*byCardReaderRecvStatus*

指纹读卡器状态，数组下标表示读卡器序号，数组值：0- 失败，1- 成功，2- 该指纹模组不在线，3- 重试或指纹质量差，4- 内存已满，5- 已存在该指纹，6- 已存在该指纹 ID，7- 非法指纹 ID，8- 该指纹模组无需配置

*byFingerPrintID*

指纹编号，有效值范围为 1~10

*byFingerType*

指纹类型：0- 普通指纹，1- 胁迫指纹

*byTotalStatus*

下发总的状态：0- 当前指纹未向所有读卡器下发完成，1- 当前指纹已向所有读卡器下发完成(指的是门禁主机往所有的读卡器下发了，不管成功与否)

*byRes1*

保留，置为 0

*byErrorMsg*

下发错误信息，当 *byCardReaderRecvStatus* 为 5 时，表示已存在指纹对应的卡号

*dwCardReaderNo*

指纹读卡器编号，可用于下发错误返回

*byRes*

保留，置为 0

#### Remarks

- 设备是否支持指纹参数配置或者支持的参数能力，可以通过设备能力集进行判断，对应门禁主机能力集(*AcsAbility*)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：ACS\_ABILITY，节点：<FingerPrint>。

## 8.68 NET\_DVR\_FIRE\_ALARM:消防报警信息

```
struct{
    DWORD                dwSize;
    NET\_DVR\_TIME\_V30    struAlarmTime;
    BYTE                 byRes[128];
}NET_DVR_FIRE_ALARM,*LPNET_DVR_FIRE_ALARM;
```

#### Members

*dwSize*

结构体大小

*struAlarmTime*

报警时间

*byRes*

保留，置为 0

## 8.69 NET\_DVR\_FTPCFG:FTP 上传参数

```
struct{
    DWORD        dwSize;
    DWORD        dwEnableFTP;
    char         sFTPIP[16];
    DWORD        dwFTPPort;
    BYTE         sUserName[NAME_LEN];
    BYTE         sPassword[PASSWD_LEN];
    DWORD        dwDirLevel;
    WORD         wTopDirMode;
    WORD         wSubDirMode;
    BYTE         byEnableAnony;
    BYTE         byRes[23];
}NET_DVR_FTPCFG, *LPNET_DVR_FTPCFG;
```

### Members

*dwSize*

结构体大小

*dwEnableFTP*

是否启用 FTP，0-不启用，1-启用

*sFTPIP*

FTP 服务器 IP 地址

*dwFTPPort*

FTP 端口号

*sUserName*

用户名

*sPassword*

密码

*dwDirLevel*

目录结构，0-不使用目录结构，直接保存在根目录，1-使用 1 级目录，2-使用 2 级目录

*wTopDirMode*

一级目录，0x1-使用设备名，0x2-使用设备号，0x3-使用设备 ip 地址，0x4-使用监测点，0x5-使用时间(年月)，0x6-自定义，0x7-违规类型，0x8-方向，0x9-地点

*wSubDirMode*

二级目录，0x1-使用通道名，0x2-使用通道号，0x3-使用时间(年月日)，0x4-使用车道号，0x5-自定义，0x6-违规类型，0x7-方向，0x8-地点

*byEnableAnony*

启用匿名：0- 否，1- 是，智能交通摄像机不支持该字段

*byRes*

保留



## 8.70 NET\_DVR\_GATE\_TIME\_CFG: 人员通道闸门时间参数结构体

```
struct{
    DWORD    dwSize;
    DWORD    dwHoldOnAlarmTime;
    DWORD    dwHoldOnGateOpenTime;
    DWORD    dwPostponeIntrusionAlarmTime;
    DWORD    dwNoLaneAccessTimeLimitTime;
    DWORD    dwSafetyZoneStayTime;
    BYTE     byRes[300];
}NET_DVR_GATE_TIME_CFG, *LPNET_DVR_GATE_TIME_CFG;
```

### Members

*dwSize*

结构体大小

*dwHoldOnAlarmTime*

延续报警器蜂鸣时间，单位：ms

*dwHoldOnGateOpenTime*

闸门收到关闭命令前继续保持打开状态时间，单位：ms

*dwPostponeIntrusionAlarmTime*

推迟触发闯入欺骗行为报警时间，单位：ms

*dwNoLaneAccessTimeLimitTime*

通道收到有效通行认证信号，但无人通行超时报警时间，单位：s

*dwSafetyZoneStayTime*

通道收到有效通行认证信号，乘客到达安全区后滞留通道超时报警时间，单位：s

*byRes*

保留，置为 0

### Remarks

- 设备是否支持人员通道闸门时间配置或者支持的参数能力，可以通过设备能力集进行判断，对应门禁主机能力集(**AcsAbility**)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：ACS\_ABILITY，节点：< GateTimeCfg >。

## 8.71 NET\_DVR\_GEOGLOCATION: 地址位置信息

```
struct{
    int      iRes[2];
    DWORD    dwCity;
}NET_DVR_GEOGLOCATION, *LPNET_DVR_GEOGLOCATION;
```

### Members

*iRes*

保留，置为 0

*dwCity*

城市

```
enum _PROVINCE_CITY_IDX_{
    ANHUI_PROVINCE           = 0,
    AOMEN_PROVINCE           = 1,
    BEIJING_PROVINCE         = 2,
    CHONGQING_PROVINCE       = 3,
    FUJIAN_PROVINCE          = 4,
    GANSU_PROVINCE           = 5,
    GUANGDONG_PROVINCE       = 6,
    GUANGXI_PROVINCE         = 7,
    GUIZHOU_PROVINCE         = 8,
    HAINAN_PROVINCE          = 9,
    HEBEI_PROVINCE           = 10,
    HENAN_PROVINCE           = 11,
    HEILONGJIANG_PROVINCE    = 12,
    HUBEI_PROVINCE           = 13,
    HUNAN_PROVINCE           = 14,
    JILIN_PROVINCE           = 15,
    JIANGSU_PROVINCE         = 16,
    JIANGXI_PROVINCE         = 17,
    LIAONING_PROVINCE        = 18,
    NEIMENGGU_PROVINCE       = 19,
    NINGXIA_PROVINCE         = 20,
    QINGHAI_PROVINCE         = 21,
    SHANDONG_PROVINCE        = 22,
    SHANXI_JIN_PROVINCE      = 23,
    SHANXI_SHAN_PROVINCE     = 24,
    SHANGHAI_PROVINCE        = 25,
    SICHUAN_PROVINCE         = 26,
    TAIWAN_PROVINCE          = 27,
    TIANJIN_PROVINCE         = 28,
    XIZANG_PROVINCE          = 29,
    XIANGGANG_PROVINCE       = 30,
    XINJIANG_PROVINCE        = 31,
    YUNNAN_PROVINCE          = 32,
    ZHEJIANG_PROVINCE        = 33
}PROVINCE_CITY_IDX
```

*ANHUI\_PROVINCE*

安徽

*AOMEN\_PROVINCE*

澳门

*BEIJING\_PROVINCE*

北京

*CHONGQING\_PROVINCE*

重庆

*FUJIAN\_PROVINCE*

福建

*GANSU\_PROVINCE*

甘肃

*GUANGDONG\_PROVINCE*

广东

*GUANGXI\_PROVINCE*

广西

*GUIZHOU\_PROVINCE*

贵州

*HAINAN\_PROVINCE*

海南

*HEBEI\_PROVINCE*

河北

*HENAN\_PROVINCE*

河南

*HEILONGJIANG\_PROVINCE*

黑龙江

*HUBEI\_PROVINCE*

湖北

*HUNAN\_PROVINCE*

湖南

*JILIN\_PROVINCE*

吉林

*JIANGSU\_PROVINCE*

江苏

*JIANGXI\_PROVINCE*

江西

*LIAONING\_PROVINCE*

辽宁

*NEIMENGGU\_PROVINCE*

内蒙古

*NINGXIA\_PROVINCE*

宁夏

*QINGHAI\_PROVINCE*

青海

*SHANDONG\_PROVINCE*

山东

*SHANXI\_JIN\_PROVINCE*

山西

*SHANXI\_SHAN\_PROVINCE*

陕西

*SHANGHAI\_PROVINCE*

上海

SICHUAN\_PROVINCE  
 四川  
 TAIWAN\_PROVINCE  
 台湾  
 TIANJIN\_PROVINCE  
 天津  
 XIZANG\_PROVINCE  
 西藏  
 XIANGGANG\_PROVINCE  
 香港  
 XINJIANG\_PROVINCE  
 新疆  
 YUNNAN\_PROVINCE  
 云南  
 ZHEJIANG\_PROVINCE  
 浙江

## 8.72 NET\_DVR\_GROUP\_CFG: 群组参数配置结构体

```

struct{
    DWORD                dwSize;
    BYTE                 byEnable;
    BYTE                 byRes1[3];
    NET\_DVR\_VALID\_PERIOD\_CFG struValidPeriodCfg;
    BYTE                 byGroupName[GROUP\_NAME\_LEN];
    BYTE                 byRes2[32];
}NET_DVR_GROUP_CFG,*LPNET_DVR_GROUP_CFG;
  
```

### Members

#### *dwSize*

结构体大小

#### *byEnable*

是否启用该群组: 0- 不启用, 1- 启用

#### *byRes1*

保留, 置为 0

#### *struValidPeriodCfg*

群组有效期参数

#### *byGroupName*

群组名称

#### *byRes2*

保留, 置为 0

## 8.73 NET\_DVR\_GROUP\_COMBINATION\_INFO:群组组合参数结构体

```
struct{
    BYTE        byEnable;
    BYTE        byMemberNum;
    BYTE        bySequenceNo;
    BYTE        byRes;
    DWORD       dwGroupNo;
}NET_DVR_GROUP_COMBINATION_INFO,*LPNET_DVR_GROUP_COMBINATION_INFO;
```

### Members

#### *byEnable*

是否启用该群组组合：0- 不启用，1- 启用

#### *byMemberNum*

刷卡成员数量

#### *bySequenceNo*

群组刷卡次序号

#### *byRes*

保留，置为 0

#### *dwGroupNo*

群组编号，0xffffffff 表示远程开门，0xffffffe 表示超级密码

### Remarks

多重卡刷卡开门功能：

有权限的任意 n 张卡刷卡之后才能开门，不限制先后次序，则只需要设置一个群组组合（卡号都配置关联该群组），byMemberNum 设为 n，dwTemplateNo 设为 1。

有权限的 n 张 A 类卡和 m 张 B 类卡刷卡之后才能开门，而且先刷 A 类卡再刷 B 类卡，则需要设置 2 个群组组合，2 个群组组合的 byMemberNum 分别为 n 和 m，dwTemplateNo 分别为 1、2；如果不需要现在刷卡先后次序，则 dwTemplateNo 都设为 0，0 表示无序。

## 8.74 NET\_DVR\_HANDLEEXCEPTION\_V30:报警和异常处理

```
struct{
    DWORD       dwHandleType;
    BYTE        byRelAlarmOut[MAX_ALARMOUT_V30];
}NET_DVR_HANDLEEXCEPTION_V30,*LPNET_DVR_HANDLEEXCEPTION_V30;
```

### Members

#### *dwHandleType*

处理方式：

0x00: 无响应

0x01: 监视器上警告

0x02: 声音警告

0x04: 上传中心

0x08: 触发报警输出

0x10: Jpeg 抓图并上传 Email

0x20: 无线声光报警器联动

0x200: 抓图并上传 ftp

*byRelAlarmOut*

报警触发的输出通道，0-不触发，1-触发输出，按位表示输出通道，例如 `byRelAlarmOut[0]==1` 表示触发输出通道 1，`byRelAlarmOut[1]==1` 表示触发输出通道 2，依此类推

## 8.75 NET\_DVR\_HANDLEEXCEPTION\_V41:报警和异常处理结构体

```
struct{
    DWORD        dwHandleType;
    DWORD        dwMaxAlarmOutChannelNum;
    DWORD        dwRelAlarmOut[MAX_ALARMOUT_V40];
    BYTE         byRes[64];
}NET_DVR_HANDLEEXCEPTION_V41,*LPNET_DVR_HANDLEEXCEPTION_V41;
```

### Members

*dwHandleType*

处理方式，各种异常处理方式的"或"结果，异常处理方式：

0x00: 无响应

0x01: 监视器上警告

0x02: 声音警告

0x04: 上传中心

0x08: 触发报警输出

0x10: Jpeg 抓图并上传 EMail

0x20: 无线声光报警器联动

0x40: 联动电子地图(目前仅 PCNVR 支持)

0x200: 抓图并上传 ftp E.g. `dwHandleType==0x01|0x04` 表示配置报警发生时联动监视器上警告并且将报警信息上传中心

0x400: 虚交侦测 联动 聚焦模式（提供可配置项，原先设备自动完成）IPC5.1.0

0x800: PTZ 联动跟踪(球机跟踪目标)

*dwMaxAlarmOutChannelNum*

设备最大支持的触发报警输出通道数（只读）

*byRelAlarmOut*

触发报警输出通道，数组元素值表示报警输出号（从 0 开始），例如：`dwRelAlarmOut[i]==3` 表示触发第 4 个报警输出通道。中间遇到 0xffffffff 则后续无效。

