# Device Network SDK (Card-Based Access Control)

## Developer Guide

# Legal Information

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, THE DOCUMENT IS PROVIDED "AS IS" AND "WITH ALL FAULTS AND ERRORS". OUR COMPANY MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT. IN NO EVENT WILL OUR COMPANY BE LIABLE FOR ANY SPECIAL, CONSEQUENTIAL, INCIDENTAL, OR INDIRECT DAMAGES, INCLUDING, AMONG OTHERS, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION OR LOSS OF DATA, CORRUPTION OF SYSTEMS, OR LOSS OF DOCUMENTATION, WHETHER BASED ON BREACH OF CONTRACT, TORT (INCLUDING NEGLIGENCE), OR OTHERWISE, IN CONNECTION WITH THE USE OF THE DOCUMENT, EVEN IF OUR COMPANY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES OR LOSS.

# Contents

# Chapter 1 Overview

Access Control is the selective restriction of access to a place or other resources. The access control applications integrated via Device Network SDK (hereafter referred to as "HCNetSDK") in this manual take the card as the management and control unit, which indicates that all applications are integrated around the basic unit. That is, the fingerprints, faces, and other attributes will be linked to a card for management and control.

## 1.1 Introduction

This manual mainly provides the integration flows and related APIs for access controller, access control terminal, fingerprint access control terminal, face recognition terminal, elevator controller, swing barrier, and so on, to implement the following functions : schedule configuration, card/fingerprint/face information management, alarm/event configuration, and door control.

## 1.2 Update History

### Summary of Changes in Version 6.1.5.20_Oct., 2020

1. Added two query parameters **security** (the version No. of encryption scheme) and **iv** (the initialization vector) to the request URI ***/ISAPI/AccessControl/CapturePresetParam?format=json*** (related API: ***NET_DVR_STDXMLConfig*** ).
2. Added two query parameters **security** (the version No. of encryption scheme) and **iv** (the initialization vector) to the request URI ***/ISAPI/AccessControl/CaptureCardInfo?format=json*** (related API: ***NET_DVR_STDXMLConfig*** ).
3. Extended the capability message of collecting card information ***JSON_CardInfoCap*** and card information message ***JSON_CardInfo_Collection*** (related URIs: ***/ISAPI/AccessControl/CaptureCardInfo/capabilities?format=json*** and ***/ISAPI/AccessControl/CaptureCardInfo?format=json*** ; related API: ***NET_DVR_STDXMLConfig*** ):
   added two card types "FelicaCard" (FeliCa card) and "DesfireCard" (DESFire card) to the node **cardType**.
4. Added two query parameters **security** (the version No. of encryption scheme) and **iv** (the initialization vector) to the request URI ***/ISAPI/AccessControl/CaptureIDInfo?format=json*** (related API: ***NET_DVR_STDXMLConfig*** ).
5. Added a URI of getting details of failing to upload the user list of offline collection (related API: ***NET_DVR_STDXMLConfig*** ): GET ***/ISAPI/AccessControl/OfflineCapture/uploadFailedDetails?format=json*** .
6. Extended the result message of searching for the collected data ***JSON_SearchTaskResponse*** (related URI: ***/ISAPI/AccessControl/OfflineCapture/DataCollections/searchTask?format=json*** ; related API: ***NET_DVR_StartRemoteConfig*** ):

added two sub nodes **cardNo** (card No.) and **cardType** (card type) to the node **CardNoList** of **DataCollections**;
added two sub nodes **IdentityInfo** (identity information) and **CardIssueStatus** (issuing status list of cards containing face pictures and fingerprints) to the node **DataCollections**.

7. Extended parameter message of offline collection rules ***JSON_RuleInfo*** (related URI: ***/ISAPI/AccessControl/OfflineCapture/ruleInfo?format=json*** ; related API: ***NET_DVR_STDXMLConfig*** ):
added two nodes **enableLocalIssueCard** (whether to enable issuing smart cards locally) and **isLocalStorage** (whether to store face picture and fingerprint information in the device locally).
8. Extended parameter message of offline collection progress ***JSON_CaptureProgress*** (related URI: ***/ISAPI/AccessControl/OfflineCapture/progress?format=json*** ; related API: ***NET_DVR_STDXMLConfig*** ):
added two nodes **reqIssueNum** (number of persons to be issued with smart cards) and **IssuedNum** (number of persons that have been issued with smart cards).
9. Added two query parameters **security** (the version No. of encryption scheme) and **iv** (the initialization vector) to the request URI ***/ISAPI/AccessControl/OfflineCapture/dataOutput?format=json*** (related API: ***NET_DVR_STDXMLConfig*** ).
10. Extended parameter message for exporting offline collected data ***JSON_DataOutputCfg*** (related URI: ***/ISAPI/AccessControl/OfflineCapture/dataOutput?format=json*** ; related API: ***NET_DVR_STDXMLConfig*** ):
added a node **type** (exporting type).
11. Extended the offline collection capability message ***JSON_OfflineCaptureCap*** (related URI: ***/ISAPI/AccessControl/OfflineCapture/capabilities?format=json*** ; related API: ***NET_DVR_STDXMLConfig*** ):
added three sub nodes **maxSize** (size of the card No. list), **cardNo** (card No.), and **cardType** (card type) to the node **CardNoList** of **DataCollections** of **SearchTask**;
added two sub nodes **IdentityInfo** (identity information) and **CardIssueStatus** (issuing status list of cards containing face pictures and fingerprints) to the node **DataCollections** of **SearchTask**;
added two nodes **enableLocalIssueCard** (whether to enable issuing smart cards locally) and **isLocalStorage** (whether to store face picture and fingerprint information in the device locally) to the node **RuleInfo**;
added two nodes **reqIssueNum** (number of persons to be issued with smart cards) and **IssuedNum** (number of persons that have been issued with smart cards) to the node **CaptureProgress**.
12. Added two query parameters **security** (the version No. of encryption scheme) and **iv** (the initialization vector) to the request URI ***/ISAPI/AccessControl/CardOperations/sectionEncryption?format=json*** (related API: ***NET_DVR_STDXMLConfig*** ).
13. Added two query parameters **security** (the version No. of encryption scheme) and **iv** (the initialization vector) to the request URI ***/ISAPI/AccessControl/CardOperations/verification?format=json*** (related API: ***NET_DVR_STDXMLConfig*** ).
14. Added two query parameters **security** (the version No. of encryption scheme) and **iv** (the initialization vector) to the request URI ***/ISAPI/AccessControl/CardOperations/controlBlock?format=json*** (related API: ***NET_DVR_STDXMLConfig*** ).

15. Added a URI of deleting data from the card: PUT ***/ISAPI/AccessControl/CardOperations/clearData?format=json*** (related API: ***NET_DVR_STDXMLConfig*** ).
16. Added a URI of setting custom card information: PUT ***/ISAPI/AccessControl/CardOperations/customData?format=json*** (related API: ***NET_DVR_STDXMLConfig*** ).
17. Added a URI of searching for custom card information: POST ***/ISAPI/AccessControl/CardOperations/customData/searchTask?format=json*** (related API: ***NET_DVR_STDXMLConfig*** ).
18. Extended card operation capability message ***JSON_CardOperationsCap*** (related URI: ***/ISAPI/AccessControl/CardOperations/capabilities?format=json*** ; related API: ***NET_DVR_STDXMLConfig*** ):
    added seven nodes: **Issue** (capability of sending a request for card issuing and getting the current card issuing status and real-time card issuing results), **localIssueCfg** (capability of configuring rule parameters for issuing smart cards), **ClearData** (capability of deleting data from the card), **CustomData** (capability of setting custom card information), **CustomDataSearchCond** (condition configuration capability of searching for custom card information), **CustomDataResult** (result capability of searching for custom card information), and **CardIssueStatus** (capability of getting the smart card issuing status).
19. Added 10 additional information logs to ***HCNetSDK Log Types*** :
    0x423-"MINOR_USB_LOGIN" (Log in via USB), 0x424-"MINOR_USB_LOGOUT" (Log out via USB), 0x425-"MINOR_ISAPI_HTTP_LOGIN" (Log in via ISAPI (HTTP)),
    0x426-"MINOR_ISAPI_HTTP_LOGOUT" (Log out via ISAPI (HTTP)),
    0x427-"MINOR_ISAPI_HTTPS_LOGIN" (Log in via ISAPI (HTTPS)),
    0x428-"MINOR_ISAPI_HTTPS_LOGOUT" (Log out via ISAPI (HTTPS)),
    0x429-"MINOR_ISUP_ONLINE" (ISUP online), 0x42a-"MINOR_ISUP_OFFLINE" (ISUP offline),
    0x42b-"MINOR_FP_ISSUE_REC" (Issuing record of card containing fingerprint information), and
    0x42c-"MINOR_FACE_ISSUE_REC" (Issuing record of card containing face picture information).

## Summary of Changes in Version 6.1.5.10_July, 2020

1. Extended configuration capability message ***XML_Cap_ChannelControllerCfg*** and parameter message ***XML_ChannelControllerCfg*** of the lane controller (related URIs: ***/ISAPI/AccessControl/ChannelControllerCfg/capabilities*** and ***/ISAPI/AccessControl/ChannelControllerCfg*** ):
   added a node <**runMode**> (running mode).
2. Added two URIs of configuring parameters of the keyfob control mode (related API: ***NET_DVR_STDXMLConfig*** ):
   Get configuration capability: GET ***/ISAPI/AccessControl/remoteCtrllerModeCfg/capabilities?format=json*** ;
   Get or set parameters: GET or PUT ***/ISAPI/AccessControl/remoteCtrllerModeCfg?format=json*** .
3. Extended functional capability message of access control ***XML_Cap_AccessControl*** (related URI: ***/ISAPI/AccessControl/capabilities*** ; related API: ***NET_DVR_STDXMLConfig*** ):
   added a node <**isSupportRemoteCtrllerModeCfg**> (whether it supports configuring parameters of the keyfob control mode).

## Summary of Changes in Version 6.1.4.10_Mar., 2020

Extended the configuration capability message ***JSON_ChannelControllerTypeCfgCap*** and the parameter message ***JSON_ChannelControllerTypeCfg*** of the lane controller's device type (related URIs: ***/ISAPI/AccessControl/channelControllerTypeCfg/capabilities?format=json*** and ***/ISAPI/AccessControl/channelControllerTypeCfg?format=json*** ):
added a device type "K3B501S-A" (DS-K3B501S series swing barrier) to the node **deviceModel**.

## Summary of Changes in Version 6.1.4.10_Mar., 2020

Extended the configuration capability message ***JSON_ChannelControllerTypeCfgCap*** and the parameter message ***JSON_ChannelControllerTypeCfg*** of the lane controller's device type (related URIs: ***/ISAPI/AccessControl/channelControllerTypeCfg/capabilities?format=json*** and ***/ISAPI/AccessControl/channelControllerTypeCfg?format=json*** ):
added a device type "K3G501" (DS-K3G501 series tripod turnstile) to the node **deviceModel**.

## Summary of Changes in Version 6.1.3.X_Sept., 2019

1. Extended access control capability message ***XML_Cap_AccessControl*** (related URL: ***/ISAPI/AccessControl/capabilities*** ; related API: ***NET_DVR_STDXMLConfig*** ):
added five nodes: <**isSupportCaptureIDInfo**> (whether it supports collecting ID card information), <**isSupportCaptureRule**> (whether it supports configuring online collection rules), <**isSupportCapturePresetParam**> (whether it supports configuring preset parameters of online collection), <**isSupportOfflineCapture**> (whether it supports offline collection), and <**isSupportCardOperations**> (whether it supports card operation).
2. Added the function of online collecting data, refer to ***Online Collect Data*** .
3. Added the function of offline collecting data, refer to ***Offline Collect Data*** .
4. Added three error codes to ***Device Network SDK Errors*** : 1927-"NET_ERR_CAPTURE_TIMEOUT" (collection timed out), 1928-"NET_ERR_LOW_SCORE" (low quality of collected data), and 1929-"NET_ERR_OFFLINE_CAPTURING" (the device is collecting data offline and cannot respond).
5. Added two sub status codes: 0x30006000-"captureTimeout" (data collection timed out) and 0x30006001-"lowScore" (low quality of collected data) to status code 3 (Device Error) in ***Response Codes of Text Protocol*** .
6. Added functions of operating cards, refer to ***Card Operation*** for details.
7. Added functions of configuring active infrared intrusion parameters (related API: ***NET_DVR_STDXMLConfig*** ):
Get configuration capability: GET ***/ISAPI/AccessControl/Configuration/IRCfg/capabilities?format=json***
Get or set parameters: GET or PUT ***/ISAPI/AccessControl/Configuration/IRCfg?format=json***
8. Added multiple log types, refer to ***HCNetSDK Log Types*** for details:
added six minor log types to the "MAJOR_EXCEPTION" log type: MINOR_AUXILIARY_BOARD_OFFLINE (0x43c), MINOR_AUXILIARY_BOARD_RESUME (0x43d), MINOR_IDCARD_SECURITY_MOUDLE_EXCEPTION (0x43e), MINOR_IDCARD_SECURITY_MOUDLE_RESUME (0x43f), MINOR_FP_PERIPHERAL_EXCEPTION (0x440), and MINOR_FP_PERIPHERAL_RESUME (0x441);

added three minor log types to the "MAJOR_OPERATION" log type: MINOR_OFFLINE_DATA_OUTPUT (0x423), MINOR_CREATE_SSH_LINK (0x42d), and MINOR_CLOSE_SSH_LINK (0x42e);
added 14 minor log types to the "MAJOR_INFORMATION" log type: MINOR_LIVE_DETECT_OPEN ( 0x400), MINOR_LIVE_DETECT_CLOSE (0x401), MINOR_CLEAR_DATA_COLLECTION (0x402), MINOR_DELETE_DATA_COLLECTION (0x403), MINOR_EXPORT_DATA_COLLECTION (0x404), MINOR_CARD_LEN_CONFIG (0x405), MINOR_DATA_BASE_INIT_FAILED (0x406), MINOR_DATA_BASE_PATCH_UPDATE (0x407), MINOR_PSAM_CARD_INSERT (0x408), MINOR_PSAM_CARD_REMOVE (0x409), MINOR_HARD_FAULT_REBOOT (0x40a), MINOR_PSAM_CARD_OCP (0x40b), MINOR_STACK_OVERFLOW (0x40c), and MINOR_PARM_CFG (0x40d).

## Summary of Changes in Version 6.1.0.20_Aug., 2019

1. Added URLs of configuring lane controller (related API: ***NET_DVR_STDXMLConfig*** ):
   Get configuration capability: GET ***/ISAPI/AccessControl/ChannelControllerCfg/capabilities*** ;
   Get or set parameters: GET or PUT ***/ISAPI/AccessControl/ChannelControllerCfg*** .
2. Added URLs of configuring device type of the lane controller (related API: ***NET_DVR_STDXMLConfig*** ):
   Get configuration capability: GET ***/ISAPI/AccessControl/channelControllerTypeCfg/capabilities?format=json*** ;
   Get or set parameters: GET or PUT ***/ISAPI/AccessControl/channelControllerTypeCfg?format=json*** .
3. Extended access control capability message ***XML_Cap_AccessControl*** (related URL: ***/ISAPI/AccessControl/capabilities*** ; related API: ***NET_DVR_STDXMLConfig*** ):
   added a node <**isSupportChannelControllerTypeCfg**> (whether it supports configuring device type of the lane controller).

## Summary of Changes in Version 6.1.0.10_July, 2019

1. Added the function of enabling or disabling NFC (Near-Field Communication) function (related API: ***NET_DVR_STDXMLConfig*** ):
   Get the configuration capability: GET ***/ISAPI/AccessControl/Configuration/NFCCfg/capabilities?format=json*** ;
   Get parameters: GET ***/ISAPI/AccessControl/Configuration/NFCCfg?format=json*** ;
   Set parameters: PUT ***/ISAPI/AccessControl/Configuration/NFCCfg?format=json*** .
2. Added the function of enabling or disabling RF (Radio Frequency) card recognition (related API: ***NET_DVR_STDXMLConfig*** ):
   Get the configuration capability: GET ***/ISAPI/AccessControl/Configuration/RFCardCfg/capabilities?format=json*** ;
   Get parameters: GET ***/ISAPI/AccessControl/Configuration/RFCardCfg?format=json*** ;
   Set parameters: PUT ***/ISAPI/AccessControl/Configuration/RFCardCfg?format=json*** .
3. Extended access control capability message ***XML_Cap_AccessControl*** (related URL: ***/ISAPI/AccessControl/capabilities*** ; related API: ***NET_DVR_STDXMLConfig*** ):

added two nodes: <**isSupportNFCCfg**> (whether the device supports enabling or disabling NFC function) and <**isSupportRFCardCfg**> (whether the device supports enabling or disabling RF card recognition).

4. Extended access control capability message ***XML_AcsAbility*** (related API: ***NET_DVR_GetDeviceAbility*** ; capability type: "0x801-ACS_ABILITY"):
   added seven event types to the sub node <**EventEntry**> (index: 3) of the node <**EventLinkage**> (event card linkage): "CPUCardEncryptVerifyFail" (verifying CPU card encryption failed), "NFCDisableVerifyFail" (disabling NFC verification failed), "EMCardRecognizeNotEnabled" (EM card recognition is disabled), "M1CardRecognizeNotEnabled" (M1 card recognition is disabled), "CPUCardRecognizeNotEnabled" (CPU card recognition is disabled), "IDCardRecognizeNotEnabled" (ID card recognition is disabled), and "CardSetSecretKeyFail" (importing key to the card failed).
5. Extended the access control event types in ***Access Control Event Types*** :
   added four operation event types to MAJOR_OPERATION: "MINOR_M1_CARD_ENCRYPT_VERIFY_OPEN" (M1 Card Encryption Verification Enabled), "MINOR_M1_CARD_ENCRYPT_VERIFY_CLOSE" (M1 Card Encryption Verification Disabled), "MINOR_NFC_FUNCTION_OPEN" (Opening Door with NFC Card Enabled), and "MINOR_NFC_FUNCTION_CLOSE" (Opening Door with NFC Card Disabled);
   added eight event types to MAJOR_EVENT: "MINOR_INFORMAL_M1_CARD_VERIFY_FAIL" (Authentication Failed: Invalid M1 Card), "MINOR_CPU_CARD_ENCRYPT_VERIFY_FAIL" (Verifying CPU Card Encryption Failed), "MINOR_NFC_DISABLE_VERIFY_FAIL" (Disabling NFC Verification Failed), "MINOR_EM_CARD_RECOGNIZE_NOT_ENABLED" (EM Card Recognition Disabled), "MINOR_M1_CARD_RECOGNIZE_NOT_ENABLED" (M1 Card Recognition Disabled), "MINOR_CPU_CARD_RECOGNIZE_NOT_ENABLED" (CPU Card Recognition Disabled), "MINOR_ID_CARD_RECOGNIZE_NOT_ENABLED" (ID Card Recognition Disabled), and "MINOR_CARD_SET_SECRET_KEY_FAIL" (Importing Key to Card Failed).
6. Extended the event linkage types in ***Event Linkage Types*** :
   added eight event linkage types of the authentication unit: "EVENT_ACS_INFORMAL_M1_CARD_VERIFY_FAIL" (Authentication Failed: Invalid M1 Card), "EVENT_ACS_CPU_CARD_ENCRYPT_VERIFY_FAIL" (Verifying CPU Card Encryption Failed), "EVENT_ACS_NFC_DISABLE_VERIFY_FAIL" (Disabling NFC Verification Failed), "EVENT_ACS_EM_CARD_RECOGNIZE_NOT_ENABLED" (EM Card Recognition Disabled), "EVENT_ACS_M1_CARD_RECOGNIZE_NOT_ENABLED" (M1 Card Recognition Disabled), "EVENT_ACS_CPU_CARD_RECOGNIZE_NOT_ENABLED" (CPU Card Recognition Disabled), "EVENT_ACS_ID_CARD_RECOGNIZE_NOT_ENABLED" (ID Card Recognition Disabled), and "EVENT_ACS_CARD_SET_SECRET_KEY_FAIL" (Importing Key to Card Failed).

## Summary of Changes in Version 6.0.2.5_03/2019

1. Extended configuration capability of intelligent identity detection terminal ***XML_Cap_IdentityTerminal*** (related API: ***NET_DVR_STDXMLConfig*** , related URL: ***/ISAPI/AccessControl/IdentityTerminal/capabilities*** ):

added a node <**ecoMode**> (ECO mode); added a value "none" to the node <**idCardReader**>.

2. Extended parameter structure of intelligent identity detection terminal ***XML_IdentityTerminal*** (related API: ***NET_DVR_STDXMLConfig*** , related URL: ***/ISAPI/AccessControl/IdentityTerminal*** ): added a node <**ecoMode**> (ECO mode); added a value "none" to the node <**idCardReader**>.
3. Extended reader configuration structure ***NET_DVR_CARD_READER_CFG_V50*** (related API: ***NET_DVR_GetDVRConfig*** and ***NET_DVR_SetDVRConfig*** , commands: 2505-NET_DVR_GET_CARD_READER_CFG_V50 and 2506-NET_DVR_SET_CARD_READER_CFG_V50): added a parameter **byFaceRecogizeEnable** (whether to enable face recognition) via 1 reserved byte.
4. Extended schedule parameter structure ***NET_DVR_SINGLE_PLAN_SEGMENT*** :
added two values to the parameter **byVerifyMode** (authentication mode), i.e., 25 (card or face), 26 (card or face or fingerprint).
5. Extended access control capability (related API: ***NET_DVR_GetDeviceAbility*** ; capability type: "0x801-ACS_ABILITY"):
added two values to the sub node <**verifyType**> (authentication mode) of node <**CardReaderVerifyTypePlan**>, i.e., "cardOrFace" (card or face) and "cardOrFaceOrFp" (card or face or fingerprint);
added three values to the sub node <**modifyParamType**> (parameter types supported to be edited) of node <**Card**> (card parameter capability), i.e., "roomNo", "simNo", and "floorNo".
added two sub nodes <**nightFaceMatchThresholdN**> (1:N face picture comparison threshold at night) and <**faceRecogizeEnable**> (whether to enable face recognition) to the node <**CardReaderCfg**>;
added two event names "PeopleAndIdCardComparePass" (face and ID card authenticated) and "PeopleAndIdCardCompareFail" (face and ID card authentication failed) to the sub node <**SubEventNameList**> of <node <**EventList**>
6. Extended the total capability of access control ***XML_Cap_AccessControl*** (related API: ***NET_DVR_STDXMLConfig*** , related URL: ***/ISAPI/AccessControl/capabilities*** ):
added a node <**FactoryReset**> (restore to factory settings by condition).

## Summary of Changes in Version 5.3.2.10_02/2018

New document.

# Chapter 2 Typical Applications

## 2.1 Data Collection

### 2.1.1 Online Collect Data

When the access control device is connected to the client software or platform via the network, you can collect data (including ID card information, card information, face data, and fingerprint) on the client software or platform remotely. The online collected data will be uploaded to the client software or platform in real time.

**Before You Start**

- Make sure you have called ***NET_DVR_Init*** to initialize the development environment.
- Make sure you have called ***NET_DVR_Login_V40*** to log in to the device.

### Steps

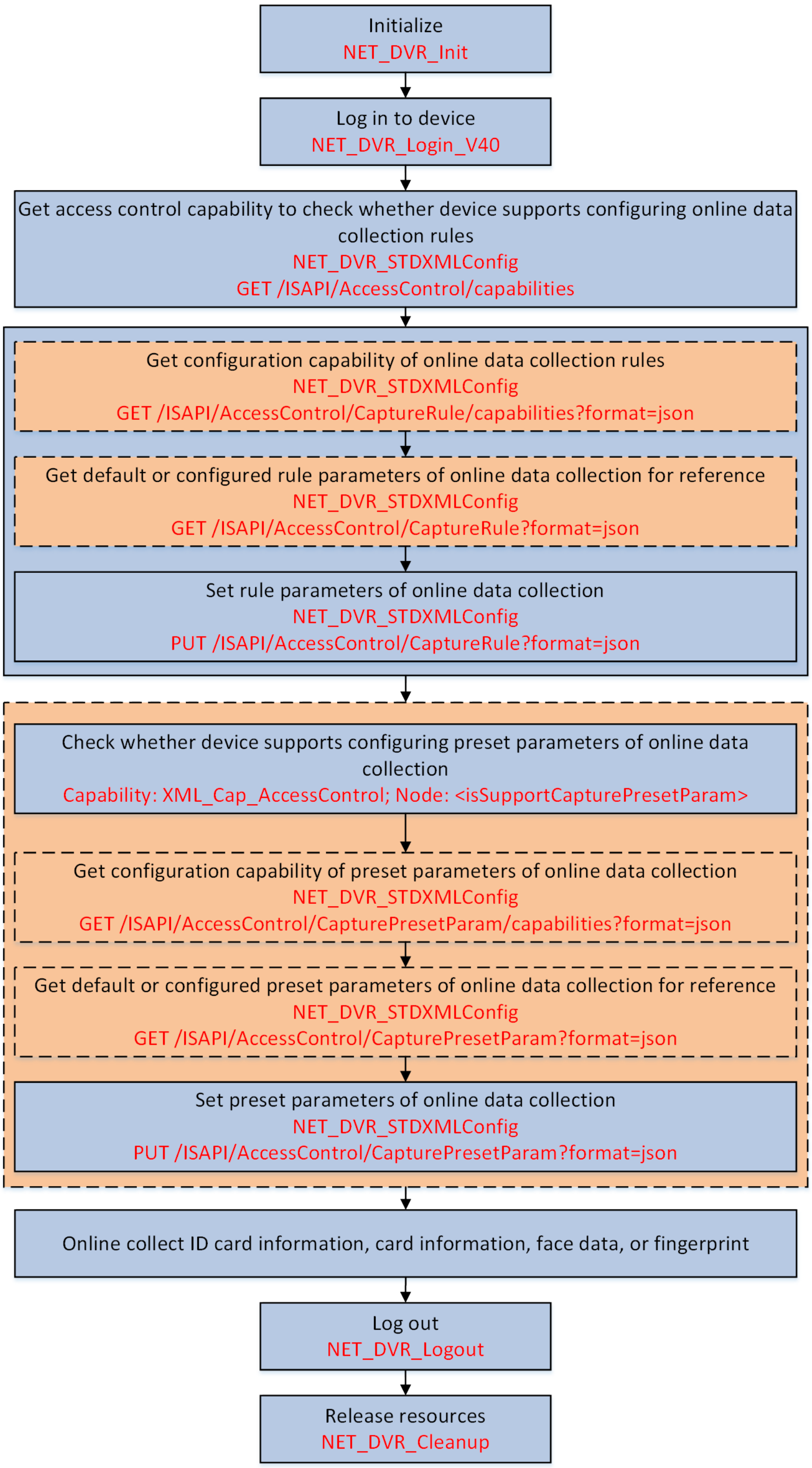
Initialize
NET_DVR_Init
Log in to device
NET_DVR_Login_V40
Get access control capability to check whether device supports configuring online data collection rules
NET_DVR_STDXMLConfig
GET /ISAPI/AccessControl/capabilities
Get configuration capability of online data collection rules
NET_DVR_STDXMLConfig
GET /ISAPI/AccessControl/CaptureRule/capabilities?format=json
Get default or configured rule parameters of online data collection for reference
NET_DVR_STDXMLConfig
GET /ISAPI/AccessControl/CaptureRule?format=json
Set rule parameters of online data collection
NET_DVR_STDXMLConfig
PUT /ISAPI/AccessControl/CaptureRule?format=json
Check whether device supports configuring preset parameters of online data collection
Capability: XML_Cap_AccessControl; Node: <isSupportCapturePresetParam>
Get configuration capability of preset parameters of online data collection
NET_DVR_STDXMLConfig
GET /ISAPI/AccessControl/CapturePresetParam/capabilities?format=json
Get default or configured preset parameters of online data collection for reference
NET_DVR_STDXMLConfig
GET /ISAPI/AccessControl/CapturePresetParam?format=json
Set preset parameters of online data collection
NET_DVR_STDXMLConfig
PUT /ISAPI/AccessControl/CapturePresetParam?format=json
Online collect ID card information, card information, face data, or fingerprint
Log out
NET_DVR_Logout
Release resources
NET_DVR_Cleanup

**Figure 2-1 Programming Flow of Online Collecting Data**

1. Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/capabilities*** for getting the access control capability to check whether the device supports configuring online data collection rules.

   The access control capability is returned in ***XML_Cap_AccessControl*** by **lpOutputParam**.

   If the device supports, the node <**isSupportCaptureRule**> is returned in the capability message and its value is "true", and then you can perform the following steps.

   Otherwise, rule configuration of online data collection is not supported, please end this task.
2. Configure online data collection rules.
   1) **Optional:** Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/CaptureRule/capabilities?format=json*** for getting the configuration capability of online data collection rules.

      The capability is returned in the message ***JSON_CaptureRuleCap*** by **lpOutputParam**.
   2) **Optional:** Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/CaptureRule?format=json*** for getting default or configured rule parameters of online data collection for reference.

      The rule parameters are returned in the message ***JSON_CaptureRule*** by **lpOutputParam**.
   3) Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: PUT ***/ISAPI/AccessControl/CaptureRule?format=json*** and set **lpInputParam** to ***JSON_CaptureRule*** for setting rule parameters of online data collection.
3. **Optional:** Configure preset parameters of online data collection.
   1) Check the access control capability ***XML_Cap_AccessControl*** to know whether the device supports configuring preset parameters of online data collection.

      If the device supports, the node <**isSupportCapturePresetParam**> is in the capability message and its value is "true", and then you can continue to set preset parameters.

      Otherwise, preset configuration of online data collection is not supported.
   2) **Optional:** Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/CapturePresetParam/capabilities?format=json*** for getting the configuration capability of preset parameters of online data collection.

      The configuration capability is returned in the message ***JSON_CapturePresetCap*** by **lpOutputParam**.
   3) **Optional:** Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/CapturePresetParam?format=json*** for getting default or configured preset parameters of online data collection for reference.

      The preset parameters are returned in the message ***JSON_CapturePreset*** by **lpOutputParam**.
   4) Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: PUT ***/ISAPI/AccessControl/CapturePresetParam?format=json*** and set **lpInputParam** to the message ***JSON_CapturePreset*** for setting preset parameters of online data collection.

**Note**

The preset parameters are used to display custom information on the device UI during data collection. Currently, it only supports displaying the name of the person whose data is being collected. The preset parameters should be configured again for each collection.

4. Perform the following operation(s) to collect ID card information, card information, face data, or fingerprint online.

**Collect ID Card Information**

a. Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/capabilities*** for getting the access control capability to check whether the device supports online collecting ID card information.
The capability is returned in the message ***XML_Cap_AccessControl*** by **lpOutputParam**. If the device supports, the node <**isSupportCaptureCardInfo**> will be returned and its value is "true", and then you can perform the following steps.
Otherwise, online collecting ID card information is not supported by the device, please end this task.

b. Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/CaptureIDInfo/capabilities?format=json*** for getting the capability of online collecting ID card information.
The capability is returned in the message ***JSON_IdentityInfoCap*** by **lpOutputParam**.

c. Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: POST ***/ISAPI/AccessControl/CaptureIDInfo?format=json*** and set **lpInputParam** to the message ***JSON_IdentityInfoCond*** for online collecting ID card information.
The online collected ID card information is returned in the message ***JSON_IdentityInfo*** by **lpOutputParam**.

**Collect Card Information**

a. Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/capabilities*** for getting the access control capability to check whether the device supports online collecting card information.
The capability is returned in the message ***XML_Cap_AccessControl*** by **lpOutputParam**. If the device supports, the node <**isSupportCaptureIDInfo**> will be returned and its value is "true", and then you can perform the following steps.
Otherwise, online collecting card information is not supported by the device, please end this task.

b. Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/CaptureCardInfo/capabilities?format=json*** for getting the capability of online collecting card information.

| | |
|---|---|
| | The capability is returned in the message ***JSON_CardInfoCap*** by **lpOutputParam**.<br>c. Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/CaptureCardInfo?format=json*** for online collecting the card information.<br>The online collected card information is returned in the message ***JSON_CardInfo_Collection*** by **lpOutputParam**. |
| **Collect Face Data** | a. Call ***NET_DVR_GetDeviceAbility*** , set **dwAbilityType** to "ACS_ABILITY", and set **pInBuf** to ***XML_Desc_AcsAbility*** for getting the access control capability to know the supported parameters of online collecting face data.<br>The capability is returned in the message ***XML_AcsAbility*** by **pOutBuf**. The related node is <**CaptureFace**>.<br>b. Call ***NET_DVR_StartRemoteConfig*** with "NET_DVR_CAPTURE_FACE_INFO" (command No.: 2510) and set **lpInBuffer** to the structure ***NET_DVR_CAPTURE_FACE_COND*** for setting up persistent connection and set callback function ( ***fRemoteConfigCallback*** ) for online collecting face data.<br>The online collected face data is returned in the structure ***NET_DVR_CAPTURE_FACE_CFG*** by **lpBuffer** of the callback function.<br>c. Call ***NET_DVR_StopRemoteConfig*** to disconnect the persistent connection and finishing online collecting face data. |
| **Collect Fingerprint** | Refer to ***Collect Fingerprint*** |

**What to do next**

Call ***NET_DVR_Logout*** and ***NET_DVR_Cleanup*** to log out of the device and release the resources.

### 2.1.2 Offline Collect Data

When the access control device is not connected to the client software or platform via the network, you can collect data (including ID card information, card information, face data, and fingerprint) locally on the stand-alone device by importing description of the information that needs to be collected. The offline collected data will be stored on the device and can also be downloaded, exported, or deleted from the device.

**Before You Start**

- Make sure you have called ***NET_DVR_Init*** to initialize the development environment.
- Make sure you have called ***NET_DVR_Login_V40*** to log in to the device.

## Steps

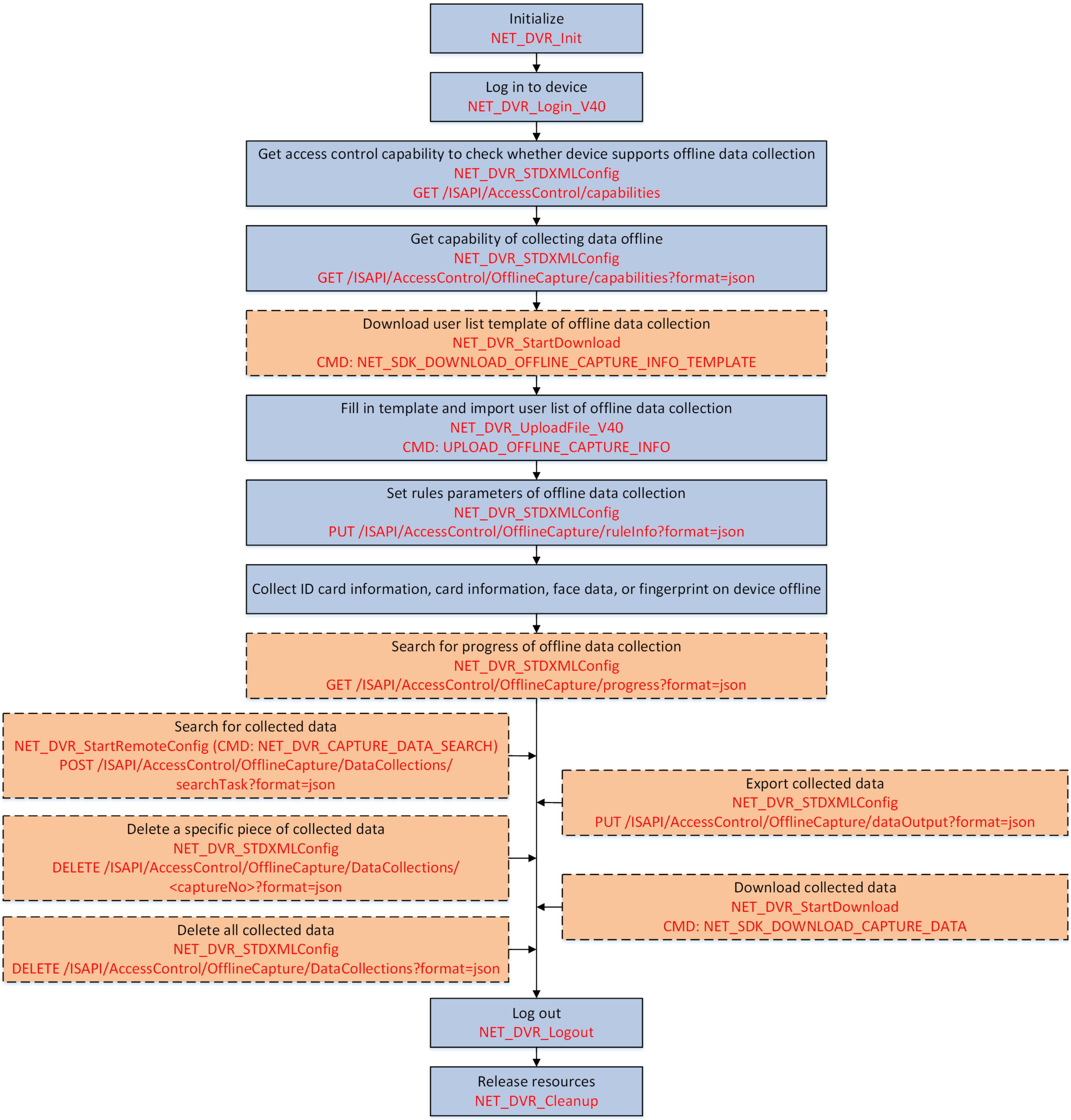
Initialize
NET_DVR_Init
Log in to device
NET_DVR_Login_V40
Get access control capability to check whether device supports offline data collection
NET_DVR_STDXMLConfig
GET /ISAPI/AccessControl/capabilities
Get capability of collecting data offline
NET_DVR_STDXMLConfig
GET /ISAPI/AccessControl/OfflineCapture/capabilities?format=json
Download user list template of offline data collection
NET_DVR_StartDownload
CMD: NET_SDK_DOWNLOAD_OFFLINE_CAPTURE_INFO_TEMPLATE
Fill in template and import user list of offline data collection
NET_DVR_UploadFile_V40
CMD: UPLOAD_OFFLINE_CAPTURE_INFO
Set rules parameters of offline data collection
NET_DVR_STDXMLConfig
PUT /ISAPI/AccessControl/OfflineCapture/ruleInfo?format=json
Collect ID card information, card information, face data, or fingerprint on device offline
Search for progress of offline data collection
NET_DVR_STDXMLConfig
GET /ISAPI/AccessControl/OfflineCapture/progress?format=json
Search for collected data
NET_DVR_StartRemoteConfig (CMD: NET_DVR_CAPTURE_DATA_SEARCH)
POST /ISAPI/AccessControl/OfflineCapture/DataCollections/searchTask?format=json
Export collected data
NET_DVR_STDXMLConfig
PUT /ISAPI/AccessControl/OfflineCapture/dataOutput?format=json
Delete a specific piece of collected data
NET_DVR_STDXMLConfig
DELETE /ISAPI/AccessControl/OfflineCapture/DataCollections/<captureNo>?format=json
Download collected data
NET_DVR_StartDownload
CMD: NET_SDK_DOWNLOAD_CAPTURE_DATA
Delete all collected data
NET_DVR_STDXMLConfig
DELETE /ISAPI/AccessControl/OfflineCapture/DataCollections?format=json
Log out
NET_DVR_Logout
Release resources
NET_DVR_Cleanup

**Figure 2-2 Programming Flow of Offline Collecting Data**

1. Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/capabilities*** for getting the access control capability to check whether the device supports offline data collection.

   The capability is returned in the message ***XML_Cap_AccessControl*** by **lpOutBuffer** of **lpOutputParam**.

   If this function is supported, the node <**isSupportOfflineCapture**> will be returned and its value is "true". Otherwise, please end this task.
2. Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/OfflineCapture/capabilities?format=json*** for getting the capability of collecting data offline to know the supported parameters.

   The capability is returned in the message ***JSON_OfflineCaptureCap*** **lpOutBuffer** of **lpOutputParam**.
3. **Optional:** Download the user list template of offline data collection.
   1) Call ***NET_DVR_StartDownload*** and set the input parameter **dwDownloadType** to "NET_SDK_DOWNLOAD_OFFLINE_CAPTURE_INFO_TEMPLATE" (macro definition value: 40) to start downloading.
   2) Call ***NET_DVR_GetDownloadState*** to get the downloading progress.
   3) Call ***NET_DVR_StopDownload*** to stop downloading.
4. Import the user list of offline data collection filled in the template.
   1) Call ***NET_DVR_UploadFile_V40*** , set **dwUploadType** to "UPLOAD_OFFLINE_CAPTURE_INFO" (macro definition value: 56), and set **lpInBuffer** to the structure ***NET_DVR_DOOR_FILE_UPLOAD_PARAM*** for start uploading the file.
   2) Call ***NET_DVR_GetUploadState*** to get the file uploading progress.
   3) Call ***NET_DVR_UploadClose*** to stop uploading.

   **Note**

   If importing failed, you can call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/OfflineCapture/uploadFailedDetails?format=json*** for getting the details of failing to upload the user list of offline data collection.
   The uploading failure details are returned in the message ***JSON_UploadFailedDetails*** by **lpOutputParam**.
5. Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: PUT ***/ISAPI/AccessControl/OfflineCapture/ruleInfo?format=json*** and set **lpInBuffer** of **lpInputParam** to the message ***JSON_RuleInfo*** for setting rule parameters of offline data collection.

   **Note**

   Before setting rule parameters of offline data collection, you'd better call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/OfflineCapture/ruleInfo?format=json*** for getting the existing or configured parameters for reference. The parameters are returned in the message ***JSON_RuleInfo*** by **lpOutBuffer** of **lpOutputParam**.
6. Collect ID card information, card information, face data, or fingerprint on the stand-alone device offline.

7. **Optional:** Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/OfflineCapture/progress?format=json*** for getting the progress of offline data collection.

   The collection progress is returned in the message ***JSON_CaptureProgress*** by **lpOutBuffer** of **lpOutputParam**.
8. **Optional:** Perform the following operation(s) after collecting data offline.

|  |  |
|---|---|
| **Export Collected Data** | Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: PUT ***/ISAPI/AccessControl/OfflineCapture/dataOutput?format=json*** and set **lpInBuffer** of **lpInputParam** to the message ***JSON_DataOutputCfg*** .<br>**Note**<br>During exporting, you can call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/OfflineCapture/dataOutput/progress?format=json*** for getting the progress of exporting the offline collected data. |
| **Download Collected Data** | a. Call ***NET_DVR_StartDownload*** and set the **dwDownloadType** to "NET_SDK_DOWNLOAD_CAPTURE_DATA" (macro definition value: 41) to start downloading.<br>b. Call ***NET_DVR_GetDownloadState*** to get the downloading status.<br>c. Call ***NET_DVR_StopDownload*** to stop downloading. |
| **Search for Collected Data** | a. Call ***NET_DVR_StartRemoteConfig*** with "NET_DVR_CAPTURE_DATA_SEARCH" (command No.: 2554) and set the **lpInBuffer** to the request URI POST ***/ISAPI/AccessControl/OfflineCapture/DataCollections/searchTask?format=json*** for setting up persistent connection and set callback function ( ***fRemoteConfigCallback*** ) for searching for the collected data.<br>b. Call ***NET_DVR_SendRemoteConfig*** to send the search condition message ***JSON_SearchTaskCond*** via the persistent connection.<br>The collected data is returned in the structure ***NET_DVR_JSON_DATA_CFG*** by the output buffer (**lpBuffer**) of the callback function.<br>**Note**<br>- The type of data to be sent (**dwDataType**) should be set to "ENUM_SEND_JSON_DATA".<br>- After a search condition message ***JSON_SearchTaskCond*** is applied by calling ***NET_DVR_SendRemoteConfig*** , the next piece of data can be searched only when ***NET_DVR_JSON_DATA_CFG*** is returned by the callback function ***fRemoteConfigCallback*** .<br>c. Call ***NET_DVR_StopRemoteConfig*** to disconnect the persistent connection and finishing searching. |

| | |
|---|---|
| **Delete A Specific Piece of Collected Data** | Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: DELETE ***/ISAPI/AccessControl/OfflineCapture/DataCollections/<captureNo>?format=json*** . |
| **Delete All Collected Data** | Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: DELETE ***/ISAPI/AccessControl/OfflineCapture/DataCollections?format=json*** . |

**What to do next**

Call ***NET_DVR_Logout*** and ***NET_DVR_Cleanup*** to log out of the device and release the resources.

### 2.1.3 Collect Fingerprint

The fingerprint information for further management and applying should be collected by fingerprint recorder first. The following contents about the process and parameter settings of fingerprint collection.

1. Call ***NET_DVR_GetDeviceAbility*** , set **dwAbilityType** to "ACS_ABILITY", and set **pInBuf** to ***XML_Desc_AcsAbility*** for getting the access control capability to know the supported parameters of collecting fingerprint.
   The capability is returned in the message ***XML_AcsAbility*** by **pOutBuf**. The related node is <**CaptureFingerPrint**>.
2. Call ***NET_DVR_StartRemoteConfig*** with "NET_DVR_CAPTURE_FINGERPRINT_INFO" (command No.: 2504) and set the input buffer to the structure ***NET_DVR_CAPTURE_FINGERPRINT_COND*** for setting up persistent connection and setting status callback function ( ***fRemoteConfigCallback*** ).
   The collected fingerprint information are returned in the structure ***NET_DVR_CAPTURE_FINGERPRINT_CFG*** by the output buffer (**lpBuffer**) of the status callback function.
3. Call ***NET_DVR_StopRemoteConfig*** to disconnect the persistent connection and finish the fingerprint collection.

## 2.2 Manage Card Information

A card is a basic unit, which can link with multiple face pictures and fingerprints, for access control in this manual. So, before starting any other operations, you should add cards and apply the card information (e.g., card number, card type, group, permissions, and so on) to access control devices.

**Before You Start**

- Make sure you have called ***NET_DVR_Init*** to initialize the development resources.
- Make sure you have called ***NET_DVR_Login_V40*** to log in to device.

**Steps**

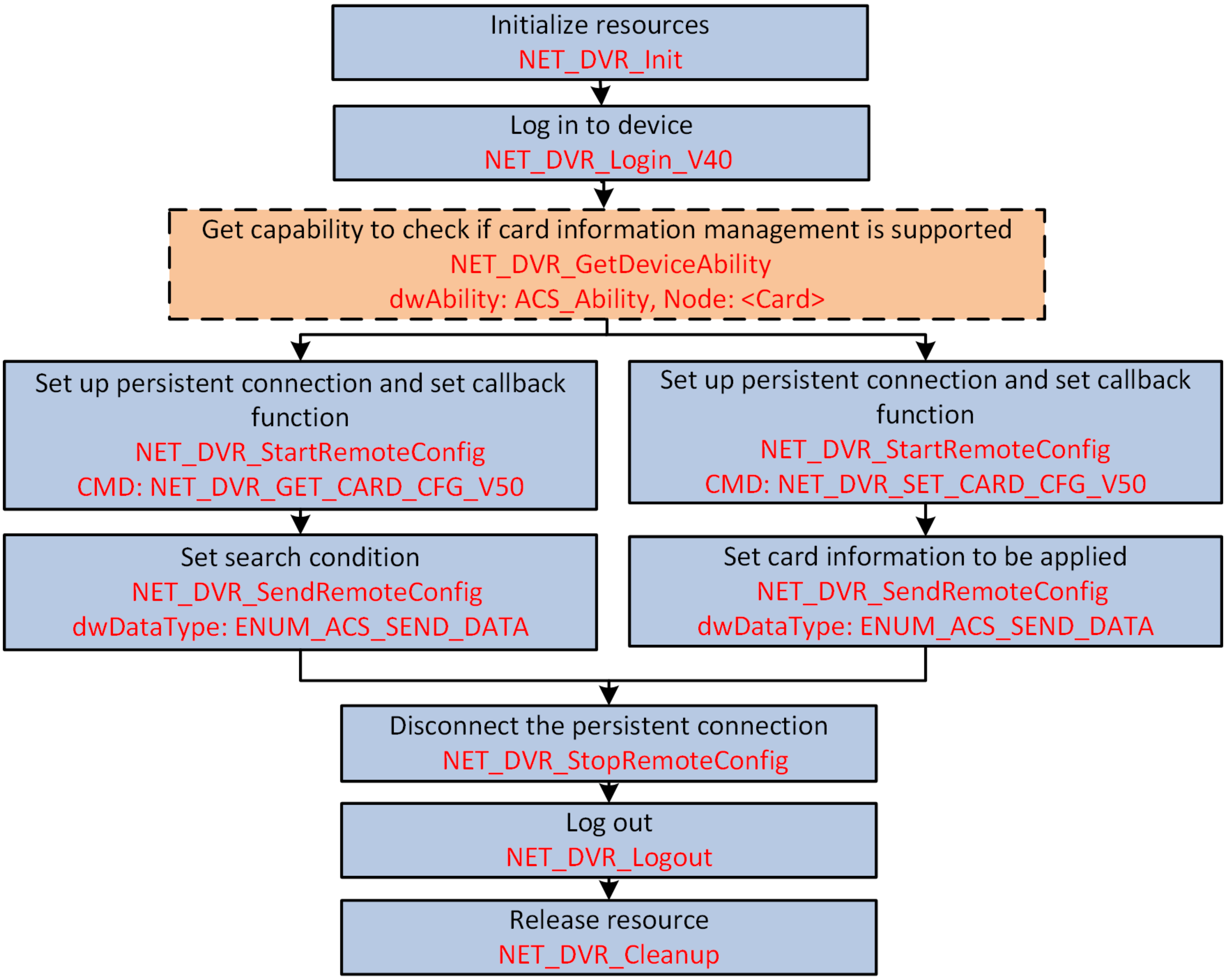
Initialize resources
NET_DVR_Init
Log in to device
NET_DVR_Login_V40
Get capability to check if card information management is supported
NET_DVR_GetDeviceAbility
dwAbility: ACS_Ability, Node: <Card>
Set up persistent connection and set callback function
NET_DVR_StartRemoteConfig
CMD: NET_DVR_GET_CARD_CFG_V50
Set up persistent connection and set callback function
NET_DVR_StartRemoteConfig
CMD: NET_DVR_SET_CARD_CFG_V50
Set search condition
NET_DVR_SendRemoteConfig
dwDataType: ENUM_ACS_SEND_DATA
Set card information to be applied
NET_DVR_SendRemoteConfig
dwDataType: ENUM_ACS_SEND_DATA
Disconnect the persistent connection
NET_DVR_StopRemoteConfig
Log out
NET_DVR_Logout
Release resource
NET_DVR_Cleanup

**Figure 2-3 Programming Flow of Managing Card Information**

1. Call ***NET_DVR_GetDeviceAbility*** , specify the capability type (**dwAbilityType**) to "ACS_ABILITY" (macro definition value: 0x801), and set the condition buffer (**pCondBuffer**) to the message ***XML_Desc_AcsAbility*** for getting access control capability to check if managing card information is supported by device.