*byRes*

保留，置为 0

## 8.76 NET\_DVR\_HDCFG:设备硬盘配置结构体

```
struct{
    DWORD        dwSize;
    DWORD        dwHDCount;
    NET_DVR_SINGLE_HD struHDInfo[MAX_DISKNUM_V30];
}
```

```
}NET_DVR_HDCFG, *LPNET_DVR_HDCFG;
```

### Members

*dwSize*

结构体大小

*dwHDCount*

硬盘数，该参数只能获取，不支持设置

*struHDInfo*

硬盘信息参数

### Remarks

本结构体中的 *dwHDCount* 参数是指设备本地的硬盘数，因此只能获取该信息，不能设置。对硬盘的信息进行设置后需要重启设备才生效。

## 8.77 NET\_DVR\_HOLIDAY\_PLAN\_CFG:假日计划配置结构体

```
struct{
    DWORD                dwSize;
    BYTE                 byEnable;
    BYTE                 byRes1[3];
    NET_DVR_DATE         struBeginDate;
    NET_DVR_DATE         struEndDate;
    NET_DVR_SINGLE_PLAN_SEGMENT struPlanCfg[MAX_DAYS][MAX_TIMESEGMENT_V30];
    BYTE                 byRes2[16];
}NET_DVR_HOLIDAY_PLAN_CFG,*LPNET_DVR_HOLIDAY_PLAN_CFG;
```

### Members

*dwSize*

结构体大小

*byEnable*

是否使能：0- 否，1- 是

*byRes1*

保留，置为 0

*struBeginDate*

假日开始日期

*struEndDate*

假日结束日期

*struPlanCfg*

假日计划参数，一周 7 天，每天最多 8 个时间段

*byRes2*

保留，置为 0

## 8.78 NET\_DVR\_HOLIDAY\_PLAN\_COND:卡权限假日计划配置条件结构体

```
struct{
    DWORD    dwSize;
```

```

    DWORD    dwHolidayPlanNumber;
    WORD      wLocalControllerID;
    BYTE      byRes[106];
} NET_DVR_HOLIDAY_PLAN_COND,*LPNET_DVR_HOLIDAY_PLAN_COND;

```

### Members

*dwSize*

结构体大小

*dwHolidayPlanNumber*

假日计划编号

*wLocalControllerID*

就地控制器序号[1,64]，0 表示门禁主机

*byRes*

保留，置为 0

## 8.79 NET\_DVR\_HOLIDAY\_GROUP\_CFG:假日组配置结构体

```

struct{
    DWORD    dwSize;
    BYTE      byEnable;
    BYTE      byRes1[3];
    BYTE      byGroupName[HOLIDAY_GROUP_NAME_LEN];
    DWORD     dwHolidayPlanNo[MAX_HOLIDAY_PLAN_NUM];
    BYTE      byRes2[32];
}NET_DVR_HOLIDAY_GROUP_CFG,*LPNET_DVR_HOLIDAY_GROUP_CFG;

```

### Members

*dwSize*

结构体大小

*byEnable*

是否使能：0- 否，1- 是

*byRes1*

保留，置为 0

*byGroupName*

假日组名称

*dwHolidayPlanNo*

假日计划编号，按值表示，采用紧凑型排列，中间遇到 0 则后续无效

*byRes2*

保留，置为 0

## 8.80 NET\_DVR\_HOLIDAY\_GROUP\_COND:卡权限假日组配置条件结构体

```

struct{
    DWORD    dwSize;
    DWORD     dwHolidayGroupNumber;

```



```

    WORD    wLocalControllerID;
    BYTE    byRes[106];
} NET_DVR_HOLIDAY_GROUP_COND, *LPNET_DVR_HOLIDAY_GROUP_COND;

```

### Members

*dwSize*

结构体大小

*dwHolidayGroupNumber*

假日组编号

*wLocalControllerID*

就地控制器序号[1,64]，0 表示门禁主机

*byRes*

保留，置为 0

## 8.81 NET\_DVR\_ID\_CARD\_INFO: 身份证信息结构体

```

struct{
    DWORD    dwSize;
    BYTE    byName[MAX_ID_NAME_LEN];
    NET_DVR_DATE struBirth;
    BYTE    byAddr[MAX_ID_ADDR_LEN];
    BYTE    byIDNum[MAX_ID_NUM_LEN];
    BYTE    byIssuingAuthority[MAX_ID_ISSUING_AUTHORITY_LEN];
    NET_DVR_DATE struStartDate;
    NET_DVR_DATE struEndDate;
    BYTE    byTermOfValidity;
    BYTE    bySex;
    BYTE    byNation;
    BYTE    byRes[101];
}NET_DVR_ID_CARD_INFO, *LPNET_DVR_ID_CARD_INFO;

```

### Members

*dwSize*

结构体大小

*byName*

姓名

*struBirth*

出生日期

*byAddr*

住址

*byIDNum*

身份证号码

*byIssuingAuthority*

签发机关

*struStartDate*

有效开始日期

*struEndDate*

有效截止日期

*byTermOfValidity*

是否长期有效: 0- 否, 1- 是 (有效截止日期无效)

*bySex*

性别: 1- 男, 2- 女

*byNation*

民族: 1-"汉", 2-"蒙古", 3-"回", 4-"藏", 5-"维吾尔", 6-"苗", 7-"彝", 8-"壮", 9-"布依", 10-"朝鲜", 11-"满", 12-"侗", 13-"瑶", 14-"白", 15-"土家", 16-"哈尼", 17-"哈萨克", 18-"傣", 19-"黎", 20-"傈僳", 21-"佤", 22-"畲", 23-"高山", 24-"拉祜", 25-"水", 26-"东乡", 27-"纳西", 28-"景颇", 29-"柯尔克孜", 30-"土", 31-"达斡尔", 32-"仫佬", 33-"羌", 34-"布朗", 35-"撒拉", 36-"毛南", 37-"仡佬", 38-"锡伯", 39-"阿昌", 40-"普米", 41-"塔吉克", 42-"怒", 43-"乌孜别克", 44-"俄罗斯", 45-"鄂温克", 46-"德昂", 47-"保安", 48-"裕固", 49-"京", 50-"塔塔尔", 51-"独龙", 52-"鄂伦春", 53-"赫哲", 54-"门巴", 55-"珞巴", 56-"基诺"

*byRes*

保留, 置为 0

## 8.82 NET\_DVR\_ID\_CARD\_INFO\_ALARM: 身份证刷卡信息上传结构体

```
struct{
    DWORD                dwSize;
    NET\_DVR\_ID\_CARD\_INFO struIDCardCfg;
    DWORD                dwMajor;
    DWORD                dwMinor;
    NET\_DVR\_TIME\_V30    struSwipeTime;
    BYTE                 byNetUser[MAX\_NAMELEN];
    NET\_DVR\_IPADDR      struRemoteHostAddr;
    DWORD                dwCardReaderNo;
    DWORD                dwDoorNo;
    DWORD                dwPicDataLen;
    char                 *pPicData;
    BYTE                 byCardType;
    BYTE                 byRes[207];
}NET_DVR_ID_CARD_INFO_ALARM, *LPNET_DVR_ID_CARD_INFO_ALARM;
```

### Members

*dwSize*

结构体大小

*struIDCardCfg*

身份证信息

*dwMajor*

报警主类型, 索引值含义详见 NET\_DVR\_ACS\_ALARM\_INFO 结构中 [Remarks](#) 说明。

*dwMinor*

报警次类型, 索引值含义详见 NET\_DVR\_ACS\_ALARM\_INFO 结构中 [Remarks](#) 说明。

*struSwipeTime*

刷卡时间

*byNetUser*

网络操作的用户名

*struRemoteHostAddr*

远程主机地址

*dwCardReaderNo*

读卡器编号，为 0 表示无效

*dwDoorNo*

门编号，为 0 表示无效

*dwPicDataLen*

图片数据大小，不为 0 是表示后面带数据

*pPicData*

图片数据缓冲区

*byCardType*

卡类型，1-普通卡，2-残疾人卡，3-黑名单卡，4-巡更卡，5-胁迫卡，6-超级卡，7-来宾卡，8-解除卡，为 0 无效

*byRes*

保留，置为 0

### 8.83 NET\_DVR\_IO\_INCFG:IO 输入配置参数

```
struct{
    DWORD    dwSize;
    BYTE     byIoInStatus;
    BYTE     byRes[3];
}NET_DVR_IO_INCFG, *LPNET_DVR_IO_INCFG;
```

#### Members

*dwSize*

大小

*byIoInStatus*

输入的 IO 口状态，0-下降沿，1-上升沿，2-上升沿和下降沿，3-高电平，4-低电平

*byRes*

保留，置为 0

### 8.84 NET\_DVR\_IO\_OUTCFG:IO 输出配置参数

```
struct{
    DWORD    dwSize;
    BYTE     byDefaultStatus;
    BYTE     byIoOutStatus;
    WORD     wAheadTime;
    DWORD    dwTimePluse;
    DWORD    dwTimeDelay;
    BYTE     byFreqMulti;
```

```

    BYTE    byDutyRate;
    BYTE    byRes[2];
}NET_DVR_IO_OUTCFG, *LPNET_DVR_IO_OUTCFG;

```

### Members

*dwSize*

大小

*byDefaultStatus*

IO 默认状态: 0-低电平, 1-高电平

*byIoOutStatus*

IO 起效时状态: 0-低电平, 1-高电平, 2-脉冲

*wAheadTime*

输出 IO 提前时间, 单位 us, 取值范围[0,300]

*dwTimePluse*

脉冲间隔时间, 单位 us

*dwTimeDelay*

IO 有效持续时间, 单位 us

*byFreqMulti*

倍频, 数值范围[1,15]

*byDutyRate*

占空比, [0,40%]

*byRes*

保留, 置为 0

## 8.85 NET\_DVR\_IPADDR:IP 地址

```

struct{
    char    slpV4[16];
    BYTE    slpV6[128];
}NET_DVR_IPADDR, *LPNET_DVR_IPADDR;

```

### Members

*slpV4*

设备 IPv4 地址

*slpV6*

设备 IPv6 地址

## 8.86 NET\_DVR\_JPEGPARA:JPEG 图像信息结构体

```

struct{
    WORD    wPicSize;
    WORD    wPicQuality;
}NET_DVR_JPEGPARA,*LPNET_DVR_JPEGPARA;

```

### Members

*wPicSize*

图 片 尺 寸： 0-CIF(352\*288/352\*240)， 1-QCIF(176\*144/176\*120)， 2-4CIF(704\*576/704\*480) 或 D1(720\*576/720\*486)， 3-UXGA(1600\*1200)， 4-SVGA(800\*600)， 5-HD720P(1280\*720)， 6-VGA(640\*480)， 7-XVGA(1280\*960)， 8-HD900P(1600\*900)， 9-HD1080P(1920\*1080)， 10-2560\*1920， 11-1600\*304， 12-2048\*1536， 13-2448\*2048， 14-2448\*1200， 15-2448\*800， 16-XGA(1024\*768)， 17-SXGA(1280\*1024)， 18-WD1(960\*576/960\*480)， 19-1080I(1920\*1080)， 20-576\*576， 21-1536\*1536， 22-1920\*1920， 0xff-Auto(使用当前码流分辨率)

*wPicQuality*

图片质量系数： 0-最好， 1-较好， 2-一般

## 8.87 NET\_DVR\_LIST\_INFO:子系统信息列表结构体

```
struct{
    DWORD    dwSize;
    BYTE     byIndex;
    BYTE     byRes[63];
} NET_DVR_LIST_INFO,*LPNET_DVR_LIST_INFO;
```

### Members

*dwSize*

结构体大小

*byIndex*

子系统号， 0xff 表示所有子系统

*byRes*

保留， 置为 0

## 8.88 NET\_DVR\_LOCAL\_ABILITY\_PARSE\_CFG:能力集解析库配置

```
struct{
    BYTE     byEnableAbilityParse;
    BYTE     byRes[127];
}NET_DVR_LOCAL_ABILITY_PARSE_CFG,*LPNET_DVR_LOCAL_ABILITY_PARSE_CFG;
```

### Members

*byEnableAbilityParse*

使用能力集解析库： 0-不使用， 1-使用， 默认不使用

*byRes*

保留， 置为 0

### Remarks

模拟能力集默认禁用， 调用该接口可以启用模拟能力集， 支持获取设备各种能力。如果需要获取能力集（NET\_DVR\_GetDeviceAbility）， 可以调用此接口来启用模拟能力集， 并且需要加载 LocalXml.zip（要求和 SDK 库文件放在同一个目录下）。

## 8.89 NET\_DVR\_LOCAL\_CHECK\_DEV:设备在线巡检参数

```
struct{
    DWORD    dwCheckOnlineTimeout;
    DWORD    dwCheckOnlineNetFailMax;
    BYTE     byRes[256];
}NET_DVR_LOCAL_CHECK_DEV, *LPNET_DVR_LOCAL_CHECK_DEV;
```

### Members

#### *dwCheckOnlineTimeout*

巡检时间间隔，单位：ms，取值范围：30s~120s，0 表示用默认值(120s)，推荐设置 30s

#### *dwCheckOnlineNetFailMax*

由于网络原因失败的最大累加次数，达到该次数，SDK 才回调用户异常消息，0 表示使用默认值 1，推荐设置 3 次

#### *byRes*

保留，置为 0

### Remarks

- SDK 按照该结构体中的时间间隔对设备进行自动巡检，巡检过程中如果连失败或者重连成功在 [NET\\_DVR\\_SetExceptionCallBack\\_V30](#) 设置的异常消息回调函数中返回，对应异常消息类型为：EXCEPTION\_EXCHANGE、RESUME\_EXCHANGE。
- 推荐设置 30s 时间间隔、3 次，即心跳间隔为 1.5 分钟。

## 8.90 NET\_DVR\_LOCAL\_CONTROLLER\_STATUS\_COND:就地控制器状态查询条件结构体

```
struct{
    DWORD    dwSize;
    WORD     wLocalControllerID;
    BYTE     byRes[306];
}NET_DVR_LOCAL_CONTROLLER_STATUS_COND,*LPNET_DVR_LOCAL_CONTROLLER_STATUS_COND;
```

### Members

#### *dwSize*

结构体大小

#### *wLocalControllerID*

就地控制器序号,为 0 表示所有

#### *byRes*

保留，置为 0

## 8.91 NET\_DVR\_LOCAL\_CONTROLLER\_STATUS:就地控制器状态参数结构体

```
struct{
    DWORD    dwSize;
    WORD     wLocalControllerID;
    BYTE     byLocalAntiDismantleStatus;
    BYTE     byPowerSupplyStatus;
    WORD     wBatteryVoltage;
    BYTE     byBatteryLowVoltage;
    BYTE     byFireAlarm;
    BYTE     bySerialNumber[SERIALNO\_LEN];
    BYTE     byMagneticStatus[MAX\_DOOR\_NUM];
    BYTE     byDoorLockStatus[MAX\_DOOR\_NUM];
    BYTE     byCardReaderOnlineStatus[MAX\_CARD\_READER\_NUM];
    WORD     wLocalControllerStatus;
    BYTE     byRes2[122];
} NET_DVR_LOCAL_CONTROLLER_STATUS,*LPNET_DVR_LOCAL_CONTROLLER_STATUS;
```

### Members

*dwSize*

结构体大小

*wLocalControllerID*

就地控制器序号,为 0 表示所有

*byLocalAntiDismantleStatus*

就地控制器防拆状态, 0-关闭, 1-开启

*byPowerSupplyStatus*

设备供电状态, 1-交流电供电, 2-蓄电池供电

*wBatteryVoltage*

蓄电池电压值, 实际值乘 10, 单位: 伏特

*byBatteryLowVoltage*

蓄电池是否处于低压状态, 0-否, 1-是

*byFireAlarm*

消防报警, 0-正常, 1-短接报警, 2-断开报警

*bySerialNumber*

设备序列号

*byMagneticStatus*

门磁状态: 0-正常关闭, 1-正常开启, 2-破坏短路报警, 3-破坏断路报警, 4-异常报警

*byDoorLockStatus*

门锁状态: 0-正常关闭, 1-正常开启, 2-破坏短路报警, 3-破坏断路报警, 4-异常报警

*byCardReaderOnlineStatus*

读卡器在线状态: 0-不在线, 1-在线

*wLocalControllerStatus*

只读，就地控制器在线状态：0-离线，1-网络在线，2-环路 1 上的 RS485 串口 1，3-环路 1 上的 RS485 串口 2，4-环路 2 上的 RS485 串口 1，5-环路 2 上的 RS485 串口 2，6-环路 3 上的 RS485 串口 1，7-环路 3 上的 RS485 串口 2，8-环路 4 上的 RS485 串口 1，9-环路 4 上的 RS485 串口 2

*byRes*

保留，置为 0

## 8.92 NET\_DVR\_LOCAL\_MEM\_POOL\_CFG:内存池本地配置

```
struct{
    DWORD    dwAlarmMaxBlockNum;
    DWORD    dwAlarmReleaseInterval;
    BYTE     byRes[60];
}NET_DVR_LOCAL_MEM_POOL_CFG,*LPNET_DVR_LOCAL_MEM_POOL_CFG;
```

### Members

*dwAlarmMaxBlockNum*

报警模块内存池最多向系统申请的内存块（block）个数，每个 block 为 64MB，超过这个上限则不向系统申请，0 表示无上限

*dwAlarmReleaseInterval*

报警模块空闲内存释放的间隔，单位：秒，为 0 表示不释放空闲的内存

*byRes*

保留，置为 0

## 8.93 NET\_DVR\_LOCAL\_PROTECT\_KEY\_CFG:密钥配置

```
struct{
    BYTE     byProtectKey[128];
    BYTE     byRes[128];
}NET_DVR_LOCAL_PROTECT_KEY_CFG,*LPNET_DVR_LOCAL_PROTECT_KEY_CFG;
```

### Members

*byProtectKey*

密钥，默认设置为 0

*byRes*

保留，置为 0

## 8.94 NET\_DVR\_LOCAL\_TALK\_MODE\_CFG:对讲模式配置

```
struct{
    BYTE     byEnableAbilityParse;
    BYTE     byRes[127];
}NET_DVR_LOCAL_TALK_MODE_CFG,*LPNET_DVR_LOCAL_TALK_MODE_CFG;
```

### Members



*byTalkMode*

对讲模式：0- 使用对讲库（默认），1- 使用 windows api 模式

*byRes*

保留，置为 0

**Remarks**

V4.2.2.5 及以前版本 SDK 均采用 windows API 实现相关功能。之后版本默认使用语音对讲库的方式，语音对讲库模式下必须加载 AudioIntercom.dll 和 OpenAL32.dll。

**8.95 NET\_DVR\_LOCAL\_TCP\_PORT\_BIND\_CFG:本地 TCP 端口绑定配置**

```
struct{
    WORD    wLocalBindTcpMinPort;
    WORD    wLocalBindTcpMaxPort;
    BYTE    byRes[60];
}NET_DVR_LOCAL_TCP_PORT_BIND_CFG,*LPNET_DVR_LOCAL_TCP_PORT_BIND_CFG;
```

**Members***wLocalBindTcpMinPort*

本地绑定 TCP 最小端口

*wLocalBindTcpMaxPort*

本地绑定 TCP 最大端口

*byRes*

保留，置为 0

**Remarks**

端口绑定的策略是：给一个端口段，可以保证使用的端口都是在这个段里（多播除外），但不能保证每一个段内的端口都用到，因为是循环利用的；端口池中取出的端口会去尝试绑定，如果被占用了，将取下一个，如果段内每一个都绑定不了，则连接操作返回失败。建议最好不要设置系统预留的端口（1-1024），比如 80 等。

设置的最大端口应该大于等于最小端口，[0, 0]表示清除绑定，[0, 非 0]将设置失败，因为 0 不能进行绑定。

**8.96 NET\_DVR\_LOCAL\_UDP\_PORT\_BIND\_CFG:本地 UDP 端口绑定配置**

```
struct{
    WORD    wLocalBindUdpMinPort;
    WORD    wLocalBindUdpMaxPort;
    BYTE    byRes[60];
}NET_DVR_LOCAL_UDP_PORT_BIND_CFG,*LPNET_DVR_LOCAL_UDP_PORT_BIND_CFG;
```

**Members***wLocalBindUdpMinPort*

本地绑定 UDP 最小端口

*wLocalBindUdpMaxPort*

本地绑定 UDP 最大端口

*byRes*

保留，置为 0

### Remarks

端口绑定的策略是：给一个端口段，可以保证使用的端口都是在这个段里（多播除外），但不能保证每一个段内的端口都用到，因为是循环利用的；端口池中取出的端口会去尝试绑定，如果被占用了，将取下一个，如果段内每一个都绑定不了，则连接操作返回失败。建议最好不要设置系统预留的端口（1-1024），比如 80 等。

设置的最大端口应该大于等于最小端口，[0, 0]表示清除绑定，[0, 非 0]将设置失败，因为 0 不能进行绑定。

## 8.97 NET\_DVR\_LOG\_V30:日志信息

```
struct{
    NET\_DVR\_TIME      strLogTime;
    DWORD             dwMajorType;
    DWORD             dwMinorType;
    BYTE              sPanelUser[MAX\_NAMELEN];
    BYTE              sNetUser[MAX\_NAMELEN];
    NET\_DVR\_IPADDR    struRemoteHostAddr;
    DWORD             dwParaType;
    DWORD             dwChannel;
    DWORD             dwDiskNumber;
    DWORD             dwAlarmInPort;
    DWORD             dwAlarmOutPort;
    DWORD             dwInfoLen;
    char              sInfo[LOG\_INFO\_LEN];
}NET_DVR_LOG_V30,*LPNET_DVR_LOG_V30;
```

### Members

*strLogTime*

日志时间

*dwMajorType*

报警主类型，定义请参见[设备日志主类型](#)

*dwMinorType*

报警次类型，根据不同的主类型的次类型定义请参见[设备日志次类型](#)

*sPanelUser*

操作面板的用户名

*sNetUser*

网络操作的用户名

*struRemoteHostAddr*

远程主机地址

*dwParaType*

对于 DS-90xx 设备，当日志次类型为 MINOR\_START\_VT 或者 MINOR\_STOP\_VT 时，表示语音对讲端口号。

当日志的主类型为 MAJOR\_OPERATION=03（操作），且次类型为 MINOR\_LOCAL\_CFG\_PARM=0x52（本地配置参数）或 MINOR\_REMOTE\_GET\_PARM=0x76（远程获得参数）或

MINOR\_REMOTE\_CFG\_PARM=0x77（远程配置参数）时，该参数类型有效，其含义如下：

宏定义	宏定义值	含义
PARA_VIDEOOUT	0x1	视频输出结构配置
PARA_IMAGE	0x2	图像参数结构配置
PARA_ENCODE	0x4	压缩参数结构配置
PARA_NETWORK	0x8	网络参数结构配置
PARA_ALARM	0x10	报警参数结构配置
PARA_EXCEPTION	0x20	异常参数结构配置
PARA_DECODER	0x40	解码器参数结构配置
PARA_RS232	0x80	RS232 参数结构配置
PARA_PREVIEW	0x100	本地预览参数结构配置
PARA_SECURITY	0x200	用户权限参数结构配置
PARA_DATETIME	0x400	本地系统配置
PARA_FRAME TYPE	0x800	帧信息参数结构配置

*dwChannel*

通道号

*dwDiskNumber*

硬盘号

*dwAlarmInPort*

报警输入端口

*dwAlarmOutPort*

报警输出端口

*dwInfoLen*

日志附加信息长度

*sInfo*

日志附加信息

## 8.98 NET\_DVR\_MULTI\_CARD\_CFG:多重卡参数配置结构体

```
struct{
    DWORD                dwSize;
    BYTE                 byEnable;
    BYTE                 bySwipeIntervalTimeout;
    BYTE                 byRes1[2];
    NET_DVR_MULTI_CARD_CFG struGroupCfg[MULTI_CARD_GROUP_NUM];
    BYTE                 byRes2[32];
}NET_DVR_MULTI_CARD_CFG,*LPNET_DVR_MULTI_CARD_CFG;
```

### Members

*dwSize*

结构体大小

*byEnable*

是否启用多重卡功能：0- 不启用，1- 启用

*bySwipeIntervalTimeout*

刷卡间隔超时时间，取值范围：1~255，单位：s（秒），默认：10

*byRes1*

保留，置为 0

*struGroupCfg*

群组刷卡参数

*byRes2*

保留，置为 0

## 8.99 NET\_DVR\_MULTI\_CARD\_GROUP\_CFG:群组刷卡参数结构体

```
struct{
    BYTE                byEnable;
    BYTE                byEnableOfflineVerifyMode;
    BYTE                byRes1[2];
    DWORD               dwTemplateNo;
    NET\_DVR\_GROUP\_COMBINATION\_INFO struGroupCombination[GROUP\_COMBINATION\_NUM];
}NET_DVR_MULTI_CARD_GROUP_CFG,*LPNET_DVR_MULTI_CARD_GROUP_CFG;
```

### Members

*byEnable*

是否启用该多重卡组参数：0- 不启用，1- 启用

*byEnableOfflineVerifyMode*

是否启用主机离线时验证方式（超级密码代替远程开门）：0- 不启用，1- 启用

*byRes1*

保留，置为 0

*dwTemplateNo*

启用多重卡功能的计划模板编号

*struGroupCombination*

群组组合参数

## 8.100 NET\_DVR\_MULTI\_DOOR\_INTERLOCK\_CFG:多门互锁参数配置结构体

```
struct{
    DWORD    dwSize;
    BYTE     byEnable;
    BYTE     byRes1[3];
    DWORD    dwMultiDoorGroup[MAX\_MULTI\_DOOR\_INTERLOCK\_GROUP][MAX\_INTER\_LOCK\_DOOR\_NUM];
    BYTE     byRes2[64];
}NET_DVR_MULTI_DOOR_INTERLOCK_CFG,*LPNET_DVR_MULTI_DOOR_INTERLOCK_CFG;
```