   The access control capability is returned in the message ***XML_AcsAbility*** by the output buffer (**pOutBuffer**).

   If the node <**Card**> is returned in the message, it indicates that managing card information is supported by device, and you can perform the following steps.

   Otherwise, managing card information is not supported, please end this task.

2. Perform the following operation(s) to get or apply card information.

   **Note**

   Before applying card information, you can configure the access permission schedule template, refer to ***Configure Access Permission Control Schedule*** for details, and link the template to the card. Otherwise, the default template will be linked.

   **Get Card Information**

   a. Call ***NET_DVR_StartRemoteConfig*** with ***NET_DVR_GET_CARD_CFG_V50*** (command No.: 2178) and set the input buffer (**lpInBuffer**) to ***NET_DVR_CARD_CFG_COND*** for setting up persistent connection and setting callback function ( ***fRemoteConfigCallback*** ).

      **Note**

      The card information is returned in the structure ***NET_DVR_CARD_CFG_V50*** by the output buffer (**lpBuffer**) of callback function.

   b. Call ***NET_DVR_SendRemoteConfig*** , specify data type (**dwDataType**) to "ENUM_ACS_SEND_DATA", and set sending buffer (**pSendBuf**) to ***NET_DVR_CARD_CFG_SEND_DATA*** for getting a specific card information.

   **Apply Card Information**

   a. Call ***NET_DVR_StartRemoteConfig*** with ***NET_DVR_SET_CARD_CFG_V50*** (command No.: 2179) and set input buffer (**lpInBuffer**) to ***NET_DVR_CARD_CFG_COND*** for setting up persistent connection and setting callback function ( ***fRemoteConfigCallback*** ).

   b. Call ***NET_DVR_SendRemoteConfig*** , specify data type (**dwDataType**) to "ENUM_ACS_SEND_DATA", and set sending buffer (**pSendBuf**) to ***NET_DVR_CARD_CFG_V50*** for applying the card information.

      **Note**

      The applying status is returned by the output buffer (**lpBuffer**) of callback function.

3. Call ***NET_DVR_StopRemoteConfig*** to disconnect the persistent connection.

**Example**

Sample Code for Getting and Applying Card Information

```
#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)
{
    //As the operations with long time comsumption are not allowed in the callback function,
        //do not call the API of HCNetSDK.DLL in the callback function.
    //The following code is for reference only, actually, processing data in the callback function is not suggested.
    //for example, process in the message response function as PostMessage

    if (dwType == NET_SDK_CALLBACK_TYPE_DATA)//Data information
    {
        LPNET_DVR_CARD_CFG_V50 lpCardCfg = new NET_DVR_CARD_CFG_V50;
        memcpy(lpCardCfg, lpBuffer, sizeof(*lpCardCfg)); //Copy the card information of callback function
        //PostMessage(WM_MSG_ADD_CARDCFG_TOLIST, (WPARAM)lpCardCfg,0);

        char *pCardNo;
        BYTE byCardType;
        pCardNo = (char *)lpCardCfg->byCardNo;
        byCardType = lpCardCfg->byCardType;
        //Other processes...
    }
    else if (dwType == NET_SDK_CALLBACK_TYPE_STATUS)//Status value
    {
        DWORD dwStatus = *(DWORD*)lpBuffer;
        if (dwStatus == NET_SDK_CALLBACK_STATUS_SUCCESS)
        {
            bGetCardCfgFinish = TRUE;//Getting card information complated.
            //PostMessage(WM_MSG_GETCARD_FINISH,0,0);
            //Other processes...
        }
        else if ( dwStatus == NET_SDK_CALLBACK_STATUS_FAILED )
        {
            char szCardNumber[ACS_CARD_NO_LEN + 1] = "\0";
            DWORD dwErrCode = *(DWORD*)((char *)lpBuffer + 4); //Error code
            strncpy(szCardNumber,(char*)(lpBuffer) + 8,ACS_CARD_NO_LEN);//Card No.
            printf("GetCard STATUS_FAILED, Error code %d, Card Number %s\n", dwErrCode,  szCardNumber);
            //Other processes...
        }
    }
}

void CALLBACK g_fSetGatewayCardCallback(DWORD dwType, void* lpBuffer, DWORD dwBufLen, void* pUserData)
{
    if (dwType != NET_SDK_CALLBACK_TYPE_STATUS)//Only status will be returned when applying card
    {
        return;
    }
    DWORD dwStatus = *(DWORD*)lpBuffer;//The first 4-byte is status value

    if (dwStatus == NET_SDK_CALLBACK_STATUS_PROCESSING)//Sending
    {
        char szCardNumber[ACS_CARD_NO_LEN + 1] = "\0";
        strncpy(szCardNumber,(char*)(lpBuffer) + 4,ACS_CARD_NO_LEN);
        printf("SetCard PROCESSING,CardNo: %s\n", szCardNumber);
        //Other processes...
    }
    else if (dwStatus == NET_SDK_CALLBACK_STATUS_FAILED)//Sending failed.
    {
        char szCardNumber[ACS_CARD_NO_LEN + 1] = "\0";
        DWORD dwErrCode = *((DWORD*)lpBuffer + 1);//Error code
        strncpy(szCardNumber,(char*)(lpBuffer) + 8,ACS_CARD_NO_LEN);//Card No.
        printf("SetCard Err:%d,CardNo:%s\n", dwErrCode, szCardNumber);
        //Other processes...
    }
    //The following contents should perfrom stopping remote configuration for twice.
    else if (dwStatus == NET_SDK_CALLBACK_STATUS_SUCCESS)//Sent
    {
        printf("SetCard SUCCESS!");
        bSetCardCfgFinish = TRUE;
        //Other processes...
        //PostMessage(WM_MSG_SETCARD_FINISH,0,0);
    }
    else if (dwStatus == NET_SDK_CALLBACK_STATUS_EXCEPTION)//Exception
    {
        bSetCardCfgFinish = TRUE;
        //Other processes...
        //PostMessage(WM_MSG_SETCARD_FINISH,0,0);
    }
}

void main()
{
    //---------------------------------------
    //Initialize
    NET_DVR_Init();

    //Set connection timeout and reconnection function
    NET_DVR_SetConnectTime(2000, 1);
    NET_DVR_SetReconnect(10000, true);

    //---------------------------------------
    //Log in to device
    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;
    }

    //Get card information--------------------------------
    NET_DVR_CARD_CFG_COND struCond = {0};
       struCond.dwSize  = sizeof(struCond);
       struCond.dwCardNum = 1;//Get the number of cards
       struCond.byCheckCardNo = 1;

    NET_DVR_CARD_CFG_SEND_DATA struSendData = {0};
       struSendData.dwSize = sizeof(struSendData);
    m_csCardNo = "12";//Card No.
    strncpy((char *)struSendData.byCardNo, (LPCTSTR)m_csCardNo, ACS_CARD_NO_LEN);

    //Start remote configuration
    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;
    }
    //Send search conditions remotely
    if (! NET_DVR_SendRemoteConfig(m_lGetCardCfgHandle, ENUM_ACS_SEND_DATA, (char *)(&struSendData),
sizeof(struSendData)) )
    {
        printf("NET_DVR_SendRemoteConfig fail, error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_StopRemoteConfig(m_lGetCardCfgHandle);
        NET_DVR_Logout(lUserID);
        NET_DVR_Cleanup();
        return;
    }
    //Stop remote configuration
    Sleep(1000);
    if (bGetCardCfgFinish)//Stop remote connection if getting card information completed.
    {
        NET_DVR_StopRemoteConfig(m_lGetCardCfgHandle);
    }

    //Apply card information--------------------------------
    NET_DVR_CARD_CFG_COND struCond_set = {0};
       struCond_set.dwSize  = sizeof(struCond_set);
       struCond_set.dwCardNum = 1;//Number of card to apply
       struCond_set.byCheckCardNo = 1;
    struCond_set.wLocalControllerID = 0;//Apply offline card information to distributed access controller No.
                       //0-access controller

    //Start remote configuration
    m_lSetCardCfgHandle = NET_DVR_StartRemoteConfig(lUserID,NET_DVR_SET_CARD_CFG_V50,&struCond_set,\
        sizeof(struCond_set),g_fSetGatewayCardCallback,NULL);
    if (m_lSetCardCfgHandle==-1)
    {
        printf("NET_DVR_StartRemoteConfig fail, error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(lUserID);
        NET_DVR_Cleanup();
        return;
    }

    //Set card information
    LPNET_DVR_CARD_CFG_V50 lpCardCfg =  new NET_DVR_CARD_CFG_V50;
    lpCardCfg->dwSize = sizeof(NET_DVR_CARD_CFG_V50);
    lpCardCfg->dwModifyParamType = 0x000003FF;//Card information to be edited, represented by bit,
                                      //each bit refers to one kind of parameter, 0-Not Edit, 1-Edit

    m_csCardNo = "12";//Card No.
    strncpy((char *)lpCardCfg->byCardNo, (LPCTSTR)m_csCardNo, ACS_CARD_NO_LEN);
    lpCardCfg->byCardValid = 1;//Card is valid or not: 0-invalid card, 1-valid card
    lpCardCfg->byCardType = 1;//Card type: 1-Normal card (default)
    lpCardCfg->byLeaderCard = 0;//First card or not: 1-Yes, 0-No
    lpCardCfg->byDoorRight[0] = 1;//byDoorRight[0] to byDoorRight[255]: door 1 to door 256, 1-with permission, 0-no
permission
    lpCardCfg->byDoorRight[1] = 1;//Door 1 and door 2 are with permission.

    lpCardCfg->byBelongGroup[0] = 1;//byBelongGroup[0] to byBelongGroup[127]: array 1 to array 128, 1-in range, 0-
not in range
    m_csCardPassword = "12345678";//Card password
    strncpy((char *)lpCardCfg->byCardPassword, (LPCTSTR)m_csCardPassword, CARD_PASSWORD_LEN);
    //Configure the access permission schedule template first.
    lpCardCfg->wCardRightPlan[0][0]=1;//This card is configured with access permission schedule template 1 and 2 for
door 1
    lpCardCfg->wCardRightPlan[0][1]=2;
    lpCardCfg->wCardRightPlan[1][0]=3;//This card is configured with access permission schedule template 3 and 4 for
door 2
    lpCardCfg->wCardRightPlan[1][1]=4;

    lpCardCfg->dwMaxSwipeTime = 0;//Maximum card swiping times, 0-no limit
    lpCardCfg->dwSwipeTime = 0;//Swiped times

    lpCardCfg->struValid.byEnable = 1;//Validity duration
    lpCardCfg->struValid.struBeginTime.wYear=2017;//Start time: 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;//End time: 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;
```

```
   //Send card information remotely
   if (!NET_DVR_SendRemoteConfig(m_lSetCardCfgHandle,ENUM_ACS_SEND_DATA, (char
*)lpCardCfg ,sizeof(*lpCardCfg)))
   {
      printf("NET_DVR_SendRemoteConfig fail, error:%d.\n", NET_DVR_GetLastError());
      NET_DVR_StopRemoteConfig(m_lSetCardCfgHandle);
      NET_DVR_Logout(lUserID);
      NET_DVR_Cleanup();
      return;
   }
   //Stop remote configuration
   Sleep(1000);
   if (bSetCardCfgFinish)//Stop remote configuration when applied or callback exception.
   {
      NET_DVR_StopRemoteConfig(m_lSetCardCfgHandle);
   }

   //---------------------------------------
   //Exit
   Sleep(5000);

   //Log out
   NET_DVR_Logout(lUserID);
   //Release SDK resources
   NET_DVR_Cleanup();
   return;
}
```

**What to do next**

Call ***NET_DVR_Logout*** and ***NET_DVR_Cleanup*** to log out and release the resource.

## 2.2.1 Card Operation

### Get Card Operation Capability

Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/CardOperations/capabilities?format=json*** .

The capability is returned in the message ***JSON_CardOperationsCap*** by **lpOutputParam**.

### Encrypt Specific Sections (M1 Card)

Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: PUT ***/ISAPI/AccessControl/CardOperations/sectionEncryption?format=json*** and set **lpInputParam** to ***JSON_SectionEncryption*** .

### Verify Section Password (M1 Card)

Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: PUT ***/ISAPI/AccessControl/CardOperations/verification?format=json*** and set **lpInputParam** to ***JSON_Verification*** .

### Change Control Block of Section (M1 Card)

Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: PUT ***/ISAPI/AccessControl/CardOperations/controlBlock?format=json*** and set **lpInputParam** to ***JSON_ControlBlock*** .

### Read or Write Block Data (M1 Card)

#### Read Block Data

Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/CardOperations/dataBlock/<address>?format=json*** .

The block data is returned in the message ***JSON_DataBlock*** by **lpOutputParam**.

#### Write Block Data

Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: PUT ***/ISAPI/AccessControl/CardOperations/dataBlock/<address>?format=json*** and set **lpInputParam** to ***JSON_DataBlock*** .

### Operate Data Block (M1 Card)

Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: PUT ***/ISAPI/AccessControl/CardOperations/dataBlock/control?format=json*** and set **lpInputParam** to ***JSON_DataBlockCtrl***

### Set Operation Protocol Type of Card

Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: PUT ***/ISAPI/AccessControl/CardOperations/protocol?format=json*** and set **lpInputParam** to ***JSON_CardProto*** .

### Set CPU Card Parameters

Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: PUT ***/ISAPI/AccessControl/CardOperations/cardParam?format=json*** and set **lpInputParam** to ***JSON_CardParam*** .

### Reset CPU Card

Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/CardOperations/reset?format=json*** .

And the resetting result is returned in the message ***JSON_CardResetResponse*** by **lpOutputParam**.

### Pass Through Data Package of CPU Card

Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: PUT ***/ISAPI/AccessControl/CardOperations/dataTrans?format=json*** and set **lpInputParam** to ***JSON_DataTrans*** .

### Encrypt CPU Card

Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: PUT ***/ISAPI/AccessControl/CardOperations/encryption?format=json*** and set **lpInputParam** to ***JSON_CardEncryption*** .

### Delete Data from Card

Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: PUT ***/ISAPI/AccessControl/CardOperations/clearData?format=json*** and set **lpInputParam** to ***JSON_ClearData*** .

### Set Custom Card Information

Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: PUT ***/ISAPI/AccessControl/CardOperations/customData?format=json*** and set **lpInputParam** to ***JSON_CustomData*** .

**Search for Custom Card Information**

Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: POST ***/ISAPI/AccessControl/CardOperations/customData/searchTask?format=json*** and set **lpInputParam** to ***JSON_CustomDataSearchCond*** .

## 2.3 Manage Face Information

If you want to access by face, you should add face picture and link the face picture with the card for getting the access permissions, and then apply face information to access control device.

**Before You Start**

- Make sure you have called ***NET_DVR_Init*** to initialize the development resources.
- Make sure you have called ***NET_DVR_Login_V40*** to log in to device.
- Make sure the card information linked with the face picture is applied, refer to ***Manage Card Information*** for details.

**Steps**

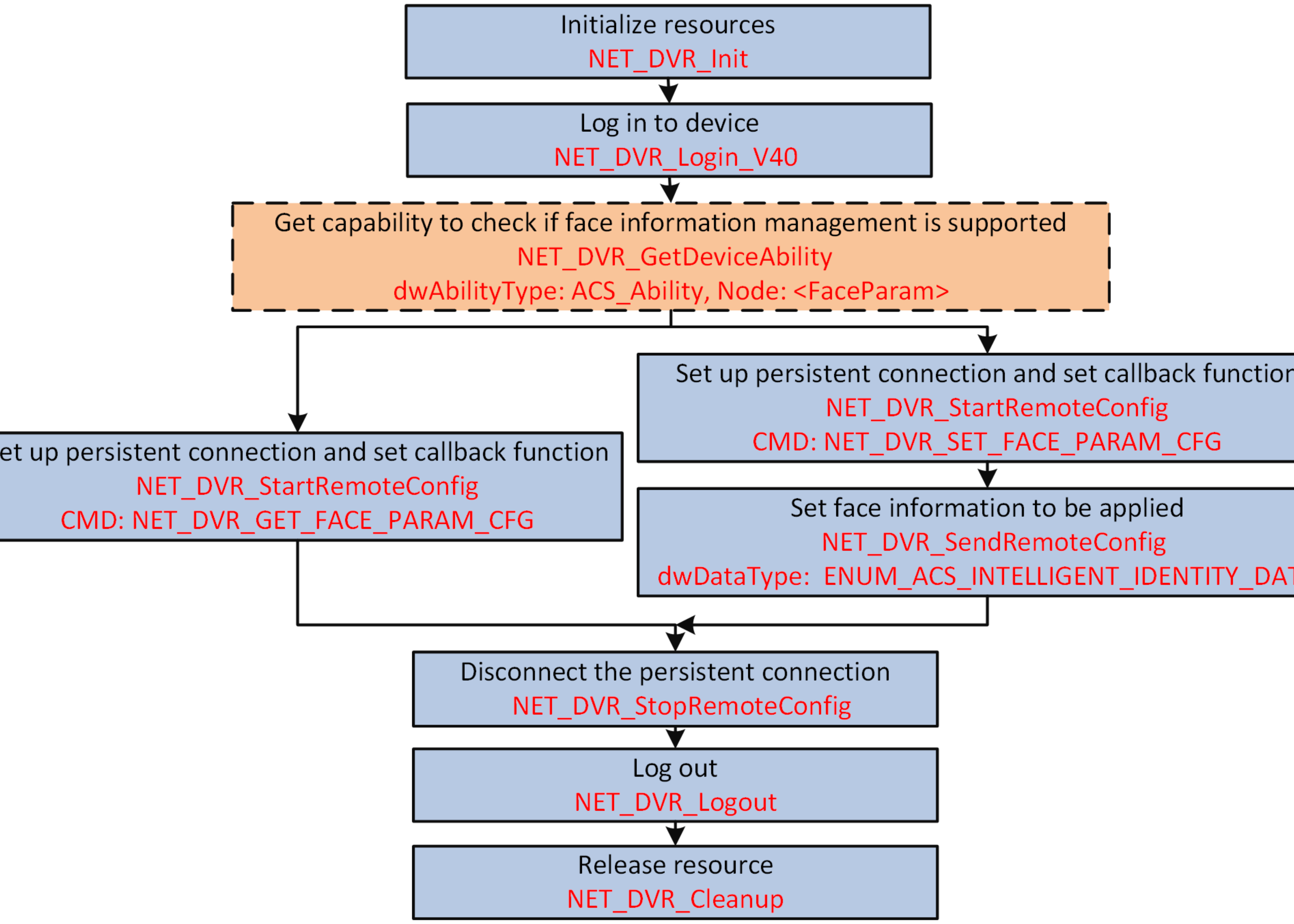
Initialize resources
NET_DVR_Init
Log in to device
NET_DVR_Login_V40
Get capability to check if face information management is supported
NET_DVR_GetDeviceAbility
dwAbilityType: ACS_Ability, Node: <FaceParam>
et up persistent connection and set callback function
NET_DVR_StartRemoteConfig
CMD: NET_DVR_GET_FACE_PARAM_CFG
Set up persistent connection and set callback functior
NET_DVR_StartRemoteConfig
CMD: NET_DVR_SET_FACE_PARAM_CFG
Set face information to be applied
NET_DVR_SendRemoteConfig
dwDataType: ENUM_ACS_INTELLIGENT_IDENTITY_DAT
Disconnect the persistent connection
NET_DVR_StopRemoteConfig
Log out
NET_DVR_Logout
Release resource
NET_DVR_Cleanup

**Figure 2-4 Programming Flow of Managing Face Information**

1. Call ***NET_DVR_GetDeviceAbility*** , specify the capability type (**dwAbilityType**) to "ACS_ABILITY" (macro definition value: 0x801), and set the condition buffer (**pCondBuffer**) to the message ***XML_Desc_AcsAbility*** for getting access control capability to check if managing face information is supported by device.

   The access control capability is returned in the message ***XML_AcsAbility*** by the output buffer (**pOutBuffer**).

   If the node <**FaceParam**> is returned in the message, it indicates that managing face information is supported by device, and you can perform the following steps.

   Otherwise, managing face information is not supported, please end this task.

2. Perform the following operation(s) to get or apply face information.

**Get Face Information**

Call ***NET_DVR_StartRemoteConfig*** with ***NET_DVR_GET_FACE_PARAM_CFG*** (command No.: 2507) and set the input buffer (**lpInBuffer**) to ***NET_DVR_FACE_PARAM_COND*** for setting up persistent connection and setting callback function ( ***fRemoteConfigCallback*** ).

**Note**

The face information is returned in the structure ***NET_DVR_FACE_PARAM_CFG*** by the output buffer (**lpBuffer**) of callback function.

**Apply Face Information**

a. Call ***NET_DVR_StartRemoteConfig*** with ***NET_DVR_SET_FACE_PARAM_CFG*** (command No.: 2508) and set input buffer (**lpInBuffer**) to ***NET_DVR_FACE_PARAM_COND*** for setting up persistent connection and setting callback function ( ***fRemoteConfigCallback*** ).

b. Call ***NET_DVR_SendRemoteConfig*** , specify data type (**dwDataType**) to "ENUM_ACS_INTELLIGENT_IDENTITY_DATA", and set sending buffer (**pSendBuf**) to ***NET_DVR_FACE_PARAM_CFG*** for applying the face information.

**Note**

The applying status is returned by the output buffer (**lpBuffer**) of callback function.

3. Call ***NET_DVR_StopRemoteConfig*** to disconnect the persistent connection.

**What to do next**

Call ***NET_DVR_Logout*** and ***NET_DVR_Cleanup*** to log out and release the resource.

## 2.4 Manage Fingerprint Information

If you want to access by fingerprint, you should collect the fingerprint data via the fingerprint recorder first, and then apply the fingerprint data and parameters (e.g., fingerprint ID, type, and so on) to the fingerprint module of access control device and link the fingerprints with the card for getting the access permissions.

**Before You Start**

- Make sure you have called ***NET_DVR_Init*** to initialize the development resources.
- Make sure you have called ***NET_DVR_Login_V40*** to log in to device.
- Make sure the card information linked with the face picture is applied, refer to ***Manage Card Information*** for details.

**Steps**

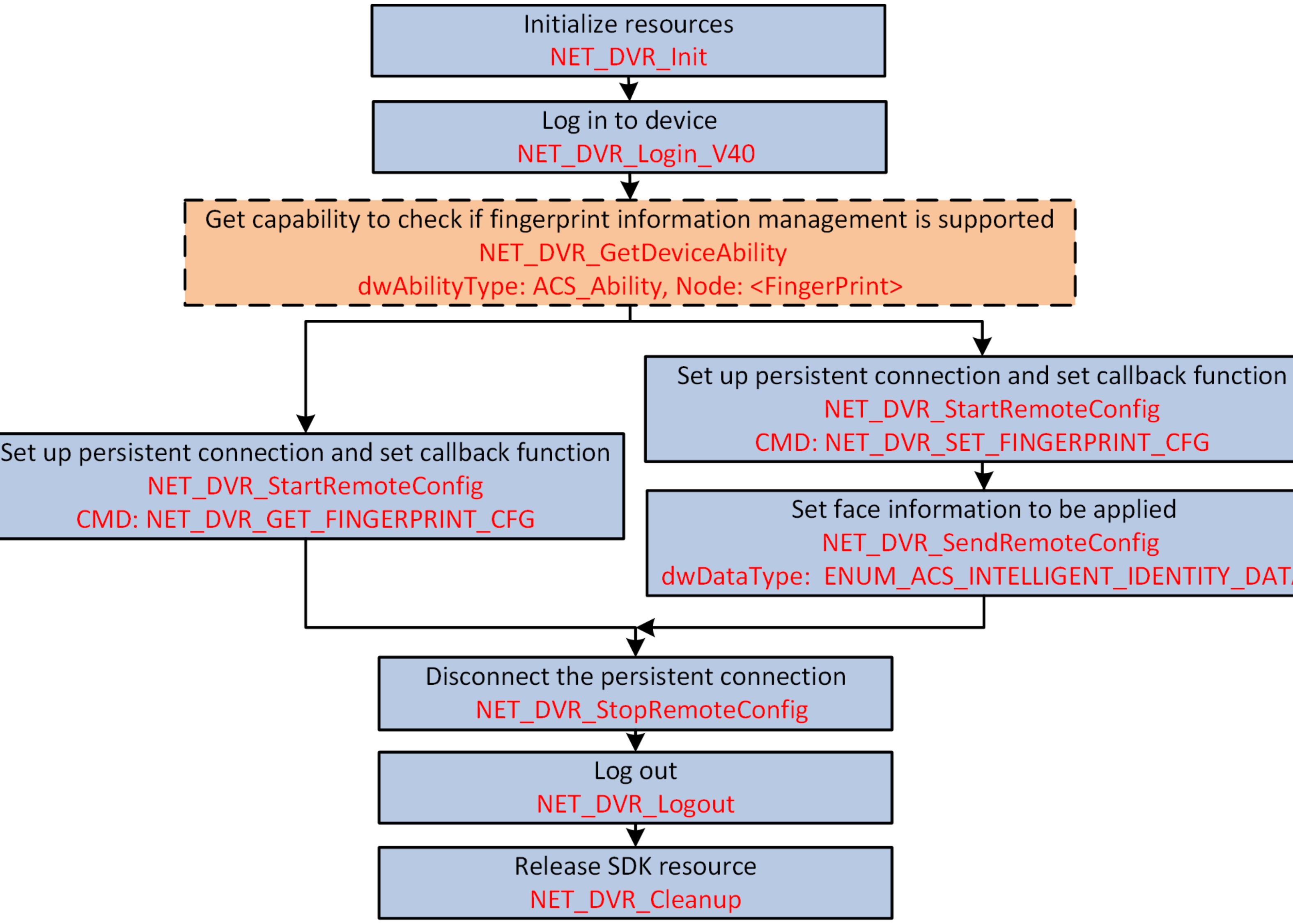
Initialize resources
NET_DVR_Init
Log in to device
NET_DVR_Login_V40
Get capability to check if fingerprint information management is supported
NET_DVR_GetDeviceAbility
dwAbilityType: ACS_Ability, Node: <FingerPrint>
Set up persistent connection and set callback function
NET_DVR_StartRemoteConfig
CMD: NET_DVR_GET_FINGERPRINT_CFG
Set up persistent connection and set callback function
NET_DVR_StartRemoteConfig
CMD: NET_DVR_SET_FINGERPRINT_CFG
Set face information to be applied
NET_DVR_SendRemoteConfig
dwDataType: ENUM_ACS_INTELLIGENT_IDENTITY_DAT
Disconnect the persistent connection
NET_DVR_StopRemoteConfig
Log out
NET_DVR_Logout
Release SDK resource
NET_DVR_Cleanup

**Figure 2-5 Programming Flow of Managing Fingerprint Information**

**Note**

To collect the fingerprint, refer to ***Collect Fingerprint*** for details.

1. Call ***NET_DVR_GetDeviceAbility*** , specify the capability type (**dwAbilityType**) to "ACS_ABILITY" (macro definition value: 0x801), and set the condition buffer (**pCondBuffer**) to the message ***XML_Desc_AcsAbility*** for getting access control capability to check if managing fingerprint information is supported by device.

   The access control capability is returned in the message ***XML_AcsAbility*** by the output buffer (**pOutBuffer**).

If the node <**FingerPrint**> is returned in the message, it indicates that managing fingerprint information is supported by device, and you can perform the following steps.

Otherwise, managing fingerprint information is not supported, please end this task.

2. Perform the following operation(s) to get or apply fingerprint information.

**Get Fingerprint Information**

Call ***NET_DVR_StartRemoteConfig*** with ***NET_DVR_GET_FINGERPRINT_CFG*** (command No.: 2150) and set the input buffer (**lpInBuffer**) to ***NET_DVR_FINGER_PRINT_INFO_COND_V50*** for setting up persistent connection and setting callback function ( ***fRemoteConfigCallback*** ).

**Note**

The fingerprint information is returned in the structure ***NET_DVR_FINGER_PRINT_CFG_V50*** by the output buffer (**lpBuffer**) of callback function.

**Apply Fingerprint Information**

a. Call ***NET_DVR_StartRemoteConfig*** with ***NET_DVR_SET_FINGERPRINT_CFG*** (command No.: 2151) and set input buffer (**lpInBuffer**) to ***NET_DVR_FINGER_PRINT_INFO_COND_V50*** for setting up persistent connection and setting callback function ( ***fRemoteConfigCallback*** ).

b. Call ***NET_DVR_SendRemoteConfig*** , specify data type (**dwDataType**) to "ENUM_ACS_SEND_DATA", and set sending buffer (**pSendBuf**) to ***NET_DVR_FINGER_PRINT_CFG_V50*** for applying the fingerprint information.

**Note**

The applying status is returned by the output buffer (**lpBuffer**) of callback function.

3. Call ***NET_DVR_StopRemoteConfig*** to disconnect the persistent connection.

**What to do next**

Call ***NET_DVR_Logout*** and ***NET_DVR_Cleanup*** to log out and release the resource.

## 2.5 Schedule Settings

### 2.5.1 Configure Authentication Mode Control Schedule

You can configure the week or holiday schedule to regularly control the authentication modes (e.g., by card, by card+password, by fingerprint, by fingerprint+card, and so on) in some specific time periods.

**Before You Start**

- Make sure you have called ***NET_DVR_Init*** to initialize the development environment.
- Make sure you have called ***NET_DVR_Login_V40*** to log in to device.

**Steps**

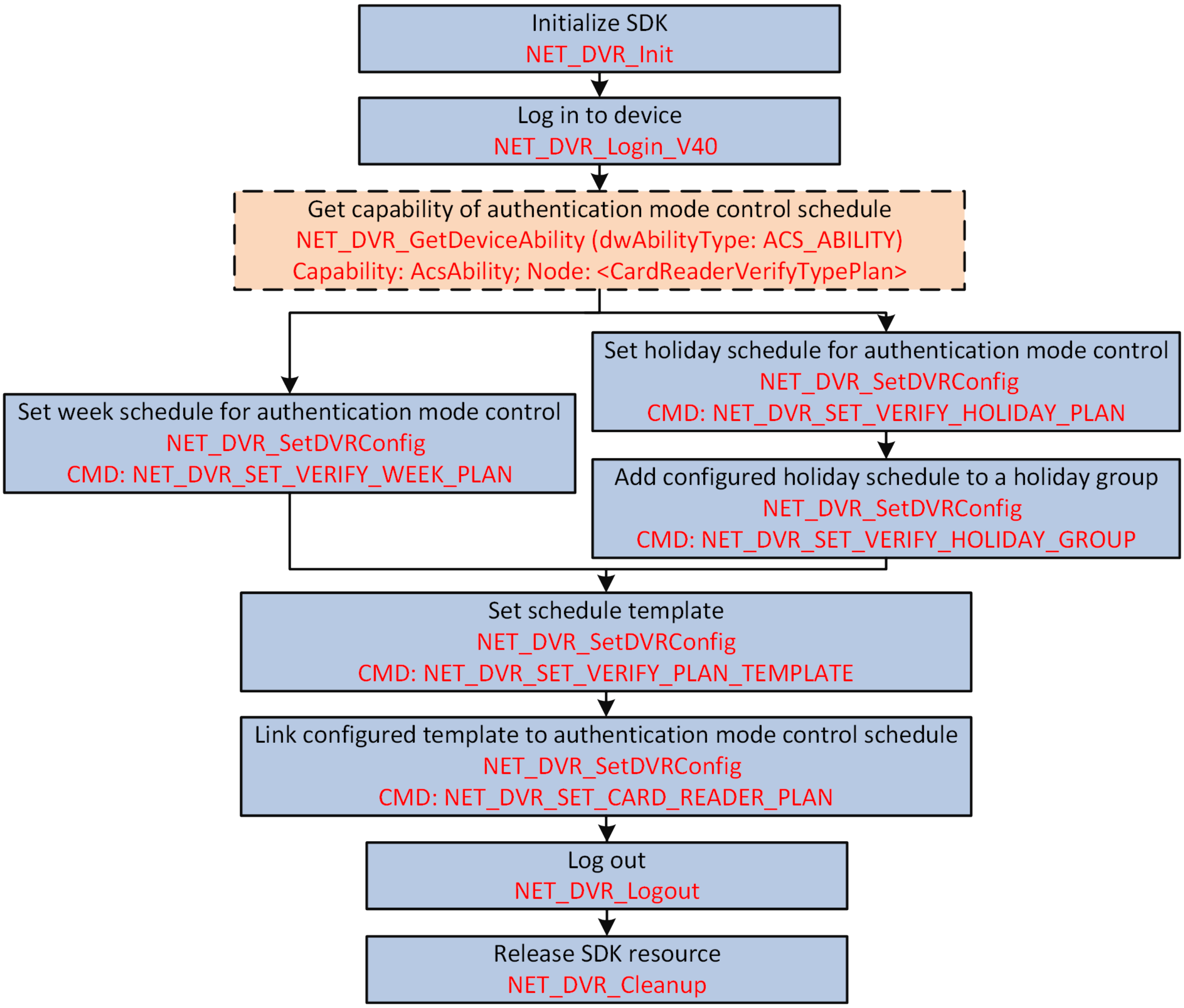
Initialize SDK
NET_DVR_Init
Log in to device
NET_DVR_Login_V40
Get capability of authentication mode control schedule
NET_DVR_GetDeviceAbility (dwAbilityType: ACS_ABILITY)
Capability: AcsAbility; Node: <CardReaderVerifyTypePlan>
Set week schedule for authentication mode control
NET_DVR_SetDVRConfig
CMD: NET_DVR_SET_VERIFY_WEEK_PLAN
Set holiday schedule for authentication mode control
NET_DVR_SetDVRConfig
CMD: NET_DVR_SET_VERIFY_HOLIDAY_PLAN
Add configured holiday schedule to a holiday group
NET_DVR_SetDVRConfig
CMD: NET_DVR_SET_VERIFY_HOLIDAY_GROUP
Set schedule template
NET_DVR_SetDVRConfig
CMD: NET_DVR_SET_VERIFY_PLAN_TEMPLATE
Link configured template to authentication mode control schedule
NET_DVR_SetDVRConfig
CMD: NET_DVR_SET_CARD_READER_PLAN
Log out
NET_DVR_Logout
Release SDK resource
NET_DVR_Cleanup

**Figure 2-6 Programming Flow of Configuring Authentication Mode Control Schedule**

1. **Optional:** Call ***NET_DVR_GetDeviceAbility*** , specify the capability type **dwAbilityType** to "ACS_ABILITY", set the input buffer (**pInBuf**) to ***XML_Desc_AcsAbility*** for getting the access control capability to check if configuring authentication mode control schedule is supported.

   The capability is returned in the message ***XML_AcsAbility*** by the output pointer (**pOutBuf**).

   If the node <**CardReaderVerifyTypePlan**> is returned, it indicates that configuring authentication mode control schedule is supported, and you can continue to perform the following steps.

Otherwise, configuring authentication mode control schedule is not supported, please end this task.

2. Perform one of the following operations to set week or holiday schedule for authentication mode control.
   - a. Call ***NET_DVR_GetDVRConfig*** with ***NET_DVR_GET_VERIFY_WEEK_PLAN*** (command No.: 2124) to get the existing week schedule configurations for reference.

     **Note**

     The week schedule parameters are returned in the structure ***NET_DVR_WEEK_PLAN_CFG*** by output buffer (**lpOutBuffer**).

     b. Call ***NET_DVR_SetDVRConfig*** with ***NET_DVR_SET_VERIFY_WEEK_PLAN*** (command No.: 2125) and set the input buffer (**lpInBuffer**) to ***NET_DVR_WEEK_PLAN_CFG*** for setting the week schedule.
   - a. Call ***NET_DVR_GetDVRConfig*** with ***NET_DVR_GET_VERIFY_HOLIDAY_PLAN*** (command No.: 2128) to get the existing holiday schedule configurations for reference.

     **Note**

     The holiday schedule parameters are returned in the structure ***NET_DVR_HOLIDAY_PLAN_CFG*** by output buffer (**lpOutBuffer**).

     b. Call ***NET_DVR_SetDVRConfig*** with ***NET_DVR_SET_VERIFY_HOLIDAY_PLAN*** (command No.: 2129) and set the input buffer (**lpInBuffer**) to ***NET_DVR_HOLIDAY_PLAN_CFG*** for setting the holiday schedule.

     c. Call ***NET_DVR_GetDVRConfig*** with ***NET_DVR_GET_VERIFY_HOLIDAY_GROUP*** (command No.: 2132) to get the existing holiday group configurations for reference.

     **Note**

     The holiday group parameters are returned in the structure ***NET_DVR_HOLIDAY_GROUP_CFG*** by output buffer (**lpOutBuffer**).

     d. Call ***NET_DVR_SetDVRConfig*** with ***NET_DVR_SET_VERIFY_HOLIDAY_GROUP*** (command No.: 2133) and set the input buffer (**lpInBuffer**) to ***NET_DVR_HOLIDAY_GROUP_CFG*** for adding the configured holiday schedule to a holiday group.
3. **Optional:** Call ***NET_DVR_GetDVRConfig*** with ***NET_DVR_GET_VERIFY_PLAN_TEMPLATE*** (command No.: 2136) to get the existing schedule template configurations for reference.

**Note**

The schedule template parameters are returned in the structure ***NET_DVR_PLAN_TEMPLATE*** by output buffer (**lpOutBuffer**).

4. Call ***NET_DVR_SetDVRConfig*** with
   ***NET_DVR_SET_VERIFY_PLAN_TEMPLATE***
   (command No.: 2137) and set the input buffer (**lpInBuffer**) to ***NET_DVR_PLAN_TEMPLATE*** for setting the schedule template.
5. **Optional:** Call ***NET_DVR_GetDVRConfig*** with
   ***NET_DVR_GET_CARD_READER_PLAN***
   (command No.: 2142) to get the existing authentication mode control schedule configurations for reference.

**Note**

The authentication mode control schedule parameters are returned in the structure ***NET_DVR_CARD_READER_PLAN*** by output buffer (**lpOutBuffer**).

6. Call ***NET_DVR_SetDVRConfig***
   ***NET_DVR_SET_CARD_READER_PLAN***
   (command No.: 2143) and set the input buffer (**lpInBuffer**) to ***NET_DVR_CARD_READER_PLAN*** for linking the configured template to the authentication mode control schedule and finishing the configuration.

**Example**

Sample Code for Configuring Authentication Mode Control Schedule

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

void main()
{
   //---------------------------------------
   //Initialize
   NET_DVR_Init();

   //Set connection timeout and reconnection function
   NET_DVR_SetConnectTime(2000, 1);
   NET_DVR_SetReconnect(10000, true);

   //---------------------------------------
   //Log in to device
   LONG lUserID;
   //Login parameters, including device IP address, user name, password, and so on
   NET_DVR_USER_LOGIN_INFO struLoginInfo = {0};
   struLoginInfo.bUseAsynLogin = 0; //Synchronous login mode
   strcpy(struLoginInfo.sDeviceAddress, "192.168.1.64"); //Device IP address
```

```
    struLoginInfo.wPort = 8000; //Device service port number
    strcpy(struLoginInfo.sUserName, "admin"); //User name
    strcpy(struLoginInfo.sPassword, "abcd1234"); //Password

    //Device information, output parameter
    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;
    }

    //---------------------------------------
    //Set card reader authentication mode schedule, template 1 linked to card reader 1
    NET_DVR_CARD_READER_PLAN struCardReaderPlan = {0};
    struCardReaderPlan.dwSize = sizeof(struCardReaderPlan);
    struCardReaderPlan.dwTemplateNo = 1;//Schedule template 1
    BOOL bRet1 = NET_DVR_SetDVRConfig(lUserID, NET_DVR_SET_CARD_READER_PLAN, 1, \
        &struCardReaderPlan, sizeof(struCardReaderPlan));
    if (!bRet1)
    {
        printf("Setting card reader authentication mode schedule failed, error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(lUserID);
        NET_DVR_Cleanup();
        return;
    }

    //Set card reader authentication mode schedule template 1, template 1 links to week schedule 1 and holiday group
1
    CString    m_csTemplateName = "card reader authentication mode schedule template 1";
    NET_DVR_PLAN_TEMPLATE struPlanTem = {0};
    struPlanTem.dwSize = sizeof(struPlanTem);
    struPlanTem.byEnable = 1;//Enable or not: 0-No, 1-Yes
    strncpy((char *)struPlanTem.byTemplateName, (LPCTSTR)m_csTemplateName, TEMPLATE_NAME_LEN);
    struPlanTem.dwWeekPlanNo = 2;//Week schedule No.2
    struPlanTem.dwHolidayGroupNo[0] = 2;//Holiday group No.2, up to 16 holiday groups can be linked to each
schedule

    BOOL bRet2 = NET_DVR_SetDVRConfig(lUserID, NET_DVR_SET_VERIFY_PLAN_TEMPLATE, 1, \
        &struPlanTem, sizeof(struPlanTem));
    if (!bRet2)
    {
        printf("Setting card reader authentication mode schedule template failed, error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(lUserID);
        NET_DVR_Cleanup();
        return;
    }

    //Set week schedule 2 for card reader authentication mode
```

```
  NET_DVR_WEEK_PLAN_CFG struWeekPlan2 = {0};
  struWeekPlan2.dwSize = sizeof(struWeekPlan2);
  struWeekPlan2.byEnable = 1;//Enable week schedule

  NET_DVR_SINGLE_PLAN_SEGMENT struSinglePlanSegment = {0};
  LPNET_DVR_SINGLE_PLAN_SEGMENT lpPlanSegment = &struSinglePlanSegment;
  struSinglePlanSegment.byEnable = 1;
  struSinglePlanSegment.byVerifyMode = 4;//Authentication mode: 0-invalid, 1-sleepy, 2-card+password, 3-card,
                              //4-card or password, 5-fingerprint, 6-fingerprint+password, 7-fingerprint or card,
                              //8-fingerprint+card, 9-fingerprint+card+password
  struSinglePlanSegment.struTimeSegment.struBeginTime.byHour = 0;//Start time
  struSinglePlanSegment.struTimeSegment.struBeginTime.byMinute = 0;
  struSinglePlanSegment.struTimeSegment.struBeginTime.bySecond = 0;

  struSinglePlanSegment.struTimeSegment.struEndTime.byHour = 23;//End time
  struSinglePlanSegment.struTimeSegment.struEndTime.byMinute = 59;
  struSinglePlanSegment.struTimeSegment.struEndTime.bySecond = 59;

  /*Up to 8 time periods can be set for each day, and you can set different authentication modes for each time period
  Here only takes setting one period for each day*/

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

  BOOL bRet3 = NET_DVR_SetDVRConfig(lUserID, NET_DVR_SET_VERIFY_WEEK_PLAN, 2, \
    &struWeekPlan2, sizeof(struWeekPlan2));
  if (!bRet3)
  {
    printf("Setting week schedule for card reader authentication mode failed,error:%d.\n", NET_DVR_GetLastError());
    NET_DVR_Logout(lUserID);
    NET_DVR_Cleanup();
    return;
  }

  //Set holiday group for card reader authentication mode
  CString    m_csGroupName = "Holiday group 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;//Holiday group 1 links to holiday schedule 1,
                                //up to 16 holiday schedules can be linked to one holiday group

  BOOL bRet4 = NET_DVR_SetDVRConfig(lUserID, NET_DVR_SET_VERIFY_HOLIDAY_GROUP, 2, \
    &struHolidayGroup2, sizeof(struHolidayGroup2));
  if (!bRet4)
  {
    printf(" Setting holiday group for card reader authentication mode failed, error:%d.\n",
NET_DVR_GetLastError());
    NET_DVR_Logout(lUserID);
```

```
        NET_DVR_Cleanup();
        return;
    }

    //Set holiday schedule for card reader authentication mode
    NET_DVR_HOLIDAY_PLAN_CFG struHolidayPlan2 = {0};
    struHolidayPlan2.dwSize = sizeof(struHolidayPlan2);
    struHolidayPlan2.byEnable = 1;
    struHolidayPlan2.struBeginDate.wYear = 2017;//Holiday start date
    struHolidayPlan2.struBeginDate.byMonth = 10;
    struHolidayPlan2.struBeginDate.byDay = 1;
    struHolidayPlan2.struEndDate.wYear = 2017;//Holiday end date
    struHolidayPlan2.struEndDate.byMonth = 10;
    struHolidayPlan2.struEndDate.byDay = 7;
    //Copy the week schedule parameters to holiday schedule of card reader authentication mode
    memcpy(struHolidayPlan2.struPlanCfg, struWeekPlan2.struPlanCfg,
sizeof(NET_DVR_SINGLE_PLAN_SEGMENT)*MAX_DAYS*MAX_TIMESEGMENT_V30);

    BOOL bRet5 = NET_DVR_SetDVRConfig(lUserID, NET_DVR_SET_VERIFY_HOLIDAY_PLAN, 2, \
        &struHolidayPlan2, sizeof(struHolidayPlan2));
    if (!bRet5)
    {
        printf("Setting holiday schedule for card reader authentication mode failed, error:%d.\n",
NET_DVR_GetLastError());
        NET_DVR_Logout(lUserID);
        NET_DVR_Cleanup();
        return;
    }
    //---------------------------------------
    //Exit
    Sleep(5000);