### Members

*dwSize*

结构体大小

*byEnable*

是否启用多门互锁功能：0- 不启用，1- 启用

*byRes1*

保留，置为 0

*dwMultiDoorGroup*多门互锁组参数，取值为门编号，即 `dwMultiDoorGroup[i][j]=k` 表示第(i+1)多门互锁组中第(j+1)个互锁门关联的门编号是 k*byRes2*

保留，置为 0

**Remarks**

互锁和反潜回功能不能同时生效。

## 8.101 **NET\_DVR\_NFSCFG:**网络文件系统参数

```

struct{
    DWORD                dwSize;
    NET\_DVR\_SINGLE\_NFS    struNfsDiskParam[MAX\_NFS\_DISK];
}NET_DVR_NFSCFG, *LPNET_DVR_NFSCFG;

```

**Members***dwSize*

结构体大小

*struNfsDiskParam*

NFS 配置子结构

**Remarks**

修改 NFS 各参数时需要重启设备才能生效。

## 8.102 **NET\_DVR\_NETCFG\_MULTI:**多网卡网络配置

```

struct{
    DWORD                dwSize;
    BYTE                byDefaultRoute;
    BYTE                byNetworkCardNum;
    BYTE                byWorkMode;
    BYTE                byRes;
    NET\_DVR\_ETHERNET\_MULTI struEtherNet[MAX\_NETWORK\_CARD];
    NET\_DVR\_IPADDR        struManageHost1IpAddr;
    NET\_DVR\_IPADDR        struManageHost2IpAddr;
    NET\_DVR\_IPADDR        struAlarmHostIpAddr;
    WORD                wManageHost1Port;
    WORD                wManageHost2Port;
    WORD                wAlarmHostIpPort;
}NET_DVR_NETCFG_MULTI, *LPNET_DVR_NETCFG_MULTI;

```

```

BYTE                byIpResolver[MAX\_DOMAIN\_NAME];
WORD                wIpResolverPort;
WORD                wDvrPort;
WORD                wHttpPortNo;
WORD                wDvrPort2;
BYTE                byRes2[4];
NET\_DVR\_IPADDR     struMulticastIpAddr;
NET\_DVR\_PPPOECFG  struPPPoE;
BYTE                byRes3[24];
}NET_DVR_NETCFG_MULTI,*LPNET_DVR_NETCFG_MULTI;

```

### Members

*dwSize*

结构体大小

*byDefaultRoute*

默认路由，0 表示 struEtherNet[0]，1 表示 struEtherNet[1]

*byNetworkCardNum*

设备实际可配置的网卡数目

*byWorkMode*

0-普通多网卡模式，1-内外网隔离模式

*byRes*

保留，置为 0

*struEtherNet*

以太网口

*struManageHost1IpAddr*

主管理主机 IP 地址

*struManageHost2IpAddr*

辅管理主机 IP 地址

*struAlarmHostIpAddr*

报警主机 IP 地址

*wManageHost1Port*

主管理主机端口号

*wManageHost2Port*

辅管理主机端口号

*wAlarmHostIpPort*

报警主机端口号

*byIpResolver*

IP 解析服务器域名或 IP 地址

*wIpResolverPort*

IP 解析服务器端口号

*wDvrPort*

通讯端口，默认 8000

*wHttpPortNo*

HTTP 端口号

*wDvrPort2*

通讯端口 2

*byRes2*

保留，置为 0

*struMulticastIpAddr*

多播组地址

*struPPPoE*

PPPoE 参数

*byRes3*

保留，置为 0

## 8.103 NET\_DVR\_NETCFG\_V30:网络配置

```
struct{
    DWORD                                dwSize;
    NET\_DVR\_ETHERNET\_V30                struEtherNet[MAX\_ETHERNET];
    NET\_DVR\_IPADDR                      struRes1[2];
    NET\_DVR\_IPADDR                      struAlarmHostIpAddr;
    WORD                                wRes2[2];
    WORD                                wAlarmHostIpPort;
    BYTE                                byUseDhcp;
    BYTE                                byRes3;
    NET\_DVR\_IPADDR                      struDnsServer1IpAddr;
    NET\_DVR\_IPADDR                      struDnsServer2IpAddr;
    BYTE                                byIpResolver[MAX\_DOMAIN\_NAME];
    WORD                                wIpResolverPort;
    WORD                                wHttpPortNo;
    NET\_DVR\_IPADDR                      struMulticastIpAddr;
    NET\_DVR\_IPADDR                      struGatewayIpAddr;
    NET\_DVR\_PPPOECFG                   struPPPoE;
    BYTE                                byRes[64];
}NET_DVR_NETCFG_V30,*LPNET_DVR_NETCFG_V30;
```

### Members

*dwSize*

大小

*struEtherNet*

以太网口

*struRes1*

保留，置为 0

*struAlarmHostIpAddr*

报警主机 IP 地址

*wRes2*

保留，置为 0

*wAlarmHostIpPort*

报警主机端口号

*byUseDhcp*

是否启用 DHCP: 0xff-无效; 0-不启用; 1-启用

*byRes3*

保留, 置为 0

*struDnsServer1IpAddr*

域名服务器 1 的 IP 地址

*struDnsServer2IpAddr*

域名服务器 2 的 IP 地址

*byIpResolver*

IP 解析服务器域名或 IP 地址 (8000 设备不支持域名)

*wIpResolverPort*

IP 解析服务器端口号

*wHttpPortNo*

HTTP 端口号

*struMulticastIpAddr*

多播组地址

*struGatewayIpAddr*

网关地址

*struPPPoE*

PPPoE 参数

*byRes*

保留, 置为 0

#### Remarks

3.0 协议以下的设备, 参数 *byUseDhcp* 为 0xff-无效, 将其 IP 地址填成空, 设备会自动去获取 DHCP。

## 8.104 NET\_DVR\_NETCFG\_V50:网络配置

```
struct{
    DWORD                                dwSize;
    NET\_DVR\_ETHERNET\_V30                 struEtherNet[MAX\_ETHERNET];
    NET\_DVR\_IPADDR                       struRes1[2];
    NET\_DVR\_IPADDR                       struAlarmHostIpAddr;
    BYTE                                wRes2[4];
    WORD                                wAlarmHostIpPort;
    BYTE                                byUseDhcp;
    BYTE                                byIPv6Mode;
    NET\_DVR\_IPADDR                       struDnsServer1IpAddr;
    NET\_DVR\_IPADDR                       struDnsServer2IpAddr;
    BYTE                                byIpResolver[MAX\_DOMAIN\_NAME];
    WORD                                wIpResolverPort;
    WORD                                wHttpPortNo;
    NET\_DVR\_IPADDR                       struMulticastIpAddr;
    NET\_DVR\_IPADDR                       struGatewayIpAddr;
    NET\_DVR\_PPPOECFG                     struPPPoE;
```



```

    BYTE                byEnablePrivateMulticastDiscovery;
    BYTE                byEnableOnvifMulticastDiscovery;
    BYTE                byEnableDNS;
    WORD                wAlarmHost2IpPort;
    NET\_DVR\_IPADDR      struAlarmHost2IpAddr;
    BYTE                byRes[600];
} NET_DVR_NETCFG_V50,*LPNET_DVR_NETCFG_V50;

```

## Members

*dwSize*

大小

*struEtherNet*

以太网口

*struRes1*

保留，置为 0

*struAlarmHostIpAddr*

报警主机 IP 地址

*wRes2*

保留，置为 0

*wAlarmHostIpPort*

报警主机端口号

*byUseDhcp*

是否启用 DHCP: 0xff-无效; 0-不启用; 1-启用

*byIPv6Mode*

IPv6 分配方式: 0-路由公告, 1-手动设置, 2-启用 DHCP 分配

*struDnsServer1IpAddr*

域名服务器 1 的 IP 地址

*struDnsServer2IpAddr*

域名服务器 2 的 IP 地址

*byIpResolver*

IP 解析服务器域名或 IP 地址 (8000 设备不支持域名)

*wIpResolverPort*

IP 解析服务器端口号

*wHttpPortNo*

HTTP 端口号

*struMulticastIpAddr*

多播组地址

*struGatewayIpAddr*

网关地址

*struPPPoE*

PPPoE 参数

*byEnablePrivateMulticastDiscovery*

私有多播搜索(SADP): 0- 默认, 1- 启用, 2- 禁用

*byEnableOnvifMulticastDiscovery*

Onvif 多播搜索(SADP): 0- 默认, 1- 启用, 2- 禁用

*byEnableDNS*

手动设置 DNS 服务器地址使能：0- 自动获取，1- 手动设置

*wAlarmHost2IpPort*

报警主机 2 端口号

*struAlarmHost2IpAddr*

报警主机 2 IP 地址

*byRes*

保留，置为 0

#### Remarks

3.0 协议以下的设备，参数 *byUseDhcp* 为 0xff-无效，将其 IP 地址填成空，设备会自动去获取 DHCP。

## 8.105 NET\_DVR\_NETWORK\_BONDING: BONDING 网卡配置

```
struct{
    DWORD                dwSize;
    BYTE                 byEnable;
    BYTE                 byNum;
    BYTE                 byRes1[2];
    NET_DVR_ONE_BONDING struOneBond[MAX_BOND_NUM];
    BYTE                 byRes2[40];
}NET_DVR_NETWORK_BONDING,*LPNET_DVR_NETWORK_BONDING;
```

#### Members

*dwSize*

结构体大小

*byEnable*

是否启用 bonding 功能

*byNum*

Bonding 网卡的个数

*byRes1*

保留

*struOneBond*

单 BONDING 网卡配置参数

*byRes2*

保留

## 8.106 NET\_DVR\_NTTPARA: 网络应用参数(NTP)

```
struct{
    BYTE                sNTPServer[64];
    WORD                wInterval;
    BYTE                byEnableNTP;
    signed char         cTimeDifferenceH;
    signed char         cTimeDifferenceM;
```

```

BYTE          res1;
WORD          wNtpPort;
BYTE          res2[8];
}NET_DVR_NTTPARA,*LPNET_DVR_NTTPARA;

```

### Members

#### *sNTPServer*

NTP 服务器域名或者 IP 地址

#### *wInterval*

校时间隔时间，以分钟或小时为单位（通过能力集 [NET\\_DVR\\_GetDeviceAbility](#) 获取，对应网络应用参数能力集：DEVICE\_NETAPP\_ABILITY）

#### *byEnableNTP*

NTP 校时是否启用：0—否，1—是

#### *cTimeDifferenceH*

与国际标准时间的时差（小时），-12 ... +13

#### *cTimeDifferenceM*

与国际标准时间的时差（分钟），0, 30, 45

#### *res1*

保留，置为 0

#### *wNtpPort*

NTP 服务器端口，设备默认为 123

#### *res2*

保留，置为 0

## 8.107 NET\_DVR\_ONE\_BONDING:单 BONDING 网卡配置

```

struct{
    BYTE          byMode;
    BYTE          byUseDhcp;
    BYTE          byMasterCard;
    BYTE          byStatus;
    BYTE          byBond[MAX\_NETWORK\_CARD];
    NET\_DVR\_ETHERNET\_V30 struEtherNet;
    NET\_DVR\_IPADDR      struGatewayIpAddr;
    BYTE          byRes[20];
}NET_DVR_ONE_BONDING,*LPNET_DVR_ONE_BONDING;

```

### Members

#### *byMode*

工作模式：0 - 网络容错，1 - 负载均衡

#### *byUseDhcp*

是否使能 dhcp

#### *byMasterCard*

指定哪张网卡为主网卡

#### *byStatus*

BONDING 的状态：0 - 异常，1-正常，只能获取不能设置

*byBond*

byBond[0]== 1 表示使用 eh0，0 表示不使用 eh0

*struEtherNet*

网卡参数

*struGatewayIpAddr*

网关地址

*byRes*

保留

## 8.108 **NET\_DVR\_ONLINE\_LOCAL\_CONTROLLER\_CFG**:在线就地控制器信息结构体

```
struct{
    DWORD                dwSize;
    BYTE                 byLocalControllerName[LOCAL_CONTROLLER_NAME_LEN];
    WORD                 wLocalControllerID;
    WORD                 wDevPort;
    NET_DVR_IPADDR       struDevIP;
    NET_DVR_IPADDR       struSubnetMask;
    NET_DVR_IPADDR       struGateway;
    BYTE                 bySearchProgress;
    BYTE                 byEffectData;
    BYTE                 byRes[302];
} NET_DVR_ONLINE_LOCAL_CONTROLLER_CFG,*LPNET_DVR_ONLINE_LOCAL_CONTROLLER_CFG;
```

### Members

*dwSize*

结构体大小

*byLocalControllerName*

就地控制器名称

*wLocalControllerID*

就地控制器序号,为 0 表示所有

*wDevPort*

设备端口号

*struDevIP*

设备 IP 地址

*struSubnetMask*

设备子网掩码

*struGateway*

设备网关

*bySearchProgress*

搜索进度，0 表示未开始，100 表示同步完成

*byEffectData*

是否为有效数据,0-表示有效，1-表示设备只返回搜索进度，作为心跳包

*byRes*

保留，置为 0

## 8.109 NET\_DVR\_PASSNUM\_INFO\_ALARM:通行人数信息上传结构体

```
struct{
    DWORD                dwSize;
    DWORD                dwAccessChannel;
    NET_DVR_TIME_V30    struSwipeTime;
    BYTE                 byNetUser[MAX_NAMELEN];
    NET_DVR_IPADDR       struRemoteHostAddr;
    DWORD                dwEntryTimes;
    DWORD                dwExitTimes;
    DWORD                dwTotalTimes;
    BYTE                 byRes[300];
}NET_DVR_PASSNUM_INFO_ALARM, *LPNET_DVR_PASSNUM_INFO_ALARM;
```

### Members

*dwSize*

结构体大小

*dwAccessChannel*

人员通道号

*struSwipeTime*

刷卡时间

*byNetUser*

网络操作的用户名

*struRemoteHostAddr*

远程主机地址

*dwEntryTimes*

人员进入次数

*dwExitTimes*

人员离开次数

*dwTotalTimes*

人员出入总次数

*byRes*

保留，置为 0

## 8.110 NET\_DVR\_PERSONNEL\_CHANNEL\_CFG:门禁主机人员通道参数配置结构体

```
struct{
    DWORD                dwSize;
    BYTE                 byInMode;
```

```

    BYTE        byOutMode;
    BYTE        byWorkMode;
    BYTE        byRes[301];
}NET_DVR_PERSONNEL_CHANNEL_CFG,*LPNET_DVR_PERSONNEL_CHANNEL_CFG;

```

### Members

*dwSize*

结构体大小

*byInMode*

进门模式：0- 受控，1- 禁止，2- 自由

*byOutMode*

出门模式：0- 受控，1- 禁止，2- 自由

*byWorkMode*

工作模式：0- 紧急，1- 维护，2- 常闭，3- 常开

*byRes*

保留，置为 0

### Remarks

- 设备是否支持人员通道参数配置或者支持的参数能力，可以通过设备能力集进行判断，对应门禁主机能力集 (**AcsAbility**)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：ACS\_ABILITY，节点：< PersonnelChannelCfg >。

## 8.111 NET\_DVR\_PERSON\_STATISTICS\_CFG:人数统计参数结构体

```

struct{
    DWORD        dwSize;
    BYTE        byEnableStatistics;
    BYTE        byEnableOfflineStatistics;
    BYTE        byRes[606];
}NET_DVR_PERSON_STATISTICS_CFG,*LPNET_DVR_PERSON_STATISTICS_CFG;

```

### Members

*dwSize*

结构体大小

*byEnableStatistics*

是否开启人数统计：0- 不开启，1- 开启

*byEnableOfflineStatistics*

是否开启离线人数统计：0- 不开启，1- 开启

*byRes*

保留，置为 0

### Remarks

- 设备是否支持人数统计功能或者支持的参数能力，可以通过设备能力集进行判断，对应门禁主机能力集 (**AcsAbility**)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：ACS\_ABILITY，节点：< PersonStatisticsCfg >。

## 8.112 NET\_DVR\_PHONECFG:电话配置结构体

```
struct{
    BYTE    byWhiteList[MAX_PHONE_NUM];
    BYTE    byPhonePerssion[32];
    BYTE    byAlarmHandler[32];
    BYTE    byRes[128];
}NET_DVR_PHONECFG,*LPNET_DVR_PHONECFG;
```

### Members

#### *byWhiteList*

白名单号码

#### *byPhonePerssion*

是否使能该号码的某功能，按字节表示：

0x0: 支持接收报警短信

0x1: 支持短信控制上下线

0x2: 支持呼叫控制上线

0x3: 支持短信重启

0x4: 支持门操作控制

#### *byAlarmHandler*

是否使能对某个报警类型的短信发送，取值：0- 否，1-是，按字节表示：

0x0: 硬盘满

0x1: 硬盘错

0x2: 网线断

0x3: IP 地址冲突

0x4: 非法访问

0x5: 视频信号异常

0x6: 输入/输出视频制式不匹配

0x7: 视频无信号

0x8: 音频无信号

0x9: 报警输入

0xa: 遮挡报警

0xb: 移动侦测

0xc: 录像异常

0xd: PIR 报警

0xe: 无线报警

0xf: 呼救报警

0x10: 音频异常侦测报警

0x11: 场景侦测报警

0x12: 虚焦侦测报警

0x13: 人脸侦测报警

0x14: 越界侦测报警

0x15: 区域入侵侦测报警

0x16: 离开区域侦测报警

0x17: 进入区域报警  
 0x18: 人员徘徊侦测报警  
 0x19: 人员聚集侦测报警  
 0x1a: 快速移动侦测报警  
 0x1b: 停车侦测报警  
 0x1c: 物品遗留侦测报警  
 0x1d: 物品拿取侦测报警

*byRes*

保留，置为 0

## 8.113 NET\_DVR\_PHONE\_DOOR\_RIGHT\_CFG:手机关联门权限参数配置

### 结构体

```
struct{
    DWORD        dwSize;
    BYTE          byOpenRight[MAX_DOOR_NUM_256];
    BYTE          byCloseRight[MAX_DOOR_NUM_256];
    BYTE          byNormalOpenRight[MAX_DOOR_NUM_256];
    BYTE          byNormalCloseRight[MAX_DOOR_NUM_256];
    BYTE          byArmRight[MAX_ALARMHOST_ALARMIN_NUM];
    BYTE          byDisarmRight[MAX_ALARMHOST_ALARMIN_NUM];
    BYTE          byRes[256];
}NET_DVR_PHONE_DOOR_RIGHT_CFG, *LPNET_DVR_PHONE_DOOR_RIGHT_CFG;
```

#### Members

*dwSize*

结构体大小

*byOpenRight*

是否有开门权限，按数组表示，数组下标表示门序号，数组值：0- 无权限，1- 有权限

*byCloseRight*

是否有关门权限，按数组表示，数组下标表示门序号，数组值：0- 无权限，1- 有权限

*byNormalOpenRight*

是否有常开权限，按数组表示，数组下标表示门序号，数组值：0- 无权限，1- 有权限

*byNormalCloseRight*

是否有常闭权限，按数组表示，数组下标表示门序号，数组值：0- 无权限，1- 有权限

*byArmRight*

是否有布防权限，按数组表示，数组下标表示报警输入序号，数组值：0- 无权限，1- 有权限

*byDisarmRight*

是否有撤防权限，按数组表示，数组下标表示报警输入序号，数组值：0- 无权限，1- 有权限

*byRes*

保留，置为 0

#### Remarks

设备是否支持手机关联门权限配置或者支持的参数能力，可以通过设备能力集进行判断，对应门禁主机能力集(**AcsAbility**)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：ACS\_ABILITY，节点：<SMS>。



## 8.114 NET\_DVR\_PICTURECFG:图片参数结构体

```
struct{
    DWORD                dwSize;
    BYTE                 byUseType;
    BYTE                 bySequence;
    BYTE                 byWallNo;
    BYTE                 byRes1;
    NET\_DVR\_BASEMAP\_CFG struBasemapCfg;
    BYTE                 sPicName[NAME\_LEN];
    BYTE                 byRes2[32];
}NET_DVR_PICTURECFG,*LPNET_DVR_PICTURECFG;
```

### Members

*dwSize*

结构体大小

*byUseType*

1- 底图, 2- GIF 图片, 3- CAD 图片, 4- 输出口图片

*bySequence*

序号。分布式不使用该参数, 设为 0

*byWallNo*

电视墙号

*byRes1*

保留, 置为 0

*struBasemapCfg*

底图参数

*sPicName*

图片名称

*byRes2*

保留, 置为 0

### Remarks

- 底图由上层切割并传送, 例如: 2\*2 的大屏, 上传底图时需要客户端将底图切割成 4 块, 并分成 4 次发送底图数据, 每次发送时都需要发送结构体 [NET\\_DVR\\_BASEMAP\\_CFG](#)。
- 设备在收到屏幕序号为 0 的底图时, 表示是新的底图。例如, 用户将一幅图切割成 4 份, 当传完前两块以后又要传另一幅底图, 此时应该将屏幕序号置为 0。这时设备就能知道是另一幅底图, 并将第一幅底图的前两块数据丢弃。
- 对于门禁主机, 不支持设置底图参数, 参数 [struBasemapCfg](#) 无效。门禁设备是否支持底图功能或者支持的参数能力, 可以通过设备能力集进行判断, 对应门禁主机能力集 ([AcsAbility](#)), 相关接口: [NET\\_DVR\\_GetDeviceAbility](#), 能力集类型: ACS\_ABILITY, 节点: <PictureCfg>。

## 8.115 NET\_DVR\_PLAN\_TEMPLATE:计划模板配置结构体

```
struct{
    DWORD                dwSize;
```

```

BYTE          byEnable;
BYTE          byRes1[3];
BYTE          byTemplateName[TEMPLATE_NAME_LEN];
DWORD         dwWeekPlanNo;
DWORD         dwHolidayGroupNo[MAX_HOLIDAY_GROUP_NUM];
BYTE          byRes2[32];
}NET_DVR_PLAN_TEMPLATE,*LPNET_DVR_PLAN_TEMPLATE;

```

### Members

*dwSize*

结构体大小

*byEnable*

是否使能：0- 否，1- 是

*byRes1*

保留，置为 0

*byTemplateName*

计划模板名称

*dwWeekPlanNo*

周计划编号，0 表示无效

*dwHolidayGroupNo*

假日组编号，按值表示，采用紧凑型排列，中间遇到 0 则后续无效

*byRes2*

保留，置为 0

## 8.116 NET\_DVR\_PLAN\_TEMPLATE\_COND:卡权限计划模板配置条件结构体

```

struct{
    DWORD    dwSize;
    DWORD    dwPlanTemplateName;
    WORD     wLocalControllerID;
    BYTE     byRes[106];
} NET_DVR_PLAN_TEMPLATE_COND,*LPNET_DVR_PLAN_TEMPLATE_COND;

```

### Members

*dwSize*

结构体大小

*dwPlanTemplateName*

计划模板编号，从 1 开始，最大值从门禁能力集获取

*wLocalControllerID*

就地控制器序号[1,64]，0 表示门禁主机

*byRes*

保留，置为 0

## 8.117 NET\_DVR\_PLATFORM\_VERIFY\_CFG:平台认证结果信息结构体

```
struct{
    DWORD        dwSize;
    DWORD        dwDoorNo;
    BYTE         byResultType;
    BYTE         byRes1[3];
    BYTE         byScreenDisplay[MAX_SCREEN_DISPLAY_LEN];
    BYTE         byRes[300];
}NET_DVR_PLATFORM_VERIFY_CFG, *LPNET_DVR_PLATFORM_VERIFY_CFG;
```

### Members

*dwSize*

结构体大小

*dwDoorNo*

门编号

*byResultType*

认证结果类型：0- 非法，1- 合法

*byRes1*

保留，置为 0

*byScreenDisplay*

LED 屏幕显示，字符串，用于显示认证相关信息

*byRes*

保留，置为 0

### Remarks

- 设备是否支持平台认证或者支持的参数能力，可以通过设备能力集进行判断，对应门禁能力集 (AcsAbility)，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：ACS\_ABILITY，节点：<PlatformVerifyCfg>。

## 8.118 NET\_DVR\_PPPCFG\_V30:PPP 参数结构体

```
struct{
    NET\_DVR\_IPADDR    struRemoteIP;
    NET\_DVR\_IPADDR    struLocalIP;
    char              sLocalIPMask[16];
    BYTE              sUsername[NAME_LEN];
    BYTE              sPassword[PASSWD_LEN];
    BYTE              byPPPMODE;
    BYTE              byRedial;
    BYTE              byRedialMode;
    BYTE              byDataEncrypt;
    DWORD             dwMTU;
    char              sTelephoneNumber[PHONENUMBER_LEN];
}NET_DVR_PPPCFG_V30, *LPNET_DVR_PPPCFG_V30;
```

**Members***struRemoteIP*

远端 IP 地址

*struLocalIP*

本地 IP 地址

*sLocalIPMask*

本地 IP 地址掩码

*sUsername*

用户名

*sPassword*

密码

*byPPPMODE*

PPP 模式: 0—主动, 1—被动

*byRedial*

是否回拨: 0-否, 1-是

*byRedialMode*

回拨模式: 0-由拨入者指定, 1-预置回拨号码

*byDataEncrypt*

数据加密: 0-否, 1-是

*dwMTU*

MTU

*sTelephoneNumber*

电话号码

**8.119 NET\_DVR\_PPPOECFG:PPPoE 配置**

```

struct{
    DWORD          dwPPPOE;
    BYTE           sPPPoEUser[NAME_LEN];
    char           sPPPoEPassword[PASSWD_LEN];
    NET_DVR_IPADDR struPPPoEIP;
}NET_DVR_PPPOECFG, *LPNET_DVR_PPPOECFG;

```

**Members***dwPPPOE*

是否启用 PPPoE: 0-不启用, 1-启用

*sPPPoEUser*

PPPoE 用户名

*sPPPoEPassword*

PPPoE 密码

*struPPPoEIP*

PPPoE IP 地址

## 8.120 NET\_DVR\_SCHEDULETIME:起止时间段参数

```
struct{
    BYTE    byStartHour;
    BYTE    byStartMin;
    BYTE    byStopHour;
    BYTE    byStopMin;
}NET_DVR_SCHEDULETIME, *LPNET_DVR_SCHEDULETIME;
```