    //Log out
    NET_DVR_Logout(lUserID);
    //Release SDK resource
    NET_DVR_Cleanup();
    return;
}
```

**What to do next**

Call ***NET_DVR_Logout*** and ***NET_DVR_Cleanup*** to log out and release the resource.

## 2.5.2 Configure Access Permission Control Schedule

To regularly control the access permissions for managing the accessible time duration (by default, it is 24 hours) of some important access control points, you can configure the week or holiday schedules.

**Before You Start**

- Make sure you have called ***NET_DVR_Init*** to initialize the development environment.
- Make sure you have called ***NET_DVR_Login_V40*** to log in to device.

**Steps**

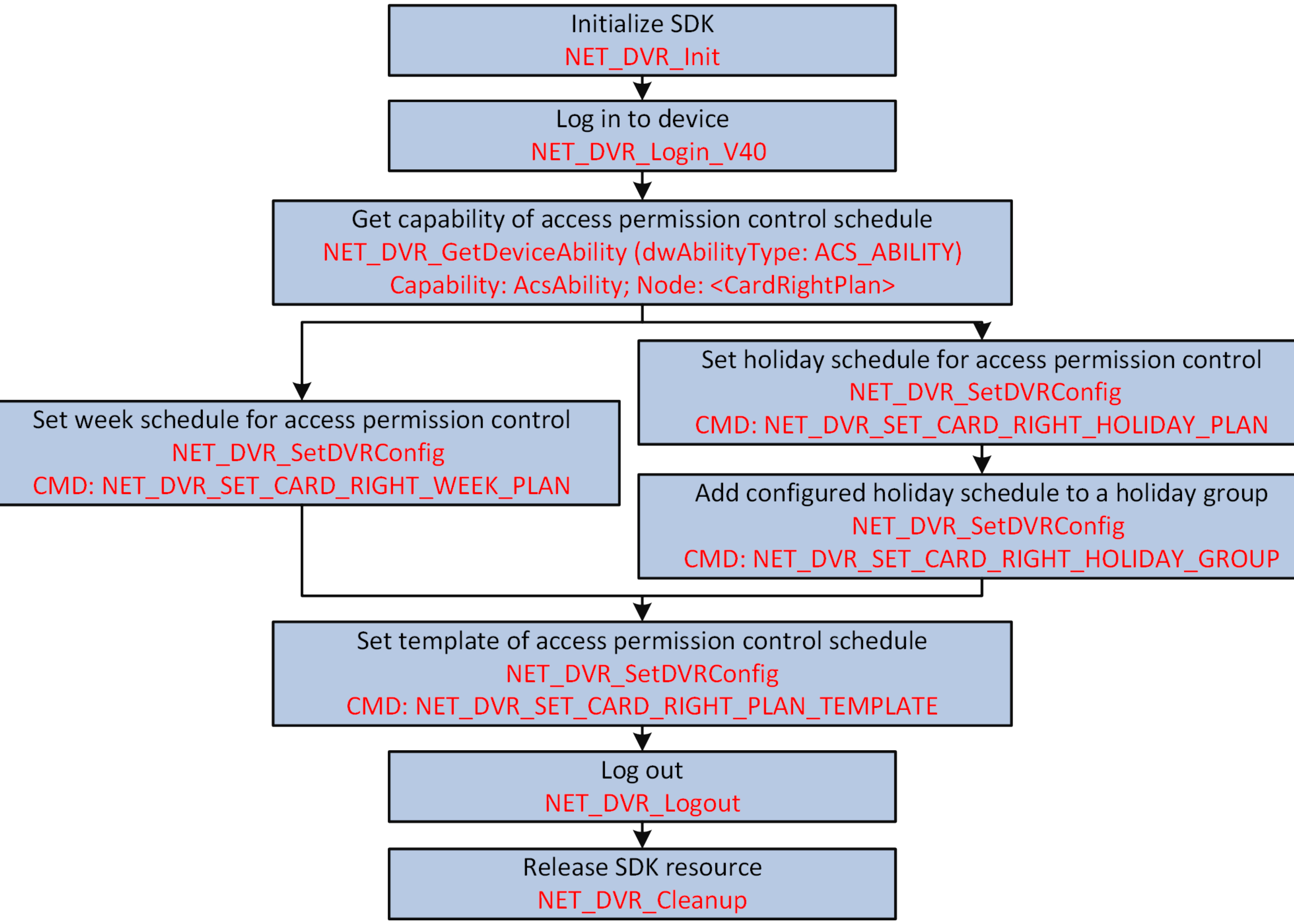
Initialize SDK
NET_DVR_Init
Log in to device
NET_DVR_Login_V40
Get capability of access permission control schedule
NET_DVR_GetDeviceAbility (dwAbilityType: ACS_ABILITY)
Capability: AcsAbility; Node: <CardRightPlan>
Set week schedule for access permission control
NET_DVR_SetDVRConfig
CMD: NET_DVR_SET_CARD_RIGHT_WEEK_PLAN
Set holiday schedule for access permission control
NET_DVR_SetDVRConfig
CMD: NET_DVR_SET_CARD_RIGHT_HOLIDAY_PLAN
Add configured holiday schedule to a holiday group
NET_DVR_SetDVRConfig
CMD: NET_DVR_SET_CARD_RIGHT_HOLIDAY_GROUP
Set template of access permission control schedule
NET_DVR_SetDVRConfig
CMD: NET_DVR_SET_CARD_RIGHT_PLAN_TEMPLATE
Log out
NET_DVR_Logout
Release SDK resource
NET_DVR_Cleanup

**Figure 2-7 Programming Flow of Configuring Access Permission Control Schedule**

1. Call ***NET_DVR_GetDeviceAbility*** , specify **dwAbilityType** to "ACS_ABILITY", set **pInBuf** to ***XML_Desc_AcsAbility*** for getting the access control capability to check if configuring access permission control schedule is supported.

   The capability is returned in the message ***XML_AcsAbility*** by **pOutBuf**.

   If the node <**CardRightPlan**> is returned, it indicates that configuring access permission control schedule is supported, and you can continue to perform the following steps.

Otherwise, configuring access permission control schedule is not supported, please end this task.

2. Perform one of the following operations to set week or holiday schedule for access permission control.
   - a. Call ***NET_DVR_GetDVRConfig*** with "NET_DVR_GET_CARD_RIGHT_WEEK_PLAN" (command No.: 2126) to get default or configured week schedule configurations for reference.

     **Note**

     The week schedule parameters are returned in the structure ***NET_DVR_WEEK_PLAN_CFG*** by **lpOutBuffer**.

     b. Call ***NET_DVR_SetDVRConfig*** with "NET_DVR_SET_CARD_RIGHT_WEEK_PLAN" (command No.: 2127) and set **lpInBuffer** to ***NET_DVR_WEEK_PLAN_CFG*** for setting the week schedule.
   - a. Call ***NET_DVR_GetDVRConfig*** with "NET_DVR_GET_CARD_RIGHT_HOLIDAY_PLAN" (command No.: 2130) to get default or configured holiday schedule configurations for reference.

     **Note**

     The holiday schedule parameters are returned in the structure ***NET_DVR_HOLIDAY_PLAN_CFG*** by **lpOutBuffer**.

     b. Call ***NET_DVR_SetDVRConfig*** with "NET_DVR_SET_CARD_RIGHT_HOLIDAY_PLAN" (command No.: 2131) and set **lpInBuffer** to ***NET_DVR_HOLIDAY_PLAN_CFG*** for setting the week schedule.

     c. Call ***NET_DVR_GetDVRConfig*** with "NET_DVR_GET_CARD_RIGHT_HOLIDAY_GROUP" (command No.: 2134) to get default or configured holiday group configurations for reference.

     **Note**

     The holiday group parameters are returned in the structure ***NET_DVR_HOLIDAY_GROUP_CFG*** by **lpOutBuffer**.

     d. Call ***NET_DVR_SetDVRConfig*** with "NET_DVR_SET_CARD_RIGHT_HOLIDAY_GROUP" (command No.: 2135) and set **lpInBuffer** to ***NET_DVR_HOLIDAY_GROUP_CFG*** for adding the configured holiday schedule to a holiday group.
3. **Optional:** Call ***NET_DVR_GetDVRConfig*** with "NET_DVR_GET_CARD_RIGHT_PLAN_TEMPLATE" (command No.: 2138) to get default or configured schedule template configurations for reference.

   **Note**

   The schedule template parameters are returned in the structure ***NET_DVR_PLAN_TEMPLATE*** by **lpOutBuffer**.
4. Call ***NET_DVR_SetDVRConfig*** with "NET_DVR_SET_CARD_RIGHT_PLAN_TEMPLATE" (command No.: 2139) and set **lpInBuffer** to ***NET_DVR_PLAN_TEMPLATE*** for setting the schedule template.

**Note**

The configured schedule template can be directly linked to person ID when applying person information. And the linked person can get the access permission configured in the template.

**Example**

Sample Code for Configuring Access Permission Control Schedule

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

void main()
{
   //---------------------------------------
   //Initialize
   NET_DVR_Init();

   //Set connection timeout and reconnection function
   NET_DVR_SetConnectTime(2000, 1);
   NET_DVR_SetReconnect(10000, true);

   //---------------------------------------
   //Log in to device
   LONG lUserID;
   //Login parameters, including device IP address, user name, password, and so on
   NET_DVR_USER_LOGIN_INFO struLoginInfo = {0};
   struLoginInfo.bUseAsynLogin = 0; //Synchronous login mode
   strcpy(struLoginInfo.sDeviceAddress, "192.168.1.64"); //Device IP address
   struLoginInfo.wPort = 8000; //Device service port number
   strcpy(struLoginInfo.sUserName, "admin"); //User name
   strcpy(struLoginInfo.sPassword, "abcd1234"); //Password

   //Device information, output parameter
   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;
   }
   //---------------------------------------
   //Set access permission schedule template, when issuing card, link to this template

   CString    m_csTemplateName = "Access permission schedule template 1";
   NET_DVR_PLAN_TEMPLATE struPlanTem = {0};
   struPlanTem.dwSize = sizeof(struPlanTem);
```

```
  struPlanTem.byEnable = 1;//Enable or not: 0-No, 1-Yes
  strncpy((char *)struPlanTem.byTemplateName, (LPCTSTR)m_csTemplateName, TEMPLATE_NAME_LEN);
  struPlanTem.dwWeekPlanNo = 1;//Week schedule No.1
  struPlanTem.dwHolidayGroupNo[0] = 1;//Holiday group No.1, up to 16 holiday groups can be linked to each
schedule
  //struPlanTem.dwHolidayGroupNo[1] = 2;//Holiday group No.2

  BOOL bRet1 = NET_DVR_SetDVRConfig(lUserID, NET_DVR_SET_CARD_RIGHT_PLAN_TEMPLATE, 1, \
    &struPlanTem, sizeof(struPlanTem));
  if (!bRet1)
  {
    printf("Setting access permission schedule template failed, error:%d.\n", NET_DVR_GetLastError());
    NET_DVR_Logout(lUserID);
    NET_DVR_Cleanup();
    return;
  }

  //Set week schedule 1 for access permission
  NET_DVR_WEEK_PLAN_CFG struWeekPlan = {0};
  struWeekPlan.dwSize = sizeof(struWeekPlan);
  struWeekPlan.byEnable = 1;//Enable week schedule

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

  struSinglePlanSegment.struTimeSegment.struBeginTime.byHour = 0;//Start time
  struSinglePlanSegment.struTimeSegment.struBeginTime.byMinute = 0;
  struSinglePlanSegment.struTimeSegment.struBeginTime.bySecond = 0;

  struSinglePlanSegment.struTimeSegment.struEndTime.byHour = 23;//End time
  struSinglePlanSegment.struTimeSegment.struEndTime.byMinute = 59;
  struSinglePlanSegment.struTimeSegment.struEndTime.bySecond = 59;

  /*Up to 8 time periods can be set for each day. Here only takes setting one period for each day*/

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

  BOOL bRet2 = NET_DVR_SetDVRConfig(lUserID, NET_DVR_SET_CARD_RIGHT_WEEK_PLAN, 1, \
    &struWeekPlan, sizeof(struWeekPlan));
  if (!bRet2)
  {
    printf("Setting week schedule for access permission failed, error:%d.\n", NET_DVR_GetLastError());
    NET_DVR_Logout(lUserID);
    NET_DVR_Cleanup();
    return;
  }

  //Set holiday group for access permission
```

```
    CString    m_csGroupName = "access permission holiday group 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;//Holiday group 1 links to holiday schedule 1,
                                              //up to 16 holiday schedules can be linked to one holiday group

    BOOL bRet3 = NET_DVR_SetDVRConfig(lUserID, NET_DVR_SET_CARD_RIGHT_HOLIDAY_GROUP, 1, \
        &struHolidayGroup1, sizeof(struHolidayGroup1));
    if (!bRet3)
    {
        printf(" Setting holiday group for access permission failed, error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(lUserID);
        NET_DVR_Cleanup();
        return;
    }

    //Set holiday schedule for access permission
    NET_DVR_HOLIDAY_PLAN_CFG struHolidayPlan = {0};
    struHolidayPlan.dwSize = sizeof(struHolidayPlan);
    struHolidayPlan.byEnable = 1;
    struHolidayPlan.struBeginDate.wYear = 2017;//Holiday start date
    struHolidayPlan.struBeginDate.byMonth = 10;
    struHolidayPlan.struBeginDate.byDay = 1;
    struHolidayPlan.struEndDate.wYear = 2017;//Holiday end date
    struHolidayPlan.struEndDate.byMonth = 10;
    struHolidayPlan.struEndDate.byDay = 7;
    //Copy the week schedule parameters to holiday schedule of access permission
    memcpy(struHolidayPlan.struPlanCfg, struWeekPlan.struPlanCfg,
sizeof(NET_DVR_SINGLE_PLAN_SEGMENT)*MAX_DAYS*MAX_TIMESEGMENT_V30);

    BOOL bRet4 = NET_DVR_SetDVRConfig(lUserID, NET_DVR_SET_CARD_RIGHT_HOLIDAY_PLAN, 1, \
        &struHolidayPlan, sizeof(struHolidayPlan));
    if (!bRet4)
    {
        printf("Setting holiday schedule for access permission failed, error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(lUserID);
        NET_DVR_Cleanup();
        return;
    }
    //---------------------------------------
    //Exit
    Sleep(5000);

    //Log out
    NET_DVR_Logout(lUserID);
    //Release SDK resource
    NET_DVR_Cleanup();
    return;
}
```

**What to do next**

Call ***NET_DVR_Logout*** and ***NET_DVR_Cleanup*** to log out and release the resource.

### 2.5.3 Configure Door Control Schedule

You can configure the week or holiday schedule to regularly control the door statuses, including Remain Open (access without authentication), Remain Closed (access is not allowed), and Normal (access with authentication), in some specific time periods.

**Before You Start**

- Make sure you have called ***NET_DVR_Init*** to initialize the development environment.
- Make sure you have called ***NET_DVR_Login_V40*** to log in to device.

**Steps**

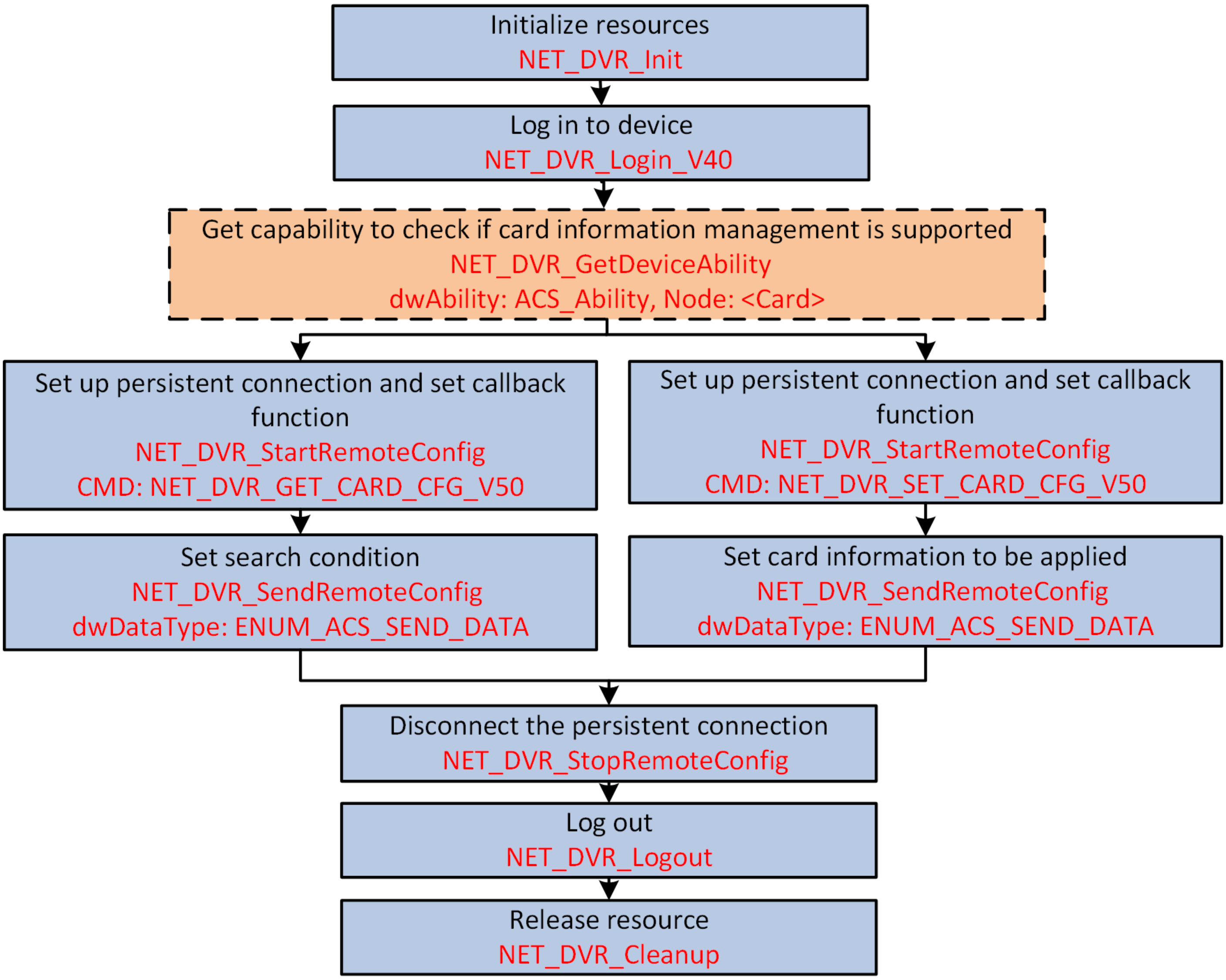
Initialize resources
NET_DVR_Init
Log in to device
NET_DVR_Login_V40
Get capability to check if card information management is supported
NET_DVR_GetDeviceAbility
dwAbility: ACS_Ability, Node: <Card>
Set up persistent connection and set callback function
NET_DVR_StartRemoteConfig
CMD: NET_DVR_GET_CARD_CFG_V50
Set up persistent connection and set callback function
NET_DVR_StartRemoteConfig
CMD: NET_DVR_SET_CARD_CFG_V50
Set search condition
NET_DVR_SendRemoteConfig
dwDataType: ENUM_ACS_SEND_DATA
Set card information to be applied
NET_DVR_SendRemoteConfig
dwDataType: ENUM_ACS_SEND_DATA
Disconnect the persistent connection
NET_DVR_StopRemoteConfig
Log out
NET_DVR_Logout
Release resource
NET_DVR_Cleanup

**Figure 2-8 Programming Flow of Configuring Door Control Schedule**

1. Call ***NET_DVR_GetDeviceAbility*** , specify the capability type **dwAbilityType** to "ACS_ABILITY", set the input buffer (**pInBuf**) to ***XML_Desc_AcsAbility*** for getting the access control capability to check if configuring door control schedule is supported.

   The capability is returned in the message ***XML_AcsAbility*** by the output pointer (**pOutBuf**).

   If the node <**DoorStatusPlan**> is returned, it indicates that configuring door control schedule is supported, and you can continue to perform the following steps.

   Otherwise, configuring door control schedule is not supported, please end this task.

2. Perform one of the following operations to set week or holiday schedule for door control.
   - a. Call ***NET_DVR_GetDVRConfig*** with "NET_DVR_GET_WEEK_PLAN_CFG" (command No.: 2100) to get the existing week schedule configurations for reference.

**Note**

The week schedule parameters are returned in the structure ***NET_DVR_WEEK_PLAN_CFG*** by output buffer (**lpOutBuffer**).

b. Call ***NET_DVR_SetDVRConfig*** with "NET_DVR_SET_WEEK_PLAN_CFG" (command No.: 2101) and set the input buffer (**lpInBuffer**) to ***NET_DVR_WEEK_PLAN_CFG*** for setting the week schedule.

- a. Call ***NET_DVR_GetDVRConfig*** with "NET_DVR_GET_DOOR_STATUS_HOLIDAY_PLAN" (command No.: 2102) to get the existing holiday schedule configurations for reference.

**Note**

The holiday schedule parameters are returned in the structure ***NET_DVR_HOLIDAY_PLAN_CFG*** by output buffer (**lpOutBuffer**).

b. Call ***NET_DVR_SetDVRConfig*** with "NET_DVR_SET_DOOR_STATUS_HOLIDAY_PLAN" (command No.: 2103) and set the input buffer (**lpInBuffer**) to ***NET_DVR_HOLIDAY_PLAN_CFG*** for setting the week schedule.

c. Call ***NET_DVR_GetDVRConfig*** with "NET_DVR_GET_DOOR_STATUS_HOLIDAY_GROUP" (command No.: 2104) to get the existing holiday group configurations for reference.

**Note**

The holiday group parameters are returned in the structure ***NET_DVR_HOLIDAY_GROUP_CFG*** by output buffer (**lpOutBuffer**).

d. Call ***NET_DVR_SetDVRConfig*** with "NET_DVR_SET_DOOR_STATUS_HOLIDAY_GROUP" (command No.: 2105) and set the input buffer (**lpInBuffer**) to ***NET_DVR_HOLIDAY_GROUP_CFG*** for adding the configured holiday schedule to a holiday group.

3. **Optional:** Call ***NET_DVR_GetDVRConfig*** with "NET_DVR_GET_DOOR_STATUS_PLAN_TEMPLATE" (command No.: 2106) to get the existing schedule template configurations for reference.

**Note**

The schedule template parameters are returned in the structure ***NET_DVR_PLAN_TEMPLATE*** by output buffer (**lpOutBuffer**).

4. Call ***NET_DVR_SetDVRConfig*** with "NET_DVR_SET_DOOR_STATUS_PLAN_TEMPLATE" (command No.: 2107) and set the input buffer (**lpInBuffer**) to ***NET_DVR_PLAN_TEMPLATE*** for setting the schedule template.
5. **Optional:** Call ***NET_DVR_GetDVRConfig*** with "NET_DVR_GET_DOOR_STATUS_PLAN" (command No.: 2110) to get the existing door control schedule configurations for reference.

**Note**

The door control schedule parameters are returned in the structure ***NET_DVR_DOOR_STATUS_PLAN*** by output buffer (**lpOutBuffer**).

6. Call ***NET_DVR_SetDVRConfig*** with "NET_DVR_SET_DOOR_STATUS_PLAN" (command No.: 2111) and set the input buffer (**lpInBuffer**) to ***NET_DVR_DOOR_STATUS_PLAN*** for linking the configured template to the door control schedule and finishing the configuration.

**Example**

Sample Code for Configuring Door Control Schedule

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

void main()
{
  //--------------------------------------
  //Initialize
  NET_DVR_Init();

  //Set connection timeout and reconnection function
  NET_DVR_SetConnectTime(2000, 1);
  NET_DVR_SetReconnect(10000, true);

  //--------------------------------------
  //Log in to device
  LONG lUserID;
  //Login parameters, including device IP address, user name, password, and so on
  NET_DVR_USER_LOGIN_INFO struLoginInfo = {0};
  struLoginInfo.bUseAsynLogin = 0; //Synchronous login mode
  strcpy(struLoginInfo.sDeviceAddress, "192.168.1.64"); //Device IP address
  struLoginInfo.wPort = 8000; //Device service port number
  strcpy(struLoginInfo.sUserName, "admin"); //User name
  strcpy(struLoginInfo.sPassword, "abcd1234"); //Password

  //Device information, output parameter
  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;
  }

  //--------------------------------------
  //Set door status schedule, template 1 linked to door 1
  NET_DVR_DOOR_STATUS_PLAN struDoorStatusPlan = {0};
  struDoorStatusPlan.dwSize = sizeof(struDoorStatusPlan);
  struDoorStatusPlan.dwTemplateNo = 1;//Schedule template 1
```

```
    BOOL bRet1 = NET_DVR_SetDVRConfig(lUserID, NET_DVR_SET_DOOR_STATUS_PLAN, 1, \
        &struDoorStatusPlan, sizeof(struDoorStatusPlan));
    if (!bRet1)
    {
        printf("Setting door status schedule failed, error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(lUserID);
        NET_DVR_Cleanup();
        return;
    }

    //Set door status schedule template 1, template 1 links to week schedule 1 and holiday group 1
    CString    m_csTemplateName = "door status schedule template 1";
    NET_DVR_PLAN_TEMPLATE struPlanTem = {0};
    struPlanTem.dwSize = sizeof(struPlanTem);
    struPlanTem.byEnable = 1;//Enable or not: 0-No, 1-Yes
    strncpy((char *)struPlanTem.byTemplateName, (LPCTSTR)m_csTemplateName, TEMPLATE_NAME_LEN);
    struPlanTem.dwWeekPlanNo = 1;//Week schedule No.1
    struPlanTem.dwHolidayGroupNo[0] = 1;//Holiday group No.1, up to 16 holiday groups can be linked to each
schedule
    //struPlanTem.dwHolidayGroupNo[1] = 2;//Holiday group No.2

    BOOL bRet2 = NET_DVR_SetDVRConfig(lUserID, NET_DVR_SET_DOOR_STATUS_PLAN_TEMPLATE, 1, \
        &struPlanTem, sizeof(struPlanTem));
    if (!bRet2)
    {
        printf("Setting door status schedule template failed, error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(lUserID);
        NET_DVR_Cleanup();
        return;
    }

    //Set week schedule 1 for door status
    NET_DVR_WEEK_PLAN_CFG struWeekPlan = {0};
    struWeekPlan.dwSize = sizeof(struWeekPlan);
    struWeekPlan.byEnable = 1;//Enable week scheudle

    NET_DVR_SINGLE_PLAN_SEGMENT struSinglePlanSegment = {0};
    LPNET_DVR_SINGLE_PLAN_SEGMENT lpPlanSegment = &struSinglePlanSegment;
    struSinglePlanSegment.byEnable = 1;
    struSinglePlanSegment.byDoorStatus = 3;//Door status: 0-invalid, 1-sleepy, 2-remain open, 3-remain closed.
    struSinglePlanSegment.struTimeSegment.struBeginTime.byHour = 0;//Start time
    struSinglePlanSegment.struTimeSegment.struBeginTime.byMinute = 0;
    struSinglePlanSegment.struTimeSegment.struBeginTime.bySecond = 0;

    struSinglePlanSegment.struTimeSegment.struEndTime.byHour = 23;//End time
    struSinglePlanSegment.struTimeSegment.struEndTime.byMinute = 59;
    struSinglePlanSegment.struTimeSegment.struEndTime.bySecond = 59;

    /*Up to 8 time periods can be set for each day, and you can set different statuses for each time period
    Here only takes setting one period for each day*/

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

    BOOL bRet3 = NET_DVR_SetDVRConfig(lUserID, NET_DVR_SET_WEEK_PLAN_CFG, 1, \
        &struWeekPlan, sizeof(struWeekPlan));
    if (!bRet3)
    {
        printf("Setting week schedule for door status failed, error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(lUserID);
        NET_DVR_Cleanup();
        return;
    }

    //Set holiday group for door status
    CString    m_csGroupName = "door status holiday group 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;//Holiday group 1 links to holiday schedule 1,
                                          //up to 16 holiday schedules can be linked to one holiday group

    BOOL bRet4 = NET_DVR_SetDVRConfig(lUserID, NET_DVR_SET_DOOR_STATUS_HOLIDAY_GROUP, 1, \
        &struHolidayGroup1, sizeof(struHolidayGroup1));
    if (!bRet4)
    {
        printf("Setting holiday group for door status failed, error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(lUserID);
        NET_DVR_Cleanup();
        return;
    }

    //Set holiday schedule for door status
    NET_DVR_HOLIDAY_PLAN_CFG struHolidayPlan = {0};
    struHolidayPlan.dwSize = sizeof(struHolidayPlan);
    struHolidayPlan.byEnable = 1;
    struHolidayPlan.struBeginDate.wYear = 2017;//Holiday start date
    struHolidayPlan.struBeginDate.byMonth = 10;
    struHolidayPlan.struBeginDate.byDay = 1;
    struHolidayPlan.struEndDate.wYear = 2017;//Holiday end date
    struHolidayPlan.struEndDate.byMonth = 10;
    struHolidayPlan.struEndDate.byDay = 7;
    //Copy the week schedule parameters to holiday schedule of door status
    memcpy(struHolidayPlan.struPlanCfg, struWeekPlan.struPlanCfg,
sizeof(NET_DVR_SINGLE_PLAN_SEGMENT)*MAX_DAYS*MAX_TIMESEGMENT_V30);

    BOOL bRet5 = NET_DVR_SetDVRConfig(lUserID, NET_DVR_SET_DOOR_STATUS_HOLIDAY_PLAN, 1, \
        &struHolidayPlan, sizeof(struHolidayPlan));
    if (!bRet5)
    {
        printf("Setting holiday schedule for door status failed, error:%d.\n", NET_DVR_GetLastError());
        NET_DVR_Logout(lUserID);
        NET_DVR_Cleanup();
        return;
    }
    //--------------------------------------
    //Exit
    Sleep(5000);

    //Log out
    NET_DVR_Logout(lUserID);
    //Release SDK resource
    NET_DVR_Cleanup();
    return;
}
```

**What to do next**
Call ***NET_DVR_Logout*** and ***NET_DVR_Cleanup*** to log out and release the resource.

## 2.6 Alarm and Event Receiving

The alarm/event information from the device can be received in third-party platform or system when the alarms are triggered or event occurred. Two modes are available for receiving alarms, including arming mode and listening mode.

**Arming Mode**

The third-party platform connects to device automatically, when the alarm is triggered, the platform sends alarm uploading command to the device, and then the device will upload the alarm to the platform.

**Listening Mode**

When alarm is triggered, the device automatically uploads the alarm, and then the third-party platform receives the uploaded alarm via the configured listening host (listening address and port should be configured). This mode is applicable for multiple devices uploading alarm/event information to one third-party platform without logging in to devices, and the restart of devices will not affect the alarm/event uploading. But a device can only support the configuration of one or two listening addresses and ports.

### 2.6.1 Configure Access Control Event

The access control events include device events, alarm input events, door events, card reader events, card swiping events, and so on. You can configure the linkage types (i.e., event linkage, card linkage, MAC linkage, and person linkage) and linkage actions (e.g., recording, alarm output, buzzing, capture, etc.) of event card linkage to execute the linked actions when the corresponding events occurred (e.g., door open or closed, card swiped, etc.). And then you can receive the event information from event sources in arming or listening mode.

**Before You Start**

- Make sure you have called ***NET_DVR_Init*** to initialize the development environment.
- Make sure you have called ***NET_DVR_Login_V40*** to log in to device.

**Steps**

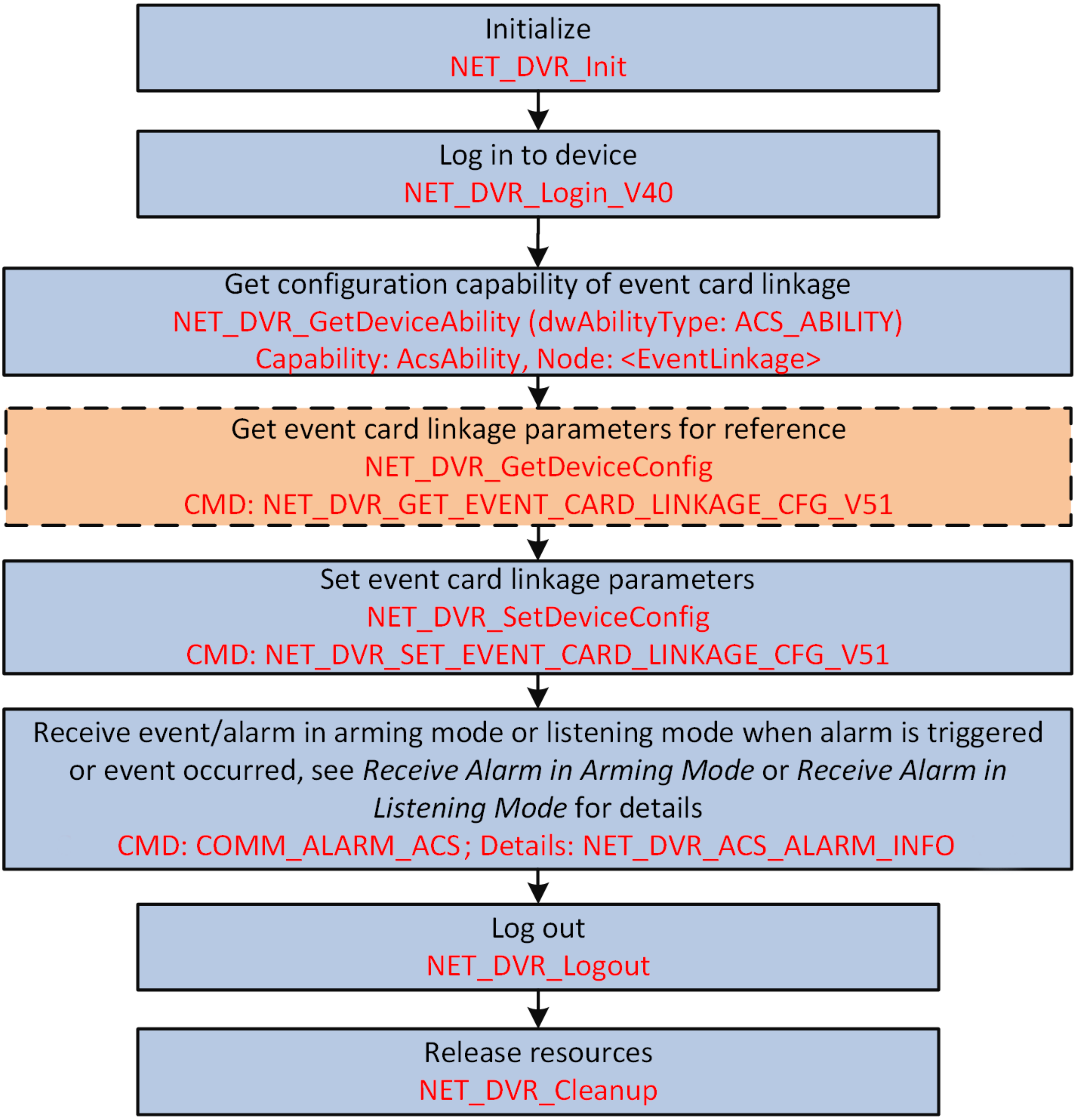
Initialize
NET_DVR_Init
Log in to device
NET_DVR_Login_V40
Get configuration capability of event card linkage
NET_DVR_GetDeviceAbility (dwAbilityType: ACS_ABILITY)
Capability: AcsAbility, Node: <EventLinkage>
Get event card linkage parameters for reference
NET_DVR_GetDeviceConfig
CMD: NET_DVR_GET_EVENT_CARD_LINKAGE_CFG_V51
Set event card linkage parameters
NET_DVR_SetDeviceConfig
CMD: NET_DVR_SET_EVENT_CARD_LINKAGE_CFG_V51
Receive event/alarm in arming mode or listening mode when alarm is triggered or event occurred, see *Receive Alarm in Arming Mode* or *Receive Alarm in Listening Mode* for details
CMD: COMM_ALARM_ACS; Details: NET_DVR_ACS_ALARM_INFO
Log out
NET_DVR_Logout
Release resources
NET_DVR_Cleanup

**Figure 2-9 Programming Flow of Configuring Access Control Event**

1. Call ***NET_DVR_GetDeviceAbility*** , specify the capability type **dwAbilityType** to "ACS_ABILITY", set the input buffer (**pInBuf**) to ***XML_Desc_AcsAbility*** for getting the access control capability to check if setting event or card No. linkage is supported.

   The capability is returned in the message ***XML_AcsAbility*** by the output pointer (**pOutBuf**).

   If the node <**EventLinkage**> is returned, it indicates that setting event or card linkage is supported, and you can continue to perform the following steps.

   Otherwise, setting event or card linkage is not supported, please end this task.
2. **Optional:** Call ***NET_DVR_GetDeviceConfig*** with "NET_DVR_GET_EVENT_CARD_LINKAGE_CFG_V51" (command No.: 2518) and set the condition buffer (**lpInBuffer**) to ***NET_DVR_EVENT_CARD_LINKAGE_COND*** for getting the existing event card linkage parameters for reference.

   **Note**

   The parameter **dwCount** should be set to 1.

   The event card linkage parameters are returned in the structure ***NET_DVR_EVENT_CARD_LINKAGE_CFG_V51*** by output buffer (**lpOutBuffer**).
3. Call ***NET_DVR_SetDeviceConfig*** with "NET_DVR_SET_EVENT_CARD_LINKAGE_CFG_V51" (command No.: 2519), set the condition buffer (**lpInBuffer**) to ***NET_DVR_EVENT_CARD_LINKAGE_COND*** , and set the input parameter (**lpInParamBuffer**) to ***NET_DVR_EVENT_CARD_LINKAGE_CFG_V51*** for setting the event card linkage parameters.

   **Note**

   The parameter **dwCount** should be set to 1.
4. Receive event/alarm in arming mode (see ***Receive Alarm/Event in Arming Mode*** ) or listening mode (see ***Receive Alarm/Event in Listening Mode*** ) when alarm is triggered or event occurred.

   **Note**

   The command to receive access control alarms/events should be set to COMM_ALARM_ACS (command No.: 0x5002) in the alarm callback function ( ***MSGCallBack*** ), and refer to the data structure ***NET_DVR_ACS_ALARM_INFO*** for the alarm/event details.

**Example**

Sample Code for Enabling Capture Linkage

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

BOOL CALLBACK MSesGCallback(LONG lCommand, NET_DVR_ALARMER *pAlarmer, char *pAlarmInfo, DWORD
dwBufLen, void* pUser)
{
   //As the operations with long time comsumption are not allowed in the callback function,
      //do not call the API of HCNetSDK.DLL in the callback function.
   //The following code is for reference only, actually, processing data in the callback function is not suggested.
   //for example, process in the message response function as PostMessage
   switch (lCommand)
   {
   case COMM_ALARM_ACS://Alarm information of access controller
      {
         NET_DVR_ACS_ALARM_INFO struAcsAlarmInfo = {0};
         memcpy(&struAcsAlarmInfo, pAlarmInfo, sizeof(struAcsAlarmInfo));

         char szTime[50] = {0};//Alarm time
         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};//Card No.
         sprintf(szCardNo, "CardNo:%s", (char *)struAcsAlarmInfo.struAcsEventInfo.byCardNo);
         BYTE byCardType = struAcsAlarmInfo.struAcsEventInfo.byCardType;//Card type
         DWORD dwCardReaderNo = struAcsAlarmInfo.struAcsEventInfo.dwCardReaderNo;//Card reader No.
         DWORD dwDoorNo = struAcsAlarmInfo.struAcsEventInfo.dwDoorNo;//Door No.

         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);

            //Save picture
            fSnapPic=fopen(filename,"wb");
            fwrite(struAcsAlarmInfo.pPicData,struAcsAlarmInfo.dwPicDataLen,1,fSnapPic);
            fclose(fSnapPic);
         }
         //Handle other information in the alarm structure as desired...
         break;
      }
   default:
      break;
   }
   return true;
}
void main()
{
   //---------------------------------------
   //Initialize
   NET_DVR_Init();

   //Set connection timeout and reconnection function
```

```
NET_DVR_SetConnectTime(2000, 1);
NET_DVR_SetReconnect(10000, true);
//---------------------------------------
//Log in to device
LONG lUserID;
//Login parameters, including device IP address, user name, password, and so on
NET_DVR_USER_LOGIN_INFO struLoginInfo = {0};
struLoginInfo.bUseAsynLogin = 0; //Synchronous login mode
strcpy(struLoginInfo.sDeviceAddress, "192.168.1.64"); //Device IP address
struLoginInfo.wPort = 8000; //Device service port number
strcpy(struLoginInfo.sUserName, "admin"); //User name
strcpy(struLoginInfo.sPassword, "abcd1234"); //Password
//Device information, output parameter
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;
}

//Set alarm callback function for capture linkage
NET_DVR_SetDVRMessageCallBack_V31(MSesGCallback, NULL);
//Set up channel for uploading alarm information
NET_DVR_SETUPALARM_PARAM struSetupParam={0};
struSetupParam.dwSize=sizeof(NET_DVR_SETUPALARM_PARAM);

LONG  lHandle = NET_DVR_SetupAlarmChan_V41(lUserID,&struSetupParam);
if (lHandle < 0)
{
   printf("NET_DVR_SetupAlarmChan_V41 error: %d\n", NET_DVR_GetLastError());
   NET_DVR_Logout(lUserID);
   NET_DVR_Cleanup();
   return;
}
//---------------------------------------
//Configure capture parameters
NET_DVR_SNAPCFG struSnapCfg = {0};
struSnapCfg.dwSize = sizeof(NET_DVR_SNAPCFG);
struSnapCfg.bySnapTimes = 2;//Capture times: 0-Not capture, non-0-Continuous capture, up to 5 times are allowed.
struSnapCfg.wIntervalTime[0] = 1000;//Time interval of continuous capture, unit: ms, value range: [67,60000]
struSnapCfg.struJpegPara.wPicSize = 5;//Picture resolution: 5-1280*720

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

//---------------------------------------
```

```
   //Set event and card No. linkage parameters
   NET_DVR_EVENT_CARD_LINKAGE_COND struEventCardLinkageCond = {0};
   struEventCardLinkageCond.dwSize = sizeof(NET_DVR_EVENT_CARD_LINKAGE_COND);
   struEventCardLinkageCond.dwEventID = 1;//Event ID, starts from 1, increase when setting different event/card No.
linkages

   NET_DVR_EVENT_CARD_LINKAGE_CFG_V50 struEventCardLinkageCfgV50 = {0};
   struEventCardLinkageCfgV50.dwSize = sizeof(NET_DVR_EVENT_CARD_LINKAGE_CFG_V50);
   struEventCardLinkageCfgV50.byProMode = 0;//Linked event source: 0-event, 1-card No.
   struEventCardLinkageCfgV50.byCapturePic = 1;//Enable capture linkage?: 0-No, 1-Yes

   //Event source ID, 0xffffffff-all, other values: invalid, when the major type is device event;
      //door No., when the major type is door event; card reader ID, when the major type is card reader event;
      //zone alarm input ID or event alarm input ID, when the major type is alarm input event.
   struEventCardLinkageCfgV50.dwEventSourceID = 0xffffffff;

   //Event major type: 0-device event, 1-alarm input event, 2-door event, 3-card reader event
   struEventCardLinkageCfgV50.uLinkageInfo.struEventLinkage.wMainEventType = 2;
   //Event minor type: 10-open door by magnetic switch, here takes capturing triggered by door open as an example.
   struEventCardLinkageCfgV50.uLinkageInfo.struEventLinkage.wSubEventType = 10;

   DWORD dwStatus = 0;
   if (!NET_DVR_SetDeviceConfig(lUserID,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;
   }

   //---------------------------------------
   //Set access controller parameters
   NET_DVR_ACS_CFG struAcsCfg = {0};
   struAcsCfg.dwSize = sizeof(NET_DVR_ACS_CFG);
   struAcsCfg.byUploadCapPic = 1;//Upload picture or not when capture is triggered: 0-No, 1-Yes

   BOOL bRet = NET_DVR_SetDVRConfig(lUserID, NET_DVR_SET_ACS_CFG, 0, \
      &struAcsCfg, sizeof(struAcsCfg));
   if (!bRet)
   {
      printf("NET_DVR_SET_ACS_CFG, error:%d.\n", NET_DVR_GetLastError());
      return;
   }
   //---------------------------------------
   //Wait for 30s for receiving captured picture uploaded by device
   Sleep(30000);
   //Close alarm uploading channel
   if (!NET_DVR_CloseAlarmChan_V30(lHandle))
   {
      printf("NET_DVR_CloseAlarmChan_V30 error, %d\n", NET_DVR_GetLastError());
      NET_DVR_Logout(lUserID);
      NET_DVR_Cleanup();
```

```
        return;
    }
    //Log out
    NET_DVR_Logout(lUserID);
    //Release SDK resource
    NET_DVR_Cleanup();
    return;
}
```

**What to do next**

Call ***NET_DVR_Logout*** and ***NET_DVR_Cleanup*** to log out and release the resource.

## 2.6.2 Receive Alarm/Event in Arming Mode

When the alarm is triggered or the event occurred, the secondarily developed third-party platform can automatically connect and send alarm/event uploading command to the device, and then the device uploads the alarm/event information to the platform for receiving.

**Before You Start**

- Make sure you have called ***NET_DVR_Init*** to initialize the development environment.
- Make sure you have called ***NET_DVR_Login_V40*** to log in to the device.
- Make sure you have configured the alarm/event parameters, refer to the typical alarm/event configurations for details.

**Steps**

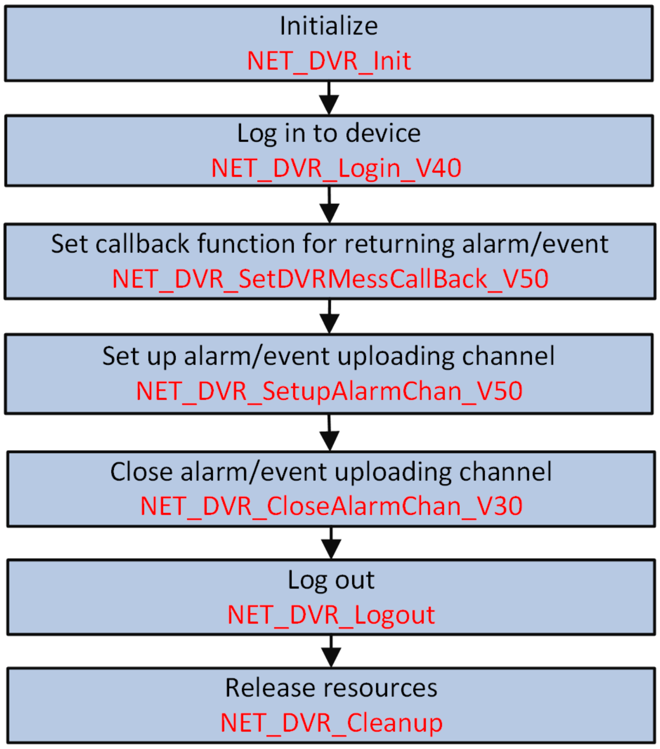
Initialize
NET_DVR_Init
Log in to device
NET_DVR_Login_V40
Set callback function for returning alarm/event
NET_DVR_SetDVRMessCallBack_V50
Set up alarm/event uploading channel
NET_DVR_SetupAlarmChan_V50
Close alarm/event uploading channel
NET_DVR_CloseAlarmChan_V30
Log out
NET_DVR_Logout
Release resources
NET_DVR_Cleanup

**Figure 2-10 Programming Flow of Receiving Alarm/Event in Arming Mode**

1. Call ***NET_DVR_SetDVRMessageCallBack_V50*** to set callback function for returning alarm/event information.

**Note**

- If the configured alarm is triggered or event occurred, the alarm/event information will be uploaded by device and returned in the callback function. You can view the alarm/event and do some processing operations.
- For the integration via device network SDK (HCNetSDK), to receive different types of alarm/event information, the parameter **lCommand** (data type to be uploaded) in the configured callback function should be different (refer to the typical alarm/event configurations). For the integration via text protocol, the **lCommand** should be set to "COMM_ISAPI_ALARM" (command No.: 0x6009) and the input parameter **pAlarmInfo** in the callback function ***MSGCallBack*** should be set to ***NET_DVR_ALARM_ISAPI_INFO*** .

2. Call ***NET_DVR_SetupAlarmChan_V50*** to set up uploading channel.
3. Call ***NET_DVR_CloseAlarmChan_V30*** to close uploading channel and stop receiving alarm or event information.