### Members

*byStartHour*

开始时间：时

*byStartMin*

开始时间：分

*byStopHour*

结束时间：时

*byStopMin*

结束时间：分

## 8.121 NET\_DVR\_SDKABL:SDK 功能信息

```
struct{
    DWORD    dwMaxLoginNum;
    DWORD    dwMaxRealPlayNum;
    DWORD    dwMaxPlayBackNum;
    DWORD    dwMaxAlarmChanNum;
    DWORD    dwMaxFormatNum;
    DWORD    dwMaxFileSearchNum;
    DWORD    dwMaxLogSearchNum;
    DWORD    dwMaxSerialNum;
    DWORD    dwMaxUpgradeNum;
    DWORD    dwMaxVoiceComNum;
    DWORD    dwMaxBroadCastNum;
    DWORD    dwRes[10];
}NET_DVR_SDKABL, *LPNET_DVR_SDKABL;
```

### Members

*dwMaxLoginNum*

最大注册用户数

*dwMaxRealPlayNum*

最大实时预览的路数

*dwMaxPlayBackNum*

最大回放或下载的路数

*dwMaxAlarmChanNum*

最大建立报警通道的路数

*dwMaxFormatNum*

最大硬盘格式化的路数

*dwMaxFileSearchNum*

最大文件搜索的路数

*dwMaxLogSearchNum*

最大日志搜索的路数

*dwMaxSerialNum*

最大建立透明通道的路数

*dwMaxUpgradeNum*

最大升级的路数

*dwMaxVoiceComNum*

最大语音转发的路数

*dwMaxBroadCastNum*

最大语音广播的路数

*dwRes*

保留，置为 0

## 8.122 NET\_DVR\_SDKLOCAL\_CFG:模拟能力集参数

```
struct{
    BYTE    byEnableAbilityParse;
    BYTE    byVoiceComMode;
    BYTE    byRes[510];
}NET_DVR_SDKLOCAL_CFG,*LPNET_DVR_SDKLOCAL_CFG;
```

### Members

*byEnableAbilityParse*

使用能力集解析库：0- 不使用，1- 使用，默认不使用

*byVoiceComMode*

对讲模式：0- 使用对讲库（默认），1- 使用 windows api 模式

*byRes*

保留

## 8.123 NET\_DVR\_SDKSTATE:SDK 状态信息

```
struct{
    DWORD    dwTotalLoginNum;
    DWORD    dwTotalRealPlayNum;
    DWORD    dwTotalPlayBackNum;
    DWORD    dwTotalAlarmChanNum;
    DWORD    dwTotalFormatNum;
    DWORD    dwTotalFileSearchNum;
    DWORD    dwTotalLogSearchNum;
    DWORD    dwTotalSerialNum;
```

```

    DWORD    dwTotalUpgradeNum;
    DWORD    dwTotalVoiceComNum;
    DWORD    dwTotalBroadCastNum;
    DWORD    dwRes[10];
}NET_DVR_SDKSTATE,*LPNET_DVR_SDKSTATE;

```

### Members

*dwTotalLoginNum*

当前注册的用户数

*dwTotalRealPlayNum*

当前实时预览的路数

*dwTotalPlayBackNum*

当前回放或下载的路数

*dwTotalAlarmChanNum*

当前建立报警通道的路数

*dwTotalFormatNum*

当前硬盘格式化的路数

*dwTotalFileSearchNum*

当前文件搜索的路数

*dwTotalLogSearchNum*

当前日志搜索的路数

*dwTotalSerialNum*

当前建立透明通道的路数

*dwTotalUpgradeNum*

当前升级的路数

*dwTotalVoiceComNum*

当前语音转发的路数

*dwTotalBroadCastNum*

当前语音广播的路数

*dwRes*

保留，置为 0

## 8.124 NET\_DVR\_SECURITY\_CFG:安全认证配置结构体

```

struct{
    DWORD    dwSize;
    BYTE     byRes1[3];
    BYTE     byWebAuthentication;
    BYTE     byRtspAuthentication;
    BYTE     byTelnetServer;
    BYTE     bySSHServer;
    BYTE     byIllegalLoginLock;
    BYTE     byRes[28];
}NET_DVR_SECURITY_CFG,*LPNET_DVR_SECURITY_CFG;

```

### Members

*dwSize*

结构体大小

*byRes1*

保留，置为 0

*byWebAuthentication*

web 认证配置：0- Digest，1- basic(默认)

*byRtspAuthentication*

rtsp 认证配置：0- 禁用，1-basic(默认)

*byTelnetServer*

telnet 设置：0- 禁用，1- 启用

*bySSHServer*

SSH 设置：0- 禁用（默认），1- 启用

*byIllegalLoginLock*

开启登录锁定：0- 启用（默认），1- 禁用

*byRes*

保留，置为 0

#### Remarks

- 安全认证配置，只有管理员用户（admin）才有权限。
- 安全认证参数配置能力，对应安全认证配置能力集 (SecurityAbility)，相关接口：  
[NET\\_DVR\\_GetDeviceAbility](#)(能力集类型：DEVICE\_ABILITY\_INFO)。

## 8.125 NET\_DVR\_SETUPALARM\_PARAM:报警布防参数

struct{

DWORD dwSize;

BYTE byLevel;

BYTE byAlarmInfoType;

BYTE byRetAlarmTypeV40;

BYTE byRetDevInfoVersion;

BYTE byRetVQDAlarmType;

BYTE byFaceAlarmDetection;

BYTE bySupport;

BYTE byRes[9];

}NET\_DVR\_SETUPALARM\_PARAM, \*LPNET\_DVR\_SETUPALARM\_PARAM;

#### Members

*dwSize*

结构体大小

*byLevel*

布防优先级：0- 一等级（高），1- 二等级（中），2- 三等级（低）

*byAlarmInfoType*

智能交通报警信息上传类型：0- 老报警信息（[NET\\_DVR\\_PLATE\\_RESULT](#)），1- 新报警信息  
([NET\\_ITS\\_PLATE\\_RESULT](#))

*byRetAlarmTypeV40*

保留，置为 0



*byRetDevInfoVersion*

保留，置为 0

*byRetVQDAlarmType*

保留，置为 0

*byFaceAlarmDetection*

保留，置为 0

*bySupport*

按位表示，值：0-上传，1-不上传

bit0- 表示二级布防是否上传图片

*byRes*

保留，置为 0

#### Remarks

- 一级布防最大连接数为 1 个，二级最大连接数为 3 个，三级最大连接数为 5 个，设备支持一级、二级和三级布防同时进行，一级布防优先上传信息。
- *byAlarmInfoType* 是否支持新报警信息可从注册返回的能力获知，详见 [NET\\_DVR\\_DEVICEINFO\\_V30](#) 结构中 *bySupport1*（表示是否支持车牌新报警信息），如果注册返回能力不支持，设备仅支持老报警信息上传。布防时设置 *byAlarmInfoType* 值为 0 或者 1，则 [NET\\_DVR\\_SetDVRMessageCallBack\\_V31](#) 设置的回调或者函数中上传的报警信息类型（*ICommand*）对应 *COMM\_UPLOAD\_PLATE\_RESULT* 或者 *COMM\_ITS\_PLATE\_RESULT*。

## 8.126 NET\_DVR\_SIMPLE\_DAYTIME:时间点信息结构体

```
struct{
    BYTE    byHour;
    BYTE    byMinute;
    BYTE    bySecond;
    BYTE    byRes;
}NET_DVR_SIMPLE_DAYTIME,*LPNET_DVR_SIMPLE_DAYTIME;
```

#### Members

*byHour*

时

*byMinute*

分

*bySecond*

秒

*byRes*

保留，置为 0

## 8.127 NET\_DVR\_SINGLE\_NFS:NFS 配置子结构

```
struct{
    char    sNfsHostIPAddr[16];
    BYTE    sNfsDirectory[PATHNAME\_LEN];
```

```
}NET_DVR_SINGLE_NFS, *LPNET_DVR_SINGLE_NFS;
```

### Members

*sNfsHostIPAddr*

网络文件系统主机 IP 地址

*sNfsDirectory*

NFS 路径

## 8.128 NET\_DVR\_SINGLE\_RS232:RS232 串口参数子结构

```
struct{
    DWORD        dwBaudRate;
    BYTE         byDataBit;
    BYTE         byStopBit;
    BYTE         byParity;
    BYTE         byFlowcontrol;
    DWORD        dwWorkMode;
}NET_DVR_SINGLE_RS232, *LPNET_DVR_SINGLE_RS232;
```

### Members

*dwBaudRate*

波特率(bps): 0-50, 1-75, 2-110, 3-150, 4-300, 5-600, 6-1200, 7-2400, 8-4800, 9-9600, 10-19200, 11-38400, 12-57600, 13-76800, 14-115.2k

*byDataBit*

数据有几位: 0-5 位, 1-6 位, 2-7 位, 3-8 位

*byStopBit*

停止位: 0-1 位, 1-2 位

*byParity*

是否校验: 0-无校验, 1-奇校验, 2-偶校验

*byFlowcontrol*

是否流控: 0-无, 1-软流控, 2-硬流控

*dwWorkMode*

工作模式, 0—窄带传输(232 串口用于 PPP 拨号), 1—控制台(232 串口用于参数控制), 2—透明通道, 3-ptz 模式 (审讯温湿度传感器), 4-报警盒模式

## 8.129 NET\_DVR\_SINGLE\_PLAN\_SEGMENT:计划参数结构体

```
struct{
    BYTE         byEnable;
    BYTE         byDoorStatus;
    BYTE         byVerifyMode;
    BYTE         byRes[5];
    NET\_DVR\_TIME\_SEGMENT struTimeSegment;
}NET_DVR_SINGLE_PLAN_SEGMENT,*LPNET_DVR_SINGLE_PLAN_SEGMENT;
```

### Members

*byEnable*

是否使能：0- 否，1- 是

*byDoorStatus*

门状态模式：0- 无效，1- 休眠，2- 常开状态，3- 常闭状态。（设置门状态计划时有效）

*byVerifyMode*

验证方式：0- 无效，1- 休眠，2- 刷卡+密码，3- 刷卡，4- 刷卡或密码，5- 指纹，6- 指纹+密码，7- 指纹或刷卡，8- 指纹+刷卡，9- 指纹+刷卡+密码（无先后顺序）。（读卡器验证方式计划时有效）

*byRes*

保留，置为 0

*struTimeSegment*

计划时间段，包括开始时间和结束时间

## 8.130 **NET\_DVR\_SMSRELATIVEPARAM**:短信相关参数结构体

```
struct{
    DWORD                dwSize;
    BYTE                 bEnableSmsAlarm;
    BYTE                 byRes1[7];
    NET\_DVR\_PHONECFG     struWhiteList[MAX\_WHITELIST\_NUM];
    BYTE                 byRes2[32];
}NET_DVR_SMSRELATIVEPARAM, *LPNET_DVR_SMSRELATIVEPARAM;
```

### Members

*dwSize*

结构体大小

*bEnableSmsAlarm*

是否启动短信报警，0- 禁用，1- 启用

*byRes1*

保留，置为 0

*struWhiteList*

白名单列表

*byRes2*

保留，置为 0

### Remarks

- 报警及异常需增加短信报警联动方式，当使能为真时，该联动方式便能生效。
- 短信报警需配合白名单使用，短信报警发生时将短信发送给白名单中的号码。白名单也可以配合短信控制设备使用，只有白名单上的号码才允许发送控制短信给设备，使设备进行相应操作。
- 设备是否支持短信相关配置或者支持的参数能力，可以通过设备能力集进行判断，对应无线网络能力集([NetworkSetting](#))，相关接口：[NET\\_DVR\\_GetDeviceAbility](#)，能力集类型：DEVICE\_NETWORK\_ABILITY，节点：<MessageConfig>。

## 8.131 **NET\_DVR\_SNAPCFG**:触发参数

```
struct{
    DWORD    dwSize;
```

```

    BYTE    byRelatedDriveWay;
    BYTE    bySnapTimes;
    WORD    wSnapWaitTime;
    WORD    wIntervalTime[MAX\_INTERVAL\_NUM];
    DWORD    dwSnapVehicleNum;
    BYTE    byRes2[20];
}NET_DVR_SNAPCFG, *LPNET_DVR_SNAPCFG;

```

### Members

*dwSize*

结构体大小

*byRelatedDriveWay*

触发 IO 关联的车道号，取值范围[0,9]

*bySnapTimes*

线圈抓拍次数，0-不抓拍，非 0-连拍次数，目前最大 5 次

*wSnapWaitTime*

抓拍等待时间，单位 ms，取值范围[0,60000]

*wIntervalTime*

连拍间隔时间，单位 ms，取值范围[67,60000]

*dwSnapVehicleNum*

抓拍匹配序号，与车牌报警上传 [NET ITS PLATE RESULT](#) 中 dwMatchNo 字段对应，网络触发连拍时有效

*byRes2*

保留，置为 0

## 8.132 NET\_DVR\_SYSTEM\_TIME:时间信息

```

struct{
    WORD    wYear;
    WORD    wMonth;
    WORD    wDay;
    WORD    wHour;
    WORD    wMinute;
    WORD    wSecond;
    WORD    wMilliSec;
    BYTE    byRes[2];
}NET_DVR_SYSTEM_TIME,*LPNET_DVR_SYSTEM_TIME;

```

### Members

*wYear*

年

*wMonth*

月

*wDay*

日

*wHour*

时  
*wMinute*  
 分  
*wSecond*  
 秒  
*wMilliSec*  
 毫秒  
*byRes*  
 保留

### 8.133 NET\_DVR\_TIME:时间参数

```

struct{
    DWORD    dwYear;
    DWORD    dwMonth;
    DWORD    dwDay;
    DWORD    dwHour;
    DWORD    dwMinute;
    DWORD    dwSecond;
}NET_DVR_TIME, *LPNET_DVR_TIME;
  
```

#### Members

*dwYear*  
 年  
*dwMonth*  
 月  
*dwDay*  
 日  
*dwHour*  
 时  
*dwMinute*  
 分  
*dwSecond*  
 秒

### 8.134 NET\_DVR\_TIME\_EX:时间参数

```

struct{
    WORD    wYear;
    BYTE    byMonth;
    BYTE    byDay;
    BYTE    byHour;
    BYTE    byMinute;
    BYTE    bySecond;
}
  
```

```

    BYTE    byRes;
}NET_DVR_TIME_EX, *LPNET_DVR_TIME_EX;

```

#### Members

```

wYear
    年
byMonth
    月
byDay
    日
byHour
    时
byMinute
    分
bySecond
    秒
byRes
    保留

```

## 8.135 NET\_DVR\_TIME\_V30:时间参数(扩展)

```

struct{
    WORD    wYear;
    BYTE    byMonth;
    BYTE    byDay;
    BYTE    byHour;
    BYTE    byMinute;
    BYTE    bySecond;
    BYTE    byRes;
    WORD    wMilliSec;
    BYTE    byRes1[2];
}NET_DVR_TIME_V30, *LPNET_DVR_TIME_V30;

```

#### Members

```

wYear
    年
byMonth
    月
byDay
    日
byHour
    时
byMinute
    分
bySecond
    秒

```

*byRes*

保留

*wMilliSec*

毫秒

*byRes1*

保留

### 8.136 **NET\_DVR\_TIME\_SEGMENT**:时间段参数结构体

```
struct{
    NET_DVR_SIMPLE_DAYTIME    struBeginTime;
    NET_DVR_SIMPLE_DAYTIME    struEndTime;
}NET_DVR_TIME_SEGMENT,*LPNET_DVR_TIME_SEGMENT;
```

#### Members

*struBeginTime*

开始时间点（时分秒）

*struEndTime*

开始时间点（时分秒）

### 8.137 **NET\_DVR\_TIMEPOINT**:时间点参数结构体

```
struct{
    DWORD    dwMonth;
    DWORD    dwWeekNo;
    DWORD    dwWeekDate;
    DWORD    dwHour;
    DWORD    dwMin;
}NET_DVR_TIMEPOINT,*LPNET_DVR_TIMEPOINT;
```

#### Members

*dwMonth*

月：[0,11]取值分别表示第 1 个月到第 12 个月

*dwWeekNo*

周：0—第 1 周，1—第 2 周，2—第 3 周，3—第 4 周，4—最后一周

*dwWeekDate*

星期：0—星期日，1—星期一，2—星期二，3—星期三，4—星期四，5—星期五，6—星期六

*dwHour*

小时：开始时间 0~23，结束时间 1~23

*dwMin*

分：0~59

## 8.138 NET\_DVR\_USER\_INFO\_V30:单用户参数

```

struct{
    BYTE                sUserName[NAME_LEN];
    BYTE                sPassword[PASSWD_LEN];
    BYTE                byLocalRight[MAX_RIGHT];
    BYTE                byRemoteRight[MAX_RIGHT];
    BYTE                byNetPreviewRight[MAX_CHANNUM_V30];
    BYTE                byLocalPlaybackRight[MAX_CHANNUM_V30];
    BYTE                byNetPlaybackRight[MAX_CHANNUM_V30];
    BYTE                byLocalRecordRight[MAX_CHANNUM_V30];
    BYTE                byNetRecordRight[MAX_CHANNUM_V30];
    BYTE                byLocalPTZRight[MAX_CHANNUM_V30];
    BYTE                byNetPTZRight[MAX_CHANNUM_V30];
    BYTE                byLocalBackupRight[MAX_CHANNUM_V30];
    NET_DVR_IPADDR      struUserIP;
    BYTE                byMACAddr[MACADDR_LEN];
    BYTE                byPriority;
    BYTE                byAlarmOnRight;
    BYTE                byAlarmOffRight;
    BYTE                byBypassRight;
    BYTE                byRes[14];
}NET_DVR_USER_INFO_V30,*LPNET_DVR_USER_INFO_V30;

```

### Members

*sUserName*

用户名

*sPassword*

密码

*byLocalRight*

本地操作权限，参数取值为 1 表示使能：

数组 0—本地控制云台

数组 1—本地手动录象

数组 2—本地回放

数组 3—本地设置参数

数组 4—本地查看状态、日志

数组 5—本地高级操作（升级，硬盘管理（格式化、设置硬盘属性、设置盘组、阵列扩容、RAID 固件升级））

数组 6—本地查看参数

数组 7—本地管理模拟和 IP camera

数组 8—本地备份

数组 9—本地关机/重启

*byRemoteRight*

远程操作权限，参数取值为 1 表示使能：



- 数组 0—远程控制云台
- 数组 1—远程手动录象
- 数组 2—远程回放
- 数组 3—远程设置参数（恢复默认参数，写日志）
- 数组 4—远程查看状态、日志
- 数组 5—远程高级操作（升级，硬盘管理（格式化、设置硬盘属性、设置盘组、阵列扩容、RAID 固件升级），JPEG 抓图，前面板锁定与解锁）
- 数组 6—远程发起语音对讲
- 数组 7—远程预览
- 数组 8—远程请求报警上传、报警输出
- 数组 9—远程控制，本地输出
- 数组 10—远程控制串口
- 数组 11—远程查看参数
- 数组 12—远程管理模拟和 IP camera
- 数组 13—远程关机/重启

**byNetPreviewRight**

远程可以预览的通道：1-有权限，0-无权限

**byLocalPlaybackRight**

本地可以回放的通道：1-有权限，0-无权限

**byNetPlaybackRight**

远程可以回放的通道：1-有权限，0-无权限

**byLocalRecordRight**

本地可以录像的通道：1-有权限，0-无权限

**byNetRecordRight**

远程可以录像的通道：1-有权限，0-无权限

**byLocalPTZRight**

本地可以控制 PTZ 的通道：1-有权限，0-无权限

**byNetPTZRight**

远程可以控制 PTZ 的通道：1-有权限，0-无权限

**byLocalBackupRight**

本地备份权限通道：1-有权限，0-无权限

**struUserIP**

用户 IP 地址(为 0 时表示允许任何地址)

**byMACAddr**

物理地址

**byPriority**

优先级：0xff-无，0-低，1-中，2-高

无（表示不支持优先级的设置）

低（默认权限：包括本地和远程回放，本地和远程查看日志和状态，本地和远程关机/重启）

中（包括本地和远程控制云台，本地和远程手动录像，本地和远程回放，语音对讲和远程预览，本地备份，本地/远程关机/重启）

高（管理员）

**byAlarmOnRight**

报警输入口布防权限

*byAlarmOffRight*

报警输入口撤防权限

*byBypassRight*

报警输入口旁路权限

*byRes*

保留，置为 0

## 8.139 NET\_DVR\_USER\_LOGIN\_INFO: 用户登录参数

```
struct{
    char                sDeviceAddress[NET_DVR_DEV_ADDRESS_MAX_LEN];
    BYTE                byRes1;
    WORD                wPort;
    char                sUserName[NET_DVR_LOGIN_USERNAME_MAX_LEN];
    char                sPassword[NET_DVR_LOGIN_PASSWD_MAX_LEN];
    fLoginResultCallBack cbLoginResult;
    void                *pUser;
    BOOL                bUseAsynLogin;
    BYTE                byRes2[128];
}NET_DVR_USER_LOGIN_INFO,*LPNET_DVR_USER_LOGIN_INFO;
```

### Members

*sDeviceAddress*

设备地址，IP 或者普通域名

*byRes1*

保留，设为 0

*wPort*

设备端口号，例如：8000

*sUserName*

登录用户名，例如：admin

*sPassword*

登录密码，例如：12345

*cbLoginResult*

登录状态回调函数，bUseAsynLogin 为 1 时有效

*pUser*

用户数据

*bUseAsynLogin*

是否异步登录：0- 否，1- 是

*byRes2*

保留，置为 0

### Callback Function

```
typedef void(CALLBACK *fLoginResultCallBack)(
    LONG                lUserID,
    DWORD               dwResult,
    LPNET_DVR_DEVICEINFO_V30 lpDeviceInfo,
```

```
void                                     *pUser);
```

### Callback Function Parameters

*lUserID*

[out] 用户 ID, NET\_DVR\_Login\_V40 的返回值

*dwResult*

[out] 登录状态: 0- 异步登录失败, 1- 异步登录成功

*lpDeviceInfo*

[out] 设备信息, 设备序列号、通道、能力等参数

*pUser*

[out] 用户数据

## 8.140 NET\_DVR\_USER\_V30:用户参数

```
struct{
    DWORD                dwSize;
    NET_DVR_USER_INFO_V30 struUser[MAX_USERNUM_V30];
}NET_DVR_USER_V30,*LPNET_DVR_USER_V30;
```

### Members

*dwSize*

结构体大小

*struUser*

用户信息参数

## 8.141 NET\_DVR\_VIDEO\_CALL\_COND:可视对讲信令处理条件参数

```
struct{
    DWORD                dwSize;
    BYTE                 byRes[128];
}NET_DVR_VIDEO_CALL_COND,*LPNET_DVR_VIDEO_CALL_COND;
```

### Members

*dwSize*

结构体大小

*byRes*

保留, 置为 0

## 8.142 NET\_DVR\_VIDEO\_CALL\_PARAM:可视对讲信令处理参数

```
struct{
    DWORD                dwSize;
    DWORD                dwCmdType;
    WORD                 wPeriod;
    WORD                 wBuildingNumber;
```

```
WORD    wUnitNumber;
SHORT   wFloorNumber;
WORD    wRoomNumber;
BYTE    byRes[118];
```

```
}NET_DVR_VIDEO_CALL_PARAM, *LPNET_DVR_VIDEO_CALL_PARAM;
```

### Members

*dwSize*

结构体大小

*dwCmdType*

信令类型：0- 请求呼叫，1- 取消本次呼叫，2- 接听本次呼叫，3- 拒绝本地来电呼叫，4- 被叫响铃超时，5- 结束本次通话，6- 设备正在通话中，7- 客户端正在通话中

*wPeriod*

期号，取值范围：[0,9]

*wBuildingNumber*

楼号

*wUnitNumber*

单元号

*wFloorNumber*

层号

*wRoomNumber*

房间号

*byRes*

保留，置为 0

## 8.143 NET\_DVR\_VALID\_PERIOD\_CFG:有效期参数结构体

```
struct{
    BYTE                byEnable;
    BYTE                byRes1[3];
    NET\_DVR\_TIME\_EX     struBeginTime;
    NET\_DVR\_TIME\_EX     struEndTime;
    BYTE                byRes2[32];
}NET_DVR_VALID_PERIOD_CFG,*LPNET_DVR_VALID_PERIOD_CFG;
```

### Members

*byEnable*

是否启用该有效期：0- 不启用，1- 启用

*byRes1*

保留，置为 0

*struBeginTime*

有效期起始时间

*struEndTime*

有效期结束时间

*byRes2*

保留，置为 0

## 8.144 NET\_DVR\_WEEK\_PLAN\_CFG:周计划配置结构体

```
struct{
    DWORD                dwSize;
    BYTE                 byEnable;
    BYTE                 byRes1[3];
    NET_DVR_SINGLE_PLAN_SEGMENT struPlanCfg[MAX_DAYS][MAX_TIMESEGMENT_V30];
    BYTE                 byRes2[16];
}NET_DVR_WEEK_PLAN_CFG,*LPNET_DVR_WEEK_PLAN_CFG;
```

### Members

*dwSize*

结构体大小

*byEnable*

是否使能：0- 否，1- 是

*byRes1*

保留，置为 0

*struPlanCfg*

周计划参数，一周 7 天，每天最多 8 个时间段

*byRes2*

保留，置为 0

## 8.145 NET\_DVR\_WEEK\_PLAN\_COND:卡权限周计划配置条件结构体

```
struct{
    DWORD    dwSize;
    DWORD    dwWeekPlanNumber;
    WORD     wLocalControllerID;
    BYTE     byRes[106];
}NET_DVR_WEEK_PLAN_COND,*LPNET_DVR_WEEK_PLAN_COND;
```

### Members

*dwSize*

结构体大小

*dwWeekPlanNumber*

周计划编号

*wLocalControllerID*

就地控制器序号[1,64]，0 表示门禁主机

*byRes*

保留，置为 0

## 8.146 NET\_DVR\_WIFI\_CFG:IP 监控设备无线参数结构体

```
struct{
    DWORD                dwSize;
    NET_DVR_WIFI_CFG_EX  struWifiCfg;
}NET_DVR_WIFI_CFG,*LPNET_DVR_WIFI_CFG;
```