**Example**

Sample Code of Receiving Alarm or Event in Arming Mode

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

void main() {
 //---------------------------------------
 // Initialize
 NET_DVR_Init();
 //Set connection time and reconnection time
 NET_DVR_SetConnectTime(2000, 1);
 NET_DVR_SetReconnect(10000, true);
 //---------------------------------------
 // Log in to device
 LONG lUserID;
 //Login parameters, including device IP address, user name, password, and so on.
 NET_DVR_USER_LOGIN_INFO struLoginInfo = {0};
 struLoginInfo.bUseAsynLogin = 0; //Synchronous login mode
 strcpy(struLoginInfo.sDeviceAddress, "192.0.0.64"); //Device IP address
 struLoginInfo.wPort = 8000; //Service port No.
 strcpy(struLoginInfo.sUserName, "admin"); //User name
 strcpy(struLoginInfo.sPassword, "abcd1234"); //Password
 //Device information, output parameter
 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;
 }

 //Set alarm callback function
 NET_DVR_SetDVRMessageCallBack_V50(0, MessageCallbackNo1, NULL);
 NET_DVR_SetDVRMessageCallBack_V50(1, MessageCallbackNo2, NULL);

 //Enable arming
 NET_DVR_SETUPALARM_PARAM_V50 struSetupParamV50={0};
 struSetupParamV50.dwSize=sizeof(NET_DVR_SETUPALARM_PARAM_V50);
 //Alarm category to be uploaded
 struSetupParamV50.byAlarmInfoType=1;
 //Arming level
 struSetupParamV50.byLevel=1;

 char szSubscribe[1024] = {0};
 //The following code is for alarm subscription (subscribe all)
 memcpy(szSubscribe, "<SubscribeEvent version=\"2.0\" xmlns=\"http://www.isapi.org/ver20/XMLSchema\">\r
\n<eventMode>all</eventMode>\r\n", 1024);
 LONG lHandle = -1;
 if (0 == strlen(szSubscribe))
 {
  //Arm
  lHandle = NET_DVR_SetupAlarmChan_V50(lUserID, &struSetupParamV50, NULL, strlen(szSubscribe));
 }
 else
 {
  //Subscribe
  LlHandle = NET_DVR_SetupAlarmChan_V50(lUserID, &struSetupParamV50, szSubscribe, strlen(szSubscribe));
 }

 if (lHandle < 0)
 {
  printf("NET_DVR_SetupAlarmChan_V50 error, %d\n", NET_DVR_GetLastError());
  NET_DVR_Logout(lUserID);
  NET_DVR_Cleanup();
  return;
 }

 Sleep(20000);
 //Disarm the uploading channel
 if (!NET_DVR_CloseAlarmChan_V30(lHandle))
 {
   printf("NET_DVR_CloseAlarmChan_V30 error, %d\n", NET_DVR_GetLastError());
   NET_DVR_Logout(lUserID);
   NET_DVR_Cleanup();
   return;
 }

 //Log out
 NET_DVR_Logout(lUserID);
 //Release resources
 NET_DVR_Cleanup();
 return;
}
```

**What to do next**

Call ***NET_DVR_Logout*** and ***NET_DVR_Cleanup*** to log out and release resources.

### 2.6.3 Receive Alarm/Event in Listening Mode

When alarm is triggered or event occurred, the device uploads the alarm/event information automatically, so you can configure the listening address and port for listening and receiving the alarm/event in the secondarily developed third-part platform.

**Before You Start**

- Make sure you have called ***NET_DVR_Init*** to initialize the development environment.
- Make sure you have configured the alarm/event parameters, refer to the typical alarm/event configurations for details.

**Steps**

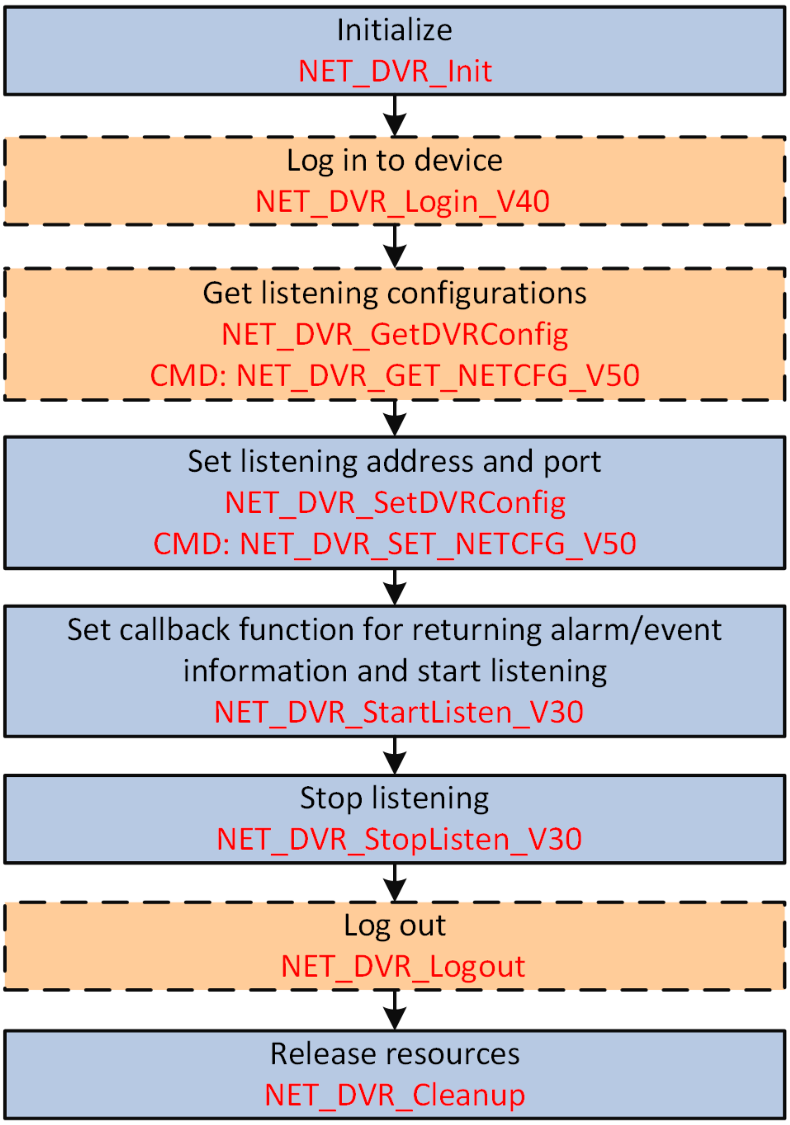

**Figure 2-11 Programming Flow of Receiving Alarm/Event in Listening Mode**

1. **Optional:** Call ***NET_DVR_Login_V40*** to log in to device.
2. **Optional:** Call ***NET_DVR_GetDVRConfig*** with "NET_DVR_GET_NETCFG_V50" (command No.: 1015) to get the existing listening configurations (i.e., listening address and port) for reference.

   The listening parameters are retruned in the structure ***NET_DVR_NETCFG_V50*** by the output parameter pointer **lpOutBuffer**.
3. Call ***NET_DVR_SetDVRConfig*** with "NET_DVR_SET_NETCFG_V50" (command No.: 1016) and specify the input parameter pointer **lpInBuffer** to the structure ***NET_DVR_NETCFG_V50*** for setting the listening address and port.
4. Call ***NET_DVR_StartListen_V30*** to set callback function for returning alarm/event information and start the listening.

**Note**

For the integration via device network SDK (HCNetSDK), to receive different types of alarm/ event information, the parameter **lCommand** (data type to be uploaded) in the configured callback function should be different (refer to the typical alarm/event configurations). For the integration via text protocol, the **lCommand** should be set to "COMM_ISAPI_ALARM" and the input parameter **pAlarmInfo** in the callback function ***MSGCallBack*** should be set to ***NET_DVR_ALARM_ISAPI_INFO*** .

The alarm/event information is automatically uploaded by the device when the configured alarm is triggered or event occurred, and the third-party platform or system gets the alarm/ event information from the configured callback function.

5. Call ***NET_DVR_StopListen_V30*** to stop listening and receiving alarm or event information.

**Example**

Sample Code of Receiving Alarm/Event in Listening Mode

```
#include <stdio.h>
#include <iostream>
#include "Windows.h"
#include "HCNetSDK.h"
using namespace std;
void main() {
  //---------------------------------------
  // Initialize
  NET_DVR_Init();
  //Set connection time and reconnection time
  NET_DVR_SetConnectTime(2000, 1);
  NET_DVR_SetReconnect(10000, true);
  //---------------------------------------
  // Log in to device
  LONG lUserID;
  NET_DVR_DEVICEINFO_V30 struDeviceInfo;
  lUserID = NET_DVR_Login_V30("172.0.0.100", 8000, "admin", "12345", &struDeviceInfo);
  if (lUserID < 0)
  {
      printf("Login error, %d\n", NET_DVR_GetLastError());
      NET_DVR_Cleanup();
      return;
  }
  //Enable listening
  LONG lHandle;
  lHandle = NET_DVR_StartListen_V30(NULL,7200, MessageCallback, NULL);
  if (lHandle < 0)
  {
     printf("NET_DVR_StartListen_V30 error, %d\n", NET_DVR_GetLastError());
     NET_DVR_Logout(lUserID);
     NET_DVR_Cleanup();
     return;
  }
  Sleep(5000);
  //Disable listening
  if (!NET_DVR_StopListen_V30(lHandle))
  {
     printf("NET_DVR_StopListen_V30 error, %d\n", NET_DVR_GetLastError());
     NET_DVR_Logout(lUserID);
     NET_DVR_Cleanup();
     return;
  }
  //Log out
  NET_DVR_Logout(lUserID);
  //Release SDK resource
  NET_DVR_Cleanup();
  return;
}
```

**What to do next**

Call ***NET_DVR_Logout*** (if logged in) and ***NET_DVR_Cleanup*** to log out and release resources.

### 2.6.4 Search for Access Control Events

If the access control alarms or events are received and stored in the third-party platform, you can search for the alarms or events by setting different search conditions.

**Before You Start**

- Make sure you have called ***NET_DVR_Init*** to initialize the development environment.
- Make sure you have called ***NET_DVR_Login_V40*** to log in to device.

**Steps**

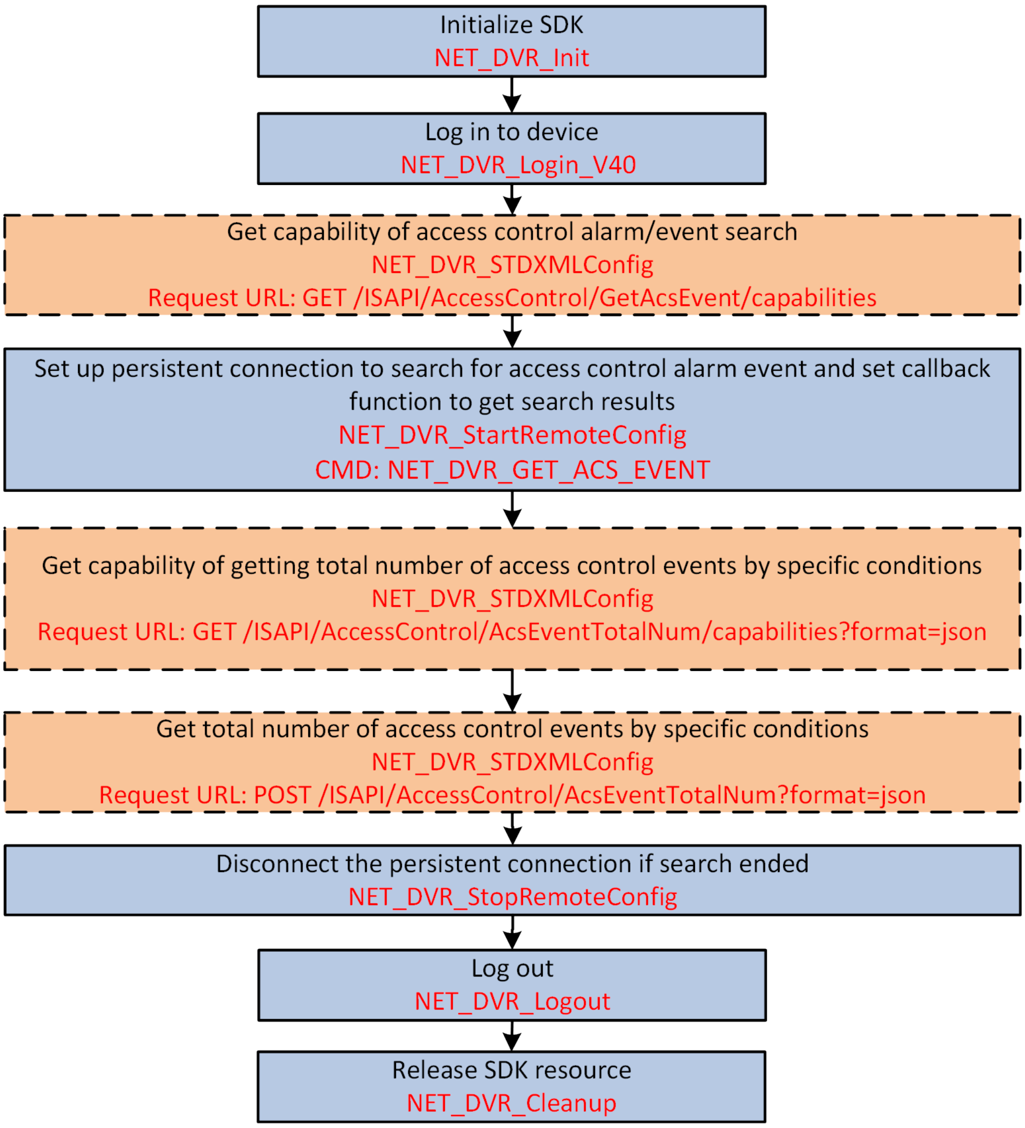
Initialize SDK
NET_DVR_Init
Log in to device
NET_DVR_Login_V40
Get capability of access control alarm/event search
NET_DVR_STDXMLConfig
Request URL: GET /ISAPI/AccessControl/GetAcsEvent/capabilities
Set up persistent connection to search for access control alarm event and set callback function to get search results
NET_DVR_StartRemoteConfig
CMD: NET_DVR_GET_ACS_EVENT
Get capability of getting total number of access control events by specific conditions
NET_DVR_STDXMLConfig
Request URL: GET /ISAPI/AccessControl/AcsEventTotalNum/capabilities?format=json
Get total number of access control events by specific conditions
NET_DVR_STDXMLConfig
Request URL: POST /ISAPI/AccessControl/AcsEventTotalNum?format=json
Disconnect the persistent connection if search ended
NET_DVR_StopRemoteConfig
Log out
NET_DVR_Logout
Release SDK resource
NET_DVR_Cleanup

**Figure 2-12 Programming Flow of Searching for Access Control Events**

1. **Optional:** Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/GetAcsEvent/capabilities*** for getting the capability of access control alarm/event search to know the details or notices about search.

**Note**

To check whether the device supports searching for access control events, you can call ***NET_DVR_GetDeviceAbility*** , set the capability type **dwAbilityType** to "ACS_ABILITY" (macro definition value: 0x801), and set the input parameter pointer **pInBuf** to the message ***XML_Desc_AcsAbility*** for getting the access control capability.

The capaility is returned in the message ***XML_AcsAbility*** by the output parameter pointer **pOutBuf**. The related node is <**isSupportGetDeviceEvent**>.

---

The capability message ***XML_Cap_GetAcsEvent*** is returned.

2. Call ***NET_DVR_StartRemoteConfig*** with "NET_DVR_GET_ACS_EVENT" (command No: 2514) and set **lpInBuffer** to ***NET_DVR_ACS_EVENT_COND*** for setting up persistent connection and set callback function ( ***fRemoteConfigCallback*** ).

   The access control event details are returned in the structure ***NET_DVR_ACS_EVENT_CFG*** by the output buffer (**lpBuffer**) of callback function.
3. **Optional:** Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/AcsEventTotalNum/capabilities?format=json*** to get the capability of getting total number of access control events by specific conditions.
4. **Optional:** Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: POST ***/ISAPI/AccessControl/AcsEventTotalNum?format=json*** and set **lpInBuffer** to the message ***JSON_AcsEventTotalNumCond*** for getting the total number of access control events by specific conditions.
5. Call ***NET_DVR_StopRemoteConfig*** to disconnect the persistent connection and finish searching.

**Example**

Sample Code of Searching for Access Control Event

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

BOOL CALLBACK MSesGCallback(LONG lCommand, NET_DVR_ALARMER *pAlarmer, char *pAlarmInfo, DWORD
dwBufLen, void* pUser)
{
  //As the operations with long time consumption are not allowed in the callback function,
    //do not call the API of HCNetSDK.DLL in the callback function.
  //The following code is for reference only, actually, processing data in the callback function is not suggested.
  //for example, process in the message response function as PostMessage
  switch (lCommand)
  {
  case COMM_ALARM_ACS://Alarm information of access controller
    {
      NET_DVR_ACS_ALARM_INFO struAcsAlarmInfo = {0};
      memcpy(&struAcsAlarmInfo, pAlarmInfo, sizeof(struAcsAlarmInfo));
      //Handle other information in the alarm structure as desired...
      break;
    }
  case COMM_PASSNUM_INFO_ALARM://Number of passed persons
    {
      NET_DVR_PASSNUM_INFO_ALARM struPassnumInfo = {0};
      memcpy(&struPassnumInfo, pAlarmInfo, sizeof(struPassnumInfo));
      //Handle other information in the alarm structure as desired...
      break;
    }
```

```
   default:
      break;
   }
   return true;
}
void main()
{
   //--------------------------------------
   //Initialize
   NET_DVR_Init();

   //Set connection timeout and reconnection function
   NET_DVR_SetConnectTime(2000, 1);
   NET_DVR_SetReconnect(10000, true);
   //--------------------------------------
   //Log in to device
   LONG lUserID;
   //Login parameters, including device IP address, user name, password, and so on
   NET_DVR_USER_LOGIN_INFO struLoginInfo = {0};
   struLoginInfo.bUseAsynLogin = 0; //Synchronous login mode
   strcpy(struLoginInfo.sDeviceAddress, "192.168.1.64"); //Device IP address
   struLoginInfo.wPort = 8000; //Device service port number
   strcpy(struLoginInfo.sUserName, "admin"); //User name
   strcpy(struLoginInfo.sPassword, "abcd1234"); //Password

   //Device information, output parameter
   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;
   }

   //Set alarm callback function for card swiping event
   NET_DVR_SetDVRMessageCallBack_V31(MSesGCallback, NULL);
   //Set up channel for uploading alarm information
   NET_DVR_SETUPALARM_PARAM struSetupParam={0};
   struSetupParam.dwSize=sizeof(NET_DVR_SETUPALARM_PARAM);

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

   //Wait for 60s for receiving captured picture uploaded by device
```

```
    Sleep(60000);
    //Close alarm uploading channel
    if (!NET_DVR_CloseAlarmChan_V30(lHandle))
    {
      printf("NET_DVR_CloseAlarmChan_V30 error, %d\n", NET_DVR_GetLastError());
      NET_DVR_Logout(lUserID);
      NET_DVR_Cleanup();
      return;
    }
    //Log out
    NET_DVR_Logout(lUserID);
    //Release SDK resource
    NET_DVR_Cleanup();
    return;
}
```

**What to do next**

Call ***NET_DVR_Logout*** and ***NET_DVR_Cleanup*** to log out and release the resource.

## 2.7 Remotely Control Door

You can remotely control door, i.e., open, close, remain open, and remain close, by the access controller.

Call ***NET_DVR_ControlGateway*** to remotely control door.

**Note**

- If the device is configured with multi-factor authentication mode, this API is available only when the parameter **dwGroupNo** in the structure ***NET_DVR_GROUP_COMBINATION_INFO_V50*** is set to "0xffffffff".
- Before controlling door by calling the above API, you should call ***NET_DVR_Init*** and ***NET_DVR_Login_V40*** to initialize the resources and log in to device.
- After controlling door, you should call ***NET_DVR_Logout*** and ***NET_DVR_Cleanup*** to log out and release the resource.

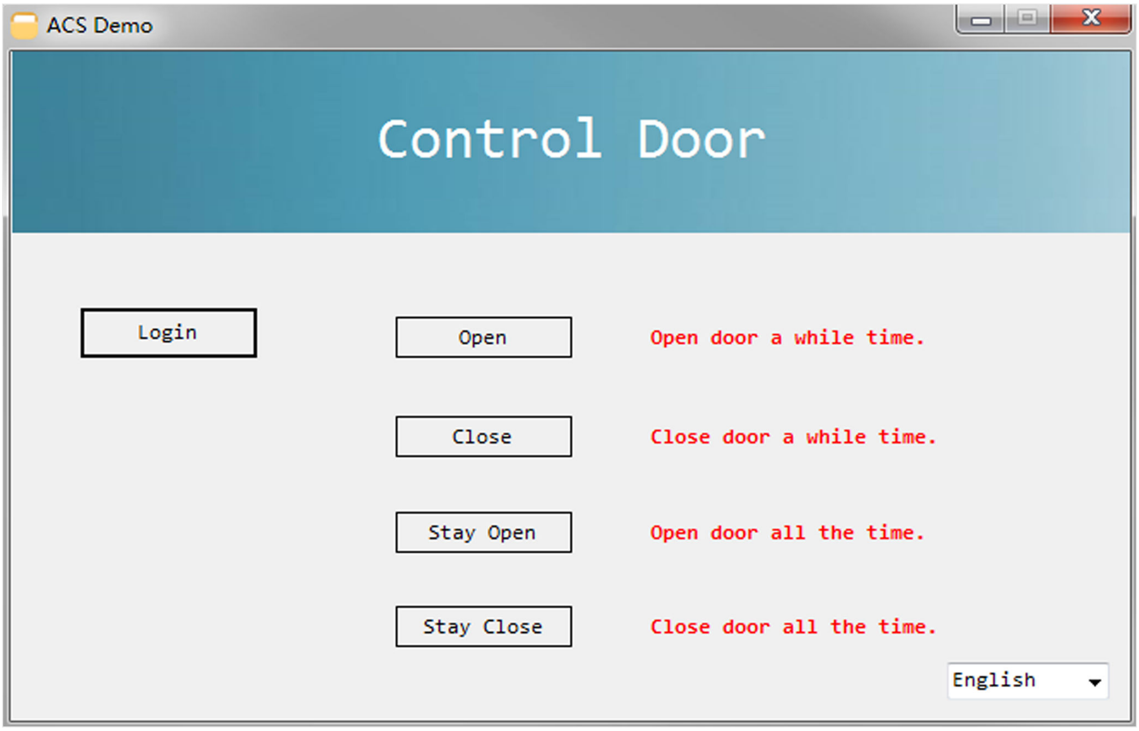

**Figure 2-13 Door Control Example Page**

**Example**

Sample Code for Remotely Controlling Door

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

void main()
{
   //---------------------------------------
   //Initialize
   NET_DVR_Init();

   //Set connection timeout and reconnection function
   NET_DVR_SetConnectTime(2000, 1);
   NET_DVR_SetReconnect(10000, true);

   //---------------------------------------
   //Initialize
   NET_DVR_Init();

   //Set connection timeout and reconnection function
   NET_DVR_SetConnectTime(2000, 1);
```

```
    NET_DVR_SetReconnect(10000, true);
    //--------------------------------------
    //Log in to device
    LONG lUserID;
    //Login parameters, including device IP address, user name, password, and so on
    NET_DVR_USER_LOGIN_INFO struLoginInfo = {0};
    struLoginInfo.bUseAsynLogin = 0; //Synchronous login mode
    strcpy(struLoginInfo.sDeviceAddress, "192.168.1.64"); //Device IP address
    struLoginInfo.wPort = 8000; //Device service port number
    strcpy(struLoginInfo.sUserName, "admin"); //User name
    strcpy(struLoginInfo.sPassword, "abcd1234"); //Password

    //Device information, output parameter
    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;
    }

    //Open door, take door 1 as an example
    BOOL bRet;
    LONG lGatewayIndex = 1;//Access controller No., starts from 1, -1: control all doors
    DWORD dwStaic = 1;//Command No.: 0-Close, 1-Open, 2-Remain Open, 3-Remain Closed

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

    //--------------------------------------
    //Exit
    Sleep(5000);

    //Log out
    NET_DVR_Logout(lUserID);
    //Release SDK resource
    NET_DVR_Cleanup();
    return;
}
```

## 2.8 Status Monitoring

You can get the status of access controllers, turnstiles, and other devices to monitor their operation.

### Working Status of Access Controller

Call ***NET_DVR_GetDVRConfig*** with the command "NET_DVR_GET_ACS_WORK_STATUS_V50" (command No.: 2180).
The working status is returned in the structure ***NET_DVR_ACS_WORK_STATUS_V50*** by the output buffer **lpOutBuffer**.

**Note**

To check whether getting working status of the access controller is supported, you can call ***NET_DVR_GetDeviceAbility*** , set the capability type **dwAbilityType** to "ACS_ABILITY" (macro definition value: 0x801), and set the input parameter pointer **pInBuf** to the message ***XML_Desc_AcsAbility*** for getting the access control capability.
The capability is returned in the message ***XML_AcsAbility*** by the output parameter pointer **pOutBuf**. The related node is <**AcsWorkStatus**>.

### General Turnstile Status

| Function | Description |
|---|---|
| Get Capability of Getting General Turnstile Status | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/GateStatus/capabilities*** .<br>The capability is returned in the message ***XML_Cap_GateStatus*** by **lpOutBuffer**. |
| Get General Turnstile Status | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/GateStatus*** .<br>The status is returned in the message ***XML_GateStatus*** by **lpOutBuffer**. |

**Note**

To check whether the device supports getting general turnstile status, you can call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET x ***/ISAPI/AccessControl/capabilities*** to get the access control capability.
The capability is returned in the message ***XML_Cap_AccessControl*** by **lpOutBuffer**. If this function is supported by the device, the node <**isSupportGateStatus**> will be returned in the message and its value is "true".

### Status of Active Infrared Intrusion Detector of Turnstile

| Function | Description |
|---|---|
| Get Capability of Getting Status of Active Infrared Intrusion Detector of Turnstile | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/GateIRStatus/capabilities*** .<br>The capability is returned in the message ***XML_Cap_GateIRStatus*** by **lpOutBuffer**. |
| Get Status of Active Infrared Intrusion Detector of Turnstile | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/GateIRStatus*** .<br>The status is returned in the message ***XML_GateIRStatus*** by **lpOutBuffer**. |

**Note**

To check whether the device supports getting the status of the active infrared intrusion detector of the turnstile, you can call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/capabilities*** to get the access control capability.
The capability is returned in the message ***XML_Cap_AccessControl*** by **lpOutBuffer**. If this function is supported by the device, the node <**isSupportGateIRStatus**> will be returned in the message and its value is "true".

### Turnstile Component Status

| Function | Description |
|---|---|
| Get Capability of Getting Related Components' Status of Turnstile | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/GateRelatedPartsStatus/capabilities*** .<br>The capability is returned in the message ***XML_Cap_GateRelatedPartsStatus*** by **lpOutBuffer**. |
| Get Related Components' Status of Turnstile | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/GateRelatedPartsStatus*** .<br>The status is returned in the message ***XML_GateRelatedPartsStatus*** by **lpOutBuffer**. |

**Note**

To check whether the device supports getting related components' status of the turnstile, you can call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/capabilities*** to get the access control capability.

The capability is returned in the message ***XML_Cap_AccessControl*** by **lpOutBuffer**. If this function is supported by the device, the node <**isSupportGateRelatedPartsStatus**> will be returned in the message and its value is "true".

### 2.8.1 Configure Attendance Status

The time and attendance refers to tracking and monitoring when employees start and stop working, and working hours (including late arrivals, early departures, time taken on breaks and absenteeism, etc.). You can set the manual or automatic time and attendance mode, or disable the attendance mode. You can also set check in, check out, break out, break in, overtime in, or overtime out to manually change the attendance status as needed.

**Before You Start**

- Make sure you have called ***NET_DVR_Init*** to initialize the development environment.
- Make sure you have called ***NET_DVR_Login_V40*** to log in to the device.
- Make sure you have added at least one card, refer to ***Manage Card Information*** for details.

**Steps**

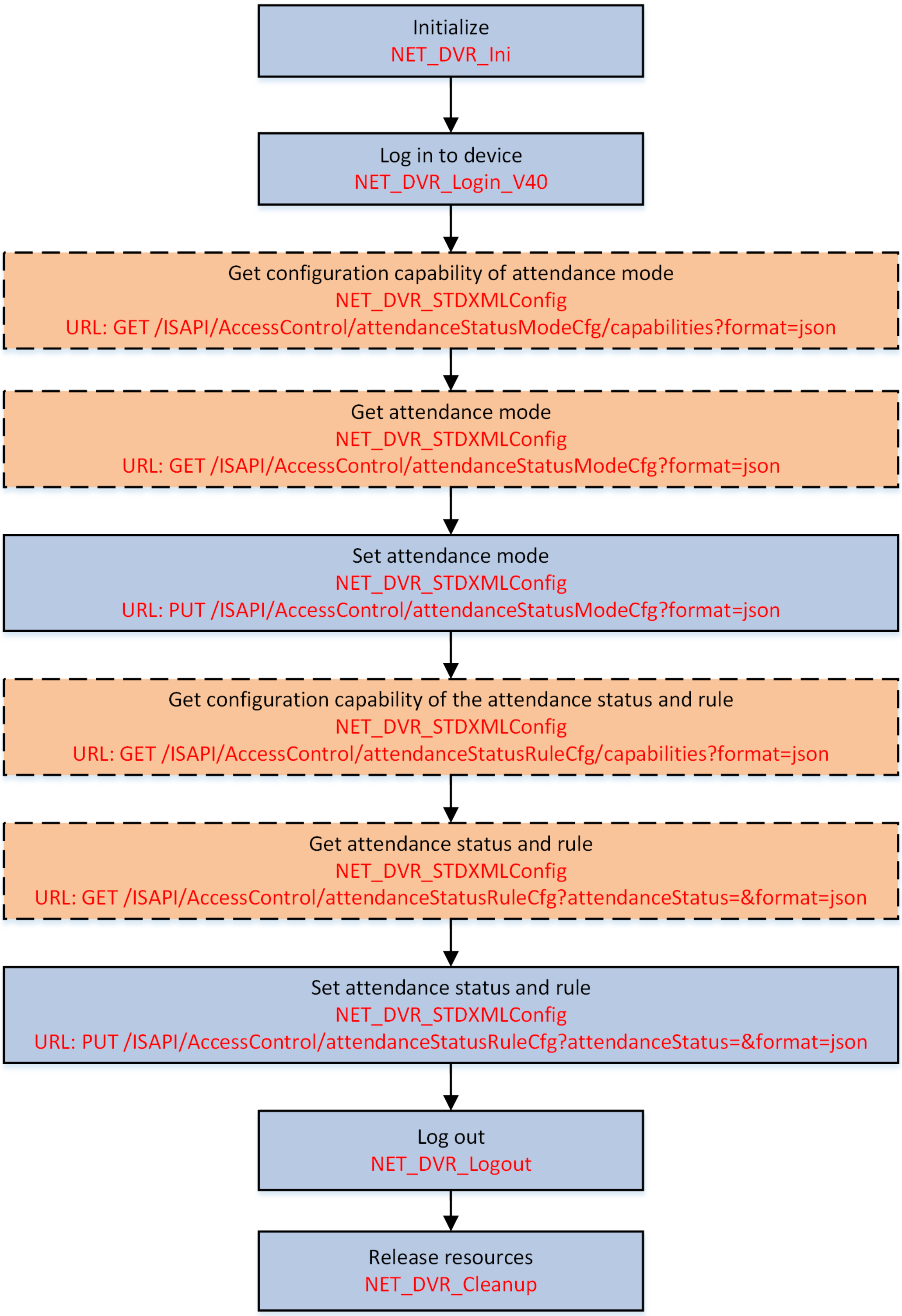
Initialize
NET_DVR_Ini
Log in to device
NET_DVR_Login_V40
Get configuration capability of attendance mode
NET_DVR_STDXMLConfig
URL: GET /ISAPI/AccessControl/attendanceStatusModeCfg/capabilities?format=json
Get attendance mode
NET_DVR_STDXMLConfig
URL: GET /ISAPI/AccessControl/attendanceStatusModeCfg?format=json
Set attendance mode
NET_DVR_STDXMLConfig
URL: PUT /ISAPI/AccessControl/attendanceStatusModeCfg?format=json
Get configuration capability of the attendance status and rule
NET_DVR_STDXMLConfig
URL: GET /ISAPI/AccessControl/attendanceStatusRuleCfg/capabilities?format=json
Get attendance status and rule
NET_DVR_STDXMLConfig
URL: GET /ISAPI/AccessControl/attendanceStatusRuleCfg?attendanceStatus=&format=json
Set attendance status and rule
NET_DVR_STDXMLConfig
URL: PUT /ISAPI/AccessControl/attendanceStatusRuleCfg?attendanceStatus=&format=json
Log out
NET_DVR_Logout
Release resources
NET_DVR_Cleanup

**Figure 2-14 Programming Flow of Configuring Attendance Status**

1. **Optional:** Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/attendanceStatusModeCfg/capabilities?format=json*** to get the configuration capability of the attendance mode and know the supported attendance modes.

   The configuration capability is returned in the message ***JSON_Cap_AttendanceStatusModeCfg*** by **lpOutBuffer**.
2. **Optional:** Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/attendanceStatusModeCfg?format=json*** to get the default or configured attendance mode for reference.

   The attendance mode is returned in the message ***JSON_AttendanceStatusModeCfg*** by **lpOutBuffer**.
3. Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: PUT ***/ISAPI/AccessControl/attendanceStatusModeCfg?format=json*** and set **lpInBuffer** to the message ***JSON_AttendanceStatusModeCfg*** to configure the attendance mode.
4. **Optional:** Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/attendanceStatusRuleCfg/capabilities?format=json*** to get the configuration capability of the attendance status and rule and know the supported attendance status and rules.

   The configuration capability of the attendance status and rule is returned in the message ***JSON_Cap_AttendanceStatusRuleCfg*** by **lpOutBuffer**.
5. **Optional:** Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: GET ***/ISAPI/AccessControl/attendanceStatusRuleCfg?attendanceStatus=&format=json*** to get the default or configured attendance status and rule for reference.

   The attendance status and rules are returned in the message ***JSON_AttendanceStatusRuleCfg*** by **lpOutBuffer**.
6. Call ***NET_DVR_STDXMLConfig*** to pass through the request URL: PUT ***/ISAPI/AccessControl/attendanceStatusRuleCfg?attendanceStatus=&format=json*** and set **lpInBuffer** to the message ***JSON_AttendanceStatusRuleCfg*** to configure the attendance status and rule.

**What to do next**

Call ***NET_DVR_Logout*** and ***NET_DVR_Cleanup*** to log out of the device and release the resources.

## 2.9 Turnstile Settings

The turnstile is a lane management device that is used to manage the entrance and exit of people in places such as office buildings, subways, residences, and so on. By adopting the turnstile integrated with the access control system, people should authenticate to pass through the lane by swiping ID card, scanning QR code, etc. With alarm linkage and output configured, unauthorized entrance and exit can trigger and upload alarms. Common turnstiles include swing barrier, flap barrier, tripod turnstile, and so on.

### 2.9.1 Lane Controller Settings

The lane controller, including master lane controller and slave lane controller, is mainly used to control infrared or motor components of the turnstile.

#### Basic Configuration

| Function | Description |
|---|---|
| Get Configuration Capability | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/ChannelControllerCfg/capabilities*** .<br>The configuration capability is returned in the message ***XML_Cap_ChannelControllerCfg*** by **lpOutputParam**. |
| Get Parameters | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/ChannelControllerCfg*** .<br>The parameters are returned in the message ***XML_ChannelControllerCfg*** by **lpOutputParam**. |
| Set Parameters | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: PUT ***/ISAPI/AccessControl/ChannelControllerCfg*** and set **lpInputParam** to the message ***XML_ChannelControllerCfg*** . |

**Note**

To check whether configuring lane controller is supported, you can call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/capabilities*** to get the access control capability.

The access control capability is returned in the message ***XML_Cap_AccessControl*** by **lpOutputParam**. If configuring lane controller is supported, the node <**isSupportChannelControllerCfg**> will be returned and its value is "true".

#### Device Type Configuration

| Function | Description |
|---|---|
| Get Configuration Capability of Device Type | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/channelControllerTypeCfg/capabilities?format=json*** .<br>The configuration capability is returned in the message ***JSON_ChannelControllerTypeCfgCap*** by **lpOutputParam**. |
| Get Device Type | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/channelControllerTypeCfg?format=json*** . |

| Function | Description |
|---|---|
| | The device type parameters are returned in the message ***JSON_ChannelControllerTypeCfg*** by **lpOutputParam**. |
| Set Device Type | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: PUT ***/ISAPI/AccessControl/channelControllerTypeCfg?format=json*** and set **lpInputParam** to the message ***JSON_ChannelControllerTypeCfg*** . |

**Note**

To check whether configuring device type of the lane controller is supported, you can call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/capabilities*** to get the access control capability.
The access control capability is returned in the message ***XML_Cap_AccessControl*** by **lpOutputParam**. If configuring device type of the lane controller is supported, the node <**isSupportChannelControllerTypeCfg**> will be returned and its value is "true".

## Alarm Linkage Configuration

| Function | Description |
|---|---|
| Get Configuration Capability of Alarm Linkage | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/ChannelControllerAlarmLinkage/capabilities*** .<br>The configuration capability is returned in the message ***XML_Cap_ChannelControllerAlarmLinkage*** by **lpOutputParam**. |
| Get Alarm Linkage Parameters | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/ChannelControllerAlarmLinkage*** .<br>The parameters are returned in the message ***XML_ChannelControllerAlarmLinkage*** by **lpOutputParam**. |
| Set Alarm Linkage Parameters | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: PUT ***/ISAPI/AccessControl/ChannelControllerAlarmLinkage*** and set **lpInputParam** to the message ***XML_ChannelControllerAlarmLinkage*** . |

**Note**

To check whether the device supports configuring alarm linkage of the lane controller, you can call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/capabilities*** to get the access control capability.

The capability is returned in the message ***XML_Cap_AccessControl*** by **lpOutputParam**. If this function is supported by the device, the node <**isSupportChannelControllerAlarmLinkage**> will be returned in the message and its value is "true".

## Alarm Output Configuration

| Function | Description |
|---|---|
| Get Configuration Capability of Alarm Output | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/ChannelControllerAlarmOut/capabilities*** .<br>The configuration capability is returned in the message ***XML_Cap_ChannelControllerAlarmOut*** by **lpOutputParam**. |
| Get Alarm Output Parameters | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/ChannelControllerAlarmOut?controllerType=&alarmOutNo=*** .<br>The parameters are returned in the message ***XML_ChannelControllerAlarmOut*** by **lpOutputParam**. |
| Set Alarm Output Parameters | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: PUT ***/ISAPI/AccessControl/ChannelControllerAlarmOut?controllerType=&alarmOutNo=*** and set **lpInputParam** to the message ***XML_ChannelControllerAlarmOut*** . |

**Note**

To check whether the device supports configuring alarm output of the lane controller, you can call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/capabilities*** to get the access control capability.
The capability is returned in the message ***XML_Cap_AccessControl*** by **lpOutputParam**. If this function is supported by the device, the node <**isSupportChannelControllerAlarmOut**> will be returned in the message and its value is "true".

## Alarm Output Control

| Function | Description |
|---|---|
| Get Capability of Controlling Alarm Output | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/ChannelControllerAlarmOutControl/capabilities*** . |

| Function | Description |
|---|---|
| | The capability is returned in the message ***XML_Cap_ChannelControllerAlarmOutControl*** by **lpOutputParam**. |
| Control Alarm Output | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: PUT ***/ISAPI/AccessControl/ChannelControllerAlarmOutControl*** and set **lpInputParam** to the message ***XML_ChannelControllerAlarmOutControl*** . |

**Note**

To check whether the device supports controlling alarm output of the lane controller, you can call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/capabilities*** for getting the access control capability.
The capability is returned in the message ***XML_Cap_AccessControl*** by **lpOutputParam**. If this function is supported, the node <**isSupportChannelControllerAlarmOutControl**> will be returned in the message and its value is "true".

### 2.9.2 Main Controller Settings

The main controller is mainly used to authenticate access permissions, connect to peripherals, and communicate with the lane controller and the upper-level platform.

| Function | Description |
|---|---|
| Upload Audio File of Main Controller | 1. Call ***NET_DVR_GetDeviceAbility*** , set the capability type **dwAbilityType** to "ACS_ABILITY" (macro definition value: 0x801), and set the input parameter pointer **pInBuf** to the message ***XML_Desc_AcsAbility*** for getting the access control capability to check whether the device supports uploading the audio file of the main controller.<br>The capability is returned in the message ***XML_AcsAbility*** by the output parameter pointer **pOutBuf**. The related node is <**UploadRightControllerAudio**>.<br>2. Call ***NET_DVR_UploadFile_V40*** , set **dwUploadType** to "UPLOAD_RIGHT_CONTROLLER_AUDIO" (macro definition value: 42), and set **lpInBuffer** to the structure ***NET_DVR_RIGHT_CONTROLLER_AUDIO_PARAM*** for uploading the audio file of the main controller.<br>3. Call ***NET_DVR_GetUploadState*** to get file uploading progress. |

| Function | Description |
|---|---|
| | **Note**<br>If the progress is "4" (network disconnected), you should stop uploading first and perform step 2 again when the network is restored.<br>4. Call ***NET_DVR_UploadClose*** to stop uploading the audio file. |
| Download Audio File of Main Controller | 1. Call ***NET_DVR_GetDeviceAbility*** , set the capability type **dwAbilityType** to "ACS_ABILITY" (macro definition value: 0x801), and set the input parameter pointer **pInBuf** to the message ***XML_Desc_AcsAbility*** for getting the access control capability to check whether the device supports downloading the audio file of the main controller.<br>The capability is returned in the message ***XML_AcsAbility*** by the output parameter pointer **pOutBuf**. The related node is <**DownloadRightControllerAudio**>.<br>2. Call ***NET_DVR_StartDownload*** , set **dwDownloadType** to "NET_SDK_DOWNLOAD_RIGHT_CONTROLLER_AUDIO" (macro definition value: 24), and set **lpInBuffer** to the structure ***NET_DVR_RIGHT_CONTROLLER_AUDIO_PARAM*** for downloading the audio file of the main controller.<br>3. Call ***NET_DVR_GetDownloadState*** to get file downloading progress.<br>**Note**<br>If the progress is "4" (network disconnected), you should stop downloading first and perform step 2 again when the network is restored.<br>4. Call ***NET_DVR_StopDownload*** to stop downloading the audio file. |
| Get Configuration Capability of Audio File of Main Controller | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/RightControllerAudio/capabilities*** .<br>The configuration capability is returned in the message ***XML_Cap_RightControllerAudio*** by **lpOutBuffer**. |
| Get Audio File Parameters of Main Controller | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/RightControllerAudio/<ID>*** .<br>The parameters are returned in the message ***XML_RightControllerAudio*** by **lpOutBuffer**. |

| Function | Description |
|---|---|
| Set Audio File Parameters of Main Controller | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: PUT ***/ISAPI/AccessControl/RightControllerAudio/<ID>*** and set **lpInBuffer** to the message ***XML_RightControllerAudio*** . |
| Delete Audio File of Main Controller | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: DELETE ***/ISAPI/AccessControl/RightControllerAudio/<ID>*** .<br>**Note**<br>To check whether the device supports configuring audio file parameters of the main controller, you can call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/capabilities*** for getting the access control capability.<br>The capability is returned in the message ***XML_Cap_AccessControl*** by **lpOutBuffer**. If this function is supported by the device, the node <**isSupportRightControllerAudio**> will be returned in the message and its value is "true". |

### 2.9.3 Other Settings

### Local DIP (Dual In-line Package) and Information

| Function | Description |
|---|---|
| Get Capability of Getting Local DIP and Information | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/GateDialAndInfo/capabilities*** .<br>The capability is returned in the message ***XML_Cap_GateDialAndInfo*** by **lpOutputParam**. |
| Get Local DIP and Information | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/GateDialAndInfo*** .<br>The information is returned in the message ***XML_GateDialAndInfo*** by **lpOutputParam**. |

**Note**

To check whether the device supports getting local DIP and information of the turnstile, you can call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/capabilities*** to get the access control capability.

The capability is returned in the message ***XML_Cap_AccessControl*** by **lpOutputParam**. If this function is supported by the device, the node <**isSupportGateDialAndInfo**> will be returned in the message and its value is "true".

## People Counting

| Function | Description |
|---|---|
| Get People Counting Parameters | Call ***NET_DVR_GetDVRConfig*** with the command "NET_DVR_ GET_PERSON_STATISTICS_CFG" (command No.: 2170).<br>The parameters are returned in the structure ***NET_DVR_PERSON_STATISTICS_CFG*** by the output buffer **lpOutBuffer**. |
| Set People Counting Parameters | Call ***NET_DVR_SetDVRConfig*** with the command "NET_DVR_ SET_PERSON_STATISTICS_CFG" (command No.: 2171) and set the input buffer **lpInBuffer** to the structure ***NET_DVR_PERSON_STATISTICS_CFG*** . |

**Note**

To check whether the device supports configuring people counting parameters, you can call ***NET_DVR_GetDeviceAbility*** , set the capability type **dwAbilityType** to "ACE_ABILITY" (macro definition value: 0x801), and set the input parameter pointer **pInBuf** to the message ***XML_Desc_AcsAbility*** for getting the access control capability.
The capability is returned in the message ***XML_AcsAbility*** by the output parameter pointer **pOutBuf**. The related node is <**PersonStatisticsCfg**>.

## Turnstile Barrier Time

| Function | Description |
|---|---|
| Get Barrier Time Parameters of Turnstile | Call ***NET_DVR_GetDVRConfig*** with the command "NET_DVR_ GET_GATE_TIME_CFG" (command No.: 2174).<br>The parameters are returned in the structure ***NET_DVR_GATE_TIME_CFG*** by the output buffer **lpOutBuffer**. |
| Set Barrier Time Parameters of Turnstile | Call ***NET_DVR_SetDVRConfig*** with the command "NET_DVR_ SET_GATE_TIME_CFG" (command No.: 2175) and set the input buffer **lpInBuffer** to the structure ***NET_DVR_GATE_TIME_CFG*** . |

**Note**

To check whether the device supports configuring barrier time parameters of the turnstile, you can call ***NET_DVR_GetDeviceAbility*** , set the capability type **dwAbilityType** to "ACE_ABILITY" (macro definition value: 0x801), and set the input parameter pointer **pInBuf** to the message ***XML_Desc_AcsAbility*** for getting the access control capability.
The capability is returned in the message ***XML_AcsAbility*** by the output parameter pointer **pOutBuf**. The related node is <**GateTimeCfg**>.

### Keyfob Control Mode

| Function | Description |
|---|---|
| Get configuration capability of keyfob control mode | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/remoteCtrllerModeCfg/capabilities?format=json*** .<br>The capability is returned in the message ***JSON_RemoteCtrllerModeCfgCap*** by **lpOutputParam**. |
| Get parameters of keyfob control mode | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/remoteCtrllerModeCfg?format=json*** .<br>The parameters are returned in the message ***JSON_RemoteCtrllerModeCfg*** by **lpOutputParam**. |
| Set parameters of keyfob control mode | Call ***NET_DVR_STDXMLConfig*** to transmit the request URI: PUT ***/ISAPI/AccessControl/remoteCtrllerModeCfg?format=json*** and set **lpInputParam** to the message ***JSON_RemoteCtrllerModeCfg*** . |

**Note**

To check whether the device supports configuring parameters of the keyfob control mode, you can call ***NET_DVR_STDXMLConfig*** to transmit the request URI: GET ***/ISAPI/AccessControl/capabilities*** to get the access control capability.
The capability is returned in the message ***XML_Cap_AccessControl*** by **lpOutputParam**. If this function is supported by the device, the node <**isSupportRemoteCtrllerModeCfg**> will be returned in the message and its value is "true".

## 2.10 Other Applications

### Intelligent Identity Detection Terminal

| Function | Description |
|---|---|
| Get Configuration Capability | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/IdentityTerminal/capabilities*** .<br>And the configuration capability is returned in the message ***XML_Cap_IdentityTerminal*** by output parameter (**lpOutputParam**). |
| Get Parameters | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/IdentityTerminal*** .<br>And the parameters are returned in the message ***XML_IdentityTerminal*** by the output buffer (**lpOutBuffer**) of output parameter (**lpOutputParam**). |
| Set Parameters | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: PUT ***/ISAPI/AccessControl/IdentityTerminal*** and set the input buffer (**lpInBuffer**) of input parameter (**lpInputParam**) to the message ***XML_IdentityTerminal*** . |

### Access Control Device No.

| Function | Description |
|---|---|
| Get Device No. | Call ***NET_DVR_GetDVRConfig*** with "NET_DVR_GET_VIDEO_ INTERCOM_DEVICEID_CFG" (command No.: 16001).<br>And the access control device No. will be returned in the structure ***NET_DVR_VIDEO_INTERCOM_DEVICEID_CFG*** by the output buffer (**lpOutBuffer**). |
| Set Device No. | Call ***NET_DVR_SetDVRConfig*** with "NET_DVR_SET_VIDEO_ INTERCOM_DEVICEID_CFG" (command No.: 16002) and set the input buffer (**lpInBuffer**) to the structure ***NET_DVR_VIDEO_INTERCOM_DEVICEID_CFG*** . |

### Linked Network Device

| Function | Description |
|---|---|
| Get Parameters of Linked Network Device | Call ***NET_DVR_GetDVRConfig*** with "NET_DVR_GET_VIDEO_ INTERCOM_RELATEDEV_CFG" (command No.: 16006). |

| Function | Description |
|---|---|
| | And the parameters of linked network device are returned in the structure ***NET_DVR_VIDEO_INTERCOM_RELATEDEV_CFG*** by the output buffer (**lpOutBuffer**). |
| Set Parameters of Linked Network Device | Call ***NET_DVR_SetDVRConfig*** with "NET_DVR_SET_VIDEO_INTERCOM_RELATEDEV_CFG" (command No.: 16007) and set the input buffer (**lpInBuffer**) to the structure ***NET_DVR_VIDEO_INTERCOM_RELATEDEV_CFG*** . |

## Reader

| Function | Description |
|---|---|
| Get Reader Parameters | Call ***NET_DVR_GetDVRConfig*** with "NET_DVR_GET_CARD_READER_CFG_V50" (command No.: 2505).<br>And the reader parameters are returned in the structure ***NET_DVR_CARD_READER_CFG_V50*** by the output buffer (**lpOutBuffer**). |
| Set Reader Parameters | Call ***NET_DVR_SetDVRConfig*** with "NET_DVR_SET_CARD_READER_CFG_V50" (command No.: 2506) and set the input buffer (**lpInBuffer**) to the structure ***NET_DVR_CARD_READER_CFG_V50*** . |

**Table 2-1 NFC (Near-Field Communication) Function**

| Function | Description |
|---|---|
| Get Configuration Capability of Enabling or Disabling NFC Function | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/Configuration/NFCCfg/capabilities?format=json*** .<br>And the configuration capability is returned in the message ***JSON_NFCCfgCap*** by the output parameter (**lpOutputParam**). |
| Get Parameters of Enabling or Disabling NFC Function | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/Configuration/NFCCfg?format=json*** .<br>And the parameters are returned in the message ***JSON_NFCCfg*** by **lpOutBuffer** of **lpOutputParam**. |
| Set Parameters of Enabling or Disabling NFC Function | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: PUT ***/ISAPI/AccessControl/Configuration/NFCCfg?format=json*** and set **lpInBuffer** of **lpInputParam** to the message ***JSON_NFCCfg*** . |

**Table 2-2 RF (Radio Frequency) Card Recognition**

| Function | Description |
|---|---|
| Get Configuration Capability of Enabling or Disabling RF Card Recognition | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/Configuration/RFCardCfg/capabilities?format=json*** .<br>And the configuration capability is returned in the message ***JSON_RFCardCfgCap*** by the output parameter (**lpOutputParam**). |
| Get Parameters of Enabling or Disabling RF Card Recognition | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/Configuration/RFCardCfg?format=json***<br>And the parameters are returned in the message ***JSON_RFCardCfg*** by **lpOutBuffer** of **lpOutputParam**. |
| Set Parameters of Enabling or Disabling RF Card Recognition | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: PUT ***/ISAPI/AccessControl/Configuration/RFCardCfg?format=json*** and set **lpInBuffer** of **lpInputParam** to the message ***JSON_RFCardCfg*** . |

**Table 2-3 Active Infrared Intrusion Detection**

| Function | Description |
|---|---|
| Get Configuration Capability of Active Infrared Intrusion Detection | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/Configuration/IRCfg/capabilities?format=json*** .<br>And the configuration capability is returned in the message ***JSON_IRCfgCap*** by **lpOutputParam**. |
| Get Parameters of Active Infrared Intrusion Detection | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/Configuration/IRCfg?format=json*** .<br>And the parameters are returned in the message ***JSON_IRCfg*** by **lpOutputParam**. |
| Set Parameters of Active Infrared Intrusion Detection | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: PUT ***/ISAPI/AccessControl/Configuration/IRCfg?format=json*** and set the **lpInputParam** to ***JSON_IRCfg*** . |

**Table 2-4 Condition Parameters of Face Picture Comparison**

| Function | Description |
|---|---|
| Get Condition Configuration Capability of Face Picture Comparison | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/FaceCompareCond/capabilities*** .<br>And the configuration capability is returned in the message ***XML_Cap_FaceCompareCond*** by **lpOutputParam**. |
| Get Conditions of Face Picture Comparison | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: GET ***/ISAPI/AccessControl/FaceCompareCond*** .<br>And the conditions are returned in the message ***XML_FaceCompareCond*** by **lpOutputParam**. |
| Set Conditions of Face Picture Comparison | Call ***NET_DVR_STDXMLConfig*** to transmit the request URL: PUT ***/ISAPI/AccessControl/FaceCompareCond*** and set **lpInputParam** to ***XML_FaceCompareCond*** . |

## Peripherals Connected via Serial Port

| Function | Description |
|---|---|
| Get Peripheral Parameters | Call ***NET_DVR_GetDVRConfig*** with the command "NET_DVR_GET_ACS_EXTERNAL_DEV_CFG" (command No.: 2165) and set the **lChannel** to 4-byte RS-485 serial port No.<br>The parameters are returned in the structure ***NET_DVR_ACS_EXTERNAL_DEV_CFG*** by the output buffer **lpOutBuffer**. |
| Set Peripheral Parameters | Call ***NET_DVR_SetDVRConfig*** with the command "NET_DVR_SET_ACS_EXTERNAL_DEV_CFG" (command No.: 2166), set the **lChannel** to 4-byte RS-485 serial port No., and set the input buffer **lpInBuffer** to the structure ***NET_DVR_ACS_EXTERNAL_DEV_CFG*** . |

**Note**

- The 4-byte RS-485 serial port No. starts from 1.
- To check whether the device supports configuring peripheral parameters connected to the access controller via serial port, you can call ***NET_DVR_GetDeviceAbility*** , set the capability type **dwAbilityType** to "ACE_ABILITY" (macro definition value: 0x801), and set the input parameter pointer **pInBuf** to the message ***XML_Desc_AcsAbility*** for getting the access control capability. The capability is returned in the message ***XML_AcsAbility*** by the output parameter pointer **pOutBuf**. The related node is <**ExternalDevCfg**>.

# Chapter 3 API Reference

## 3.1 NET_DVR_Cleanup

Release the resources after the program is ended.

### API Definition

```
BOOL NET_DVR_Cleanup(
);
```

### Return Values

Returns *TURE* for success, and returns *FALSE* for failure.
If *FALSE* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.
The available error codes may be returned by this API are 0 and 3. See details in ***Device Network SDK Errors*** .

### Remarks

- When calling this API, you cannot call other APIs at the same time.
- ***NET_DVR_Init*** and this API should be called by pair. That is, once the NET_DVR_Init is called, you should call NET_DVR_Cleanup to release the resources when exiting the program.

## 3.2 NET_DVR_GetErrorMsg

Return the error information of the last operation.

### API Definition

```
char *NET_DVR_GetErrorMsg(
  LONG   *pErrorNo
);
```

### Parameters

**pErrorNo**

[OUT] Error code pointer.

### Return Values

The return values are the pointers of error information, see ***Device Network SDK Errors*** for details.

### Remarks

You can call ***NET_DVR_GetLastError*** to get the error codes.

## 3.3 NET_DVR_GetLastError

Return the error code of the last operation.

### API Definition

```
DWORD NET_DVR_GetLastError(
);
```

### Return Values

The return values are error codes, see ***Device Network SDK Errors*** for details.

### Remarks

You can also call ***NET_DVR_GetErrorMsg*** to directly get the error information.

## 3.4 NET_DVR_Init

Initialize the programming environment before calling other APIs.

### API Definition

```
BOOL NET_DVR_Init(
);
```

### Return Values

Returns *TURE* for success, and returns *FALSE* for failure.
If *FALSE* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.
The available error codes of this API are 0, 41, and 53. See details in ***Device Network SDK Errors*** .

### Remarks

Before initializing, you can call ***NET_DVR_SetSDKInitCfg*** to set the initialization parameters, such as supported capabilities, loading path of component libraries (only supported by Linux system), and so on.

### See Also

***NET_DVR_Cleanup***

## 3.5 NET_DVR_Login_V40

Log in to the device (supports asynchronous login).

## API Definition

```
LONG NET_DVR_Login_V40(
  NET_DVR_USER_LOGIN_INFO   pLoginInfo,
  NET_DVR_DEVICEINFO_V40    lpDeviceInfo
);
```

## Parameters

**pLoginInfo**

[IN] Login parameters, including device address, user name, password, and so on. See details in the structure ***NET_DVR_USER_LOGIN_INFO*** .

**lpDeviceInfo**

[OUT] Device information. See details in the structure ***NET_DVR_DEVICEINFO_V40*** .

## Return Values

- For asynchronous login, the callback function ( ***fLoginResultCallBack*** ) configured in the structure ( ***NET_DVR_USER_LOGIN_INFO*** ) returns the asynchronous login status, user ID and device information.
- For synchronous login, this API returns *-1* for logging failed, and returns other values for the returned user IDs. The user ID is unique, and it helps to realize the further device operations.
- If *-1* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.

## Remarks

- When **bUseAsynLogin** in **pLoginInfo** is 0, it indicates that login is in synchronous mode; when **bUseAsynLogin** in **pLoginInfo** is 1, it indicates that login is in asynchronous mode.
- Up to 2048 users are allowed to log in to HCNetSDK at same time, and the values of returned **UserID** are ranging from 0 to 2047.

## See Also

***NET_DVR_Logout***

## 3.5.1 fLoginResultCallBack

### Login Status Callback Function

| Member | Data Type | Description |
|---|---|---|
| lUserID | LONG | User ID, which is returned by ***NET_DVR_Login_V40*** . |
| dwResult | DWORD | Login status: 0-asynchronously logging in failed, 1-asynchronously logged in. |
| lpDeviceInfo | ***NET_DVR_DEVICEINFO_V40*** | Device information, such as serial No., channel, capability, and so on. |
| pUser | void* | User data. |

## 3.6 NET_DVR_Logout

Log out from devices.

### API Definitions

```
BOOL NET_DVR_Logout(
  LONG    lUserID
);
```

### Parameters

**lUserID**

[IN] User ID, which is returned by ***NET_DVR_Login_V40*** .

### Return Values

Returns *TURE* for success, and returns *FALSE* for failure.
If *FALSE* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.
The available error codes may be returned by this API are 0, 3, 7, 8, 9, 10, 14, 17, 41, 44, 47, 72, and 73. See details in ***Device Network SDK Errors*** .

## 3.7 NET_DVR_SetSDKInitCfg

Set initialization parameters.

### API Parameters

```
BOOL NET_DVR_SetSDKInitCfg(
  NET_SDK_INIT_CFG_TYPE    enumType,
  void* const          lpInBuff
);
```

## Parameters

### enumType

[IN] Initialization parameter type. Different type values correspond to different parameters, see details in the table below.

**Table 3-1 NET_SDK_INIT_CFG_TYPE**

| enumType | Value | Description | lpInBuff |
|---|---|---|---|
| NET_SDK_INIT_CFG_ABILITY | 1 | Capability supported by SDK. | ***NET_DVR_INIT_CFG_ABILITY*** |
| NET_SDK_INIT_CFG_SDK_PATH | 2 | Set loading path for component libraries (supported by both Linux and Windows system). | ***NET_DVR_LOCAL_SDK_PATH*** |
| NET_SDK_INIT_CFG_LIBEAY_PATH | 3 | Set path (including library name) for libeay32.dll (Windows), libcrypto.so (Linux), and libcrypto.dylib (Mac) of OpenSSL in version 1.1.1 and 1.0.2. | Path in string format, e.g., ***C:\\libeay32.dll***. |
| NET_SDK_INIT_CFG_SSLEAY_PATH | 4 | Set path (including library name) for ssleay32.dll (Windows), libssl.so (Linux), libssl.dylib (Mac) of OpenSSL in version 1.1.1 and 1.0.2. | Path in string format, e.g., ***C:\\ssleay32.dll***. |

### lpInBuff

[IN] Input parameter. Different parameter types correspond to different structures, see details in the table above.

## Return Values

Returns *TURE* for success, and returns *FALSE* for failure.
If *FALSE* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.

### Remarks

This API should be called before calling ***NET_DVR_Init*** to initialize and check the dependent libraries or capabilities.

## 3.8 NET_DVR_GetDeviceAbility

Get the device capabilities.

### API Definition

```
BOOL NET_DVR_GetDeviceAbility(
 LONG    lUserID,
 DWORD   dwAbilityType,
 char    *pInBuf,
 DWORD   dwInLength,
 char    *pOutBuf,
 DWORD   dwOutLength
);
```

### Parameters

**lUserID**

[IN] Value returned by ***NET_DVR_Login_V40*** .

**dwAbilityType**

[IN] Capability types, which are different according to different devices and functions.

**pInBuf**

[IN] Input parameter buffer pointer, which are different according to different devices and functions, and they are returned in the structure or messages.

**dwInLength**

[IN] Size of input buffer.

**pOutBuf**

[OUT] Output parameter buffer pointer, which are different according to different devices and functions, and they are returned in the structure or messages.

**dwOutLength**

[OUT] Size of buffer for receiving data.

### Return Values

Returns *TRUE* for success, and returns *FALSE* for failure.
If *FALSE* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.

## 3.9 NET_DVR_GetDeviceConfig

Get device configuration information in batch (with sending data).

### API Definition

```
BOOL NET_DVR_GetDeviceConfig(
  LONG      lUserID,
  DWORD     dwCommand,
  DWORD     dwCount,
  LPVOID    lpInBuffer,
  DWORD     dwInBufferSize,
  LPVOID    lpStatusList,
  LPVOID    lpOutBuffer,
  DWORD     dwOutBufferSize
);
```

### Parameters

**lUserID**

[IN] Value returned by ***NET_DVR_Login_V40*** .

**dwCommand**

[IN] Device getting commands. The commands are different for different getting functions.

**dwCount**

[IN] Number of configurations (cameras) to get at a time. 0, 1-one camera, 2-two cameras, 3-three cameras, and so on. Up to 64 cameras' configuration information can be obtained at a time.

**lpInBuffer**

[IN] Pointer of configuration condition buffer, which specifies the number (**dwCount**) of configurations to get, and relates to the getting commands.

**dwInBufferSize**

[IN] Size of configuration condition buffer, which saves the obtained configuration information (the number is **dwCount**).

**lpStatusList**

[OUT] Error information list, and its memory is allocated by user, each error information contains 4 bytes (a unsigned 32-bit integer).

There is a one-to-one correspondence between the errors in the list and the cameras need to search, e.g., **lpStatusList[2]** corresponds to **lpInBuffer[2]**.

If the parameter value is 0 or 1, it refers to getting succeeded, otherwise, this parameter value is the error code.

**lpOutBuffer**

[OUT] Parameters returned by device, which relates to the getting commands. And there is a one-to-one correspondence between the parameters and the cameras need to search.

If the **lpStatusList** of one camera is larger than 1, the corresponding **lpOutBuffer** is invalid.

**dwOutBufferSize**

[IN] Total size of returned results (the number is **dwCount**).

### Return Values

Returns *TRUE* for success, and returns *FALSE* for failure. If returns *TRUE*, it does not mean that all configurations are obtained, you can check the value of **lpStatusList[n]** to judge which one is succeeded.
If *FALSE* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.

### See Also

***NET_DVR_SetDeviceConfig***

## 3.10 NET_DVR_GetDownloadState

Get the file downloading progress and status.

### API Definition

```
LONG NET_DVR_GetDownloadState(
  LONG      lDownloadHandle,
  DWORD     *pProgress
);
```

### Parameters

**lDownloadHandle**

[IN] Handle for downloading files, which is returned by ***NET_DVR_StartDownload*** .

**pProgress**

[OUT] Returned progress value, which is ranging from 1 to 100.

### Return Values

Returns *-1* for calling failed, and returns other values as the downloading status codes: 1-Downloaded, 2-Downloading, 3-Downloading Failed, 4-Network Disconnected, Unknown Status.
If returning failed, you can call ***NET_DVR_GetLastError*** to get the error code.

## 3.11 NET_DVR_GetNextRemoteConfig

Get the next search result.

## API Definition

```
LONG NET_DVR_GetNextRemoteConfig(
  LONG     lHandle,
  void     *lpOutBuff,
  DWORD    dwOutBuffSize
);
```

## Parameters

**lHandle**

[IN] Search handle, which is the value returned by ***NET_DVR_StartRemoteConfig*** .

**lpOutBuff**

[OUT] Output parameter buffer pointer, which relates to the commands (**dwCommand**) of ***NET_DVR_StartRemoteConfig*** .

**dwOutBuffSize**

[IN] Buffer size.

## Return Values

Returns *-1* for failure, and returns other values for the current statuses, see details in the following table.

| Status | Value | Description |
|---|---|---|
| NET_SDK_GET_NEXT_STATUS_SUCCESS | 1000 | The data is obtained. The API NET_DVR_GetNextRemoteConfig should be called again to get the next item of data. |
| NET_SDK_GET_NETX_STATUS_NEED_WAIT | 1001 | Waiting. The API NET_DVR_GetNextRemoteConfig can be called again. |
| NET_SDK_GET_NEXT_STATUS_FINISH | 1002 | All data is obtained. The API ***NET_DVR_StopRemoteConfig*** can be called to end. |
| NET_SDK_GET_NEXT_STATUS_FAILED | 1003 | Getting data exception. The API ***NET_DVR_StopRemoteConfig*** can be called to end. |

If *-1* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.

## Remarks

To get all information, you should call this API repeatedly.

## 3.12 NET_DVR_GetUploadState

Get the file uploading progress and status.

### API Definition

```
LONG NET_DVR_GetUploadState(
  LONG      lUploadHandle,
  DWORD     *pProgress
);
```

### Parameters

**lUploadHandle**

[IN] Handling for uploading files, which is returned by ***NET_DVR_UploadFile_V40*** .

**pProgress**

[OUT] Returned progress value.

### Return Values

Return *-1* for failure, and return other values as the uploading status codes, see details in the following table.

**Table 3-2 Uploading Status Code**

| Return Value | Description |
|---|---|
| 1 | Uploaded successfully. |
| 2 | Uploading. |
| 3 | Uploading failed. |
| 4 | Network disconnected. Unknown status. |
| 6 | HDD error. |
| 7 | No HDD for saving inquest files. |
| 8 | Insufficient capacity. |
| 9 | Insufficient device resource. |
| 10 | No more files can be uploaded. |
| 11 | Too large file size. |
| 15 | File type error. |
| 19 | Invalid file format. |
| 20 | Incorrect file content. |

| Return Value | Description |
|---|---|
| 21 | The uploaded audio sampling rate is not supported. |
| 22 | Insufficient storage in the face library. |
| 26 | Name error. |
| 27 | Invalid picture resolution. |
| 28 | Too many targets on the picture. |
| 29 | No target is recognized on the picture. |
| 30 | Picture recognition failed. |
| 31 | Analysis engine exception. |
| 32 | Analyzing additional information on the picture failed. |
| 33 | Thumbnail modeling failed. |
| 34 | Incorrect security verification key. |
| 35 | Downloading picture via URL has not started. |
| 36 | Duplicate custom ID of different persons. |
| 37 | Person ID error (The ID is saved in **customHumanID** of **FaceAppendData**). |
| 38 | Modeling failed. Device inner error. |
| 39 | Modeling failed. Face modeling error. |
| 40 | Modeling failed. Face score error. |
| 41 | Modeling failed. Feature collection error. |
| 42 | Modeling failed. Attribute collection error. |
| 43 | Picture data error. |
| 44 | Picture additional information error. |
| 45 | Certificate has already existed. |

## 3.13 NET_DVR_ControlGateway

Call this API to remotely control the door or elevator.

### API Definition

```
BOOL NET_DVR_ControlGateway(
  LONG     lUserID,
  LONG     lGatewayIndex,
  DWORD    dwStaic
);
```

### Parameters

**lUserID**

[IN] Value returned by ***NET_DVR_Login_V40*** .

**lGatewayIndex**

[IN] Door No. or floor No., which starts from 1, -1: Control all doors or elevators of floors.

**dwStaic**

[IN] Command No.: 0-Close (Under Control), 1-Open, 2-Remain Open (Free), 3-Remain Closed (Disabled), 4-Recovery (only for elevator), 5-Vistor Call Elevator (only for elevator), 6-Resident Call Elevator (only for elevator).

### Return Values

Returns *TRUE* for success, and returns *FALSE* for failure.
If returning failed, you can call ***NET_DVR_GetLastError*** to get the error code.

## 3.14 NET_DVR_SendRemoteConfig

Send data via the persistent connection.

### API Definition

```
BOOL NET_DVR_SendRemoteConfig(
  LONG     lHandle,
  DWORD    dwDataType,
  char     *pSendBuf,
  DWORD    dwBufSize
);
```

### Parameters

**lHandle**

Persistent configuration handle, which is returned by ***NET_DVR_StartRemoteConfig*** .

**dwDataType**

[IN] Data type, which relates to the commands of ***NET_DVR_StartRemoteConfig*** .

**pSendBuf**

[IN] Buffer for saving data to be sent, which relates to **dwDataType**.

**dwBufSize**

[IN] Size of data to be sent.

### Return Values

Returns *TRUE* for success, and returns *FALSE* for failure.
If *FALSE* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.

### Remarks

Before calling this API, you must call ***NET_DVR_StartRemoteConfig*** to get the persistent connection handle.

## 3.15 NET_DVR_SetDeviceConfig

Set device parameters in batch (sending data is supported).

### API Definition

```
BOOL NET_DVR_SetDeviceConfig(
  LONG      lUserID,
  DWORD     dwCommand,
  DWORD     dwCount,
  LPVOID    lpInBuffer,
  DWORD     dwInBufferSize,
  LPVOID    lpStatusList,
  LPVOID    lpInParamBuffer,
  DWORD     dwInParamBufferSize
);
```

### Parameters

**lUserID**

[IN] Value returned by ***NET_DVR_Login_V40*** .

**dwCommand**

[IN] Device configuration commands, which are different according to different configurations.

**dwCount**

[IN] Number of cameras to be set at a time. 0,1-one camera, 2-two cameras, 3-three cameras, and so on. Up to 256 cameras can be configured at a time.

**lpInBuffer**

[IN] Pointer of configuration condition buffer, e.g., stream ID, which specifies the number (**dwCount**) of cameras to set, and relates to the configuration commands.

**dwInBufferSize**

[IN] Size of configuration condition buffer, which saves the configured information of cameras with the number of **dwCount**.

**lpStatusList**

[OUT] Error information list, and its memory is allocated by user, each error information contains 4 bytes (a unsigned 32-bit integer).

There is a one-to-one correspondence between the errors in the list and the cameras that need to be searched, e.g., **lpStatusList[2]** corresponds to **lpInBuffer[2]**.

If the parameter value is 0, it refers to setting succeeded, otherwise, this parameter value is the error code.

**lpInParamBuffer**

[IN] Device parameters to set, which relates to the configuration commands. And there is a one-to-one correspondence between the parameters and the cameras that need to be searched.

**dwInParamBufferSize**

[IN] Set the size of content buffer.

### Return Values

Returns *TRUE* for success, and returns *FALSE* for all failed. If returns *TRUE*, it does not indicate that all settings are succeeded, you can get the value of **lpStatusList[n]** to check which one is succeeded.
If *FALSE* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.

### See Also

***NET_DVR_GetDeviceConfig***

## 3.16 NET_DVR_StartDownload

Start downloading files

### API Definition

```
LONG NET_DVR_StartDownload(
  LONG         lUserID,
  DWORD        dwDownloadType,
  LPVOID       lpInBuffer,
  DWORD        dwInBufferSize,
  char const   *sFileName
);
```

### Parameters

**lUserID**

[IN] Value returned by ***NET_DVR_Login_V40*** .

**dwDownloadType**

[IN] Downloading commands which specify the file type to download, see details in the enumeration ***NET_SDK_DOWNLOAD_TYPE*** .

**lpInBuffer**

[IN] Input parameters, which are different according to different downloading commands.

**dwInBufferSize**

[IN] Input buffer size.

**sFileName**

[IN] Path for saving downloaded files (absolute path, includes file name).

### Return Values

Returns *-1* for failure, and returns other values as the parameters of ***NET_DVR_StopDownload*** and ***NET_DVR_GetDownloadState*** .
If returning failed, you can call ***NET_DVR_GetLastError*** to get the error code.

## 3.17 NET_DVR_StartRemoteConfig

Enable remote configuration.

### API Definition

```
LONG NET_DVR_StartRemoteConfig(
  LONG                   lUserID,
  DWORD                  dwCommand,
  LPVOID                 lpInBuffer,
  DWORD                  dwInBufferLen,
  fRemoteConfigCallback  cbStateCallback,
  LPVOID                 pUserData
);
```

### Parameters

**lUserID**

[IN] Value returned by ***NET_DVR_Login_V40*** .

**dwCommand**

[IN] Configuration commands. For different functions, the commands and **lpInBuffer** are different, see the detailed relation in the table below:

| dwCommand Macro Definition | Value | Description | lpInBuffer Related Structure | lpBuffer Related Structure |
|---|---|---|---|---|
| NET_DVR_GET_ALL_RECORD_PASSBACK_TASK_MANUAL | 6235 | Get tasks of manually copying back videos | ***NET_DVR_RECORD_PASSBACK_MANUAL_COND*** | ***NET_DVR_RECORD_PASSBACK_MANUAL_TASK_RET*** |

**lpInBuffer**

Input parameter buffer pointer, which relates to the configuration command.

**dwInBufferLen**

[IN] Size of input buffer.

**cbStateCallback**

[IN] Status callback function, see the definition in ***fRemoteConfigCallback*** .

**pUserData**

[OUT] User data.

## Return Values

Returns *-1* for failure, and returns other values for the handles of ***NET_DVR_GetNextRemoteConfig*** and ***NET_DVR_StopRemoteConfig*** .
If *-1* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.

## Remarks

This API specifies the information to search. After calling this API, you can call ***NET_DVR_GetNextRemoteConfig*** to get the information one by one.

## 3.17.1 fRemoteConfigCallback

Function for calling back the persistent connection status and data to be transmitted.

## Callback Function Definition

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

## Parameters

**dwType**

[OUT] Connection statuses, see the macro definitions below:

```
enum _NET_SDK_CALLBACK_TYPE_{
 NET_SDK_CALLBACK_TYPE_STATUS   = 0,
 NET_SDK_CALLBACK_TYPE_PROGRESS = 1,
 NET_SDK_CALLBACK_TYPE_DATA     = 2
}NET_SDK_CALLBACK_TYPE
```

**NET_SDK_CALLBACK_TYPE_STATUS**

Connection status.

**NET_SDK_CALLBACK_TYPE_PROGRESS**

Connection progress.

**NET_SDK_CALLBACK_TYPE_DATA**

Related data to be called back.

**lpBuffer**

[OUT] Pointer of buffer for saving progress, status, and related data to be called back, which relates to **dwType**, see details in the following table.

| dwType | lpBuffer |
|---|---|
| NET_SDK_CALLBACK_TYPE_STATUS | If **dwBufLen** is 4, **lpBuffer** is 4-byte connection status; if **dwBufLen** is 8, **lpBuffer** consists of 4-byte connection status and 4-byte error code.<br>The connection status is enumerated in ***NET_SDK_CALLBACK_STATUS_NORMAL*** |
| NET_SDK_CALLBACK_TYPE_PROGRESS | Connection progress value. |
| NET_SDK_CALLBACK_TYPE_DATA | Data structures to be returned, which are different according to different commands (**dwCommand**) in ***NET_DVR_StartRemoteConfig*** . |

**dwBufLen**

[OUT] Buffer size.

**pUserData**

[OUT] User data.

## 3.18 NET_DVR_STDXMLConfig

Transmit request URL with XML or JSON format to implement some typical functions.

### API Definition

```
BOOL NET_DVR_STDXMLConfig(
  LONG                              lUserID,
  const NET_DVR_XML_CONFIG_INPUT      *lpInputParam,
  NET_DVR_XML_CONFIG_OUTPUT          *lpOutputParam
);
```

### Parameters

**lUserID**

[IN] Value returned by ***NET_DVR_Login_V40*** .

**lpInputParam**

[IN] Input parameters, refer to the structure ***NET_DVR_XML_CONFIG_INPUT*** for details.

**lpOutputParam**

[IN][OUT] Output parameters, refer to the structure ***NET_DVR_XML_CONFIG_OUTPUT*** for details.

### Return Values

Return *TRUE* for success, and return *FALSE* for failure.
If *FALSE* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.

### Remarks

The input parameter **lpInputParam** and output parameter **lpOutputParam** are different when transmitting text protocol for implementing different functions, and each parameter corresponds to a component of text protocol, see the relations below:

| Parameter of NET_DVR_STDXMLConfig | | Component of Text Protocol |
|---|---|---|
| **lpInputParam** | **lpRequestUrl** (see in structure ***NET_DVR_XML_CONFIG_INPUT*** ) | Method+URL<br>E.g., GET /ISAPI/System/capabilities |
| | **lpInBuffer** (see in structure ***NET_DVR_XML_CONFIG_INPUT*** ) | Request Message |

| Parameter of NET_DVR_STDXMLConfig | | Component of Text Protocol |
|---|---|---|
| **lpOutputParam** | **lpOutBuffer** (see in structure ***NET_DVR_XML_CONFIG_OUTPUT*** ) | Response Message |
| | **lpStatusBuffer** (see in structure ***NET_DVR_XML_CONFIG_OUTPUT*** ) | Response Message |

## 3.19 NET_DVR_StopDownload

Stop downloading files.

### API Definition

```
BOOL NET_DVR_StopDownload(
  LONG   lHandle
);
```

### Parameters

**lHandle**

[IN] Handle for downloading files, which is returned by ***NET_DVR_StartDownload*** .

### Return Values

Returns *TRUE* for success, and returns *FALSE* for failure.
If *FALSE* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.

## 3.20 NET_DVR_StopRemoteConfig

Disconnect the persistent connection to stop remote configuration, and release resources.

### API Definition

```
BOOL NET_DVR_StopRemoteConfig(
  LONG   lHandle
);
```

### Parameters

**lHandle**

[IN] Handle, which is returned by ***NET_DVR_StartRemoteConfig*** .

### Return Values

Returns *TRUE* for success, and returns *FALSE* for failure.
If *FALSE* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.

## 3.21 NET_DVR_UploadClose

Stop uploading files.

### API Definition

```
BOOL NET_DVR_UploadClose(
  LONG    lUploadHandle
);
```

### Parameters

**lUploadHandle**

[IN] Handle for uploading files, which is returned by ***NET_DVR_UploadFile_V40*** .

### Return Values

Return *TRUE* for success, and return *FALSE* for failure.
If *FALSE* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.

## 3.22 NET_DVR_UploadFile_V40

Upload file.

### API Definition

```
LONG NET_DVR_UploadFile_V40(
  LONG      lUserID,
  DWORD     dwUploadType,
  LPVOID    lpInBuffer,
  DWORD     dwInBufferSize,
  char      *sFileName,
  LPVOID    lpOutBuffer,
  DWORD     dwOutBufferSize
);
```

### Parameters

**lUserID**

[IN] Value returned by ***NET_DVR_Login_V40*** .

**dwUploadType**

[IN] Uploading commands, which specify the file type to upload, see details in the enumeration ***NET_SDK_UPLOAD_TYPE*** .

**lpInBuffer**

[IN] Input parameters, which are different according to different uploading commands.

**dwInBufferSize**

[IN] Input buffer size.

**sFileName**

[IN] Name of the file to be uploaded. For the complete file path (including the file name), the maximum size is 128 bytes, and the maximum size of the file name is 32 bytes.

**lpOutBuffer**

[OUT] Output parameters, which are different according to different uploading commands.

**dwOutBufferSize**

[OUT] Output buffer size.

### Return Values

Return *-1* for failure, and return other values as the parameter of ***NET_DVR_UploadClose*** and ***NET_DVR_GetUploadState*** .
If *-1* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.

## 3.23 NET_DVR_CloseAlarmChan_V30

Close alarm uploading channel.

### API Definition

```
BOOL NET_DVR_CloseAlarmChan_V30(
  LONG   lAlarmHandle
);
```

### Parameters

**lAlarmHandle**

Value returned by ***NET_DVR_SetupAlarmChan_V50*** .

### Return Values

Return *TURE* for success, and return *FALSE* for failure.
If *FALSE* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.
The available error codes of this API are 0, 3, 6, 12, 17, 41, and 47. See details in the ***Device Network SDK Errors*** .

## 3.24 NET_DVR_GetDVRConfig

Get the device configuration information.

### API Definition

```
BOOL NET_DVR_GetDVRConfig(
  LONG       lUserID,
  DWORD      dwCommand,
  LONG       lRuleID,
  LONG       lChannel,
  LPVOID     lpOutBuffer,
  DWORD      dwOutBufferSize,
  LPDWORD    lpBytesReturned
);
```

### Parameters

**lUserID**

[IN] Value returned by ***NET_DVR_Login_V40*** .

**dwCommand**

[IN] Device getting commands, which are different according to different getting functions.

**lRuleID**

[IN] Rule ID.

**lChannel**

[IN] Channel No. (NIC No.), which varies with different commands. 0xffffffff-invalid or all channels, 1-main NIC, 2-extended NIC.

**lpOutBuffer**

[OUT] Pointer of buffer to receive data. For different getting functions, the structures of this parameter are different.

**dwOutBufferSize**

[IN] Size of buffer to receive data (unit: byte). It cannot be 0.

**lpBytesReturned**

[OUT] Pointer of actually received data size. It cannot be NULL.

### Return Values

Returns *TRUE* for success, and returns *FALSE* for failure.
If *FALSE* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.
The following error codes may be returned by this API: 0, 3, 6, 7, 8, 9, 10, 12, 17, 41, 43, 44, 47, 72, 73, and 76. See the corresponding error types and descriptions in the ***Device Network SDK Errors*** .

### See Also

***NET_DVR_SetDVRConfig***

## 3.25 NET_DVR_SetDVRConfig

Set the device parameters.

### API Definition

```
BOOL NET_DVR_SetDVRConfig(
  LONG     lUserID,
  DWORD    dwCommand,
  LONG     lChannel,
  LPVOID   lpInBuffer,
  DWORD    dwInBufferSize
);
```

### Parameters

**lUserID**

[IN] Value returned by ***NET_DVR_Login_V40*** .

**dwCommand**

[IN] Device configuration commands, which are different according to different configuration functions.

**lChannel**

[IN] Channel No. (NIC No.), which varies with different commands. 0xFFFFFFFF-invalid, 1-main NIC, 2-extended NIC.

**lpInBuffer**

[IN] Pointer of input data buffer. For different configuration functions, the structures of this parameter are different.

**dwInBufferSize**

[IN] Size of input data buffer (unit: byte).

### Return Values

Returns *TRUE* for success, and returns *FALSE* for failure.
If *FALSE* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.
The following error codes may be returned by this API: 0, 3, 6, 7, 8, 9, 10, 12, 17, 41, 43, 44, 47, 72, 73, and 76. See the corresponding error types and descriptions in the ***Device Network SDK Errors*** .

### See Also

***NET_DVR_GetDVRConfig***

## 3.26 NET_DVR_SetDVRMessageCallBack_V50

Set callback functions for getting the video data.

### API Definition

```
BOOL NET_DVR_SetDVRMessageCallBack_V50(
  int           iIndex,
  MSGCallBack    fMessageCallBack,
  void           *pUser
);
```

### Parameters

**iIndex**

[IN] Callback function index No., which ranges from 0 to 15.

**fMessageCallBack**

[IN] Callback function, see details in ***MSGCallBack*** .

**pUser**

[IN] User data.

### Return Values

Return *TRUE* for success, and return *FALSE* for failure.
If *FALSE* returned, call ***NET_DVR_GetLastError*** to get the error code.

### Remarks

- This API supports setting multiple callback functions for different channels (up to 16 channels are supported) at same time, and the configured callback functions are distinguished by the index No.
- All alarm/event information will be returned in each configured callback function, and you can distinguish the devices via the **pAlarmInfo** in the callback function ( ***MSGCallBack*** ).

**Example**

Sample Code of Setting Multiple Callback Functions to Receive Different Alarms/Events in Arming Mode

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

int iNum=0;
void CALLBACK MessageCallbackNo1(LONG lCommand, NET_DVR_ALARMER *pAlarmer, char *pAlarmInfo, DWORD
dwBufLen, void* pUser)
{
  int i=0;
  char filename[100];
  FILE *fSnapPic=NULL;
  FILE *fSnapPicPlate=NULL;

  //This sample code is for reference only. Actually, it is not recommended to process the data and save file in the
callback function directly.
  //You'd better process the data in the message response funcion via message mode (PostMessage).

  switch(lCommand)
  {
    case COMM_ALARM:
    {
      NET_DVR_ALARMINFO struAlarmInfo;
      memcpy(&struAlarmInfo, pAlarmInfo, sizeof(NET_DVR_ALARMINFO));
      switch (struAlarmInfo.dwAlarmType)
      {
        case 3: //Motion detection alarm
          for (i=0; i<16; i++)   //#define MAX_CHANNUM   16  //The maximum number of channels
          {
            if (struAlarmInfo.dwChannel[i] == 1)
            {
              printf("Channel Number with Motion Detection Alarm %d\n", i+1);
            }
          }
          break;
        default:
          break;
      }
      break;
    }
    case COMM_UPLOAD_PLATE_RESULT:
    {
      NET_DVR_PLATE_RESULT struPlateResult={0};
      memcpy(&struPlateResult, pAlarmInfo, sizeof(struPlateResult));
      printf("License Plate Number: %s\n", struPlateResult.struPlateInfo.sLicense);//License plate number

      switch(struPlateResult.struPlateInfo.byColor)//License plate color
      {
      case VCA_BLUE_PLATE:
        printf("Vehicle Color: Blue\n");
        break;
      case VCA_YELLOW_PLATE:
        printf("Vehicle Color: Yellow\n");
        break;
      case VCA_WHITE_PLATE:
        printf("Vehicle Color: White\n");
        break;
      case VCA_BLACK_PLATE:
        printf("Vehicle Color: Black\n");
        break;
      default:
```

```
                break;
            }
            //Scene picture
            if (struPlateResult.dwPicLen != 0 && struPlateResult.byResultType == 1 )
            {
                sprintf(filename,"testpic_%d.jpg",iNum);
                fSnapPic=fopen(filename,"wb");
                fwrite(struPlateResult.pBuffer1,struPlateResult.dwPicLen,1,fSnapPic);
                iNum++;
                fclose(fSnapPic);
            }
            //License plate picture
            if (struPlateResult.dwPicPlateLen != 0 && struPlateResult.byResultType == 1)
            {
                sprintf(filename,"testPicPlate_%d.jpg",iNum);
                fSnapPicPlate=fopen(filename,"wb");
                fwrite(struPlateResult.pBuffer1,struPlateResult.dwPicLen,1,fSnapPicPlate);
                iNum++;
                fclose(fSnapPicPlate);
            }
            //Processing other data...
            break;
        }
        case COMM_ITS_PLATE_RESULT:
        {
            NET_ITS_PLATE_RESULT struITSPlateResult={0};
            memcpy(&struITSPlateResult, pAlarmInfo, sizeof(struITSPlateResult));

            for (i=0;i<struITSPlateResult.dwPicNum;i++)
            {
                printf("License Plate Number: %s\n", struITSPlateResult.struPlateInfo.sLicense);//License plate number
                switch(struITSPlateResult.struPlateInfo.byColor)//License plate color
                {
                case VCA_BLUE_PLATE:
                    printf("Vehicle Color: Blue\n");
                    break;
                case VCA_YELLOW_PLATE:
                    printf("Vehicle Color: Yellow\n");
                    break;
                case VCA_WHITE_PLATE:
                    printf("Vehicle Color: White\n");
                    break;
                case VCA_BLACK_PLATE:
                    printf("Vehicle Color: Black\n");
                    break;
                default:
                    break;
                }
                //Save scene picture
                if ((struITSPlateResult.struPicInfo[i].dwDataLen != 0)&&(struITSPlateResult.struPicInfo[i].byType== 1)||
(struITSPlateResult.struPicInfo[i].byType == 2))
                {
                    sprintf(filename,"testITSpic%d_%d.jpg",iNum,i);
                    fSnapPic=fopen(filename,"wb");
                    fwrite(struITSPlateResult.struPicInfo[i].pBuffer, struITSPlateResult.struPicInfo[i].dwDataLen,1,fSnapPic);
                    iNum++;
                    fclose(fSnapPic);
                }
                //License plate thumbnails
                if ((struITSPlateResult.struPicInfo[i].dwDataLen != 0)&&(struITSPlateResult.struPicInfo[i].byType == 0))
                {
                    sprintf(filename,"testPicPlate%d_%d.jpg",iNum,i);
                    fSnapPicPlate=fopen(filename,"wb");
                    fwrite(struITSPlateResult.struPicInfo[i].pBuffer, struITSPlateResult.struPicInfo[i].dwDataLen, 1, \
fSnapPicPlate);
                    iNum++;
                    fclose(fSnapPicPlate);
                }
                //Processing other data...
            }
            break;
        }
    default:
        break;
    }
}

void CALLBACK MessageCallbackNo2(LONG lCommand, NET_DVR_ALARMER *pAlarmer, char *pAlarmInfo, DWORD
dwBufLen, void* pUser)
{
    int i=0;
    char filename[100];
    FILE *fSnapPic=NULL;
    FILE *fSnapPicPlate=NULL;

    //This sample code is for reference only. Actually, it is not recommended to process the data and save file in the
callback function directly.
    //You'd better process the data in the message response funcion via message mode (PostMessage).

    switch(lCommand)
    {
        case COMM_ALARM:
        {
            NET_DVR_ALARMINFO struAlarmInfo;
            memcpy(&struAlarmInfo, pAlarmInfo, sizeof(NET_DVR_ALARMINFO));
            switch (struAlarmInfo.dwAlarmType)
            {
                case 3: //Motion detection alarm
                    for (i=0; i<16; i++)   //#define MAX_CHANNUM   16  //The maximum number of channel
                    {
                        if (struAlarmInfo.dwChannel[i] == 1)
                        {
                            printf("Channel No. with Motion Detection Alarm %d\n", i+1);
                        }
```

```
         }
         break;
      default:
         break;
   }
   break;
}
case COMM_UPLOAD_PLATE_RESULT:
{
   NET_DVR_PLATE_RESULT struPlateResult={0};
   memcpy(&struPlateResult, pAlarmInfo, sizeof(struPlateResult));
   printf("License Plate Number: %s\n", struPlateResult.struPlateInfo.sLicense);//License plate number

   switch(struPlateResult.struPlateInfo.byColor)//License plate color
   {
   case VCA_BLUE_PLATE:
      printf("Vehicle Color: Blue\n");
      break;
   case VCA_YELLOW_PLATE:
      printf("Vehicle Color: Yellow\n");
      break;
   case VCA_WHITE_PLATE:
      printf("Vehicle color: White\n");
      break;
   case VCA_BLACK_PLATE:
      printf("Vehicle Color: Black\n");
      break;
   default:
      break;
   }
   //Scene picture
   if (struPlateResult.dwPicLen != 0 && struPlateResult.byResultType == 1 )
   {
      sprintf(filename,"testpic_%d.jpg",iNum);
      fSnapPic=fopen(filename,"wb");
      fwrite(struPlateResult.pBuffer1,struPlateResult.dwPicLen,1,fSnapPic);
      iNum++;
      fclose(fSnapPic);
   }
   //License plate picture
   if (struPlateResult.dwPicPlateLen != 0 && struPlateResult.byResultType == 1)
   {
      sprintf(filename,"testPicPlate_%d.jpg",iNum);
      fSnapPicPlate=fopen(filename,"wb");
      fwrite(struPlateResult.pBuffer1,struPlateResult.dwPicLen,1,fSnapPicPlate);
      iNum++;
      fclose(fSnapPicPlate);
   }
   //Processing other data...
   break;
}
case COMM_ITS_PLATE_RESULT:
        {
            NET_ITS_PLATE_RESULT struITSPlateResult={0};
            memcpy(&struITSPlateResult, pAlarmInfo, sizeof(struITSPlateResult));

            for (i=0;i<struITSPlateResult.dwPicNum;i++)
            {
                printf("License Plate Number: %s\n", struITSPlateResult.struPlateInfo.sLicense);//License plate number
                switch(struITSPlateResult.struPlateInfo.byColor)//License plate color
                {
                case VCA_BLUE_PLATE:
                    printf("Vehicle Color: Blue\n");
                    break;
                case VCA_YELLOW_PLATE:
                    printf("Vehicle Color: Yellow\n");
                    break;
                case VCA_WHITE_PLATE:
                    printf("Vehicle Color: White\n");
                    break;
                case VCA_BLACK_PLATE:
                    printf("Vehicle Color: Black\n");
                    break;
                default:
                    break;
                }
                //Save scene picture
                if ((struITSPlateResult.struPicInfo[i].dwDataLen != 0)&&(struITSPlateResult.struPicInfo[i].byType== 1)||
(struITSPlateResult.struPicInfo[i].byType == 2))
                {
                    sprintf(filename,"testITSpic%d_%d.jpg",iNum,i);
                    fSnapPic=fopen(filename,"wb");
                    fwrite(struITSPlateResult.struPicInfo[i].pBuffer, struITSPlateResult.struPicInfo[i].dwDataLen,1,fSnapPic);
                    iNum++;
                    fclose(fSnapPic);
                }
                //License plate thumbnails
                if ((struITSPlateResult.struPicInfo[i].dwDataLen != 0)&&(struITSPlateResult.struPicInfo[i].byType == 0))
                {
                    sprintf(filename,"testPicPlate%d_%d.jpg",iNum,i);
                    fSnapPicPlate=fopen(filename,"wb");
                    fwrite(struITSPlateResult.struPicInfo[i].pBuffer, struITSPlateResult.struPicInfo[i].dwDataLen, 1, \
fSnapPicPlate);
                    iNum++;
                    fclose(fSnapPicPlate);
                }
                //Processing other data...
            }
            break;
        }
    default:
        break;
    }
}
```

```
void main() {

  //---------------------------------------
  //Initialize
  NET_DVR_Init();
  //Set the connection time and reconnection time
  NET_DVR_SetConnectTime(2000, 1);
  NET_DVR_SetReconnect(10000, true);

  //---------------------------------------
  //Log in to device
  LONG lUserID;
  NET_DVR_DEVICEINFO_V30 struDeviceInfo;
  lUserID = NET_DVR_Login_V30("172.0.0.100", 8000, "admin", "12345", &struDeviceInfo);
  if (lUserID < 0)
  {
       printf("Login error, %d\n", NET_DVR_GetLastError());
       NET_DVR_Cleanup();
       return;
  }

  //Set alarm callback function
  NET_DVR_SetDVRMessageCallBack_V50(0, MessageCallbackNo1, NULL);
  NET_DVR_SetDVRMessageCallBack_V50(1, MessageCallbackNo2, NULL);

  //Enable arming
  NET_DVR_SETUPALARM_PARAM struSetupParam={0};
  struSetupParam.dwSize=sizeof(NET_DVR_SETUPALARM_PARAM);

  //Alarm information type to upload: 0-History Alarm (NET_DVR_PLATE_RESULT), 1-Real-Time Alarm
(NET_ITS_PLATE_RESULT)
  struSetupParam.byAlarmInfoType=1;
  //Arming Level: Level-2 arming (for traffic device)
  struSetupParam.byLevel=1;

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

  Sleep(20000);
  //Disarm uploading channel
  if (!NET_DVR_CloseAlarmChan_V30(lHandle))
  {
     printf("NET_DVR_CloseAlarmChan_V30 error, %d\n", NET_DVR_GetLastError());
     NET_DVR_Logout(lUserID);
     NET_DVR_Cleanup();
```

```
    return;
  }

  //User logout
  NET_DVR_Logout(lUserID);
  //Release SDK resource
  NET_DVR_Cleanup();
  return;
}
```

## See Also

***NET_DVR_SetupAlarmChan_V50***

### 3.26.1 MSGCallBack

Alarm/event information callback function.

## Callback Function Definition

```
typedef void(CALLBACK *MSGCallBack)(
  LONG             lCommand,
  NET_DVR_ALARMER    *pAlarmer,
  char             *pAlarmInfo,
  DWORD            dwBufLen,
  void             *pUser
);
```

## Parameters

**lCommand**

[OUT] Uploaded message type. You can distinguish the alarm/event information via the type.

**pAlarmer**

[OUT] Alarm device information, including serial No., IP address, login handle, and so on, see details in ***NET_DVR_ALARMER*** .

**pAlarmInfo**

[OUT] Alarm/event information, the details are returned in different structures according to **lCommand**.

**dwBufLen**

[OUT] Size of alarm/event information buffer.

**pUser**

[OUT] User data.

## 3.27 NET_DVR_SetupAlarmChan_V50

Set up persistent connection to receive alarm/event information (supports alarm/event subscription).

### API Definition

```
LONG NET_DVR_SetupAlarmChan_V50(
  LONG                    lUserID,
  NET_DVR_SETUPALARM_PARAM_V50    lpSetupParam,
  char                   *pData,
  DWORD                   dwDataLen,
);
```

### Parameters

**lUserID**

[IN] Value returned by ***NET_DVR_Login_V40*** .

**lpSetupParam**

[IN] Arming parameters, refer to the structure ***NET_DVR_SETUPALARM_PARAM_V50*** for details.

**pData**

[IN] Alarm/event subscription conditions.

**dwDataLen**

[IN] Length of alarm/event subscription conditions.

### Return Values

Return *-1* for failure, and return other values as the handles of ***NET_DVR_CloseAlarmChan_V30*** . If *-1* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.

### Remarks

This API supports alarm/event subscription, you can specify the types of alarm or event to be uploaded by device by setting **pData** and **dwDataLen**.

## 3.28 NET_DVR_StartListen_V30

Register callback function for receiving alarm/event information and start listening (supports multiple threads).

### API Definition

```
LONG NET_DVR_StartListen_V30(
  char         *sLocalIP,
  WORD         wLocalPort,
  MSGCallBack   DataCallback,
  void         *pUserData
);
```

### Parameters

**sLocalIP**

[IN] IP address of local PC. It can be set to null.

**wLocalPort**

[IN] Listening port No. of local PC. It is configured by user, and it should be the same with that of device.

**DataCallback**

[IN] Alarm/event information callback function, see details in ***MSGCallBack*** .

**pUserData**

[IN] User data.

### Return Values

Return *-1* for failure, and return other values for the handle parameters of ***NET_DVR_StopListen_V30*** .
If *-1* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.
The available error codes of this API are 0, 3, 6, 12, 17, 41, 44, 47, 72, and 75. See details in the ***Device Network SDK Errors*** .

### Remarks

- To receive the alarm/event information sent by device, you should set the management host server address or listening host server address of device to the IP address of PC (which is same with the **sLocalIP**), or set the management host server port or listening host server port to the listening port No. of PC (which is same with the **wLocalPort**).
- The callback function in this API is prior to other callback functions, that is, if the callback function is configured in this API, other callback functions will not receive the alarm information. All the device alarm information is returned in same callback function, and you can distinguish the devices via the alarm device information (**pAlarmInfo**).

## 3.29 NET_DVR_StopListen_V30

Stop listening (supports multiple threads).

### API Definition

```
BOOL NET_DVR_StopListen_V30(
  LONG    lListenHandle
);
```

## Parameters

**lListenHandle**

Listening handle, which is returned by ***NET_DVR_StartListen_V30*** .

## Return Values

Return *TRUE* for success, and return *FALSE* for failure.
If *FALSE* is returned, you can call ***NET_DVR_GetLastError*** to get the error code.
The available error codes of this API are 0, 3, 12, and 17. See details in the ***Device Network SDK Errors*** .

# Chapter 4 Structure and Enumeration

## 4.1 Data Structure

### 4.1.1 NET_ALARM_CVR_SUBINFO_UNION

### Union about CVR Alarm Information

| Member | Data Type | Description |
|---|---|---|
| **byLen** | BYTE[] | Union size, the maximum array length is 492 bytes. |
| **struRecordLost** | ***NET_ALARM_RECORDFILE_LOSS*** | Video loss alarm information, the value of **dwAlarmType** in ***NET_DVR_ALARMINFO_DEV_V40*** is 8. |
| **struStreamException** | ***NET_ALARM_STREAM_EXCEPTION*** | Streaming exception alarm information, the value of **dwAlarmType** in ***NET_DVR_ALARMINFO_DEV_V40*** is 9. |
| **struResourceUsage** | ***NET_ALARM_RESOURCE_USAGE*** | Resource usage alarm information, the value of **dwAlarmType** in ***NET_DVR_ALARMINFO_DEV_V40*** is 10. |
| **struRecordException** | ***NET_ALARM_RECORD_EXCEPTION*** | Recording exception alarm information, the value of **dwAlarmType** in ***NET_DVR_ALARMINFO_DEV_V40*** is 12. |

### 4.1.2 NET_ALARM_RECORD_EXCEPTION

### Structure about Recording Exception Alarm Information

| Member | Data Type | Description |
|---|---|---|
| **byReason** | BYTE | Exception reason: 0-video volume full, 1-video volume exception, 2-no available video volume. |
| **byRes1** | BYTE[] | Reserved, set to 0. The maximum array length is 3 bytes. |

| Member | Data Type | Description |
|---|---|---|
| **sVolumeName** | BYTE[] | Video volume name, the maximum array length is "MAX_VOLUMENAME_LEN" (32 bytes). |
| **dwVolumeID** | DWORD | Video volume ID, or HDD No. |
| **byRes** | BYTE[] | Reserved, set to 0. The maximum array length is 452 bytes. |

## 4.1.3 NET_ALARM_RECORDFILE_LOSS

### Structure about Video Loss Alarm Information

| Member | Data Type | Description |
|---|---|---|
| **struInspectStart** | ***NET_DVR_TIME_EX*** | Start time of video loss check. |
| **struInspectEnd** | ***NET_DVR_TIME_EX*** | End time of video loss check. |
| **struIP** | ***NET_DVR_IPADDR_UNION*** | IP address of video loss channel. |
| **dwChanNo** | DWORD | Channel No. |
| **dwIDIndex** | DWORD | Encoder ID. |
| **sName** | BYTE[] | Encoder name, the maximum array length is "STREAM_ID_LEN" (32 bytes). |
| **struLossStartTime** | ***NET_DVR_TIME_EX*** | Start time of video loss. |
| **struLossEndTime** | ***NET_DVR_TIME_EX*** | End time of video loss. |
| **dwLostNum** | DWORD | Number of lost video files, 0xffffffff-all video files are lost. |
| **byRes** | BYTE[] | Reserved, set to 0. The maximum array length is 240 bytes. |

## 4.1.4 NET_ALARM_RESOURCE_USAGE

### Structure about Resource Usage Alarm Information

| Member | Data Type | Description |
|---|---|---|
| **byLevel** | BYTE | Usage alarm level: 0-normal, 1-alarm level 1, 2-alarm level 2, 3-alarm level 3. |
| **byRes** | BYTE[] | Reserved, set to 0. The maximum array length is 491 bytes. |

## 4.1.5 NET_ALARM_STREAM_EXCEPTION

### Structure about Video Exception Alarm Information

| Member | Data Type | Description |
|---|---|---|
| **struIP** | ***NET_DVR_IPADDR_UNION*** | IP address of video exception channel. |
| **dwChanNo** | DWORD | Channel No. |
| **dwIDIndex** | DWORD | Encoder ID. |
| **sName** | BYTE[] | Encoder name, the maximum array length is "STREAM_ID_LEN" (32 bytes). |
| **byExceptionCase** | BYTE | Exception reason: 0-data writing exception, 1-network exception. |
| **byRes** | BYTE[] | Reserved, set to 0. The maximum array length is 307 bytes. |

## 4.1.6 NET_DVR_ALARMER

### Alarm Device Information Structure

| Member | Data Type | Description |
|---|---|---|
| byUserIDValid | BYTE | Whether the user ID is valid: 0-no, 1-yes |
| bySerialValid | BYTE | Whether the serial No. is valid: 0-no, 1-yes |
| byVersionValid | BYTE | Whether the version No. is valid: 0-no, 1-yes |
| byDeviceNameValid | BYTE | Whether the device name is valid: 0-no, 1-yes |

| Member | Data Type | Description |
|---|---|---|
| byMacAddrValid | BYTE | Whether the MAC address is valid: 0-no, 1-yes |
| byLinkPortValid | BYTE | Whether the login port No. is valid: 0-no, 1-yes |
| byDeviceIPValid | BYTE | Whether the device IP address is valid: 0-no, 1-yes |
| bySocketIPValid | BYTE | Whether the Socket IP address is valid: 0-no, 1-yes |
| lUserID | LONG | Value returned by ***NET_DVR_Login_V40*** , it is valid when arming. |
| sSerialNumber | Array of BYTE | Serial No. |
| dwDeviceVersion | DWORD | Version information |
| sDeviceName | Array of char | Device name |
| byMacAddr | Array of BYTE | MAC address |
| wLinkPort | WORD | Device communication port No. |
| sDeviceIP | Array of char | Device IP address |
| sSocketIP | Array of char | Socket IP address when actively uploading alarm. |
| byIpProtocol | BYTE | Network protocol: 0-IPv4, 1-IPv6 |
| byRes2 | Array of BYTE | Reserved, set to 0. |

### 4.1.7 NET_DVR_ALARMINFO_DEV

### Device Alarm Information Structure

| Memeber | Data Type | Description |
|---|---|---|
| **dwAlarmType** | DWORD | Alarm types: 0-alarm input alarm of encoder, 1-second private volume damaged, 2-NVR disconnected, 3-encoder exception, 4-system clock exception, 5-the remaining capacity of the recording volume is too low, 6-motion detection alarm of encoder or encoding channel, 7-video |

| Memeber | Data Type | Description |
|---|---|---|
| | | tampering alarm of encoder or encoding channel. |
| **struTime** | | Alarm time |
| **byRes** | Array of BYTE | Reserved, set to 0. |
| **dwNumber** | DWORD | Number of alarm triggered channels. |
| **pNO** | WORD* | Channel No. or disk No., which ranges from 0 to 65535. |

### Remarks

For **pNO**: if **dwAlarmType** is 0, 3, 6, or 7, it may be channel No.; if **dwAlarmType** is 5, it may be disk No.

## 4.1.8 NET_DVR_ALARMINFO_DEV_V40

### Structure about CVR Alarm Information

| Member | Data Type | Description |
|---|---|---|
| **dwAlarmType** | DWORD | Alarm categories: 0-alarm input alarm of encoder, 1-second private volume damaged, 2-NVR disconnected, 3-encoder exception, 4-system clock exception, 5-the remaining capacity of the recording volume is too low, 6-motion detection alarm of encoder or encoding channel, 7-video tampering alarm of encoder or encoding channel, 8-video loss alarm, 9-real-time health monitoring alarm, 10-usage alarm, 11-CVR exception recovered, 12-recording exception. |
| **struTime** | ***NET_DVR_TIME*** | Alarm time |
| **uSubAlarmInfo** | ***NET_ALARM_CVR_SUBINFO_UNION*** | CVR alarm information structure, and it is valid when the alarm type is 8, 9, 10, and 12. |
| **byRes** | Array of BYTE | Reserved, set to 0. The maximum size is 256 bytes. |

| Member | Data Type | Description |
|---|---|---|
| **dwNumber** | DWORD | Number of alarm triggered channels. |
| **pNO** | WORD* | Channel No. or disk No., which ranges from 0 to 65535. |

### Remarks

For **pNO**: if **dwAlarmType** is 0, 3, 6, or 7, it may be channel No.; if **dwAlarmType** is 5, it may be disk No.