### Members

*dwSize*

结构体大小

*struWifiCfg*

无线参数子结构体

## 8.147 NET\_DVR\_WIFI\_CFG\_EX:IP 监控设备无线参数的子结构体

```
struct{
    NET_DVR_WIFIETHERNET  struEtherNet;
    char                   sEssid[IW_ESSID_MAX_SIZE];
    DWORD                  dwMode;
    DWORD                  dwSecurity;
    union{
        struct{
            DWORD  dwAuthentication;
            DWORD  dwKeyLength;
            DWORD  dwKeyType;
            DWORD  dwActive;
            char    sKeyInfo[WIFI_WEP_MAX_KEY_COUNT][WIFI_WEP_MAX_KEY_LENGTH];
        }wep;
        struct{
            DWORD  dwKeyLength;
            char    sKeyInfo[WIFI_WPA_PSK_MAX_KEY_LENGTH];
            BYTE    byEncryptType;
            char    sNewKeyInfo[WIFI_WPA_PSK_MAX_HEXKEY_LENGTH];
            BYTE    byKeyType;
            BYTE    byRes[7];
        }wpa_psk;
        struct{
            BYTE    byEncryptType;
            BYTE    byAuthType;
            BYTE    byRes[2];
            union{
                struct{
                    BYTE    byEapolVersion;
                    BYTE    byAuthType;
                }
            }
        }
    }
}
```

```

        BYTE    byRes1[2];
        BYTE    byAnonyIdentity[NAME_LEN];
        BYTE    byUserName[NAME_LEN];
        BYTE    byPassword[NAME_LEN];
        BYTE    byRes[44];
    }EAP_TTLS;
    struct{
        BYTE    byEapolVersion;
        BYTE    byAuthType;
        BYTE    byPeapVersion;
        BYTE    byPeapLabel;
        BYTE    byAnonyIdentity[NAME_LEN];
        BYTE    byUserName[NAME_LEN];
        BYTE    byPassword[NAME_LEN];
        BYTE    byRes[44];
    }EAP_PEAP;
    struct{
        BYTE    byEapolVersion;
        BYTE    byRes1[3];
        BYTE    byIdentity[NAME_LEN];
        BYTE    byPrivateKeyPswd[NAME_LEN];
        BYTE    byRes[76];
    }EAP_TLS;
    }auth_param;
    }wpa_wpa2;
    }key;
}NET_DVR_WIFI_CFG_EX,*LPNET_DVR_WIFI_CFG_EX;

```

## Members

*struEtherNet*

WIFI 网口参数

*sEssid*

SSID

*dwMode*

工作模式：0-mange 模式，1-ad-hoc 模式

*dwSecurity*

加密模式：0- 不加密，1- WEP 加密，2- WPA-personal，3- WPA-enterprise，4- WPA2-personal，5- WPA2-enterprise

**wep 为 WEP 加密参数结构体**

*dwAuthentication*

权限类型：0-开放式，1-共享式

*dwKeyLength*

密钥长度：0-64 位；1-128 位；2-152 位

*dwKeyType*

密钥类型：0-16 进制；1-ASCII

*dwActive*

激活哪个密钥（值有 0、1、2、3），0 表示激活第一个，以此类推

*sKeyInfo*

密钥信息

**wpa\_psk 为 WPA-personal/WPA2-personal 加密参数结构体**

*dwKeyLength*

字符加密的长度，允许 8-63 个 ASCII 字符

*sKeyInfo*

字符密钥的信息，byKeyType 为 0 时有效

*byEncryptType*

WPA-personal/WPA2-personal 模式下加密类型：0- AES，1- TKIP

*sNewKeyInfo*

新类型密钥（支持 8-63 个 ASCII 字符以及 64 个十六制字符密钥），byKeyType 为 1 时有效

*byKeyType*

密钥类型：0- 老密钥类型（只支持 8-63 个 ASCII 字符），1- 新密钥类型（支持 8-63 个 ASCII 字符以及 64 个十六制字符密钥）

*byRes*

保留，置为 0

**wpa\_wpa2 为 WPA-enterprise/WPA2-enterpris 加密参数结构体**

*byEncryptType*

加密类型：0- AES，1- TKIP

*sKeyInfo*

认证类型：0- EAP\_TTLS，1- EAP\_PEAP，2- EAP\_TLS

*byRes*

保留，置为 0

[EAP\\_TTLS 为 EAP\\_TTLS 认证参数结构体](#)

*byEapolVersion*

EAPOL 版本：0- 版本 1，1- 版本 2

*byAuthType*

内部认证方式：0- PAP，1- MSCHAPV2

*byRes1*

保留，置为 0

*byAnonylIdentity*

匿名身份

*byUserName*

用户名

*byPassword*

密码

*byRes*

保留，置为 0

[EAP\\_PEAP 为 EAP\\_PEAP 认证参数结构体](#)

*byEapolVersion*

EAPOL 版本：0- 版本 1，1- 版本 2

*byAuthType*



内部认证方式：0- GTC，1- MD5，2- MSCHAPV2

*byPeapVersion*

PEAP 版本：0- 版本 0，1- 版本 1

*byPeapLabel*

PEAP 标签：0- 老标签，1- 新标签

*byAnonylIdentity*

匿名身份

*byUserName*

用户名

*byPassword*

密码

*byRes*

保留，置为 0

*EAP\_TLS* 为 *EAP\_TLS* 认证参数结构体

*byEapolVersion*

EAPOL 版本：0- 版本 1，1- 版本 2

*byRes1*

保留，置为 0

*byIdentity*

身份

*byPrivateKeyPswd*

私钥密码

*byRes*

保留，置为 0

## 8.148 **NET\_DVR\_WIFIETHERNET:**无线网口参数结构体

```
struct{
    char        slpAddress[16];
    char        slpMask[16];
    BYTE        byMACAddr[MACADDR_LEN];
    BYTE        byCloseWifi;
    BYTE        bRes;
    DWORD       dwEnableDhcp;
    DWORD       dwAutoDns;
    char        sFirstDns[16];
    char        sSecondDns[16];
    char        sGatewayIpAddr[16];
    BYTE        bRes2[8];
}NET_DVR_WIFIETHERNET,*LPNET_DVR_WIFIETHERNET;
```

### Members

*slpAddress*

设备 IP 地址

*slpMask*

掩码

*byMACAddr*

物理地址，仅获取不能设置

*byCloseWifi*

是否关闭 wifi 连接：0- 不关闭，1- 关闭

*bRes*

保留，置为 0

*dwEnableDhcp*

是否启动 DHCP：0-不启动，1-启动

*dwAutoDns*

如果启动 DHCP 是否自动获取 DNS：0-不自动获取，1-自动获取；对于有线如果启动 DHCP，目前自动获取 DNS

*sFirstDns*

第一个 DNS 域名

*sSecondDns*

第二个 DNS 域名

*sGatewayIpAddr*

网关地址

*bRes2*

保留，置为 0

## 8.149 NET\_DVR\_XML\_CONFIG\_INPUT:透传接口输入参数

```
struct{
    DWORD          dwSize;
    void*          lpRequestUrl;
    DWORD          dwRequestUrlLen;
    void*          lpInBuffer;
    DWORD          dwInBufferSize;
    DWORD          dwRecvTimeOut;
    BYTE           byRes[32];
} NET_DVR_XML_CONFIG_INPUT, *LPNET_DVR_XML_CONFIG_INPUT;
```

### Members

*dwSize*

结构体大小

*lpRequestUrl*

请求信令，字符串格式

*dwRequestUrlLen*

请求信令长度，字符串长度

*lpInBuffer*

输入参数缓冲区，XML 格式

*dwInBufferSize*

输入参数缓冲区大小

*dwRecvTimeOut*

接收超时时间，单位：ms，填 0 则使用默认超时 5s

*byRes*

保留，置为 0

## 8.150 **NET\_DVR\_XML\_CONFIG\_OUTPUT:**透传接口输出参数

```
struct{
    DWORD        dwSize;
    void*        lpOutBuffer;
    DWORD        dwOutBufferSize;
    DWORD        dwReturnedXMLSize;
    void*        lpStatusBuffer;
    DWORD        dwStatusSize;
    BYTE         byRes[32];
} NET_DVR_XML_CONFIG_OUTPUT, *LPNET_DVR_XML_CONFIG_OUTPUT;
```

### Members

*dwSize*

结构体大小

*lpOutBuffer*

输出参数缓冲区，XML 格式

*dwOutBufferSize*

输出参数缓冲区大小(内存大小)

*dwReturnedXMLSize*

实际输出的 XML 内容大小

*lpStatusBuffer*

返回的状态参数(XML 格式: [ResponseStatus](#)), 获取失败时返回, 获取成功时不会赋值, 如果不需要, 可以置 NULL; 设置时均会返回。

*dwStatusSize*

状态缓冲区大小(内存大小)

*byRes*

保留，置为 0

## 8.151 **ResponseStatus XML Block:**设备返回的响应状态的 XML 描述

### ResponseStatus XML Block

```
<ResponseStatus version="1.0" xmlns="http://www.std-cgi.org/ver20/XMLSchema">
  <requestURL/><!-- req, ro,xs:string --> </requestURL>
  <statusCode><!-- req, ro,xs:integer --></statusCode>
  <!-- O=1-OK, 2-Device Busy, 3-Device Error, 4-Invalid Operation, 5-Invalid XML Format,
  6-Invalid XML Content; 7-Reboot Required -->
  <statusString><!-- req, ro,xs:string --></statusString>
  <subStatusCode><!-- req, ro,xs:string --></subStatusCode>
```

```
<ID><!-- opt,xs:string --><ID>
</ResponseStatus>
```

## 8.152 NET\_DVR\_ZONEANDDST:夏令时参数结构体

```
struct{
    DWORD          dwSize;
    BYTE           byRes1[16];
    DWORD          dwEnableDST;
    BYTE           byDSTBias;
    BYTE           byRes2[3];
    NET\_DVR\_TIMEPOINT struBeginPoint;
    NET\_DVR\_TIMEPOINT struEndPoint;
}NET_DVR_ZONEANDDST,*LPNET_DVR_ZONEANDDST;
```

### Members

*dwSize*

结构体大小

*byRes1*

保留，置为 0

*dwEnableDST*

是否启用夏时制：0—不启用，1—启用

*byDSTBias*

夏令时偏移值（以分钟计）：30min，60min，90min，120min

*byRes2*

保留，置为 0

*struBeginPoint*

夏时制开始时间

*struEndPoint*

夏时制停止时间

## 8.153 NET\_VCA\_POINT:点坐标参数结构体

```
struct{
    float  fX;
    float  fY;
}NET_VCA_POINT,*LPNET_VCA_POINT;
```

**Members***fX*

X 轴坐标, 取值范围[0.001,1]

*fY*

Y 轴坐标, 取值范围[0.001,1]

**8.154 NET\_VCA\_POLYGON: 多边形结构体**

```
struct{
    DWORD          dwPointNum;
    NET_VCA_POINT  struPos[VCA_MAX_POLYGON_POINT_NUM];
}NET_VCA_POLYGON,*LPNET_VCA_POLYGON;
```

**Members***dwPointNum*

有效点（大于等于 3），若是 3 点在一条线上认为是无效区域，线交叉认为是无效区域

*struPos*

多边形边界点，最大值为 10

**8.155 NET\_VCA\_RECT: 区域框参数结构体**

```
struct{
    float  fX;
    float  fY;
    float  fWidth;
    float  fHeight;
}NET_VCA_RECT,*LPNET_VCA_RECT;
```

**Members***fX*

边界框左上角点的 X 轴坐标, 取值范围[0.001,1]

*fY*

边界框左上角点的 Y 轴坐标, 取值范围[0.001,1]

*fWidth*

边界框的宽度, 取值范围[0.001,1]

*fHeight*

边界框的高度, 取值范围[0.001,1]

**8.156 NET\_VCA\_SIZE\_FILTER: 尺寸过滤器参数结构体**

```
struct{
    BYTE          byActive;
    BYTE          byMode;
    BYTE          byRes[2];
```

[NET\\_VCA\\_RECT](#)      `struMiniRect;`

[NET\\_VCA\\_RECT](#)      `struMaxRect;`

`}NET_VCA_SIZE_FILTER,*LPNET_VCA_SIZE_FILTER;`

## Members

### *byActive*

是否激活尺寸过滤器，0-否，非 0-是

### *byMode*

过滤器模式，具体定义如下：

`enum _VCA_SIZE_FILTER_MODE_{`

`IMAGE_PIX_MODE,`

`REAL_WORLD_MODE,`

`DEFAULT_MODE`

`}SIZE_FILTER_MODE`

*IMAGE\_PIX\_MODE*

根据像素大小设置

*REAL\_WORLD\_MODE*

根据实际大小设置

*DEFAULT\_MODE*

默认模式(目前 ATM 支持)

### *byRes*

保留，置为 0

### *struMiniRect*

最小目标框，全 0 表示不设置（瞳距）。实际模式时取值范围为 0~50，单位为 m

### *struMaxRect*

最大目标框，全 0 表示不设置