## 4.1.9 NET_DVR_ALARMINFO_V30

### Structure About Uploaded Alarm Information

| Member | Data Type | Description |
|---|---|---|
| dwAlarmType | DWORD | Alarm types: 0-alarm input alarm of encoder, 1-second private volume damaged, 2-NVR disconnected, 3-encoder exception, 4-system clock exception, 5-the remaining capacity of the recording volume is too low, 6-motion detection alarm of encoder or encoding channel, 7-video tampering alarm of encoder or encoding channel, 8-video loss alarm, 9-real-time health monitoring alarm, 10-usage alarm, 11-CVR exception recovered, 12-recording exception. |
| dwAlarmInputNumber | DWORD | Alarm input No., it is valid when alarm type is 0 or 23 |
| byAlarmOutputNumber | Array of BYTE | The triggered alarm output No. E.g. dwAlarmOutputNumber[0]==1 indicates that alarm output No.1 is triggered; dwAlarmOutputNumber[1]==1 indicates that alarm output No.2 is triggered. |
| byAlarmRelateChannel | Array of BYTE | The triggered recording channel No.: 0-not triggered, 1-triggered. E.g. dwAlarmRelateChannel[0]==1 indicates that the channel No.1 is triggered to record. |

| Member | Data Type | Description |
|---|---|---|
| byChannel | Array of BYTE | Alarm channel, it is valid when alarm type is 2, 3, 6, 9, 10 or 11. E.g. dwChannel[0]==1 indicates that the channel No. is in alarm. |
| byDiskNumber | Array of BYTE | Alarm HDD, it is valid when alarm type is 1, 4, or 5. E.g. dwDiskNumber [0]==1 indicates that the HDD No.1 is abnormal. |

## Remarks

The time interval to upload the alarm of face picture library changed is 1 hour; for other alarm type, the alarm information is uploaded in real-time, and the time interval is 1s. Currently, editing the time interval is not supported.

## 4.1.10 NET_DVR_ALARMINFO_V40

## Structure About Uploaded Alarm Information

| Member | Data Type | Description |
|---|---|---|
| struAlarmFixedHeader | ***NET_DVR_ALRAM_FIXED_HEADER*** | Constant content in alarm information, see details in the structure . |
| pAlarmData | DWORD* | Variable content in alarm information |

## Remarks

- The time interval to upload the alarm of face picture library changed is 1 hour; for other alarm type, the alarm information is uploaded in real-time, and the time interval is 1s. Currently, editing the time interval is not supported.
- The content of **pAlarmData** varies with the value of **dwAlarmType** in the structure ***NET_DVR_ALRAM_FIXED_HEADER*** , see details in the table below:

**Table 4-1 Relations Between pAlarmData and dwAlarmType**

| dwAlarmType | Description | pAlarmData |
|---|---|---|
| 0, 23 | Alarm input alarm, pulse alarm | dwTrigerAlarmOutNum*(DWORD) Alarm output No., |

| dwAlarmType | Description | pAlarmData |
|---|---|---|
| | | +dwTrigerRecordChanNum*(DWORD) Channel No. |
| 2, 3, 6, 9, 10, 11, 13, 15, 16, 19 | Video loss, motion detection, video tampering alarm, video exception, recording exception, scene change, resolution mismatched, VCA detection, PoE power supply exception, audio loss | dwAlarmChanNum*(DWORD) channel No. |
| 1, 4, 5 | HDD full, HDD uninitialized, writing to HDD failed | dwAlarmHardDiskNum*(DWORD) HDD No. |
| 7, 8, 12, 17, 18, 24, 25, 26 | Standard mismatches, invalid login, array exception, education sharing system alarm, two-way audio request alarm, face library HDD exception, face library changed, picture changed in face picture library | None |

### 4.1.11 NET_DVR_ALRAM_FIXED_HEADER

### Structure About Constant Alarm Information

| Member | Data Type | Description |
|---|---|---|
| dwAlarmType | DWORD | Alarm information type: 0-alarm input alarm, 1-HDD full, 2-video loss, 3-motion detection, 4-HDD unformatted, 5-writing to HDD failed, 6-video tampering alarm, 7-standard mismatched, 8-invalid login, 9-video exception, 10-recording exception, 11-scene change, 12-RAID exception, 13-resolution mismatched, 15-VCA detection, 16- PoE power supply exception, 17-education sharing system alarm, 18-two-way audio request alarm, 23-pulse alarm, 24-face picture library HDD exception, 25-face picture library changed, 26-picture of face picture library changed, 27-POC exception, 28-camera FOV |

| Member | Data Type | Description |
|---|---|---|
| | | exception, 30-no SD card, 31-supply voltage exception, 32-PTZ locked |
| struAlarmTime | ***NET_DVR_TIME_EX*** | Alarm time |
| uStruAlarm | Union ( ***Table 4-2*** ) | Alarm information union |
| pRes | DWORD* | Reserved. |
| byTimeDiffFlag | BYTE | Whether the time difference parameter is valid: 0-invalid, 1-valid. |
| cTimeDifferenceH | char | Time difference between time and UTC time, unit: hour, the value is between -12 and +14 ("+" indicates the east time zone), it is valid when **byISO8601** is "1". |
| cTimeDifferenceM | char | Time difference between time and UTC time, unit: minute, the value is -30, +30, or +45 ("+" indicates the east time zone), it is valid when **byISO8601** is "1". |
| byRes | Array of BYTE | Reserved, set to 0. The maximum size is 5 bytes. |

**Table 4-2 Union about Alarm Information Structures (uStruAlarm)**

| Member | Data Type | Description |
|---|---|---|
| byUnionLen | Array of BYTE | Union size, which is 116 bytes. |
| struIOAlarm | Struct ( ***Table 4-3*** ) | Structure about alarm input parameters |
| struAlarmChannel | Struct ( ***Table 4-4*** ) | Structure about alarm channel parameters |
| struAlarmHardDisk | Struct ( ***Table 4-5*** ) | Structure about HDD alarm parameters |
| struRecordingHost | Struct ( ***Table 4-6*** ) | Structure about alarm parameters of education sharing system |
| struVoltageInstable | Struct ( ***Table 4-7*** ) | Structure about alarm parameters of supply voltage exception |
| struPTLocking | Struct ( ***Table 4-8*** | Structure about parameters of PTZ locked alarm |

**Table 4-3 Structure about Alarm Input Parameters (struIOAlarm)**

| Member | Data Type | Description |
|---|---|---|
| dwAlarmInputNo | DWORD | Alarm input No. |
| dwTrigerAlarmOutNum | DWORD | The number of triggered alarm outputs. It is used for calculating the number of all triggered alarm outputs by **pAlarmData** in ***NET_DVR_ALARMINFO_V40*** , each alarm output is represented by 4 bytes. |
| dwTrigerRecordChanNum | DWORD | The number of triggered recording channels. It is used for calculating the number of all triggered recording channels by **pAlarmData** of ***NET_DVR_ALARMINFO_V40*** , each channel is represented by 4 bytes. |

**Table 4-4 Structure about Alarm Channel Parameters (struAlarmChannel)**

| Member | Data Type | Description |
|---|---|---|
| dwAlarmChanNum | DWORD | The number of alarm channels. It is used for calculating the number of all alarm channels by **pAlarmData** of ***NET_DVR_ALARMINFO_V40*** , each alarm channel is represented by 4 bytes. |
| dwPicLen | DWORD | Size of JPEG picture. |
| byPicURL | BYTE | Picture data format: 0-binary data, 1-URL. |
| byTarget | BYTE | Detection target type: 0-not supported, 1-person, 2-vehicle. |
| byRes1 | Array of BYTE | Reserved, the maximum size is 2 bytes. |
| pDataBuff | char* | Alarm picture data or URL. The pointer size is 8 bytes. |
| byRes3 | Array of BYTE | Reserved, the maximum size is 4 bytes. This member is only available for 64-bit Window operating system and 64-bit Linux operating system. |

**Table 4-5 Structure about HDD Alarm Parameters (struAlarmHardDisk)**

| Member | Data Type | Description |
|---|---|---|
| dwAlarmHardDiskNum | DWORD | The number of alarm HDD. It is used for calculating the number of all alarm HDDs by **pAlarmData** of ***NET_DVR_ALARMINFO_V40*** , each alarm HDD is represented by 4 bytes. |

**Table 4-6 Structure about Alarm Parameters of Education Sharing System (struRecordingHost)**

| Member | Data Type | Description |
|---|---|---|
| bySubAlarmType | BYTE | Alarm minor type: 1-one-touch post-record |
| byRes1 | Array of BYTE | Reserved, set to 0. The maximum size is 3 bytes. |
| struRecordEndTime | ***NET_DVR_TIME_EX*** | Recording end time. |

**Table 4-7 Structure about Alarm Parameters of Supply Voltage Exception (struVoltageInstable)**

| Member | Data Type | Description |
|---|---|---|
| fVoltageValue | float | Supply voltage, unit: V, corrects to one decimal place. |
| byVoltageAlarmType | BYTE | Supply voltage exception type: 0-high supply voltage, 1-low supply voltage |
| byRes1 | Array of BYTE | Reserved, set to 0. The maximum size is 3 bytes. |

**Table 4-8 Structure about Parameters of PTZ Locked Alarm (struPTLocking)**

| Member | Data Type | Description |
|---|---|---|
| fTemperature | float | Sensor temperature, which is accurate to one decimal place. |
| dwCustomInfoLength | DWORD | Custom information length. |
| pCustomInfo | BYTE* | Custom information. |
| byType | BYTE | PTZ locked direction: 1-panning is locked, 2-tilting is locked. |
| byDeicingEnabled | BYTE | Whether to enable heat for PTZ: 0-no, 1-yes. |

## Remarks

**dwAlarmType**==0, 23 corresponds to the structure struIOAlarm; **dwAlarmType**== 2/3/6/9/10/11/13/15/16/28 corresponds to the structure struAlarmChannel; **dwAlarmType**==

1/4/5 corresponds to the structure struAlarmHardDisk; **dwAlarmType**== 17 corresponds to the structure struRecordingHost; **dwAlarmType**== 31 corresponds to the structure struVoltageInstable; for other value, the union is not available.

### 4.1.12 NET_DVR_ALARM_ISAPI_INFO

#### Structure about Alarm Information Transmitted Based on Text Protocol

| Member | Data Type | Description |
|---|---|---|
| **pAlarmData** | char* | Alarm information based on text protocol (XML or JSON message without binary data). |
| **dwAlarmDataLen** | DWORD | Alarm data length. |
| **byDataType** | BYTE | Alarm data type: 0-invalid, 1-XML, 2-JSON. |
| **byPicturesNumber** | BYTE | The number of pictures (number of **pPicPackData** returned). When this member is 1, only one structure of ***NET_DVR_ALARM_ISAPI_PICDATA*** will be returned by **pPicPackData**. When this member is larger than 1, multiple structures of ***NET_DVR_ALARM_ISAPI_PICDATA*** will be returned by **pPicPackData**. |
| **byRes** | Array of BYTE | Reserved, set to 0. The maximum size is 2 bytes. |
| **pPicPackData** | void* | Alarm picture structure, see ***NET_DVR_ALARM_ISAPI_PICDATA*** for details. |
| **byRes** | Array of BYTE | Reserved. The maximum size is 32 bytes. |

#### Remarks

When enabling the listening mode, you should call the network configuration API based on text protocol to set the IP address for the listening service.

### 4.1.13 NET_DVR_ALARM_ISAPI_PICDATA

#### Structure about Alarm Picture Data Transmitted Based on Text Protocol

| Member | Data Type | Description |
|---|---|---|
| **dwPicLen** | DWORD | Alarm picture data length. |
| **byRes** | Array of BYTE | Reserved, set to 0. The maximum size is 4 bytes. |
| **szFilename** | Array of char | Picture file saving path, including file name. The maximum size is 256 bytes. |
| **pPicData** | BYTE* | Pointer that pointing to the uploaded image data. |

### 4.1.14 NET_DVR_ACS_ALARM_INFO

#### Structure about Access Control Alarm/Event Information

| Member | Data Type | Description |
|---|---|---|
| **dwSize** | DWORD | Structure size. |
| **dwMajor** | DWORD | Major alarm/event types, see details in ***Access Control Event Types*** . |
| **dwMinor** | DWORD | Minor alarm/event types, see details in ***Access Control Event Types*** . |
| **struTime** | ***NET_DVR_TIME*** | Alarm time information. |
| **sNetUser** | Array [BYTE] | User name for network operation. The maximum size is 16 bytes (the value of the macro definition "MAX_ NAMELEN"). |
| **struRemoteHostAddr** | ***NET_DVR_IPADDR_UNION*** | IP address of the remote access controller. |

| Member | Data Type | Description |
|---|---|---|
| **struAcsEventInfo** | ***NET_DVR_ACS_EVENT_INFO*** | Access control event details. |
| **dwPicDataLen** | DWORD | Picture size, 0: no picture, non-0: picture data exist. |
| **pPicData** | char* | Picture data. |
| **wInductiveEventType** | WORD | Inductive event type, 0-invalid. The alarm event types will be distinguished according to the inductive event type if **wInductiveEventType** is not 0; otherwise, the alarm event types will be distinguished according to **dwMajor** and **dwMinor**. |
| **byPicTransType** | BYTE | Picture data transmission mode: 0-binary, 1-URL. |
| **byRes1** | BYTE | Reserved. |
| **dwIOTChannelNo** | DWORD | IOT channel No. |
| **pAcsEventInfoExtend** | char* | When **byAcsEventInfoExtend** is set to 1, it points to the structure ***NET_DVR_ACS_EVENT_INFO_EXTEND*** . |
| **byAcsEventInfoExtend** | BYTE | Whether **pAcsEventInfoExtend** is valid: 0-no, 1-yes. |
| **byTimeType** | BYTE | Time type: 0-device's local time, 1-UTC time (it is the same as **struTime**). |
| **byRes2** | BYTE | Reserved. |
| **byAcsEventInfoExtendV20** | BYTE | Whether the member **pAcsEventInfoExtendV20** is valid: 0-invalid, 1-valid. If this member is valid, the member **byAcsEventInfoExtend** must be valid. |

| Member | Data Type | Description |
|---|---|---|
| **pAcsEventInfoExtendV20** | char* | When **byAcsEventInfoExtendV20** is set to 1, it points to the structure ***NET_DVR_ACS_EVENT_INFO_EXTEND_V20*** . |
| **byRes** | Array [BYTE] | Reserved, set to 0. The maximum size is 4 bytes. |

### 4.1.15 NET_DVR_ACS_EVENT_CFG

Access control event parameter structure.

## Structure Definition

```
struct{
  DWORD                dwSize;
  DWORD                dwMajor;
  DWORD                dwMinor;
  NET_DVR_TIME             struTime;
  BYTE               sNetUser[MAX_NAMELEN/*16*/];
  NET_DVR_IPADDR       struRemoteHostAddr;
  NET_DVR_ACS_EVENT_DETAIL  struAcsEventInfo;
  DWORD                dwPicDataLen;
  char               *pPicData;
  BYTE                byTimeType;
  BYTE                byRes1;
  DWORD                dwQRCodeInfoLen;
  DWORD                dwVisibleLightDataLen;
  DWORD                dwThermalDataLen;
  char               *pQRCodeInfo;
  char               *pVisibleLightData;
  char               *pThermalData;
  BYTE                byRes[36];
}NET_DVR_ACS_EVENT_CFG, *LPNET_DVR_ACS_EVENT_CFG;
```

## Members

**dwSize**

Structure size.

**dwMajor**

Event major types, see details in ***Access Control Event Types*** .

**dwMinor**

Event minor types, see details in ***Access Control Event Types*** .

**struTime**

Time information, see ***NET_DVR_TIME*** for details.

**sNetUser**

User name.

**struRemoteHostAddr**

IP address of remote access controller, see ***NET_DVR_IPADDR_UNION*** for details.

**struAcsEventInfo**

Access control event details, see ***NET_DVR_ACS_EVENT_DETAIL*** for details.

**dwPicDataLen**

Picture size, non-0: picture data exists.

**pPicData**

Picture data.

**byTimeType**

Time type: 0-device local time (default), 1-UTC time (which is same as **struTime**).

**byRes1**

Reserved.

**dwQRCodeInfoLen**

Length of the QR code information. If this member is not 0, it indicates that there is QR code information data following after.

**dwVisibleLightDataLen**

Length of the visible light picture captured by the thermal camera. If this member is not 0, it indicates that there is visible light picture data following after.

**dwThermalDataLen**

Length of the thermal picture. If this member is not 0, it indicates that there is thermal picture data following after.

**pQRCodeInfo**

Pointer of the QR code information.

**pVisibleLightData**

Pointer of the visible light picture captured by the thermal camera.

**pThermalData**

Pointer of the thermal picture.

**byRes**

Reserved, set to 0.

### 4.1.16 NET_DVR_ACS_EVENT_COND

Condition structure about getting access control events.

#### Structure Definition

```
struct{
 DWORD          dwSize;
 DWORD          dwMajor;
 DWORD          dwMinor;
 NET_DVR_TIME   struStartTime;
 NET_DVR_TIME   struEndTime;
 BYTE        byCardNo[ACS_CARD_NO_LEN/*32*/];
 BYTE        byName[NAME_LEN/*32*/];
 BYTE        byPicEnable;
 BYTE        byTimeType;
 BYTE        byRes2[2];
 DWORD          dwBeginSerialNo;
 DWORD          dwEndSerialNo;
 DWORD          dwIOTChannelNo;
 WORD          wInductiveEventType;
 BYTE        bySearchType;
 BYTE        byEventAttribute;
 char        szMonitorID[NET_SDK_MONITOR_ID_LEN/*64*/];
 BYTE        byEmployeeNo[NET_SDK_EMPLOYEE_NO_LEN/*32*/];
 BYTE        byRes[140];
}NET_DVR_ACS_EVENT_COND,*LPNET_DVR_ACS_EVENT_COND;
```

#### Members

**dwSize**

Structure size.

**dwMajor**

Event major types, see details in ***Access Control Event Types*** , 0-all.

**dwMinor**

Event minor types, see details in ***Access Control Event Types*** , 0-all.

**struStartTime**

Start time, see ***NET_DVR_TIME*** for details.

**struEndTime**

End time, see ***NET_DVR_TIME*** for details.

**byCardNo**

Card No.

**byName**

Cardholder name.

**byPicEnable**

Whether contain pictures: 0-no, 1-yes. If this member is set to 0, all events that meet the requirements will be uploaded without pictures. If this member is set to 1, for all events that meet the requirements, the event information will be uploaded if there is no linkage picture, and the event information along with the linkage pictures will be uploaded if there are any.

**byTimeType**

Time type: 0-device local time (default), 1-UTC time (which is same as **struStartTime** and **struEndTime**).

**byRes2**

Reserved, set to 0.

**dwBeginSerialNo**

Start serial No.: 0-all.

**dwEndSerialNo**

End serial No.: 0-all.

**dwIOTChannelNo**

IOT channel No., 0-invalid.

**wInductiveEventType**

Inductive event type, 0-invalid. The alarm event types will be distinguished according to the inductive event type if **wInductiveEventType** is not 0; otherwise, the alarm event types will be distinguished according to **dwMajor** and **dwMinor**.

**bySearchType**

Search mode: 0-reserved, 1-search by event source (the channel No. is the non-video channel No.), 2-search by monitoring resource ID.

**byEventAttribute**

Event attribute: 0-undefined, 1-valid authentication, 2-other.

**szMonitorID**

Monitoring resource ID which consists of device serial No., channel type, and No. For example, the access point ID is device serial No.+"DOOR"+door No.

**byEmployeeNo**

Employee No. (person ID)

**byRes**

Reserved, set to 0.

## 4.1.17 NET_DVR_ACS_EVENT_DETAIL

Access control event details structure.

## Structure Definition

```
struct{
  DWORD        dwSize;
  BYTE         byCardNo[ACS_CARD_NO_LEN/*32*/];
  BYTE         byCardType;
  BYTE         byAllowListNo;
  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         byMACAddr[MACADDR_LEN/*6*/];
  BYTE         bySwipeCardType;
  BYTE         byEventAttribute;
  DWORD        dwSerialNo;
  BYTE         byChannelControllerID;
  BYTE         byChannelControllerLampID;
  BYTE         byChannelControllerIRAdaptorID;
  BYTE         byChannelControllerIREmitterID;
  DWORD        dwRecordChannelNum;
  char         *pRecordChannelData;
  BYTE         byUserType;
  BYTE         byCurrentVerifyMode;
  BYTE         byAttendanceStatus;
  BYTE         byStatusValue;
  BYTE         byEmployeeNo[NET_SDK_EMPLOYEE_NO_LEN/*32*/];
  BYTE         byRes1;
  BYTE         byMask;
  BYTE         byThermometryUnit;
  BYTE         byIsAbnomalTemperature;
  float        fCurrTemperature;
  NET_VCA_POINT  struRegionCoordinates;
  BYTE      byHealthCode;
  BYTE      byRes[47];
}NET_DVR_ACS_EVENT_DETAIL, *LPNET_DVR_ACS_EVENT_DETAIL;
```

## Members

**dwSize**

Structure size.

**byCardNo**

Card No.: 0-invalid.

**byCardType**

Card types: 0-invalid, 1-normal card, 2-disabled card, 3-blocklist card, 4-patrol card, 5-duress card, 6-super card, 7-visitor card.

**byAllowListNo**

Allowlist No., which is between 1 and 8, but if the value is 0, it is invalid.

**byReportChannel**

Event uploading channel types: 0-invalid, 1-upload in arming mode, 2-upload by central group 1, 3-upload by central group 2.

**byCardReaderKind**

Authentication device types: 0-invalid, 1-IC card reader, 2-ID card reader, 3-QR code scanner, 4-fingerprint module.

**dwCardReaderNo**

Authentication device No.: 0-invalid.

**dwDoorNo**

Door or floor No.: 0-invalid. For Turnstile (swing barrier), door No.1 refers to entrance, and door No.2 refers to exist.

**dwVerifyNo**

Multiple authentication No.: 0-invalid

**dwAlarmInNo**

Alarm input No.: 0-invalid

**dwAlarmOutNo**

Alarm output No.: 0-invalid

**dwCaseSensorNo**

Event trigger No.

**dwRs485No**

RS485 channel No.: 0-invalid.

**dwMultiCardGroupNo**

Group No.

**wAccessChannel**

Turnstile No.

**byDeviceNo**

Device No.: 0-invalid.

**byDistractControlNo**

Distributed controller No.: 0-invalid.

**dwEmployeeNo**

Employee No.: 0-invalid.

**wLocalControllerID**

Distributed access controller No.: 0-access controller, 0 to 64: distributed access controller.

**byInternetAccess**

Network interface No.: 1-upstream network interface No.1, 2-upstream network interface No.2, 3-downstream network interface No.1.

**byType**

Zone types: 0-instant alarm zone, 1-24-hour alarm zone, 2-delayed zone, 3-internal zone, 4-key zone, 5-fire alarm zone, 6-perimeter protection, 7-24-hour silent alarm zone, 8-24-hour auxiliary zone, 9-24-hour shock alarm zone, 10-emergency door open alarm zone, 11-emergency door closed alarm zone, off-none

**byMACAddr**

Physical address, 0-invalid.

**bySwipeCardType**

Card swiping type: 0-invalid, 1-QR code.

**byEventAttribute**

Event attribute: 0-undefined, 1-valid authentication, 2-other.

**dwSerialNo**

Event serial No.: 0-invalid, which is used to judge whether the event loss occurred.

**byChannelControllerID**

Lane controller No.: 0-invalid, 1-main lane controller, 2-sub-lane controller.

**byChannelControllerLampID**

Light board No. of lane controller, which is between 1 and 255, 0-invalid

**byChannelControllerIRAdaptorID**

IR adaptor No. of lane controller, which is between 1 and 255, 0-invalid.

**byChannelControllerIREmitterID**

Active infrared intrusion detector No. of lane controller, which is between 1 and 255, 0-invalid.

**dwRecordChannelNum**

Number of recording channels.

**pRecordChannelData**

Recording channel, the size depends on **dwRecordChannelNum**.

**byUserType**

Person type: 0-invalid, 1-resident, 2-visitor, 3-person in blocklist, 4-administrator.

**byCurrentVerifyMode**

Authentication mode: 0-invalid, 1-sleepy, 2-card+password, 3-card, 4-card or password, 5-fingerprint, 6-fingerprint+password, 7-fingerprint or card, 8-fingerprint+card, 9-fingerprint+card +password, 10-face or fingerprint or card or password, 11-face+fingerprint, 12-face+password, 13-face+card, 14-face, 15-employee No.+password, 16-fingerprint or password, 17-employee No.+fingerprint, 18-employee No.+fingerprint+password, 19-face+fingerprint+card, 20-face +password+fingerprint, 21-employee No.+face, 22-face or face+card, 23-fingerprint or face, 24-card or face or password, 25-card or face, 26-card or face or fingerprint, 27-card or fingerprint or password.

**byAttendanceStatus**

Attendance status: 0-undefined, 1-check in, 2-check out, 3-break out, 4-break in, 5-overtime in, 6-overtime out.

**byStatusValue**

Attendance status value.

**byEmployeeNo**

Employee No. (person ID). Both **byEmployeeNo** and **dwEmployeeNo** should be transferred by the device. The **byEmployeeNo** will be parsed by the upper-level platform or client first. If the **byEmployeeNo** is NULL, the **dwEmployeeNo** will be parsed.

**byRes1**

Reserved.

**byMask**

Whether the person is wearing mask or not: 0-reserved, 1-unknown, 2-not wearing mask, 3-wearing mask.

**byThermometryUnit**

Temperature unit: 0-Celsius (default), 1-Fahrenheit, 2-Kelvin.

**byIsAbnomalTemperature**

Whether the face temperature is abnormal: 1-yes, 0-no.

**fCurrTemperature**

Face temperature which is accurate to one decimal place.

**struRegionCoordinates**

Face temperature's coordinates, see details in the structure ***NET_VCA_POINT*** .

**byHealthCode**

Health code status: 0 (no request), 1 (no health code), 2 (green QR code), 3 (yellow QR code), 4 (red QR code), 5 (no such person), 6 (other error, e.g., searching failed due to API exception), 7 (searching for the health code timed out).

**byRes**

Reserved, set to 0.

## 4.1.18 NET_DVR_ACS_EVENT_INFO

### Structure about Extended Access Control Event Details

| Member | Data Type | Description |
|---|---|---|
| **dwSize** | DWORD | Structure size. |
| **byCardNo** | Array [BYTE] | Card No., 0-invalid. Some special cards' numbers are listed as the follows: "18446744073709551613"-supper card, "18446744073709551614"-duress card, "18446744073709551615"-invalid card. The maximum size is 32 bytes (the value of the macro definition "ACS_CARD_NO_LEN"). |
| **byCardType** | BYTE | Card types: 0-invalid, 1-normal card, 2-disability card, 3-blocklist card, 4-patrol card, 5-duress card, 6-super card, 7-visitor card, 8-dismiss card. |
| **byAllowListNo** | BYTE | Allowlist No., which is between 1 and 8, but if the value is 0, it is invalid. |
| **byReportChannel** | BYTE | Event uploading channel types: 0-invalid, 1-upload in arming mode, 2-upload by central group 1, 3-upload by central group 2. |

| Member | Data Type | Description |
|---|---|---|
| **byCardReaderKind** | BYTE | Authentication device types: 0-invalid, 1-IC card reader, 2-ID card reader, 3-QR code scanner, 4-fingerprint module. |
| **dwCardReaderNo** | DWORD | Authentication device No.: 0-invalid. |
| **dwDoorNo** | DWORD | Door or floor No.: 0-invalid. For turnstile (swing barrier), door No.1 refers to entrance, and door No.2 refers to exit. |
| **dwVerifyNo** | DWORD | Multiple authentication No.: 0-invalid. |
| **dwAlarmInNo** | DWORD | Alarm input No.: 0-invalid. |
| **dwAlarmOutNo** | DWORD | Alarm output No.: 0-invalid. |
| **dwCaseSensorNo** | DWORD | Event trigger No. |
| **dwRs485No** | DWORD | RS-485 channel No.: 0-invalid. |
| **dwMultiCardGroupNo** | DWORD | Group No. |
| **wAccessChannel** | WORD | Turnstile No. |
| **byDeviceNo** | BYTE | Device No.: 0-invalid. |
| **byDistractControlNo** | BYTE | Distributed controller No.: 0-invalid. |
| **dwEmployeeNo** | DWORD | Employee ID: 0-invalid. |
| **wLocalControllerID** | WORD | Distributed access controller No.: 0-access controller, 0 to 64: distributed access controller. |
| **byInternetAccess** | BYTE | Network interface No.: 1-upstream network interface No.1, 2-upstream network interface No.2, 3-downstream network interface No.1. |
| **byType** | BYTE | Zone types: 0-instant zone, 1-24-hour zone, 2-delayed zone, |

| Member | Data Type | Description |
|---|---|---|
| | | 3-internal zone, 4-key zone, 5-fire alarm zone, 6-perimeter zone, 7-24-hour silent zone, 8-24-hour auxiliary zone, 9-24-hour shock zone, 10-emergency door open alarm zone, 11-emergency door closed alarm zone, 0xff-none. |
| **byMACAddr** | Array [BYTE] | Physical address, 0-invalid. The maximum size is 6 bytes (the value of the macro definition "MACADDR_LEN"). |
| **bySwipeCardType** | BYTE | Card swiping type: 0-invalid, 1-QR code. |
| **byMask** | BYTE | Whether the person is wearing mask: 0-reserved, 1-unknown, 2-not wearing mask, 3-wearing mask. |
| **dwSerialNo** | DWORD | Event serial No.: 0-invalid, which is used to check whether the event loss occurred. |
| **byChannelControllerID** | BYTE | Lane controller No.: 0-invalid, 1-main lane controller, 2-sub-lane controller. |
| **byChannelControllerLampID** | BYTE | Light board No. of the lane controller, which is between 1 and 255, 0-invalid. |
| **byChannelControllerIRAdaptorID** | BYTE | IR adaptor No. of the lane controller, which is between 1 and 255, 0-invalid. |
| **byChannelControllerIREmitterID** | BYTE | Active infrared intrusion detector No. of the lane controller, which is between 1 and 255, 0-invalid. |

| Member | Data Type | Description |
|---|---|---|
| **byHelmet** | BYTE | Whether the person is wearing hard hat: 1-unknown, 2-no, 3-yes. |
| **byHealthCode** | BYTE | Health code status: 0 (no request), 1 (no health code), 2 (green QR code), 3 (yellow QR code), 4 (red QR code), 5 (no such person), 6 (other error, e.g., searching failed due to API exception), 7 (searching for the health code timed out). |
| **byRes** | Array [BYTE] | Reserved, set to 0. The maximum size is 2 bytes. |

## 4.1.19 NET_DVR_ACS_EVENT_INFO_EXTEND

### Structure about Extended Access Control Event Information

| Member | Data Type | Description |
|---|---|---|
| dwFrontSerialNo | DWORD | Event serial No., 0-invalid. If this member is set to 0, the platform will check whether the event is lost by **dwSerialNo**; otherwise, the platform will check whether the event is lost by both **dwFrontSerialNo** and **dwSerialNo**. This member is used for discontinuous **dwSerialNo** after alarm subscription. |
| byUserType | BYTE | Person type: 0-invalid, 1-normal person (resident), 2-visitor, 3-person in the blocklist, 4-administrator. |
| byCurrentVerifyMode | BYTE | Current authentication mode of the card reader: 0-invalid, 1-sleepy, 2-card+password, 3-card, 4-card or password, 5-fingerprint, 6-fingerprint+password, 7-fingerprint or card, 8-fingerprint+card, 9-fingerprint+card+password, 10-face or fingerprint or card or password, 11-face+fingerprint, 12-face+password, 13-face |

| Member | Data Type | Description |
|---|---|---|
| | | +card, 14-face, 15-employee No.+password, 16-fingerprint or password, 17-employee No.+fingerprint, 18-employee No.+fingerprint+password, 19-face+fingerprint+card, 20-face+password+fingerprint, 21-employee No.+face, 22-face or face+card, 23-fingerprint or face, 24-card or face or password, 25-card or face, 26-card or face or fingerprint, 27-card or fingerprint or password. |
| byCurrentEvent | BYTE | Whether it is a real-time event: 0-invalid, 1-yes (real-time event), 2-no (offline event). |
| byPurePwdVerifyEnable | BYTE | Whether the device supports opening the door only by password: 1-yes, 0-no.<br>For opening the door only by password: 1. The password in "XXX or password" in the authentication mode refers to the person's password (the value of the node **password** in JSON_UserInfo); 2. The device will not check the duplication of the password, and the upper platform should ensure that the password is unique; 3. The password cannot be added, deleted, edited, or searched for on the device locally. |
| byEmployeeNo | BYTE[] | Employee No. (person ID). Both **byEmployeeNo** and **dwEmployeeNo** should be transferred by the device. The **byEmployeeNo** will be parsed by the upper-layer platform or client first. If the **byEmployeeNo** is not configured, the **dwEmployeeNo** will be parsed. The maximum length is "NET_SDK_EMPLOYEE_NO_LEN" (32 bytes). |
| byAttendanceStatus | BYTE | Attendance status: 0-undefined, 1-check in, 2-check out, 3-break out, 4-break in, 5-overtime in, 6-overtime out. |
| byStatusValue | BYTE | Attendance status value. |
| byRes2 | BYTE[] | Reserved. The maximum length is 2 bytes. |

| Member | Data Type | Description |
|---|---|---|
| byUUID | BYTE[] | UUID, this member is only used when accessing EZVIZ platform. The maximum length is "NET_ SDK_UUID_LEN" (36 bytes). |
| byDeviceName | BYTE[] | Device serial No. The maximum length is "NET_ DEV_NAME_LEN" (64 bytes). |
| dwBodyTemp | DWORD | Skin-surface temperature, which equals to actual temperature value (a float number) × 1000. |
| byMaskEnabled | BYTE | Whether the person is wearing mask: 1 (yes), 2 (no). |
| byRes | BYTE[] | Reserved. The maximum length is 19 bytes. |

## See Also

***NET_DVR_ACS_ALARM_INFO***

## 4.1.20 NET_DVR_ACS_EVENT_INFO_EXTEND_V20

### Structure about Extended Access Control Event Information (V20)

| Member | Data Type | Description |
|---|---|---|
| **byRemoteCheck** | BYTE | Whether remote verification is required: 0-invalid, 1-no (default), 2-yes. |
| **byThermometryUnit** | BYTE | Temperature unit: 0-Celsius (default), 1-Fahrenheit, 3-Kelvin. |
| **byIsAbnomalTemperature** | BYTE | Whether the face temperature is abnormal: 1-yes, 0-no. |
| **byRes2** | BYTE | Reserved. |
| **fCurrTemperature** | float | Face temperature, it is accurate to one decimal place. |
| **struRegionCoordinates** | ***NET_VCA_POINT*** | Face temperature's coordinates. |

| Member | Data Type | Description |
|---|---|---|
| **dwQRCodeInfoLen** | DWORD | Data size of the QR code information. If this member is not 0, it indicates that the QR code information data exists. |
| **dwVisibleLightDataLen** | DWORD | Data size of the visible light picture captured by the thermal camera. If this member is not 0, it indicates that the visible light picture data exists. |
| **dwThermalDataLen** | DWORD | Data size of the thermal picture. If this member is not 0, it indicates that the thermal picture data exists. |
| **pQRCodeInfo** | char* | Pointer of the QR code information. |
| **pVisibleLightData** | char* | Pointer of the visible light picture captured by the thermal camera. |
| **pThermalData** | char* | Pointer of the thermal picture. |
| **byAttendanceLabel** | Array [BYTE] | Custom attendance name. The maximum size is 64 bytes. |
| **byRes** | Array [BYTE] | Reserved. The maximum size is 960 bytes. |

## 4.1.21 NET_DVR_ACS_EXTERNAL_DEV_CFG

Structure about the peripheral parameters of access controller.

### Structure Definition

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

## Members

**dwSize**

Structure Size.

**byIDCardUpMode**

Mode of uploading ID card information: 0-upload 18-digit ID card No., 1-upload all information.

**byRes1**

Reserved, set to 0.

**byCardVerifyMode**

Card authentication mode: 0-authenticate by remote center, 1-authenticate by client or platform.

**byACSDevType**

Device type: 1-ID card reader, 2-IC card reader, 3-QR code reader, 4-fingerprint reader, 5-character screen + QR code reader, 6-card collector, 7-character screen, 8-fingerprint scanner, 9-voice module, 10-person and ID card device.

**byDoorMode**

Door entrance or exit mode: 0-entrance, 1-exit.

**byRes2**

Reserved, set to 0.

**wDevDetailType**

External device model:

when **byACSDevType**=1: 0-iDR210, 1-IDM10, 2-ID card reader;

when **byACSDevType**=7: 0-DC48270RS043_01T, 1-DC80480B070_03T.

**byRes**

Reserved, set to 0.

## 4.1.22 NET_DVR_ACS_WORK_STATUS_V50

Access controller working status structure.

## Structure Definition

```
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/*8*/];
 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/*512*/];
 BYTE    byAlarmInStatus[MAX_ALARMHOST_ALARMIN_NUM/*512*/];
 BYTE    byAlarmOutStatus[MAX_ALARMHOST_ALARMOUT_NUM/*512*/];
 DWORD   dwCardNum;
 BYTE    byFireAlarmStatus;
 BYTE    byBatteryChargeStatus;
 BYTE    byMasterChannelControllerStatus;
 BYTE    bySlaveChannelControllerStatus;
 BYTE    byAntiSneakServerStatus;
 BYTE    byRes3[3];
 DWORD   dwAllowFaceNum;
 DWORD   dwBlockFaceNum;
 BYTE    byRes2[108];
}NET_DVR_ACS_WORK_STATUS_V50,*LPNET_DVR_ACS_WORK_STATUS_V50;
```

## Members

**dwSize**

Structure size

**byDoorLockStatus**

Lock status (or elevator relay status), 0-closed, 1-open, 2-short circuit alarm, 3-open circuit alarm, 4-exception alarm

**byDoorStatus**

Door status (or elevator status): 1-sleepy, 2-open (for elevator: free status), 3-closed (for elevator: disabled status), 4-normal (for elevator: controlled status).

**byMagneticStatus**

Magnet status: 0-closed, 1-open, 2-short circuit alarm, 3-open circuit alarm, 4-exception alarm.

**byCaseStatus**

Alarm input status: 0-no input, 1-with input.

**wBatteryVoltage**

Storage battery voltage, the actual value equals to the 10 multiples of **wBatteryVoltage**, unit: volt.

**byBatteryLowVoltage**

Whether the storage battery is in low voltage status: 0-no, 1-yes.

**byPowerSupplyStatus**

Device power supply status: 1-AC, 2-storage battery.

**byMultiDoorInterlockStatus**

Multi-door interlocking status: 0-disabled, 1-enabled.

**byAntiSneakStatus**

Anti-passing back status: 0-disabled, 1-enabled.

**byHostAntiDismantleStatus**

Controller tampering status: 0-disabled, 1-enabled.

**byIndicatorLightStatus**

Indicator status: 0-offline, 1-online.

**byCardReaderOnlineStatus**

Fingerprint and card reader status: 0-offline, 1-online.

**byCardReaderAntiDismantleStatus**

Fingerprint and card reader tampering status: 0-offline, 1-online.

**byCardReaderVerifyMode**

Authentication types: 0-invalid, 1-sleepy, 2-card+password, 3-card, 4-card or password, 5-fingerprint, 6-fingerprint+password, 7-fingerprint or card, 8-fingerprint+card, 9-fingerprint+card+password, 10-face+fingerprint+card+password, 11-face+fingerprint, 12-face+password, 13-face+card, 14-face, 15-employee ID+password, 16-fingerprint or password, 17-employee ID+fingerprint, 18-employee ID+fingerprint+password, 19-face+fingerprint+card, 20-face+fingerprint+password, 21-employee ID+face, 22-face/face+card, 23-fingerprint/face, 24-card/face/password.

**bySetupAlarmStatus**

Alarm input arming status: 0-disarmed, 1-armed

**byAlarmInStatus**

Alarm input status: 0-no alarm, 1-in alarm.

**byAlarmOutStatus**

Alarm output status: 0-no alarm, 1-in alarm.

**dwCardNum**

Number of added cards.

**byFireAlarmStatus**

Fire alarm status: 0-normal, 1-short circuit alarm, 2-open circuit alarm.

**byBatteryChargeStatus**

Battery charging status: 0-invalid, 1-charging, 2-unchanged.

**byMasterChannelControllerStatus**

Online status of main lane controller online status: 0-invalid, 1-offline, 2-online.

**bySlaveChannelControllerStatus**

Online status of sub-lane controller online status: 0-invalid, 1-offline, 2-online.

**byAntiSneakServerStatus**

Anti-passing back server status: 0-invalid, 1-disabled, 2-normal, 3-disconnected.

**byRes3**

Reserved, set to 0.

**dwAllowFaceNum**

The number of face pictures in allowlist.

**wBlockFaceNum**

The number of face pictures in blocklist.

**byRes2**

Reserved, set to 0

## 4.1.23 NET_DVR_AGAIN_RELATEDEV

Parameter structure of linked network device of doorphone

### Structure Definition

```
struct{
 NET_DVR_IPADDR   struSIPServer;
 NET_DVR_IPADDR   struCenterAddr;
 WORD             wCenterPort;
 BYTE             byRes1[2];
 NET_DVR_IPADDR   struIndoorUnit;
 NET_DVR_IPADDR   struAgainAddr;
 BYTE             byRes[444];
}NET_DVR_AGAIN_RELATEDEV,*LPNET_DVR_AGAIN_RELATEDEV;
```

### Members

**struSIPServer**

IP address of SIP server, refer to the structure ***NET_DVR_IPADDR_UNION*** for details.

**struCenterAddr**

IP address of platform or system, refer to the structure ***NET_DVR_IPADDR_UNION*** for details.

**byRes**

Reserved, set to 0.

**struIndoorUnit**

IP address of indoor station, refer to the structure ***NET_DVR_IPADDR_UNION*** for details.

**struAgainAddr**

IP address of main doorphone, refer to the structure ***NET_DVR_IPADDR_UNION*** for details.

**byRes**

Reserved, set to 0.

### See Also

## 4.1.24 NET_DVR_CAPTURE_FACE_CFG

Collected face data structure

### Structure Definition

```
struct{
  DWORD  dwSize;
  DWORD  dwFaceTemplate1Size;
  char   *pFaceTemplate1Buffer;
  DWORD  dwFaceTemplate2Size;
  char   *pFaceTemplate2Buffer;
  DWORD  dwFacePicSize;
  char   *pFacePicBuffer;
  BYTE   byFaceQuality1;
  BYTE   byFaceQuality2;
  BYTE   byCaptureProgress;
  BYTE   byFacePicQuality;
  DWORD  dwInfraredFacePicSize;
  char   *pInfraredFacePicBuffer;
  BYTE   byInfraredFacePicQuality;
  BYTE   byRes1[3];
  NET_DVR_FACE_FEATURE  struFeature;
  BYTE   byRes[56];
}NET_DVR_CAPTURE_FACE_CFG,*LPNET_DVR_CAPTURE_FACE_CFG;
```

### Members

**dwSize**

Structure size.

**dwFaceTemplate1Size**

Size of face data template 1. When its value is 0, it indicates that there is no data template 1.

**pFaceTemplate1Buffer**

Buffer to save face data template 1, the buffer size should be smaller than or equal to 2.5 KB.

**dwFaceTemplate2Size**

Size of face data template 2. When its value is 0, it indicates that there is no data template 2.

**pFaceTemplate2Buffer**

Buffer to save face data template 2, the buffer size should be smaller than or equal to 2.5 KB.

**dwFacePicSize**

Size of face picture data. When its value is 0, it indicates that there is no face picture data.

**pFacePicBuffer**

Buffer to save face picture data.

**byFaceQuality1**

Face picture quality, it is between 1 and 100.

**byFaceQuality2**

Face picture quality, it is between 1 and 100.

**byCaptureProgress**

Collection progress: 0-no face data collected, 1-collected. The face information can be parsed only when the progress value is 100.

**byFacePicQuality**

Face quality in the face picture.

**dwInfraredFacePicSize**

Size of infrared face picture data. When its value is 0, it indicates that there is no face picture data.

**pInfraredFacePicBuffer**

Buffer to save infrared face picture data.

**byInfraredFacePicQuality**

Face quality in the infrared face picture.

**byRes1**

Reserved.

**struFeature**

Feature information in the matted face picture, see details in the structure ***NET_DVR_FACE_FEATURE*** .

**byRes**

Reserved.

### 4.1.25 NET_DVR_CAPTURE_FACE_COND

Condition structure for collecting face data.

## Structure Definition

```
struct{
  DWORD   dwSize;
  BYTE    byRes[128];
}NET_DVR_CAPTURE_FACE_COND,*LPNET_DVR_CAPTURE_FACE_COND;
```

## Members

**dwSize**

Structure size.

**byRes**

Reserved.

### 4.1.26 NET_DVR_CAPTURE_FINGERPRINT_CFG

Fingerprint collection result structure

## Structure Definition

```
struct{
  DWORD   dwSize;
  DWORD   dwFingerPrintDataSize;
  BYTE    byFingerData[MAX_FINGER_PRINT_LEN/*768*/];
  DWORD   dwFingerPrintPicSize;
  char    *pFingerPrintPicBuffer;
  BYTE    byFingerNo;
  BYTE    byFingerPrintQuality;
  BYTE    byRes[62];
}NET_DVR_CAPTURE_FINGERPRINT_CFG, *LPNET_DVR_CAPTURE_FINGERPRINT_CFG;
```

## Members

**dwSize**

Structure size.

**dwFingerPrintDataSize**

Fingerprint data size.

**byFingerData**

Fingerprint details.

**dwFingerPrintPicSize**

Fingerprint picture size, 0-no fingerprint picture.

**pFingerPrintPicBuffer**

Buffer for saving fingerprint picture data.

**byFingerNo**

Finger No., which is between 1 and 10.

**byFingerPrintQuality**

Fingerprint quality, which is between 1 and 100.

**byRes**

Reserved, set to 0.

## 4.1.27 NET_DVR_CAPTURE_FINGERPRINT_COND

Fingerprint collection condition structure

### Structure Definition

```
struct{
  DWORD    dwSize;
  BYTE     byFingerPrintPicType;
  BYTE     byFingerNo;
  BYTE     byRes[126];
}NET_DVR_CAPTURE_FINGERPRINT_COND, *LPNET_DVR_CAPTURE_FINGERPRINT_COND;
```

### Members

**dwSize**

Structure size.

**byFingerPrintPicType**

Fingerprint picture type: 0-reserved.

**byFingerNo**

Finger No., which is between 1 and 10.

**byRes**

Reserved, set to 0.

## 4.1.28 NET_DVR_CARD_READER_CFG_V50

Fingerprint and card reader parameters structure.

### Structure Definition

```
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    byFingerPrintRecogizeInterval;
 BYTE    byFingerPrintMatchFastMode;
 BYTE    byFingerPrintModuleSensitive;
 BYTE    byFingerPrintModuleLightCondition;
 BYTE    byFaceMatchThresholdN;
 BYTE    byFaceQuality;
 BYTE    byFaceRecogizeTimeOut;
 BYTE    byFaceRecogizeInterval;
 WORD    wCardReaderFunction;
 BYTE    byCardReaderDescription[CARD_READER_DESCRIPTION/*32*/];
 WORD    wFaceImageSensitometry;
 BYTE    byLivingBodyDetect;
 BYTE    byFaceMatchThreshold1;
 WORD    wBuzzerTime;
 BYTE    byFaceMatch1SecurityLevel;
 BYTE    byFaceMatchNSecurityLevel;
 BYTE    byEnvirMode;
 BYTE    byLiveDetLevelSet;
 BYTE    byLiveDetAntiAttackCntLimit;
 BYTE    byEnableLiveDetAntiAttack;
 BYTE    bySupportDelFPByID;
 BYTE    byRes1;
 BYTE    byFaceContrastMotionDetLevel;
 BYTE    byDayFaceMatchThresholdN;
 BYTE    byNightFaceMatchThresholdN;
 BYTE    byFaceRecogizeEnable;
 BYTE    byBlockFaceMatchThreshold;
 BYTE    byRes3[2];
 BYTE    byDefaultVerifyMode;
 DWORD   dwFingerPrintCapacity;
 DWORD   dwFingerPrintNum;
 BYTE    byEnableFingerPrintNum;
 BYTE    byRes[231];
}NET_DVR_CARD_READER_CFG_V50,*LPNET_DVR_CARD_READER_CFG_V50;
```

## Members

**dwSize**

Structure size

**byEnable**

Whether to enable: 0-no, 1-yes.

**byCardReaderType**

Fingerprint and card reader types: 1-DS-K110XM/MK/C/CK, 7-Wiegand or RS485 offline, 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, 19-DS-K1102EM, 20- DS-K1102E, 21-DS-K1200EF, 22-DS-K1200MF, 23-DS-K1200CF, 33-DS-K1T200EF, 34- DS-K1T300EF

**byOkLedPolarity**

OK LED polarity: 0-negative pole, 1-positive pole.

**byErrorLedPolarity**

Error LED polarity: 0-negative pole, 1-positive pole.

**byBuzzerPolarity**

Buzzer polarity: 0-negative pole, 1-positive pole.

**bySwipeInterval**

Time interval of repeated authentication, which is valid for authentication modes such as fingerprint, card, face, etc., unit: second.

**byPressTimeout**

Button pressing timeout, unit: second, which is ranging from 1 to 255.

**byEnableFailAlarm**

Whether to enable excessive failed authentication attempts alarm: 0-no, 1-yes.

**byMaxReadCardFailNum**

Maximum number of failed authentication attempts, which is ranging from 1 to 10.

**byEnableTamperCheck**

Whether to enable tampering detection: 0-no, 1-yes.

**byOfflineCheckTime**

Offline detection time, unit: second, which is ranging from 0 to 255.

**byFingerPrintCheckLevel**

Fingerprint recognition level: 1-1/10 error rate, 2-1/100error rate, 3-1/1000error rate, 4-1/10000error rate, 5-1/100000error rate, 6-1/1000000error rate, 7-1/10000000error rate, 8-1/100000000error rate, 9-3/100error rate, 10-3/1000error rate, 11-3/10000error rate, 12-3/100000error rate, 13-3/1000000error rate, 14-3/10000000error rate, 15-3/100000000error rate, 16-Auto Normal, 17-Auto Secure, 18-Auto More Secure

**byUseLocalController**

Read-only, whether is it linked with distributed access controller or not? 0-no, 1-yes.

**byRes1**

Reserved, set to 0.

**wLocalControllerID**

Read-only, distributed access controller No. It is valid when **byUseLocalController** is 1, No.0 indicates that the controller is not registered, and the No. is ranging from 1 and 255.

**wLocalControllerReaderID**

Read-only, fingerprint and card reader No. of distributed access controller. It is valid when **byUseLocalController** is 1, No.0 indicates that the controller is not registered.

**wCardReaderChannel**

Read-only, communication channel No. of fingerprint an card reader: 0-Wiegand or offline, 1-RS485A, 2-RS485B. It is valid when **byUseLocalController** is 1.

**byFingerPrintImageQuality**

Fingerprint picture quality: 0-invalid, 1-low (V1), 2-medium (V1), 3-high (V1), 4-highest (V1), 5-low (V2), 6-medium (V2), 7-high (V2), 8-highest (V2).

**byFingerPrintContrastTimeOut**

Fingerprint picture comparison timeout: 0-invalid, 1 to 20-1 to 20 second, 0xff-unlimited.

**byFingerPrintRecogizeInterval**

Fingerprint picture comparison interval: 0-invalid, 1 to 10-1 to 10 second, 0xff-no delay.

**byFingerPrintMatchFastMode**

Fingerprint matching mode: 0-invalid, 1 to 5-fast mode 1 to fast mode 5, 0xff-auto.

**byFingerPrintModuleSensitive**

Fingerprint module sensitive: 0-invalid, 1 to 8-sensitive level 1 to level 8.

**byFingerPrintModuleLightCondition**

Fingerprint module light condition: 0-invalid, 1-outdoor, 2-indoor.

**byFaceMatchThresholdN**

Face picture comparison threshold. which is ranging from 0 to 100.

**byFaceQuality**

Face picture quality, which is ranging from 0 to 100.

**byFaceRecogizeTimeOut**

Face recognition timeout: 1 to 20-1s to 20s, 0xff-unlimited.

**byFaceRecogizeInterval**

Face recognition interval: 0-invalid, 1 to 10-1s to 10s, 0xff-no delay.

**wCardReaderFunction**

Read-only, fingerprint and card reader types, which is represented by bit: bit1-fingerprint, bit2-face, bit3-pulse; bit value: 0-no, 1-yes

**byCardReaderDescription**

Fingerprint and card reader description.

**wFaceImageSensitometry**

Read-only, face picture exposure, which is ranging from 0 to 65535.

**byLivingBodyDetect**

Live face detection: 0-invalid, 1-disable, 2-disable.

**byFaceMatchThreshold1**

Face picture 1:1 threshold, which is ranging from 0 to 100.

**wBuzzerTime**

Buzzing time, which is ranging from 0 to 5999s (0-long buzzing).

**byFaceMatch1SecurityLevel**

Face picture 1:1 security level: 0-invalid, 1-normal, 2-high, 3-higher

**byFaceMatchNSecurityLevel**

Face picture 1:N security level: 0-Invalid, 1-normal, 2-high, 3-higher

**byEnvirMode**

Face recognition environment mode: 0-invalid, 1-indoor, 2-other

**byLiveDetLevelSet**

Set threshold level of live face detection: 0-invalid, 1-low, 2-medium, 3-high

**byLiveDetAntiAttackCntLimit**

Anti-attacking times of live face detection: 0-invalid, ranging from 1 to 255.

**byEnableLiveDetAntiAttack**

Whether to enable the anti-attacking of live dace detection: 0-invalid, 1-no, 1-yes.

**bySupportDelFPByID**

Read-only, whether the fingerprint and card reader supports deleting fingerprint by finger ID: 0-invalid, 1-no, 2-yes.

**byRes1**

Reserved.

**byFaceContrastMotionDetLevel**

Motion detection level during face picture comparison: 0-invalid, 1-low, 2-medium, 3-high.

**byDayFaceMatchThresholdN**

1:N face picture comparison threshold in day, which is between 0 and 100.

**byNightFaceMatchThresholdN**

1:N face picture comparison threshold at night, which is between 0 and 100.

**byFaceRecogizeEnable**

Whether to enable facial recognition: 0-invalid, 1-yes (one face), 2-no, 3-yes (multiple faces).

**byBlockFaceMatchThreshold**

Face picture comparison threshold in blocklist, which is between 0 and 100.

**byRes3**

Reserved.

**byDefaultVerifyMode**

Default authentication mode of the fingerprint and card reader (factory settings), read-only: 1-sleepy, 2-card+password, 3-card, 4-card or password, 5-fingerprint, 6-fingerprint+password, 7-fingerprint or card, 8-fingerprint+card, 9-fingerprint+card+password, 10-face or fingerprint or card or password, 11-face+fingerprint, 12-face+password, 13-face+card, 14-face, 15-employee No.+password, 16-fingerprint or password, 17-employee No.+fingerprint, 18-employee No.+fingerprint+password, 19-face+fingerprint+card, 20-face+password+fingerprint, 21-employee No.+face, 22-face or face+card, 23-fingerprint or face, 24-card or face or password, 25-card or face, 26-card or face or fingerprint, 27-card or fingerprint or password.

**dwFingerPrintCapacity**

Read-only, fingerprint capability, it is valid only when **byEnableFingerPrintNum** is 1.

**dwFingerPrintNum**

Read-only, number of existing fingerprint pictures, it is valid only when **byEnableFingerPrintNum** is 1.

**byEnableFingerPrintNum**

Read-only, whether to enable fingerprint capability: 0-no, 1-yes.

**byRes**

Reserved, set to 0.

## 4.1.29 NET_DVR_CARD_CFG_SEND_DATA

Data structure to be sent for getting card information.

### Structure Definition

```
struct{
  DWORD   dwSize;
  BYTE    byCardNo[ACS_CARD_NO_LEN/*32*/];
  DWORD   dwCardUserId;
  BYTE    byRes[12];
}NET_DVR_CARD_CFG_SEND_DATA,*LPNET_DVR_CARD_CFG_SEND_DATA;
```

### Members

**dwSize**

Structure size.

**byCardNo**

Card number.

**dwCardUserId**

Card holder ID.

**byRes2**

Reserved, set to 0.

## 4.1.30 NET_DVR_CARD_CFG_V50

Card parameter structure

### Structure Definition

```
struct{
 DWORD                  dwSize;
 DWORD                  dwModifyParamType;
 BYTE                 byCardNo[ACS_CARD_NO_LEN/*32*/];
 BYTE                 byCardValid;
 BYTE                 byCardType;
 BYTE                 byLeaderCard;
 BYTE                 byUserType;
 BYTE                 byDoorRight[MAX_DOOR_NUM/*256*/];
 NET_DVR_VALID_PERIOD_CFG    struValid;
 BYTE                 byBelongGroup[MAX_GROUP_NUM/*128*/];
 BYTE                 byCardPassword[CARD_PASSWORD_LEN/*8*/];
 WORD                  wCardRightPlan[MAX_DOOR_NUM/*256*/][MAX_CARD_RIGHT_PLAN_NUM/*4*/];
 DWORD                  dwMaxSwipeTime;
 DWORD                  dwSwipeTime;
 WORD                  wRoomNumber;
 SHORT                 wFloorNumber;
 DWORD                  dwEmployeeNo;
 BYTE                 byName[NAME_LEN/*32*/];
 WORD                  wDepartmentNo;
 WORD                  wSchedulePlanNo;
 BYTE                 bySchedulePlanType;
 BYTE                 byRes2[3];
 DWORD                  dwLockID;
 BYTE                 byLockCode[MAX_LOCK_CODE_LEN/*8*/];
 BYTE                 byRoomCode[MAX_DOOR_CODE_LEN/*8*/];
 DWORD                  dwCardRight;
 DWORD                  dwPlanTemplate;
 DWORD                  dwCardUserId;
 BYTE                 byCardModelType;
 BYTE                 bySIMNum[NAME_LEN/*32*/];
 BYTE                 byRes3[51];
}NET_DVR_CARD_CFG_V50,*LPNET_DVR_CARD_CFG_V50;
```

## Members

### dwSize

Structure size.

### dwModifyParamType

Card parameters to be edited, it is valid when applying card information. Each bit represents a kind of parameters, bit value: 0-not edit. -1-edit.

| Macro Definition | Value | Description |
|---|---|---|
| CARD_PARAM_CARD_VALID | 0x00000001 | Card validation parameter |
| CARD_PARAM_VALID | 0x00000002 | Expiry date |
| CARD_PARAM_CARD_TYPE | 0x00000004 | Card type |
| CARD_PARAM_DOOR_RIGHT | 0x00000008 | Card permission |
| CARD_PARAM_LEADER_CARD | 0x00000010 | First card parameter |
| CARD_PARAM_SWIPE_NUM | 0x00000020 | Maximum card swiping times |
| CARD_PARAM_GROUP | 0x00000040 | Group parameter |
| CARD_PARAM_PASSWORD | 0x00000080 | Card password |
| CARD_PARAM_RIGHT_PLAN | 0x00000100 | Card permission control schedule |
| CARD_PARAM_SWIPED_NUM | 0x00000200 | Card swiped times |
| CARD_PARAM_EMPLOYEE_NO | 0x00000400 | Employee No. |
| CARD_PARAM_NAME | 0x00000800 | Name |
| CARD_PARAM_DEPARTMENT_NO | 0x00001000 | Department No. |
| CARD_SCHEDULE_PLAN_NO | 0x00002000 | Shift schedule No. |
| CARD_SCHEDULE_PLAN_TYPE | 0x00004000 | Shift schedule Type |
| CARD_ROOM_NUMBER | 0x00008000 | Room No. |
| CARD_SIM_NO | 0x00010000 | Mobile phone number |
| CARD_FLOOR_NUMBER | 0x00020000 | Floor No. |

### byCardNo

Card number, see the special card No. as follows:

0xFFFFFFFFFFFFFFFF: Illegal card No.

0xFFFFFFFFFFFFFFFE: Duress card No.

0xFFFFFFFFFFFFFFFD: Super card No.

0xFFFFFFFFFFFFFFFC to 0xFFFFFFFFFFFFFFF1: Reserved special card No. range

0xFFFFFFFFFFFFFFF0: Maximum valid card No.

**byCardValid**

Whether the card is valid: 0-no, 1-yes (used for deleting card).

**byCardType**

Card type, 1-normal card (default); 2-disabled card; 3-blocklist card; 4-patrol card; 5-duress card; 6-super card; 7- visitor card; 8-dismissing card; 9-employee card; 10-emergency card; 11-emergency management card (for assigning permission for temporary card, it cannot open door).

**byLeaderCard**

Whether it is the first card: 1-yes; 0-no.

**byUserType**

User type: 0-normal user, 1-admin user

**byDoorRight**

Door (elevator, lock) control permission, which is represented by bit, bit1-door (elevator, lock) 1, bit2-door (elevator, lock) 2, ..., value of bit: 1-with permission, 0-no permission.

**struValid**

Expiry date, refer to the structure ***NET_DVR_VALID_PERIOD_CFG*** for details.

**byBelongGroup**

Whether it belongs to a group, which is represented by bit, bit1-group 1, bit2-group 2, ..., bit value: 1-yes; 0-no.

**byCardPassword**

Card password.

**wCardRightPlan**

Access permission control schedule No.

**dwMaxSwipeTime**

Maximum card swiping times: 0-no limit.

**dwSwipeTime**

Card swiped times.

**wRoomNumber**

Room No.

**wFloorNumber**

Floor No.

**dwEmployeeNo**

Employee ID, which is between 1 and 99999999, it cannot be 0 and cannot be duplicated.

**byName**

Name

**wDepartmentNo**

Department No.

**wSchedulePlanNo**

Shift schedule No.

**bySchedulePlanType**

Shift schedule type: 0-reserved, 1-person, 2-department

**byRes2**

Reserved, set to 0.

**dwLockID**

Lock ID

**byLockCode**

Lock No.

**byRoomCode**

Room No., which is represented by bit, bit value: 0-no permission, 1-with permission

Bit 0: weak current alarm

Bit 1: audio prompt for open door

Bit 2: restricted guest card

Bit 3: channel

Bit 4: open double locked door

Bit 5: patrol

**dwCardRight**

Access permission.

**dwPlanTemplate**

Whether to enable the schedule: 0-no, 1-yes

**dwCardUserId**

Card holder ID

**byCardModelType**

0-reserved, 1-M1 S50, 2-M1 S70, 3- FM1208 CPU card, 4-FM1216 CPU card, 5-reserved, 6-identity card, 7-NFC

**byRes2**

Reserved, set to 0.

## Remarks

For fingerprint access control terminal (DS-K1T803F) and fingerprint time attendance terminal (DS-K1A801F), the following members **dwEmployeeNo**, **byName**, **wDepartmentNo**, **wSchedulePlanNo**, and **bySchedulePlanType** in the structure is required. For other access control device, they are optional.

## 4.1.31 NET_DVR_CARD_CFG_COND

Condition structure of card configuration.

## Structure Definition

```
struct{
  DWORD   dwSize;
  DWORD   dwCardNum;
  BYTE    byCheckCardNo;
  BYTE    byRes1[3];
  WORD    wLocalControllerID;
  BYTE    byRes2[2];
  DWORD   dwLockID;
  BYTE    byRes3[20];
}NET_DVR_CARD_CFG_COND,*LPNET_DVR_CARD_CFG_COND;
```

## Members

**dwSize**

Structure size.

**dwCardNum**

Number of cards to get or apply, 0xffffffff-all cards.

**byCheckCardNo**

Whether to enable card number verification: 0-no; 1-yes.

**byRes1**

Reserved, set to 0.

**wLocalControllerID**

Distributed access controller No., 0-access controller.

**byRes2**

Reserved, set to 0.

**dwLockID**

Lock ID.

**byRes3**

Reserved, set to 0.

## Remarks

When applying card information, if the member **byCheckCardNo** is set to "0", the device will not verify the card number applied by application layer, and the card number will be directly written to the local storage, which can improve the applying speed. But the application layer should make sure the card number is unique.

## 4.1.32 NET_DVR_CARD_READER_PLAN

Parameter structure about configuration of authentication mode control schedule.

## Structure Definition

```
struct{
  DWORD    dwSize;
  DWORD    dwTemplateNo;
  BYTE     byRes[64];
}NET_DVR_CARD_READER_PLAN,*LPNET_DVR_CARD_READER_PLAN;
```

## Members

**dwSize**

Structure size.

**dwTemplateNo**

Schedule template No.: 0-cancel linking template with schedule, and restore to the default settings (available for swiping card to open the door); non-0-link template with schedule by No.

**byRes**

Reserved, set to 0.

## 4.1.33 NET_DVR_DATE

Date information structure.

## Structure Definition

```
struct{
  WORD    wYear;
  BYTE    byMonth;
  BYTE    byDay;
}NET_DVR_DATE,*LPNET_DVR_DATE;
```

## Members

**wYear**

Year

**byMonth**

Month

**byDay**

Day

### 4.1.34 NET_DVR_DEVICEINFO_V30

Device parameter structure (V30).

## Device Parameter Structure (V30)

| Member | Data Type | Description |
|---|---|---|
| sSerialNumber | BYTE | Device serial No. |
| byAlarmInPortNum | BYTE | Number of analog alarm inputs |
| byAlarmOutPortNum | BYTE | Number of analog alarm outputs |
| byDiskNum | BYTE | Number of HDDs |
| byDVRType | BYTE | Device type |
| byChanNum | BYTE | Number of analog channels |
| byStartChan | BYTE | Start No. of analog channel, which starts from 1. |
| byAudioChanNum | BYTE | Number of two-way audio channels |
| byIPChanNum | BYTE | Number of digital channels, low 8-bit. |
| byZeroChanNum | BYTE | Number of channel-zero |
| byMainProto | BYTE | Transmission protocol type of main stream: 0-private protocol (default), 1-RTSP, 2-private protocol+RTSP |
| bySubProto | BYTE | Transmission protocol type of sub-stream: 0-private protocol (default), 1-RTSP, 2-private protocol+RTSP |
| bySupport | BYTE | Capabilities, if the result of bitwise operation is 0, it refers that the capability is not supported, |

| Member | Data Type | Description |
|---|---|---|
| | | if the result is 1, it indicates that the capability is supported.<br>• bySupport&0x1: whether supports VCA search.<br>• bySupport&0x2: whether supports backup.<br>• bySupport&0x4: whether supports getting encoding parameters.<br>• bySupport&0x8: whether supports dual-NIC.<br>• bySupport&0x10: whether supports remote SADP.<br>• bySupport&0x20: whether supports RAID card.<br>• bySupport&0x40: whether supports searching in IPSAN directory.<br>• bySupport&0x80: whether supports RTP over RTSP. |
| bySupport1 | BYTE | Extended capabilities, if the result of bitwise operation is 0, it refers that the capability is not supported, if the result is 1, it indicates that the capability is supported.<br>• bySupport1&0x1: whether supports SNMP with version 30.<br>• bySupport1&0x2: whether supports playback and downloading video files.<br>• bySupport1&0x4: whether supports setting the arming priority.<br>• bySupport1&0x8: whether supports extending the arming time period.<br>• bySupport1&0x10: whether supports multiple HDDs (more than 33).<br>• bySupport1&0x20: whether supports RTP over RTSP.<br>• bySupport1&0x80: whether supports license plate recognition alarm. |
| bySupport2 | BYTE | Extended capabilities, if the result of bitwise operation is 0, it refers that the capability is not supported, if the result is 1, it indicates that the capability is supported. |

| Member | Data Type | Description |
|---|---|---|
| | | • bySupport2&0x1: whether supports getting stream via URL.<br>• bySupport2&0x2: whether supports FTP with version 40.<br>• bySupport2&0x4: whether supports ANR.<br>• bySupport2&0x20: whether supports getting device status.<br>• bySupport2&0x40: whether supports encrypting stream. |
| wDevType | WORD | Device model |
| bySupport3 | BYTE | Extended capabilities, if the result of bitwise operation is 0, it refers that the capability is not supported, while, if the result is 1, it indicates that the capability is supported.<br>• bySupport3&0x1: whether supports multi-stream.<br>• bySupport3&0x4: whether supports configuring by group (e.g., image, alarm input, alarm output, user, device status, JPEG picture capture, continuous and scheduled capture, .HDD group management, and so on).<br>• bySupport3&0x20: whether supports getting stream via DDNS. |
| byMultiStreamProto | BYTE | Whether supports multi-stream, if the result of bitwise operation is 0, it refers to not support, if the result is 1, it refers to support.<br>• byMultiStreamProto&0x1: whether supports third-stream.<br>• byMultiStreamProto&0x2: whether supports fourth-stream.<br>• byMultiStreamProto&0x40: whether supports main stream.<br>• byMultiStreamProto&0x80: whether supports sub-stream. |
| byStartDChan | BYTE | Start No. of digital channel, 0-no digital channel (e.g., DVR, network camera). |

| Member | Data Type | Description |
|---|---|---|
| byStartDTalkChan | BYTE | Start No. of two-way audio channel, 0-no two-way audio channel. |
| byHighDChanNum | BYTE | Number of digital channels, high 8-bit. |
| bySupport4 | BYTE | Extended capabilities, if the result of bitwise operation is 0, it refers that the capability is not supported, if the result is 1, it indicates that the capability is supported.<br>• bySupport4&0x01: whether all stream types support RTSP and private protocol.<br>• bySupport4&0x02: whether the device supports transmitting form format data via API (NET_DVR_STDXMLConfig).<br>• bySupport4&0x10: whether supports loading network disk by domain name. |
| byLanguageType | BYTE | Supported language types, if the result of bitwise operation is 0, it refers to not support, if the result is 1, it refers to support.<br>• byLanguageType ==0: this field is not supported by device.<br>• byLanguageType&0x1: whether supports Chinese.<br>• byLanguageType&0x2: whether supports English. |
| byVoiceInChanNum | BYTE | Number of audio input channels |
| byStartVoiceInChanNo | BYTE | Start No. of audio input channel, 0-invalid. |
| byRes3 | Array of BYTE | Reserved, set to 0. |
| byMirrorChanNum | BYTE | Number of mirror channels |
| wStartMirrorChanNo | WORD | Start No. of mirror channel |
| byRes2 | Array of BYTE | Reserved, set to 0. |

## Remarks

- The maximum number of digital channels equal to byIPChanNum+byHighDChanNum*256.
- For login via text protocol, the following parameters are not supported: **byMainProto**, **bySubProto**, **bySupport**, **bySupport1**, **bySupport2**, **bySupport3**, **bySupport4**, **bySupport5**, **bySupport6**, **bySupport7**, **byMultiStreamProto**, **byStartDTalkChan**, **byVoiceInChanNum**, **byStartVoiceInChanNo**, **byMirrorChanNum**, and **wStartMirrorChanNo**.

## See Also

***NET_DVR_DEVICEINFO_V40***

## 4.1.35 NET_DVR_DEVICEINFO_V40

### Device Parameter Structure (V40)

| Member | Data Type | Description |
|---|---|---|
| struDeviceV30 | ***NET_DVR_DEVICEINFO_V30*** | Device parameters |
| bySupportLock | BYTE | Whether supports locking function: 1-support. |
| byRetryLoginTime | BYTE | Remaining login attempts, it is valid when the user name or password is incorrect and the **bySupportLock** is 1. |
| byPasswordLevel | BYTE | Password strength: 0-invalid, 1-default password, 2-valid password, 3-risky password. For default password or risky password, the users are reminded to change password. |
| byProxyType | BYTE | Proxy type: 0-no proxy, 1-standard proxy, 2-EHome proxy. |
| dwSurplusLockTime | DWORD | Remaining locking time, unit: second. It is valid only when **bySupportLock** is 1. During the locing time, if the user try to log in to again, the remaining locking time will resume to 30 minutes. |
| byCharEncodeType | BYTE | Character encodings. 0-no decoding information, 1-GB2312 (Simplified Chinese), 2-GBK, 3-BIG5 (Traditional Chinese), 4-Shift_JIS (Japanese), 5-EUC-KR (Korean), 6-UTF-8, 7-ISO8859-1, 8-ISO8859-2, 9-ISO8859-3, …, 21-ISO8859-15 (Western European) |
| bySupportDev5 | BYTE | Whether to support getting the parameters of devices that support HCNetSDK version 5.0 or above, the size of device name and type name are extended to 64 bytes. |

| Member | Data Type | Description |
|---|---|---|
| bySupport | BYTE | Whether it supports uploading changes, it depends on the result of bitwise AND (&) operation: 0-not support, 1-support. The result of **bySupport**&0x1 indicates that this member is reserved; the result of **bySupport**&0x2 indicates that whether it supports uploading changes: 0-not support, 1-support. This member is the capability set extension. |
| byLoginMode | BYTE | Login mode: 0-login via private protocol, 1-login via text protocol. For private protocol, the default login port number is 8000, and for text protocol, the default login port number is 80 or 443. |
| dwOEMCode | DWORD | OEM code. |
| iResidualValidity | int | Remaining valid days of the user's password, unit: day. If the negative number is returned, it indicates that the password being used has expired. For example, if -3 is returned, it indicates that the password being used has expired for three days. |
| byResidualValidity | BYTE | Whether the member **iResidualValidity** is valid: 0-invalid, 1-valid. |
| bySingleStartDTalkChan | BYTE | Start channel No. for connecting independent audio tracks to the device. The value 0 is reserved and invalid. The channel No. of audio tracks cannot start from 0. |
| bySingleDTalkChanNums | BYTE | Total number of channels of the device connected with independent tracks, 0-not support. |
| byPassWordResetLevel | BYTE | Whether to prompt the non-admin user to change the password: 0 (invalid), 1 (If the administrator creates a non-admin user account with an initial password, the non-admin user will be prompted "Please change the initial password" each time he/she logs in to the device until he/she changes the initial password), 2(If the non-admin user's password |

| Member | Data Type | Description |
|---|---|---|
| | | has been changed by the administrator, the non-admin user will be prompted "Please set a new password" each time he/she logs in to the device until he/she changes the password). |
| bySupportStreamEncrypt | BYTE | Whether it supports stream encryption, it depends on the result of bitwise AND (&) operation: 0-no, 1-yes. The result of **bySupportStreamEncrypt**&0x1 indicates whether to support RTP/TLS streaming, the result of **bySupportStreamEncrypt**&0x2 indicates whether to support SRTP/UDP streaming, and the result of **bySupportStreamEncrypt**&0x4 indicates whether to support SRTP/MULTICAST streaming. |
| byRes2 | Array of BYTE | Reserved, set to 0. |

## Remarks

- Four character types are allowed in the password, including digits, lowercase letters, uppercase letters and symbols. The maximum password length is 16 bits, and there are four password strength levels, see details below:
  - Level 0 (Risky Password): The password length is less than 8 bits, or only contains one kind of the character types. Or the password is the same with the user name, or is the mirror writing of the user name.
  - Level 1 (Weak Password): The password length is more than or equal to 8 bits, and contains two kinds of the character types. Meanwhile, the combination should be (digits + lowercase letters) or (digits + uppercase letters).
  - Level 2 (Medium Password): The password length is more than or equal to 8 bits, and contains two kinds of the character types. Meanwhile, the combination cannot be (digits + lowercase letters) and (digits + uppercase letters).
  - Level 3 (Strong Password): The password length is more than or equal to 8 bits, and at least contains three kinds of the character types.
- For login via text protocol, the following parameters are not supported: **bySupportLock**, **byRetryLoginTime**, **byPasswordLevel**, **byProxyType**, **dwSurplusLockTime**, **byCharEncodeType**, and **bySupportDev5**.

## 4.1.36 NET_DVR_DOOR_FILE_UPLOAD_PARAM

Structure about the parameters of the access control file to be uploaded.

### Structure Definition

```
struct{
 DWORD  dwSize;
 DWORD  dwFileSize;
 BYTE   byFileName[MAX_FILE_NAME_LEN/*100*/];
 BYTE   byRes1[256];
}NET_DVR_DOOR_FILE_UPLOAD_PARAM, *LPNET_DVR_DOOR_FILE_UPLOAD_PARAM;
```

### Members

**dwSize**

Structure size.

**dwFileSize**

File size.

**byFileName**

File name.

**byRes1**

Reserved.

## 4.1.37 NET_DVR_DOOR_STATUS_PLAN

Parameter structure about door control schedule configuration.

### Structure Definition

```
struct{
 DWORD    dwSize;
 DWORD    dwTemplateNo;
 BYTE     byRes[64];
}NET_DVR_DOOR_STATUS_PLAN,*LPNET_DVR_DOOR_STATUS_PLAN;
```

### Members

**dwSize**

Structure size.

**dwTemplateNo**

Schedule template No.: 0-cancel linking the configured template with schedule, and restore to the default settings; non-0-link the configured template with schedule.

**byRes**

Reserved, set to 0.

## 4.1.38 NET_DVR_ETHERNET_V30

### Ethernet Configuration Structure

| Member | Data Type | Description |
|---|---|---|
| struDVRIP | ***NET_DVR_IPADDR_UNION*** | Device IP address |
| struDVRIPMask | ***NET_DVR_IPADDR_UNION*** | Mask of device IP address |
| dwNetInterface | DWORD | Network interface type: 1-10MBase-T; 2-10MBase-T (full duplex); 3-100MBase-TX; 4-100M (full duplex); 5-10M/100M/1000M (self-adaptive); 6-1000M (full duplex) |
| wDVRPort | WORD | Device port No. |
| wMTU | WORD | MTU settings, the default is 1500. |
| byMACAddr | Array of BYTE | Device physical address. |
| byEthernetPortNo | BYTE | Network interface No.: 0-invalid, 1-interface 0, 2-interface 1, and so on. This parameter is read-only. |
| byRes | Array of BYTE | Reserved. |

## 4.1.39 NET_DVR_EVENT_CARD_LINKAGE_CFG_V51

Parameter structure about event and card linkage configuration.

### Structure Definition

```
struct{
  DWORD                    dwSize;
  BYTE                   byProMode;
  BYTE                   byRes1[3];
  DWORD                    dwEventSourceID;
  NET_DVR_EVENT_CARD_LINKAGE_UNION uLinkageInfo;
  BYTE                   byAlarmout[MAX_ALARMHOST_ALARMOUT_NUM/*512*/];
  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                    byMainDevStopBuzzer;
 WORD                      wAudioDisplayID;
 BYTE                    byAudioDisplayMode;
 BYTE                    byRes3[25];
 BYTE                    byReaderBuzzer[MAX_CARD_READER_NUM/*512*/];
 BYTE                    byAlarmOutClose[MAX_ALARMHOST_ALARMOUT_NUM/*512*/];
 BYTE                    byAlarmInSetup[MAX_ALARMHOST_ALARMOUT_NUM/*512*/];
 BYTE                    byAlarmInClose[MAX_ALARMHOST_ALARMOUT_NUM/*512*/];
 BYTE                    byReaderStopBuzzer[MAX_CARD_READER_NUM/*64*/];
 BYTE                    byRes[512];
}NET_DVR_EVENT_CARD_LINKAGE_CFG_V51, *LPNET_DVR_EVENT_CARD_LINKAGE_CFG_V51;
```

## Members

### dwSize

Structure size.

### byProMode

Linkage type: 0-event linkage, 1-card No. linkage, 2-MAC address linkage, 3- employee No. (person ID) linkage.

### byRes1

Reserved, set to 0.

### dwEventSourceID

Event triggering source ID: 0xffffffff-all. For device events, this parameter is invalid; for door events, it refers to door No.; for card reader events, it refers to card reader ID; for alarm input events, it refers to zone or event alarm input ID.

### uLinkageInfo

Linkage action parameter, see ***NET_DVR_EVETN_CARD_LINKAGE_UNION*** for details.

### byAlarmout

Linked alarm output No., which is represented by byte. 0-not link, 1-link.

### byRes2

Reserved, set to 0.

### byOpenDoor

Whether to enable door opening linkage, which is represented by byte. 0-disable, 1-enable.

### byCloseDoor

Whether to enable door closing linkage, which is represented by byte. 0-disable, 1-enable.

### byNormalOpen

Whether to enable door remaining open linkage, which is represented by byte. 0-disable, 1-enable.

**byNormalClose**

Whether to enable door remaining closed linkage, which is represented by byte. 0-disable, 1-enable.

**byMainDevBuzzer**

Whether to enable access controller buzzing, 0-disable, 1-enable.

**byCapturePic**

Whether to enable capture linkage, 0-disable, 1-enable.

**byRecordVideo**

Whether to enable recording linkage, 0-disable, 1-enable.

**byMainDevStopBuzzer**

Whether to enable access controller stopping buzzing linkage, 0-disable, 1-enable.

**wAudioDisplayID**

Linked audio prompt ID, currently it is between 1 and 32, and 0 indicates no linkage.

**byAudioDisplayMode**

Linked audio prompt mode: 0-disable, 1-play once, 2-loop playing.

**byRes3**

Reserved.

**byReaderBuzzer**

Whether to enable buzzer linkage, which is represented by byte. 0-disable, 1-enable.

**byAlarmOutClose**

Whether to enable alarm output disabling linkage, which is represented by byte. 0-disable, 1-enable.

**byAlarmInSetup**

Whether to enable zone arming linkage, which is represented by byte. 0-disable, 1-enable.

**byAlarmInClose**

Whether to enable zone disarming linkage, which is represented by byte. 0-disable, 1-enable.

**byReaderStopBuzzer**

Whether to enable card reader stopping buzzing linkage, which is represented by byte. 0-disable, 1-enable.

**byRes**

Reserved, set to 0.

## 4.1.40 NET_DVR_EVENT_CARD_LINKAGE_COND

Condition structure about the event card linkage configuration.

### Structure Definition

```
struct{
  DWORD   dwSize;
  DWORD   dwEventID;
  WORD    wLocalControllerID;
  BYTE    byRes[106];
}NET_DVR_EVENT_CARD_LINKAGE_COND,*LPNET_DVR_EVENT_CARD_LINKAGE_COND;
```

### Members

**dwSize**

Structure size.

**dwEventID**

Event ID.

**wLocalControllerID**

Distributed access controller No. which is between 1 and 64, while, 0-access controller.

**byRes**

Reserved, set to 0.

## 4.1.41 NET_DVR_EVETN_CARD_LINKAGE_UNION

Parameter union about event and card linkage configuration.

### Structure Definition

```
union{
  BYTE                      byCardNo[ACS_CARD_NO_LEN/*32*/];
  NET_DVR_EVENT_LINKAGE_INFO  struEventLinkage;
  BYTE                      byMACAddr[MACADDR_LEN/*6*/];
  BYTE                      byEmployeeNo[NET_SDK_EMPLOYEE_NO_LEN/*32*/];
}NET_DVR_EVETN_CARD_LINKAGE_UNION,*LPNET_DVR_EVETN_CARD_LINKAGE_UNION;
```

### Members

**byCardNo**

Card No.

**struEventLinkage**

Event linkage parameters, see details in the structure ***NET_DVR_EVENT_LINKAGE_INFO*** .

**byMACAddr**

Physical MAC address.

**byEmployeeNo**

Employee No. (person ID)

## See Also

***NET_DVR_EVENT_CARD_LINKAGE_CFG_V51***

## 4.1.42 NET_DVR_EVENT_LINKAGE_INFO

Event linkage parameter structure.

## Structure Definition

```
struct{
  WORD    wMainEventType;
  WORD    wSubEventType;
  BYTE    byRes[28];
}NET_DVR_EVENT_LINKAGE_INFO,*LPNET_DVR_EVENT_LINKAGE_INFO;
```

## Members

**wMainEventType**

Event major types, see ***Access Control Event Types*** for details.

**wSubEventType**

Event minor types, see ***Access Control Event Types*** for details.

**byRes**

Reserved, set to 0.

## See Also

***NET_DVR_EVETN_CARD_LINKAGE_UNION***

## 4.1.43 NET_DVR_FACE_FEATURE

Structure about facial feature parameters.

## Structure Definition

```
struct{
  NET_VCA_RECT   struFace;
  NET_VCA_POINT  struLeftEye;
  NET_VCA_POINT  struRightEye;
  NET_VCA_POINT  struLeftMouth;
  NET_VCA_POINT  struRightMouth;
  NET_VCA_POINT  struNoseTip;
}NET_DVR_FACE_FEATURE, *LPNET_DVR_FACE_FEATURE;
```

## Members

**struFace**

Face sub-picture area, see details in the structure ***NET_VCA_RECT*** .

**struLeftEye**

Coordinates of the left eye, see details in the structure ***NET_VCA_POINT*** .

**struRightEye**

Coordinates of the right eye, see details in the structure ***NET_VCA_POINT*** .

**struLeftMouth**

Coordinates of the left mouth corner, see details in the structure ***NET_VCA_POINT*** .

**struRightMouth**

Coordinates of the right mouth corner, see details in the structure ***NET_VCA_POINT*** .

**struNoseTip**

Coordinates of the nose, see details in the structure ***NET_VCA_POINT*** .

## 4.1.44 NET_DVR_FACE_PARAM_CFG

Face parameter structure

## Structure Definition

```
struct{
  DWORD    dwSize;
  BYTE     byCardNo[ACS_CARD_NO_LEN/*32*/];
  DWORD    dwFaceLen;
  char     *pFaceBuffer;
  BYTE     byEnableCardReader[MAX_CARD_READER_NUM/*512*/];
  BYTE     byFaceID;
  BYTE     byFaceDataType;
  BYTE     byRes[126];
}NET_DVR_FACE_PARAM_CFG, *LPNET_DVR_FACE_PARAM_CFG;
```

## Members

**dwSize**

Structure size.

**byCardNo**

Card number linked with the face.

**dwFaceLen**

Face data size.

**pFaceBuffer**

Pointer that points to the buffer for saving face data, it is valid when **dwFaceLen** is not 0. The data is encrypted.

**byEnableCardReader**

Face collector to be applied to, which is represented by array, each bit of array refers to one collector. Array value: 0-not apply, 1-apply.

**byFaceID**

Face picture ID, which is between 1 and 2.

**byFaceDataType**

Face data type: 0-template (default), 1-picture

**byRes**

Reserved, set to 0.

## 4.1.45 NET_DVR_FACE_PARAM_COND

Condition structure of face information configuration.

### Structure Definition

```
struct{
  DWORD    dwSize;
  BYTE     byCardNo[ACS_CARD_NO_LEN/*32*/];
  BYTE     byEnableCardReader[MAX_CARD_READER_NUM/*512*/];
  DWORD    dwFaceNum;
  BYTE     byFaceID;
  BYTE     byFaceDataType;
  BYTE     byRes[126];
}NET_DVR_FACE_PARAM_COND, *LPNET_DVR_FACE_PARAM_COND;
```

### Members

**dwSize**

Structure size.

**byCardNo**

Card number that linked with face information.

**byEnableCardReader**

Face collector status, which is represented by array, each bit of array refers to one collector. Array value: 0-invalid, 1-valid.

**dwFaceNum**

Number of face pictures to be get and apply. 0xffffffff-all face pictures.

**byFaceID**

Face picture ID, which is between 1 and 2, 0xff-all face pictures linked with the card.

**byFaceDataType**

Face data type: 0-template (default), 1-picture

**byRes**

Reserved, set to 0.

## 4.1.46 NET_DVR_FINGER_PRINT_CFG_V50

Fingerprint configuration structure.

### Structure Definition

```
struct{
  DWORD    dwSize;
  BYTE     byCardNo[ACS_CARD_NO_LEN/*32*/];
  DWORD    dwFingerPrintLen;
  BYTE     byEnableCardReader[MAX_CARD_READER_NUM/*512*/];
  BYTE     byFingerPrintID;
  BYTE     byFingerType;
  BYTE     byRes1[30];
  BYTE     byFingerData[MAX_FINGER_PRINT_LEN/*768*/];
  BYTE     byEmployeeNo[NET_SDK_EMPLOYEE_NO_LEN/*32*/];
  BYTE     byLeaderFP[MAX_DOOR_NUM_256/*256*/]
  BYTE     byRes[128];
}NET_DVR_FINGER_PRINT_CFG, *LPNET_DVR_FINGER_PRINT_CFG;
```

### Members

**dwSize**

Structure size.

**byCardNo**

Card No., which is linked with the fingerprint.

**dwFingerPrintLen**

Size of fingerprint data. The fingerprint module and fingerprint recorder will be used in pair.

**byEnableCardReader**

Whether to apply fingerprint data to the fingerprint module, which is represented by array: 0-no 1-yes

**byFingerPrintID**

Finger No., which is between 1 and 10

**byFingerType**

Fingerprint type: 0-normal fingerprint, 1-duress fingerprint, 2-patrol fingerprint, 3-super fingerprint, 4-dismiss fingerprint

**byRes1**

Reserved, set to 0

**byFingerData**

Fingerprint data

**byEmployeeNo**

Employee No. (person ID)

**byLeaderFP**

Whether to support first time authentication function (door, represented by byte): 0-no, 1-yes.

**byRes**

Reserved, set to 0

## 4.1.47 NET_DVR_FINGER_PRINT_INFO_COND_V50

Fingerprint parameter configuration structure.

### Structure Definition

```
struct{
  DWORD   dwSize;
  BYTE    byCardNo[ACS_CARD_NO_LEN/*32*/];
  BYTE    byEnableCardReader[MAX_CARD_READER_NUM/*512*/];
  DWORD   dwFingerPrintNum;
  BYTE    byFingerPrintID;
  BYTE    byCallbackMode;
  BYTE    byRes2[2];
  BYTE    byEmployeeNo[NET_SDK_EMPLOYEE_NO_LEN/*32*/];
  BYTE    byRes1[128];
}NET_DVR_FINGER_PRINT_INFO_COND_V50, *LPNET_DVR_FINGER_PRINT_INFO_COND_V50;
```

### Members

**dwSize**

Structure size.

**byCardNo**

Card No. linked with the fingerprint. This parameter is invalid when setting fingerprint parameters.

**byEnableCardReader**

Fingerprint module status: 0-invalid; 1-valid.

**dwFingerPrintNum**

Number of obtained or configured fingerprints, 0xffffffff-get all fingerprints' information.

**byFingerPrintID**

Finger No., which is between 1 and 10, 0xff indicates-all fingerprints of the card.

**byCallbackMode**

Device callback mode: 0-returned when applied all; 1-returned when applied a part.

**byRes2**

Reserved, set to 0.

**byEmployeeNo**

Employee No. (person ID).

**byRes1**

Reserved, set to 0.

## Remarks

Two fingerprint applying modes are available: blocking mode and non-blocking mode.

- Blocking Mode: Set **byCallbackMode** to "0", and the applying status will be returned for once only after applying each fingerprint.
- Non-blocking Mode: Set **byCallbackMode** to "1", and the applying status will be returned for multiple times after applying each fingerprint. And the next fingerprint can be applied until the previous fingerprint information is applied.

### 4.1.48 NET_DVR_GATE_TIME_CFG

Structure about the barrier time parameters of the turnstile.

## Structure Definition

```
struct{
  DWORD       dwSize;
  DWORD       dwHoldOnALarmTime;
  DWORD       dwHoldOnGateOpenTime;
  DWORD       dwPostponeIntrusionAlarmTime;
  DWORD       dwNoLaneAccessTimeLimitTime;
  DWORD       dwSafetyZoneStayTime;
  DWORD       byIRTriggerTimeoutTime;
  BYTE        byRes[299];
}NET_DVR_GATE_TIME_CFG, *LPNET_DVR_GATE_TIME_CFG;
```

## Members

**dwSize**

Structure Size.

**dwHoldOnALarmTime**

Extend alarm device buzzing time, unit: ms.

**dwHoldOnGateOpenTime**

Remaining open time before the barrier receives closing order, unit: ms

**dwPostponeIntrusionAlarmTime**

Delay time of triggering intrusion alarm, unit: ms.

**dwNoLaneAccessTimeLimitTime**

Timeout alarm time for no person passing when the lane receives valid passing signal, unit: s.

**dwSafetyZoneStayTime**

Timeout alarm time for persons staying in the lane after arriving at the safe area when the lane receives valid passing signal, unit: s.

**byIRTriggerTimeoutTime**

Maximum IR obstructed duration, it is between 0 and 255, unit: s.

**byRes**

Reserved, set to 0.

## 4.1.49 NET_DVR_GROUP_CFG

Group configuration structure.

### Structure Definition

```
struct{
  DWORD              dwSize;
  BYTE               byEnable;
  BYTE               byRes1[3];

                         NET_DVR_VALID_PERIOD_CFG
                         struValidPeriodCfg;
  BYTE               byGroupName[GROUP_NAME_LEN/*32*/];
  BYTE               byRes2[32];
}NET_DVR_GROUP_CFG,*LPNET_DVR_GROUP_CFG;
```

### Members

**dwSize**

Structure size

**byEnable**

Whether to enable the group: 0-no, 1-yes.

**byRes1**

Reserved, set to 0.

**struValidPeriodCfg**

Group expiry date.

**byGroupName**

Group name

**byRes2**

Reserved, set to 0.

## 4.1.50 NET_DVR_GROUP_COMBINATION_INFO_V50

Group parameters structure.

### Structure Definition

```
struct{
  BYTE     byEnable;
  BYTE     byMemberNum;
  BYTE     bySequenceNo;
  BYTE     byRes;
  DWORD    dwGroupNo;
}NET_DVR_MULTI_CARD_CFG_V50,*LPNET_DVR_MULTI_CARD_CFG_V50;
```

### Members

**byEnable**

Whether to enable the group: 0-no, 1-yes

**byMemberNum**

Number of cards should be swiped in the group.

**bySequenceNo**

Card swiping order in the group.

**byRes**

Reserved, set to 0.

**dwGroupNo**

Group No., 0xffffffff-remotely open door, 0xfffffffe-open door by super password.

### See Also

***NET_DVR_MULTI_CARD_GROUP_CFG_V50***

## 4.1.51 NET_DVR_HOLIDAY_GROUP_CFG

Holiday group configuration structure.

## Structure Definition

```
struct{
  DWORD    dwSize;
  BYTE     byEnable;
  BYTE     byRes1[3];
  BYTE     byGroupName[HOLIDAY_GROUP_NAME_LEN/*32*/];
  DWORD    dwHolidayPlanNo[MAX_HOLIDAY_PLAN_NUM/*16*/];
  BYTE     byRes2[32];
}NET_DVR_HOLIDAY_GROUP_CFG,*LPNET_DVR_HOLIDAY_GROUP_CFG;
```

## Members

**dwSize**

Structure size.

**byEnable**

Whether to enable: 1-enable, 0-disable.

**byRes1**

Reserved, set to 0.

**byGroupName**

Holiday group name.

**dwHolidayPlanNo**

Holiday schedule No.: 0-invalid.

**byRes2**

Reserved, set to 0.

## 4.1.52 NET_DVR_HOLIDAY_PLAN_CFG

Holiday schedule configuration structure.

## Structure Definition

```
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**

Structure size.

**byEnable**

Enable? 0- No; 1- Yes

**byRes1**

Reserved, set to 0.

**struBeginDate**

Holiday start time, see ***NET_DVR_DATE*** for details.

**struEndDate**

Holiday end time, see ***NET_DVR_DATE*** for details.

**struPlanCfg**

Holiday schedule parameters, up to 7 days can be set in one week, and up to 8 time periods can be set in one day, see ***NET_DVR_SINGLE_PLAN_SEGMENT*** for details.

**byRes2**

Reserved, set to 0.

### 4.1.53 NET_DVR_INDOOR_UNIT_DEVICEID

Parameter structure of indoor station No.

## Structure Definition

```
struct{
  SHORT    wFloorNumber;
  WORD     wRoomNumber;
  WORD     wDevIndex;
  BYTE     byRes[122];
}NET_DVR_INDOOR_UNIT_DEVICEID, *LPNET_DVR_INDOOR_UNIT_DEVICEID;
```

## Members

**wFloorNumber**

Floor No.

**wRoomNumber**

Room No.

**wDevIndex**

Indoor station No., which is between 0 and 10.

**byRes**

Reserved, set to 0.

## See Also

***NET_DVR_VIDEO_INTERCOM_UNIT_DEVICEID_UNION***

### 4.1.54 NET_DVR_INDOOR_UNIT_RELATEDEV

Parameter structure of linked network device of indoor station.

## Structure Definition

```
struct{
 NET_DVR_IPADDR   struOutdoorUnit;
 NET_DVR_IPADDR   struManageUnit;
 NET_DVR_IPADDR   struSIPServer;
 NET_DVR_IPADDR   struAgainUnit;
 BYTE        byOutDoorType;
 BYTE            byOutInConnectMode;
 BYTE            byIndoorConnectMode;
 BYTE            byRes1;
 NET_DVR_IPADDR   struIndoorUnit;
 BYTE            byRes[300];
}NET_DVR_INDOOR_UNIT_RELATEDEV,*LPNET_DVR_INDOOR_UNIT_RELATEDEV;
```

## Members

**struOutdoorUnit**

IP address of main door station, refer to the structure ***NET_DVR_IPADDR_UNION*** for details.

**struManageUnit**

IP address of main station, refer to the structure ***NET_DVR_IPADDR_UNION*** for details.

**struSIPServer**

IP address of SIP server, refer to the structure ***NET_DVR_IPADDR_UNION*** for details.

**struAgainUnit**

Doorphone IP address, refer to the structure ***NET_DVR_IPADDR_UNION*** for details.

**byOutDoorType**

Main door station type: 0-reserved, 1-main door station, 2-villa door station

**byOutInConnectMode**

Network connection mode of door station and indoor station: 1-in same LAN, 2-in different LAN.

**byIndoorConnectMode**

Network connection mode of indoor station and sub indoor station: 1-by wireless NIC, 2-by wired NIC

**byRes1**

Reserved, set to 0.

**struIndoorUnit**

IP address of indoor station, refer to the structure ***NET_DVR_IPADDR_UNION*** for details.

**byRes**

Reserved, set to 0.

## See Also

### 4.1.55 NET_DVR_INIT_CFG_ABILITY

## Initialization Capability Structure

| Member | Data Type | Description |
|---|---|---|
| enumMaxLoginUsersNum | INIT_CFG_MAX_NUM | Maximum number of users can log in, see details below:<br>enum _INIT_CFG_MAX_NUM_{<br>  INIT_CFG_NUM_2048  = 2048,<br>  INIT_CFG_NUM_5120  = 5120,<br>  INIT_CFG_NUM_10240 = 10240,<br>  INIT_CFG_NUM_15360 = 15360,<br>  INIT_CFG_NUM_20480 = 20480<br>}INIT_CFG_MAX_NUM |
| enumMaxAlarmNum | INIT_CFG_MAX_NUM | Maximum number of alarm channels, see details below:<br>enum _INIT_CFG_MAX_NUM_{<br>  INIT_CFG_NUM_2048  = 2048,<br>  INIT_CFG_NUM_5120  = 5120,<br>  INIT_CFG_NUM_10240 = 10240,<br>  INIT_CFG_NUM_15360 = 15360,<br>  INIT_CFG_NUM_20480 = 20480<br>}INIT_CFG_MAX_NUM |
| byRes | Array of BYTE | Reserved, set to 0. |

## Remarks

By default, up to 2048 channels are supported. More channels require higher computer performance and network bandwidth.

## See Also

***NET_DVR_SetSDKInitCfg***

## 4.1.56 NET_DVR_IPADDR_UNION

## IP Address Union

| Member | Data Type | Description |
|---|---|---|
| szIPv4 | char[] | IPv4 address. The maximum length is 16 bytes. |
| szIPv6 | char[] | IPv6 address. The maximum length is 256 bytes. |

## 4.1.57 NET_DVR_JSON_DATA_CFG

Structure about picture data in JSON format.

## Structure Definition

```
struct{
 DWORD   dwSize;
 void    *lpJsonData;
 DWORD   dwJsonDataSize;
 void    *lpPicData;
 DWORD   dwPicDataSize;
 DWORD   dwInfraredFacePicSize;
 char    *lpInfraredFacePicBuffer;
 BYTE    byRes[248];
}NET_DVR_JSON_DATA_CFG,*LPNET_DVR_JSON_DATA_CFG;
```

## Members

**dwSize**

Structure size.

**lpJsonData**

Returned message in JSON format.

**dwJsonDataSize**

Size of the message in JSON format.

**lpPicData**

Picture data. If the returned message is the response status message, this member is invalid; if the returned message in JSON format does not contain **faceURL**, this member should contain picture data in binary format.

**dwPicDataSize**

Picture data size, the maximum size is 200 KB.

**dwInfraredFacePicSize**

Data size of the infrared face picture. When this member is 0, it indicates that there is no face picture data. When the response message is ***JSON_ResponseStatus*** , this member is meaningless. When the request message in JSON format does not contain the value of **infraredFaceURL**, this member should contain the binary picture.

**lpInfraredFacePicBuffer**

Buffer of infrared face picture data.

**byRes**

Reserved.

### 4.1.58 NET_DVR_LOCAL_SDK_PATH

### Path Information Structure for Loading Component Libraries

| Member | Data Type | Description |
|---|---|---|
| sPath | Array of char | Component libraries' addresses |
| byRes | Array of BYTE | Reserved. |

### Remarks

If the path of HCNetSDKCom folder and HCNetSDK libraries are same, but the path of executable programs are different, you can call ***NET_DVR_SetSDKInitCfg*** to specify the path of HCNetSDKCom folder to make sure the component libraries are loaded normally.

### 4.1.59 NET_DVR_MANAGE_UNIT_DEVICEID

Parameter structure of main station No.

### Structure Definition

```
struct{
  DWORD   wPeriod;
  DWORD   wDevIndex;
  BYTE     byRes[124];
}NET_DVR_MANAGE_UNIT_DEVICEID, *LPNET_DVR_MANAGE_UNIT_DEVICEID;
```

## Members

**wPeriod**

Community No., range: [0,9].

**wDevIndex**

Outer door station No., which is unique in each floor, and it starts from 0.

**byRes**

Reserved, set to 0.

## See Also

***NET_DVR_VIDEO_INTERCOM_UNIT_DEVICEID_UNION***

### 4.1.60 NET_DVR_MANAGE_UNIT_RELATEDEV

Parameter structure of linked network device of main station.

## Structure Definition

```
struct{
  NET_DVR_IPADDR    struSIPServer;
  BYTE              byRes[880];
}NET_DVR_MANAGE_UNIT_RELATEDEV,*LPNET_DVR_MANAGE_UNIT_RELATEDEV;
```

## Members

**struSIPServer**

IP address of SIP server, refer to the structure ***NET_DVR_IPADDR_UNION*** for details.

**byRes**

Reserved, set to 0.

## See Also

### 4.1.61 NET_DVR_MIME_UNIT

### Input Content Details Structure of Message Transmission API (NET_DVR_STDXMLConfig)

| Member | Data Type | Description |
|---|---|---|
| szContentType | Array of char | Content type (corresponds to **Content-Type** field in the message), e.g., text/json. text/xml, and so on. The content format must be supported by HTTP. |
| szName | Array of char | Content name (corresponds to **name** field in the message), e.g., name="upload". |
| szFilename | Array of char | Content file name (corresponds to **filename** field in the message), e.g., filename="C:\Users\test\Desktop\11.txt". |
| dwContentLen | DWORD | Content size |
| pContent | char* | Data point |
| bySelfRead | BYTE | 0-External file, 1-Internal data, whose address is specified by **szFilename**. |
| byRes | Array of BYTE | Reserved. Set to 0. Maximum: 15 bytes. |

### See Also

***NET_DVR_XML_CONFIG_INPUT***

## 4.1.62 NET_DVR_MULTI_CARD_CFG_V50

Multi-factor authentication parameter structure.

### Structure Definition

```
struct{
  DWORD                   dwSize;
  BYTE                    byEnable;
  BYTE                    bySwipeIntervalTimeout;
  BYTE                    byRes1[2];
  NET_DVR_MULTI_CARD_GROUP_CFG_V50  struGroupCfg[NET_SDK_MULTI_CARD_GROUP_NUM/*20*/];
  BYTE                    byRes2[32];
}NET_DVR_MULTI_CARD_CFG_V50,*LPNET_DVR_MULTI_CARD_CFG_V50;
```

### Members

**dwSize**

Structure size

**byEnable**

Whether to enable multi-factor authentication: 0-no, 1-yes.

**bySwipeIntervalTimeout**

Card swiping interval timeout, which is ranging from 1 to 255, unit: second, default: 10s.

**byRes1**

Reserved, set to 0.

**struGroupCfg**

Card swiping parameters of group, see details in the structure ***NET_DVR_MULTI_CARD_GROUP_CFG_V50*** .

**byRes2**

Reserved, set to 0.

### 4.1.63 NET_DVR_MULTI_CARD_GROUP_CFG_V50

Card swiping parameter structure of card group.

## Structure Definition

```
struct{
  BYTE                       byEnable;
  BYTE                       byEnableOfflineVerifyMode;
  BYTE                       byRes1[2];
  DWORD                        dwTemplateNo;
  NET_DVR_GROUP_COMBINATION_INFO_V50  struGroupCombination[GROUP_COMBINATION_NUM];
}NET_DVR_MULTI_CARD_GROUP_CFG_V50,*LPNET_DVR_MULTI_CARD_GROUP_CFG_V50;
```

## Members

**byEnable**

Whether to enable card group parameters of multi-factor authentication: 0-no, 1-yes.

**bySwipeIntervalTimeout**

Whether to enable access authentication when the access controller is offline (open door by super password instead of remotely opening door): 0-no, 1-yes

**byRes1**

Reserved, set to 0.

**dwTemplateNo**

Template No. of multi-factor authentication schedule, which reuses the template of access permission control schedule.

**struGroupCombination**

Group parameters, see details in the structure ***NET_DVR_GROUP_COMBINATION_INFO_V50*** .

## See Also

***NET_DVR_MULTI_CARD_CFG_V50***

## 4.1.64 NET_DVR_NETCFG_V50

### Network Configuration Structure

| Member | Data Type | Description |
|---|---|---|
| dwSize | DWORD | Structure size. |
| struEtherNet | Array of ***NET_DVR_ETHERNET_V30*** | Ethernet interface |
| struRes1 | Array of | Reserved, set to 0. |
| struAlarmHostIpAddr | ***NET_DVR_IPADDR_UNION*** | Listening service IP address |
| byRes2 | Array of BYTE | Reserved, set as 0 |
| wAlarmHostIpPort | WORD | Listening service port No. |
| byUseDhcp | BYTE | Whether to enable DHCP: 0xff- invalid; 0-disable, 1-enable |
| byIPv6Mode | BYTE | Allocation mode of IPv6 address: 0-by router advertisement, 1-by manual setting, 2-by enabling DHCP allocation. |
| struDnsServer1IpAddr | ***NET_DVR_IPADDR_UNION*** | IP address of domain name server 1 |
| struDnsServer2IpAddr | ***NET_DVR_IPADDR_UNION*** | IP address of domain name server 2 |
| byIpResolver | Array of BYTE | IP resolver domain name or IP address (if the port No. of device is 8000, the domain name is not supported). |
| wIpResolverPort | WORD | IP resolver port No. |
| wHttpPortNo | WORD | HTTP port No. |
| struMulticastIpAddr | ***NET_DVR_IPADDR_UNION*** | Multicast group address |

| Member | Data Type | Description |
|---|---|---|
| struGatewayIpAddr | ***NET_DVR_IPADDR_UNION*** | Gateway address |
| struPPPoE | ***NET_DVR_PPPOECFG*** | PPPoE parameters |
| byEnablePrivateMulticastDiscovery | BYTE | Private multicast search (SADP): 0-default, 1-enable, 2-disable |
| byEnableOnvifMulticastDiscovery | BYTE | Onvif multicast search (SADP): 0-default, 1-enable, 2-disable |
| wAlarmHost2IpPort | WORD | Port No. of listening host 2. |
| struAlarmHost2IpAddr | ***NET_DVR_IPADDR_UNION*** | IP address of listening host 2 |
| byEnableDNS | BYTE | DNS address setting mode: 0-automatically get, 1-manually set. |
| byRes | Array of BYTE | Reserved, set to 0 |

### Remarks

- For device only supports the private protocol with version 3.0 or lower, when the parameter **byUseDhcp**="0xff", you should set the device IP address to null, and then the device will automatically get the DHCP information.
- When the parameter **byIPv6Mode** is set to 0 or 2, setting IPv6 address in the parameter **struEtherNet** is not required, it will be obtained automatically by the device; when **byIPv6Mode** is set to 1, you should set IPv6 address. As there are multiple IPv6 addresses, the IPv6 address of current logged-in device may be different with that in **struEtherNet**.

## 4.1.65 NET_DVR_OUTDOOR_FENCE_DEVICEID

Parameter structure of outer door station No.

### Structure Definition

```
struct{
  DWORD   wPeriod;
  DWORD   wDevIndex;
  BYTE    byRes[124];
}NET_DVR_OUTDOOR_FENCE_DEVICEID, *LPNET_DVR_OUTDOOR_FENCE_DEVICEID;
```

### Members

**wPeriod**

Community No., range: [0,9].

**wDevIndex**

Outer door station No., which starts from 0.

**byRes**

Reserved, set to 0.

## See Also

***NET_DVR_VIDEO_INTERCOM_UNIT_DEVICEID_UNION***

## 4.1.66 NET_DVR_OUTDOOR_UNIT_DEVICEID

Parameter structure of door station (or intelligent access control device) No.

## Structure Definition

```
struct{
  DWORD   wPeriod;
  DWORD   wBuildingNumber;
  DWORD   wUnitNumber
  DWORD   wFloorNumber
  DWORD   wDevIndex
  DWORD   byRes[118];
}NET_DVR_OUTDOOR_UNIT_DEVICEID, *LPNET_DVR_OUTDOOR_UNIT_DEVICEID;
```

## Members

**wPeriod**

Project No., range: [0,9].

**wBuildingNumber**

Building No.

**wUnitNumber**

Unit No.

**wFloorNumber**

Floor No.

**wDevIndex**

Door station No., which is unique in each floor, and it starts from 0.

**byRes**

Reserved, set to 0.

## See Also

***NET_DVR_VIDEO_INTERCOM_UNIT_DEVICEID_UNION***

### 4.1.67 NET_DVR_OUTDOOR_UNIT_RELATEDEV

Parameter structure of linked network device of door station, villa door station, or intelligent access control device.

#### Structure Definition

```
struct{
  NET_DVR_IPADDR   struMainOutdoorUnit;
  NET_DVR_IPADDR   struManageUnit;
  NET_DVR_IPADDR   struSIPServer;
  BYTE             byManageCenterID[32];
  BYTE             byRes[560];
}NET_DVR_OUTDOOR_UNIT_RELATEDEV,*LPNET_DVR_OUTDOOR_UNIT_RELATEDEV;
```

#### Members

**struMainOutdoorUnit**

IP address of door station, it is valid when sub door station exists, refer to the structure ***NET_DVR_IPADDR_UNION*** for details.

**struManageUnit**

IP address of master station, refer to the structure ***NET_DVR_IPADDR_UNION*** for details.

**struSIPServer**

IP address of SIP server, it is valid when the sub door station exits, refer to the structure ***NET_DVR_IPADDR_UNION*** for details.

**byManageCenterID**

Management center ID, which is valid in SIP mode, and it should contains digits, letters, @, and dots.

**byRes**

Reserved, set to 0.

#### See Also

### 4.1.68 NET_DVR_PERSON_STATISTICS_CFG

People counting parameters structure.

#### Structure Definition

```
struct{
  DWORD   dwSize;
  BYTE    byEnableStatistics;
```

```
 BYTE     byEnableOfflineStatistics;
 BYTE     byCountSignalStatisticalStandard;
 BYTE     byRes[605];
}NET_DVR_PERSON_STATISTICS_CFG, *LPNET_DVR_PERSON_STATISTICS_CFG;
```

## Members

**dwSize**

Structure size.

**byEnableStatistics**

Whether to enable people counting: 0-disable, 1-enable.

**byEnableOfflineStatistics**

Whether to enable offline people counting: 0-disable, 1-enable.

**byCountSignalStatisticalStandard**

People counting type: 0-invalid, 1-IR detection, 2-authentication number.

**byRes**

Reserved, set to 0.

## 4.1.69 NET_DVR_PLAN_TEMPLATE

Schedule template configuration structure.

## Structure Definition

```
struct{
 DWORD    dwSize;
 BYTE     byEnable;
 BYTE     byRes1[3];
 BYTE     byTemplateName[TEMPLATE_NAME_LEN/*32*/];
 DWORD    dwWeekPlanNo;
 DWORD    dwHolidayGroupNo[MAX_HOLIDAY_GROUP_NUM/*16*/];
 BYTE     byRes2[32];
}NET_DVR_PLAN_TEMPLATE,*LPNET_DVR_PLAN_TEMPLATE;
```

## Members

**dwSize**

Structure size.

**byEnable**

Whether to enable: 1-enable, 0-disable.

**byRes1**

Reserved, set to 0.

**byGroupName**

Schedule template name.

**byGroupName**

Week schedule No.: 0-invalid.

**dwHolidayGroupNo**

Holiday group No.: 0-invalid.

**byRes2**

Reserved, set to 0.

### 4.1.70 NET_DVR_PPPOECFG

### PPPoE Configuration Structure

| Member | Data Type | Description |
|---|---|---|
| dwPPPOE | DWORD | Whether to enable PPPoE: 0-no, 1-yes. |
| sPPPoEUser | Array of BYTE | PPPoE user name. |
| sPPPoEPassword | Array of char | PPPoE password. |
| struPPPoEIP | ***NET_DVR_IPADDR_UNION*** | PPPoE IP address |

### 4.1.71 NET_DVR_RECORD_PASSBACK_MANUAL_COND

### Structure About Conditions of Getting Task of Manually Copying Back Videos

| Member | Data Type | Description |
|---|---|---|
| **dwSize** | DWORD | Structure size. |
| **byType** | BYTE | Method of getting the task information: 0 (get remaining tasks), 1 (get remaining tasks by stream ID), 2 (get all tasks), 3 (get all tasks by stream ID). |
| **byRes1** | BYTE | Reserved, set to 0. The size is 3 bytes. |

| Member | Data Type | Description |
|---|---|---|
| **struStreamInfo** | ***NET_DVR_STREAM_INFO*** | Stream information structure. This member is valid when getting the task information by stream ID. |
| **byRes** | Array of BYTE | Reserved, set to 0. The size is 128 bytes. |

### 4.1.72 NET_DVR_RECORD_PASSBACK_MANUAL_TASK_RET

### Structure About Results of Getting Task of Manually Copying Back Videos

| Member | Data Type | Description |
|---|---|---|
| **dwSize** | DWORD | Structure size. |
| **struStreamInfo** | ***NET_DVR_STREAM_INFO*** | Stream information structure. This member is valid when getting the task information by stream ID. |
| **dwTaskID** | DWORD | Task ID |
| **struStartTime** | ***NET_DVR_TIME_EX*** | Start time of video copy-back |
| **struStopTime** | ***NET_DVR_TIME_EX*** | End time of video copy back |
| **byTaskStatus** | BYTE | Task status: 0 (not executed), 1 (pausing), 2 (executed), 3 (copying back), 4 (copy-back failed), 5 (succeeded, but only some videos are copied back), 6 (succeeded, but there is no video in the camera). |
| **byRes1** | Array of BYTE | Reserved, set to 0. The size is 3 bytes. |
| **struExecuteStartTime** | ***NET_DVR_TIME_EX*** | Actual start time of executing the task. This member is valid when the value of **byTaskStatus** is 1 or 2. |
| **struExecuteStopTime** | ***NET_DVR_TIME_EX*** | Actual end time of executing the task. This member is valid when the value of **byTaskStatus** is 1 or 2. |
| **byRes** | Array of BYTE | Reserved, set to 0. The size is 128 bytes. |

## 4.1.73 NET_DVR_RIGHT_CONTROLLER_AUDIO_PARAM

Structure about audio file parameters of main controller.

### Structure Definition

```
struct{
 DWORD       dwSize;
 DWORD       dwFileSize;
 DWORD       dwAudioID;
 BYTE       byRes[256];
}NET_DVR_RIGHT_CONTROLLER_AUDIO_PARAM,*LPNET_DVR_RIGHT_CONTROLLER_AUDIO_PARAM;
```

### Members

**dwSize**

Structure size.

**dwFileSize**

File size, unit: byte. This member is valid only when uploading audio file, and it is invalid when downloading audio file.

**dwAudioID**

Audio file ID. 0xffffffff indicates uploading all audio files. Currently, only uploading all audio files is supported, and uploading a single file by ID is not supported.

**byRes**

Reserved.

### Remarks

The audio file is uploaded from the client to the device.

## 4.1.74 NET_DVR_SETUPALARM_PARAM_V50

### Arming Parameter Structure

| Member | Data Type | Description |
|---|---|---|
| **dwSize** | DWORD | Structure size. |
| **byLevel** | BYTE | Arming priority: 0-high, 1-medium, 2-low. |
| **byAlarmInfoType** | BYTE | Intelligent traffic alarm information type: 0-old (NET_DVR_PLATE_RESULT),1-new (NET_ITS_PLATE_RESULT). |

| Member | Data Type | Description |
|---|---|---|
| **byRetAlarmTypeV40** | BYTE | 0-the motion detection, video loss, video tampering, and alarm input alarm information is uploaded in normal mode (alarm type: COMM_ALARM_V30, alarm information structure: ***NET_DVR_ALARMINFO_V30*** ); 1-alarm information is uploaded in variable size (alarm type: COMM_ALARM_V40, alarm information structure: ***NET_DVR_ALARMINFO_V40*** ). |
| **byRetDevInfoVersion** | BYTE | Alarm types of CVR: 0-COMM_ALARM_DEVICE (alarm information structure: ***NET_DVR_ALARMINFO_DEV*** ), 1-COMM_ALARM_DEVICE_V40 (alarm information structure: ***NET_DVR_ALARMINFO_DEV_V40*** ). |
| **byRetVQDAlarmType** | BYTE | VQD alarm types: 0-COMM_ALARM_VQD (alarm information structure: NET_DVR_VQD_DIAGNOSE_INFO), 1-COMM_ALARM_VQD_EX (alarm information structure: NET_DVR_VQD_ALARM, including camera information and captured pictures) |
| **byFaceAlarmDetection** | BYTE | Face detection alarm types: 1-face detection alarm (alarm type: COMM_ALARM_FACE_DETECTION, alarm information structure: NET_DVR_FACE_DETECTION), 0-face capture alarm (alarm type: COMM_UPLOAD_FACESNAP_RESULT, alarm information structure: NET_VCA_FACESNAP_RESULT). |
| **bySupport** | BYTE | Capabilities, which is represented by bit:<br>• bit0-whether to upload picture: 0-yes, 1-no<br>• bit1-whether to enable ANR: 0-no, 1-yes<br>• bit4-whether to upload abnormal event detection events of all detection targets: 0-no, 1-yes. It is used to enable the NVR to get events of all targets detected by network cameras.<br>• bit5-whether to enable all-day event or alarm uploading: 0-no, 1-yes. It is used to enable |

| Member | Data Type | Description |
|---|---|---|
| | | the NVR to receive all alarms from network cameras. |
| **byBrokenNetHttp** | BYTE | ANR type, which is represented by bit and should be supported by device:<br>• bit0-whether to enable ANR for ANPR: 0-no, 1-yes.<br>• bit1-whether to enable ANR for people counting: 0-no, 1-yes.<br>• bit2-whetehr to enable ANR for heat map: 0-no, 1-yes.<br>• bit3-whether to enable ANR for face capture: 0-no, 1-yes.<br>• bit4-whether to enable ANR for face picture comparison: 0-no, 1-yes.<br>• bit5-whether to enable ANR for JSON message transmission: 0-no, 1-yes.<br>• bit6: whether to enable ANR for uploading heat map data by dwell time duration and by people quantity: 0-no, 1-yes.<br>• bit7: whether to enable ANR for uploading intersection analysis result: 0-no, 1-yes. |
| **wTaskNo** | BYTE | Task No. |
| **byDeployType** | BYTE | Arming type: 0-arm via client software, 1-real-time arming. |
| **bySubScription** | BYTE | Subscription parameters, which is represent by bit.<br>Bit7-whether to upload picture after subscribing motion detection alarm by person or vehicle: 0-no, 1-yes. |
| **byRes1** | Array [BYTE] | Reserved, set to 0. The maximum size is 2 bytes. |
| **byAlarmTypeURL** | BYTE | Alarm picture data type, which is represented by bit, if the device supports uploading alarm pictures in binary format and URL format, you can specify the data type to be uploading via this parameter, if the device only supports URL format, this parameter is invalid. If the URL format is selected, you should set the device |

| Member | Data Type | Description |
|---|---|---|
| | | and enable the cloud storage, otherwise, the picture will still be transmitted in binary format.<br>• bit0-type of captured face pictures: 0-binary data, 1-URL<br>• bit1-type of picture uploaded in message: 0-binary, 1-URL<br>• bit2-type of picture uploaded for face picture comparison: 0-binary, 1-URL |
| **byCustomCtrl** | BYTE | Custom control type, which is represented by bit, bit0-whether to upload the face thumbnail of the front passenger: 0-no, 1-yes |
| **byRes4** | Array [BYTE] | Reserved, set to 0. The maximum size is 128 bytes. |

## Remarks

- The parameters **byLevel** and **byAlarmInfoType** are available for traffic cameras. Up to 1 cameras can be armed in the priority of level 0, up to 3 cameras can be armed in the priority of level 1, and up to 5 cameras can be armed in the priority of level 3, the alarm/event information from the camera in highest priority will be uploaded first.
- For arming via client software, only supports arming one channel, and supports uploading the alarm/event when device is offline; for real-time arming, up to four channels can be armed at same time, but uploading alarm/event when device is offline is not supported.
- The parameter **wTaskNo** is used to distinguish different arming connections. If the value of this parameter in different arming connections is same, error will be returned.

### 4.1.75 NET_DVR_SIMPLE_DAYTIME

Time parameter structure.

## Structure Definition

```
struct{
  BYTE    byHour;
  BYTE    byMinute;
  BYTE    bySecond;
  BYTE    byRes;
}NET_DVR_SIMPLE_DAYTIME,*LPNET_DVR_SIMPLE_DAYTIME;
```

## Members

**byHour**

Hour

**byMinute**

Minute

**bySecond**

Second

**byRes**

Reserved, set to 0.

### 4.1.76 NET_DVR_SINGLE_PLAN_SEGMENT

Parameter structure of a schedule

## Structure Definition

```
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**

Whether to enable: 1-no, 0-yes.

**byDoorStatus**

Door status: 0-invalid, 1-remain open (access without authentication), 2-remain closed (access is not allowed), 3-normal (access by authentication).

**byVerifyMode**

Authentication mode: 0-invalid, 1-card, 2-card+password, 3-password, 4-card or password, 5-fingerprint, 6-fingerprint+password, 7-fingerprint or password, 8-fingerprint+card, 9-fingerprint +card+password, 10-face or fingerprint or card or password, 11-face+fingerprint, 12-face +password, 13-face+card, 14-face, 15-employee ID+password, 16-fingerprint or password, 17-employee ID+fingerprint, 18-employee ID+fingerprint+password, 19-face+fingerprint+card, 20-face+password+fingerprint, 21-employee ID+face, 22-face or face+card, 23-fingerprint or face, 24-card or face or password, 25-card or face, 26-card or face or fingerprint, 27-card or fingerprint or password.

**byRes**

Reserved, set to 0.

**struTimeSegment**

Time period parameters, see ***NET_DVR_TIME_SEGMENT*** for details.

### 4.1.77 NET_DVR_STREAM_INFO

Stream information structure.

## Structure Definition

```
struct{
  DWORD    dwSize;
  BYTE     byID[STREAM_ID_LEN/*32*/];
  DWORD    dwChannel;
  BYTE     byRes[32];
}NET_DVR_STREAM_INFO,*LPNET_DVR_STREAM_INFO;
```

## Members

**dwSize**

Structure size.

**byID**

Stream ID, which consists of letters, digits, and dashes, 0-invalid.

**dwChannel**

Linked device channel. When it is 0xffffffff, if setting the stream source, this parameter indicates that no device channel is linked; if setting configuration condition, this parameter is invalid.

**byRes**

Reserved, set to 0.

## Remarks

- If the device does not support marking stream ID, e.g., DVR, the parameter **byID** should be set to 0.
- For transcoder, when setting the stream source, only one of **byID** and **dwChannel** can be valid; when transcoding, both the **byID** and **dwChannel** can be invalid, the transcoding channel or stream ID is automatically allocated by device.
- For other devices (e.g., CVR), when this structure is inputted as configuration condition, if both the **byID** and **dwChannel** are invalid, error code (17) will be returned, if they are valid, but mismatched, error may also be returned, so only setting one of these two parameters is suggested.

## 4.1.78 NET_DVR_TIME

### Time Parameter Structure

| Member | Data Type | Description |
|---|---|---|
| dwYear | DWORD | Year |
| dwMonth | DWORD | Month |
| dwDay | DWORD | Day |
| dwHour | DWORD | Hour |
| dwMinute | DWORD | Minute |
| dwSecond | DWORD | Second |

## 4.1.79 NET_DVR_TIME_EX

### Extended Time Parameter Structure

| Member | Data Type | Description |
|---|---|---|
| wYear | WORD | Year |
| byMonth | BYTE | Month |
| byDay | BYTE | Day |
| byHour | BYTE | Hour |
| byMinute | BYTE | Minute |
| bySecond | BYTE | Second |
| byRes | BYTE | Reserved. |

## 4.1.80 NET_DVR_TIME_SEGMENT

Time period parameter structure.

### Structure Definition

```
struct{
  NET_DVR_SIMPLE_DAYTIME  struBeginTime;
  NET_DVR_SIMPLE_DAYTIME  struEndTime;
}NET_DVR_TIME_SEGMENT, *LPNET_DVR_TIME_SEGMENT;
```

## Members

**struBeginTime**

Start time of time period, refer to the structure ***NET_DVR_SIMPLE_DAYTIME*** for details.

**struEndTime**

End time of time period, refer to the structure ***NET_DVR_SIMPLE_DAYTIME*** for details.

## 4.1.81 NET_DVR_USER_LOGIN_INFO

## Structure About Login Parameters

| Member | Data Type | Description |
|---|---|---|
| sDeviceAddress | char | Device IP address, or domain name. |
| byUseTransport | BYTE | Enable capability transmission or not: 0-no (default), 1-yes. |
| wPort | WORD | Device port number, e.g., 8000 (when login by private protocol), 80 (when login by text protocol). |
| sUserName | char | User name for logging in to device. |
| sPassword | char | Login password. |
| cbLoginResult | ***fLoginResultCallBack*** | Callback function used to return login status, it is valid only when **bUseAsynLogin** is "1". |
| pUser | void* | User data. |
| bUseAsynLogin | BOOL | Whether to enable asynchronous login: 0-no, 1-yes. |
| byProxyType | BYTE | Proxy server type: 0-no proxy, 1-standard proxy, 2-EHome proxy. |
| byUseUTCTime | BYTE | 0-not convert (default), 1-input or output UTC time, 2-input or output local time. |
| byLoginMode | BYTE | Login mode: 0-login by private protocol, 1-login by text protocol, 2-self-adaptive (it is available when the protocol type supported by device is unknown, and this mode does not support asynchronous login). |

| Member | Data Type | Description |
|---|---|---|
| byHttps | BYTE | Whether to enable TLS for login (by private protocol or by text protocol): 0-no, 1-yes, 2-self-adaptive (which is usually used when the protocol type supported by device is unknown. Both HTTP and HTTPS requests will be sent). |
| iProxyID | LONG | Proxy server No. |
| byVerifyMode | BYTE | Whether to enable verification mode: 0-no, 1-bidirectional verification (currently not available), 2-unidirectional verification (it is valid when **byLoginMode** is 0 and **byHttps** is 1); when **byVerifyMode** is 0, CA certificate is not required, when **byVerifyMode** is 2, you should call NET_DVR_SetSDKLocalCfg to load CA certificate, and the enumeration value is "NET_ SDK_LOCAL_CFG_CERTIFICATION". |
| byRes3 | BYTE[] | Reserved, the maximum length is 119 bytes. |

### 4.1.82 NET_DVR_VALID_PERIOD_CFG

Expiry date configuration structure.

## Structure Definition

```
struct{
  BYTE          byEnable;
  BYTE          byBeginTimeFlag;
  BYTE          byEnableTimeFlag;
  BYTE          byTimeDurationNo;
  NET_DVR_TIME_EX  struBeginTime;
  NET_DVR_TIME_EX  struEndTime;
  BYTE          byTimeType;
  BYTE          byRes2[32];
}NET_DVR_VALID_PERIOD_CFG,*LPNET_DVR_VALID_PERIOD_CFG;
```

## Members

**byEnable**

Whether to enable the expiry date: 0-no, 1-yes.

**byBeginTimeFlag**

Whether to enable the flag to limit the start time: 0-no, 1-yes.

**byEnableTimeFlag**

Whether to enable the flag to limit the end time: 0-no, 1-yes.

**byTimeDurationNo**

Expiry date index No., which starts from 0.

**struBeginTime**

Start time of the expiry date, see details in the structure ***NET_DVR_TIME_EX*** .

**struEndTime**

End time of the expiry date, see details in the structure ***NET_DVR_TIME_EX*** ..

**byTimeType**

Time type: 0-device's local time (default), 1-UTC time. This member is valid for **struBeginTime** and **struEndTime**.

**byRes2**

Reserved, set to 0.

## See Also

***NET_DVR_GROUP_CFG***

### 4.1.83 NET_DVR_VIDEO_INTERCOM_DEVICEID_CFG

Parameter structure for configuring video intercom device No.

## Structure Definition

```
struct{
  DWORD                              bySize;
  BYTE                             byUnitType;;
  BYTE                             byIsAutoReg;
  BYTE                             byRes1[2];
  NET_DVR_VIDEO_INTERCOM_UNIT_DEVICEID_UNION    uVideoIntercomUnit;
  BYTE                             byRes2[128];
}NET_DVR_VIDEO_INTERCOM_DEVICEID_CFG, *LPNET_DVR_VIDEO_INTERCOM_DEVICEID_CFG;
```

## Members

**dwSize**

Structure size.

**byUnitType**

Device type: 1-door station, 2-master station, 4-outer station, 5-villa door station, 6-doorphone, 7-intelligent access control device.

**byIsAutoReg**

Whether to enable auto registration: 0-no, 1-yes.

**byRes1**

Reserved, set to 0.

**uVideoIntercomUnit**

Device No., refer to the data union ***NET_DVR_VIDEO_INTERCOM_UNIT_DEVICEID_UNION*** for details.

**byRes2**

Reserved, set to 0.

## Remarks

Configuring No. for doorphone is not required.

## 4.1.84 NET_DVR_VIDEO_INTERCOM_RELATEDEV_CFG

Configuration structure for linked network devices of video intercom.

## Structure Definition

```
struct{
  DWORD                                   dwSize;
  BYTE                                    byUnitType;
  BYTE                                    byRes1[3];
  NET_DVR_VIDEO_INTERCOM_UNIT_RELATEDEV_UNION    uVideoIntercomUnit;
  BYTE                                    byRes2[128];
}NET_DVR_VIDEO_INTERCOM_RELATEDEV_CFG, *LPNET_DVR_VIDEO_INTERCOM_RELATEDEV_CFG;
```

## Members

**dwSize**

Structure size.

**byUnitType**

Device type: 1-door station, 2-master station, 4-outer station, 5-villa door station, 6-doorphone, 7-intelligent access control device.

**byRes1**

Reserved, set to 0.

**uVideoIntercomUnit**

Parameter union of linked network device, refer to the union ***NET_DVR_VIDEO_INTERCOM_UNIT_RELATEDEV_UNION*** for details.

**byRes2**

Reserved, set to 0.

## 4.1.85 NET_DVR_VIDEO_INTERCOM_UNIT_DEVICEID_UNION

Parameter union of video intercom device No.

### Structure Definition

```
union{
  BYTE                          byLen[128];
  NET_DVR_INDOOR_UNIT_DEVICEID      struIndoorUnit;
  NET_DVR_OUTDOOR_UNIT_DEVICEID     struOutdoorUnit;
  NET_DVR_MANAGE_UNIT_DEVICEID      struManageUnit;
  NET_DVR_OUTDOOR_FENCE_DEVICEID    struFenceUnit;
  NET_DVR_OUTDOOR_UNIT_DEVICEID     struVillaOutdoorUnit;
  NET_DVR_OUTDOOR_UNIT_DEVICEID     struAgainConfirmUnit;
}NET_DVR_VIDEO_INTERCOM_UNIT_DEVICEID_UNION, *LPNET_DVR_VIDEO_INTERCOM_UNIT_DEVICEID_UNION;
```

### Members

**byLen**

Union size, which is 128 bytes.

**struIndoorUnit**

Indoor station No., refer to the structure ***NET_DVR_INDOOR_UNIT_DEVICEID*** for details.

**struOutdoorUnit**

Door station (or intelligent access control device) No., refer to the structure ***NET_DVR_OUTDOOR_UNIT_DEVICEID*** for details.

**struManageUnit**

Main station No., refer to the structure ***NET_DVR_MANAGE_UNIT_DEVICEID*** for details.

**struFenceUnit**

Outer door station No., refer to the structure ***NET_DVR_OUTDOOR_FENCE_DEVICEID*** for details.

**struVillaOutdoorUnit**

Villa door station No., refer to the structure ***NET_DVR_OUTDOOR_UNIT_DEVICEID*** for details.

**struAgainConfirmUnit**

Doorphone No., refer to the structure ***NET_DVR_OUTDOOR_UNIT_DEVICEID*** for details.

### Remarks

- The rules to generate device No. are as the follows:

  - The long No. of main station is "*00000001XX": "*"-project No., "001"-main station, "XX"-main station No. (01, 02, ..., increased by 1).
  - The long No. of door station, villa door station, or doorphone is "*XXXXX000$$": "*"-project No.; "XXX"-building No.; "XX"-unit No.; "000"-door station; "$$"-door station No. ("00"-main door station, other values-sub door station).
  - The long No. of indoor station is "*XXXXX$$$$$": "*"-project No.; "XXX"-building No.; "XX"-unit No.; "$$$"-floor No.; "$$"-room No.
  - The long No. of outer door station is "*00000002XX": "*"-project No., "002"-outer door station, "XX"-outer door station No. (01, 02, ..., increased by 1).
- In the actual application, one main station may belong to multiple communities, so the project No. of the device can be any of the existing project No., as long as the generated device No. is unique.

## 4.1.86 NET_DVR_VIDEO_INTERCOM_UNIT_RELATEDEV_UNION

Configuration union of the network device linked with video intercom.

### Structure Definition

```
union{
  DWORD                     dwRes[256];
  NET_DVR_INDOOR_UNIT_RELATEDEV   struIndoorUnit;
  NET_DVR_OUTDOOR_UNIT_RELATEDEV  struOutdoorUnit;
  NET_DVR_MANAGE_UNIT_RELATEDEV   struManageUnit;
  NET_DVR_OUTDOOR_UNIT_RELATEDEV  struVillaUnit;
  NET_DVR_AGAIN_RELATEDEV         struAgainUnit;
}NET_DVR_VIDEO_INTERCOM_UNIT_RELATEDEV_UNION,*LPNET_DVR_VIDEO_INTERCOM_UNIT_RELATEDEV_UNION;
```

### Members

**dwRes**

Union size, which is 1024 bytes (256 × 4).

**struIndoorUnit**

Linked network device parameters of indoor station, refer to the structure ***NET_DVR_INDOOR_UNIT_RELATEDEV*** for details.

**struOutdoorUnit**

Linked network device parameters of door station, outer door station, or intelligent access control device, refer to the structure ***NET_DVR_OUTDOOR_UNIT_RELATEDEV*** for details.

**struManageUnit**

Linked network device parameters of master station, refer to the structure ***NET_DVR_MANAGE_UNIT_RELATEDEV*** for details.

**struVillaUnit**

Linked network device parameters of villa door station, refer to the structure ***NET_DVR_OUTDOOR_UNIT_RELATEDEV*** for details.

**struAgainUnit**

Linked network device parameters of doorphone, refer to the structure ***NET_DVR_AGAIN_RELATEDEV*** for details.

## See Also

***NET_DVR_VIDEO_INTERCOM_RELATEDEV_CFG***

### 4.1.87 NET_DVR_WEEK_PLAN_CFG

Week schedule parameter structure.

## Structure Definition

```
struct{
  DWORD                      dwSize;
  BYTE                       byEnable;
  BYTE                       byRes1[3];
  NET_DVR_SINGLE_PLAN_SEGMENT    struPlanCfg[MAX_DAYS/*7*/][MAX_TIMESEGMENT_V30/*8*/];
  BYTE                       byRes2[16];
}NET_DVR_WEEK_PLAN_CFG, *LPNET_DVR_WEEK_PLAN_CFG;
```

## Members

**dwSize**

Structure size.

**byEnable**

Whether to enable: 1-no, 0-yes.

**byRes1**

Reserved, set to 0.

**struPlanCfg**

Week schedule parameters, up to 7 days can be set in one week, and up to 8 time periods can be set in one day, see ***NET_DVR_SINGLE_PLAN_SEGMENT*** for details.

**byRes2**

Reserved, set to 0.

### 4.1.88 NET_DVR_XML_CONFIG_INPUT

### Input Parameter Structure of Message Transmission API (NET_DVR_STDXMLConfig)

| Member | Data Type | Description |
|---|---|---|
| **dwSize** | DWORD | Structure size. |
| **lpRequestUrl** | void* | Request URL (command) for implement different functions, and it is in string format. |
| **dwRequestUrlLen** | DWORD | Request URL size. |
| **lpInBuffer** | void* | Buffer for storing input parameters (request messages), see the input content details structure in ***NET_DVR_MIME_UNIT*** . |
| **dwInBufferSize** | DWORD | Input buffer size. |
| **dwRecvTimeOut** | DWORD | Receiving timeout, unit: ms, 0-5000ms (default). |
| **byForceEncrpt** | BYTE | Whether to enable force encryption (the messages will be encrypted by AES algorithm for transmission): 0-no, 1-yes. |
| **byNumOfMultiPart** | BYTE | Number of message segments: 0-invalid; other values-number of message segments, which is transmitted by the parameter **lpInBuffer** in the structure ***NET_DVR_MIME_UNIT*** . |
| **byRes** | Array of BYTE | Reserved, set to 0. |

### Related API

***NET_DVR_STDXMLConfig***

## 4.1.89 NET_DVR_XML_CONFIG_OUTPUT

### Output Parameter Structure of Message Transmission API (NET_DVR_STDXMLConfig)

| Member | Data Type | Description |
|---|---|---|
| dwSize | DWORD | Structure size. |
| lpOutBuffer | void* | Buffer for storing output parameters (response messages), which is allocated when passing through URL by GET method. |

| Member | Data Type | Description |
|---|---|---|
| dwOutBufferSize | DWORD | Output buffer size. |
| dwReturnedXMLSize | DWORD | Actual size of response message. |
| lpStatusBuffer | void* | Response status (ResponseStatus message). This parameter will not be assigned if performing GET operation succeeded, and you can also set it to "NULL" if not required. |
| dwStatusSize | DWORD | Size of response status buffer. |
| lpDataBuffer | HPR_VOIDPTR | Buffer for transmitted data. This parameter is valid when the value of **byNumOfMultiPart** is larger than 0. |
| byNumOfMultiPart | HPR_UINT8 | Number of parts that the message is divided into. |
| byRes<br>[23] | BYTE | Reserved, set to 0. |

### Related API

***NET_DVR_STDXMLConfig***

## 4.1.90 NET_SDK_CALLBACK_STATUS_NORMAL

### Enumeration About Persistent Connection Status

| Enumeration Type | Marco Definition Value | Description |
|---|---|---|
| NET_SDK_CALLBACK_STATUS_ SUCCESS | 1000 | Succeeded. |
| NET_SDK_CALLBACK_STATUS_ PROCESSING | 1001 | Connecting. The **lpBuffer** is 4-byte status. |
| NET_SDK_CALLBACK_STATUS_ FAILED | 1002 | Failed. The **lpBuffer** is the value of 4-byte status and 4-byte error code. |

## 4.1.91 NET_VCA_POINT

### Structure About Point Coordinates Parameters

| Member | Data Type | Description |
|---|---|---|
| fX | float | X-coordinate, it is a normalized value ranging from 0.000 to 1. The floating-point number is the percentage of the current image size and is accurate to three decimal places. |
| fY | float | Y-coordinate, it is a normalized value ranging from 0.000 to 1. The floating-point number is the percentage of the current image size and is accurate to three decimal places. |

## 4.1.92 NET_VCA_RECT

### Structure About Rectangle Region Coordinate Parameters

| Member | Data Type | Description |
|---|---|---|
| fX | float | X-coordinate of frame's upper-left corner, it ranges from 0.000 to 1. |
| fY | float | Y-coordinate of frame' upper-left corner, it ranges from 0.000 to 1. |
| fWidth | float | Frame width, it ranges from 0.000 to 1. |
| fHeight | float | Frame height, it ranges from 0.000 to 1. |

# 4.2 Enumeration

## 4.2.1 NET_SDK_DOWNLOAD_TYPE

Enumerate file types to be downloaded.

### Enumeration Definition

```
typedef enum {
  NET_SDK_DOWNLOAD_CERT                    = 0,
  NET_SDK_DOWNLOAD_IPC_CFG_FILE            = 1,
  NET_SDK_DOWNLOAD_BASELINE_SCENE_PIC      = 2,
  NET_SDK_DOWNLOAD_VQD_ALARM_PIC           = 3,
  NET_SDK_DOWNLOAD_CONFIGURATION_FILE              = 4,
  NET_SDK_DOWNLOAD_SCENE_CONFIGURATION_FILE        = 5,
  NET_SDK_DOWNLOAD_FILE_FORM_DB                    = 6,
  NET_SDK_DOWNLOAD_TME_FILE                        = 7,
  NET_SDK_DOWNLOAD_VEHICLE_BLOCKALLOWLIST_FILE     = 8,
  NET_SDK_DOWNLOAD_GUID_FILE                       = 9,
  NET_SDK_DOWNLOAD_FILE_FORM_CLOUD                 = 10,
  NET_SDK_DOWNLOAD_PICTURE                         = 11,
  NET_SDK_DOWNLOAD_VIDEO                           = 12,
  NET_DVR_DOWNLOAD_SCREEN_FILE                     = 13,
  NET_SDK_DOWNLOAD_PUBLISH_MATERIAL                = 14,
  NET_SDK_DOWNLOAD_THERMOMETRIC_FILE               = 15,
  NET_SDK_DOWNLOAD_LED_CHECK_FILE                  = 16,
  NET_SDK_DOWNLOAD_VEHICLE_INFORMATION             = 17,
  NET_SDK_DOWNLOAD_CERTIFICATE_BLOCKLIST_TEMPLET   = 18,
  NET_SDK_DOWNLOAD_LOG_FILE                        = 19,
  NET_SDK_DOWNLOAD_FILEVOLUME_DATA                 = 20,
  NET_SDK_DOWNLOAD_FD_DATA                         = 21,
  NET_SDK_DOWNLOAD_SECURITY_CFG_FILE               = 22,
  NET_SDK_DOWNLOAD_PUBLISH_SCHEDULE                = 23,
  NET_SDK_DOWNLOAD_RIGHT_CONTROLLER_AUDIO          = 24,
  NET_SDK_DOWNLOAD_MODBUS_CFG_FILE                 = 25,
  NET_SDK_DOWNLOAD_RS485_PROTOCOL_DLL_FILE         = 26,
  NET_SDK_DOWNLOAD_CLUSTER_MAINTENANCE_LOG         = 27,
  NET_SDK_DOWNLOAD_SQL_ARCHIVE_FILE                = 28,
  NET_SDK_DOWNLOAD_SUBWIND_STREAM                  = 29,
  NET_SDK_DOWNLOAD_DEVTYPE_CALIBFILE               = 30,
  NET_SDK_DOWNLOAD_HD_CAMERA_CORRECT_TABLE         = 31,
  NET_SDK_DOWNLOAD_CLIENT_CALIBFILE                = 32,
  NET_SDK_DOWNLOAD_FOUE_CAMERAS_PICTURES           = 33,
  NET_SDK_DOWNLOAD_DOOR_CONTENT                    = 34,
  NET_SDK_DOWNLOAD_PUBLISH_MATERIAL_THUMBNAIL      = 35,
  NET_SDK_DOWNLOAD_PUBLISH_PROGRAM_THUMBNAIL       = 36,
  NET_SDK_DOWNLOAD_PUBLISH_TEMPLATE_THUMBNAIL      = 37,
  NET_SDK_DOWNLOAD_DARK_FIGHTER_X_CORRECT_TABLE_MAIN   = 38,
  NET_SDK_DOWNLOAD_DARK_FIGHTER_X_CORRECT_TABLE_BACKUP  = 39,
  NET_SDK_DOWNLOAD_OFFLINE_CAPTURE_INFO_TEMPLATE   = 40,
  NET_SDK_DOWNLOAD_CAPTURE_DATA                    = 41,
  NET_SDK_DOWNLOAD_HD_CAMERA_CORRECT_TABLE_FILE    = 42,
  NET_SDK_DOWNLOAD_CLIENT_CALIBFILE_FILE           = 43,
  NET_SDK_DOWNLOAD_FOUR_CAMERAS_PICTURES_FILE      = 44,
  NET_SDK_DOWNLOAD_SCENE_FILE                      = 45,
  NET_SDK_DOWNLOAD_OPEN_SOURCE_CERT                = 46,
  NET_SDK_DOWNLOAD_RATIOSTITCHING_FILE             = 47,
  NET_SDK_DOWNLOAD_LENS_PARAM_FILE                 = 48,
  NET_SDK_DOWNLOAD_SELECT_DEVTYPE_CALIBFILE        = 49
} NET_SDK_DOWNLOAD_TYPE;
```

## Enumeration Type

### NET_SDK_DOWNLOAD_CERT

Certificate.

**NET_SDK_DOWNLOAD_IPC_CFG_FILE**

Network camera configuration file.

**NET_SDK_DOWNLOAD_BASELINE_SCENE_PIC**

Base scene picture.

**NET_SDK_DOWNLOAD_VQD_ALARM_PIC**

VQD (video quality diagnosis) alarm picture.

**NET_SDK_DOWNLOAD_CONFIGURATION_FILE**

Configuration file.

**NET_SDK_DOWNLOAD_SCENE_CONFIGURATION_FILE**

Scene configuration file.

**NET_SDK_DOWNLOAD_FILE_FORM_DB**

File in the image and video library.

**NET_SDK_DOWNLOAD_TME_FILE**

Entrance and exit management file.

**NET_SDK_DOWNLOAD_VEHICLE_BLOCKALLOWLIST_FILE**

Blocklist and allowlist configuration file.

**NET_SDK_DOWNLOAD_GUID_FILE**

GUID file.

**NET_SDK_DOWNLOAD_FILE_FORM_CLOUD**

Picture in the cloud storage.

**NET_SDK_DOWNLOAD_PICTURE**

Picture.

**NET_SDK_DOWNLOAD_VIDEO**

Video.

**NET_DVR_DOWNLOAD_SCREEN_FILE**

Screen server file.

**NET_SDK_DOWNLOAD_PUBLISH_MATERIAL**

Local material file of information release.

**NET_SDK_DOWNLOAD_THERMOMETRIC_FILE**

Thermometry calibration file.

**NET_SDK_DOWNLOAD_LED_CHECK_FILE**

LED correction file.

**NET_SDK_DOWNLOAD_VEHICLE_INFORMATION**

Vehicle information to be exported.

**NET_SDK_DOWNLOAD_CERTIFICATE_BLOCKLIST_TEMPLET**

ID card blocklist template.

**NET_SDK_DOWNLOAD_LOG_FILE**

Log to be exported.

**NET_SDK_DOWNLOAD_FILEVOLUME_DATA**

File volume data file, currently it is only supported by CVR (central video recorder) devices.

**NET_SDK_DOWNLOAD_FD_DATA**

Data in a specific face picture library to be exported.

**NET_SDK_DOWNLOAD_SECURITY_CFG_FILE**

Configuration file to be securely exported.

**NET_SDK_DOWNLOAD_PUBLISH_SCHEDULE**

Schedule to be exported.

**NET_SDK_DOWNLOAD_RIGHT_CONTROLLER_AUDIO**

Audio file of the main controller.

**NET_SDK_DOWNLOAD_MODBUS_CFG_FILE**

Configuration file of Modbus protocol.

**NET_SDK_DOWNLOAD_RS485_PROTOCOL_DLL_FILE**

Dynamic library file of RS-485 protocol.

**NET_SDK_DOWNLOAD_CLUSTER_MAINTENANCE_LOG**

Cluster maintenance log to be exported.

**NET_SDK_DOWNLOAD_SQL_ARCHIVE_FILE**

Archived record in the database to be exported.

**NET_SDK_DOWNLOAD_SUBWIND_STREAM**

Sub-window stream to be exported.

**NET_SDK_DOWNLOAD_DEVTYPE_CALIBFILE**

Model calibration file to be exported (*.cal).

**NET_SDK_DOWNLOAD_HD_CAMERA_CORRECT_TABLE**

24 MP/32 MP correction list to be exported (*.cal).

**NET_SDK_DOWNLOAD_CLIENT_CALIBFILE**

Client calibration file to be exported (*.pto).

**NET_SDK_DOWNLOAD_FOUE_CAMERAS_PICTURES**

Four-channel picture package to be exported (.tar).

**NET_SDK_DOWNLOAD_DOOR_CONTENT**

Door contact information.

**NET_SDK_DOWNLOAD_PUBLISH_MATERIAL_THUMBNAIL**

Thumbnail of local information release material.

**NET_SDK_DOWNLOAD_PUBLISH_PROGRAM_THUMBNAIL**

Thumbnail of information release program.

**NET_SDK_DOWNLOAD_PUBLISH_TEMPLATE_THUMBNAIL**

Thumbnail of information release template.

**NET_SDK_DOWNLOAD_DARK_FIGHTER_X_CORRECT_TABLE_MAIN**

DarkfighterX correction list file (main partition).

**NET_SDK_DOWNLOAD_DARK_FIGHTER_X_CORRECT_TABLE_BACKUP**

DarkfighterX correction list file (backup partition).

**NET_SDK_DOWNLOAD_OFFLINE_CAPTURE_INFO_TEMPLATE**

User list template of collection.

**NET_SDK_DOWNLOAD_CAPTURE_DATA**

Offline collected data.

**NET_SDK_DOWNLOAD_HD_CAMERA_CORRECT_TABLE_FILE**

HD camera correction sheet (CAL format).

**NET_SDK_DOWNLOAD_CLIENT_CALIBFILE_FILE**

User calibration file (PTO format).

**NET_SDK_DOWNLOAD_FOUR_CAMERAS_PICTURES_FILE**

Channel pictures package (TAR format).

**NET_SDK_DOWNLOAD_SCENE_FILE**

Scene file.

**NET_SDK_DOWNLOAD_OPEN_SOURCE_CERT**

Open source license compliance.

**NET_SDK_DOWNLOAD_RATIOSTITCHING_FILE**

Ratio stitching file.

**NET_SDK_DOWNLOAD_LENS_PARAM_FILE**

Lens parameters file.

**NET_SDK_DOWNLOAD_SELECT_DEVTYPE_CALIBFILE**

Calibration file in CAL format.

### 4.2.2 NET_SDK_UPLOAD_TYPE

## Enumeration about File Types to Be Uploaded

| Enumeration Type | Macro Definition Value | Description |
|---|---|---|
| **UPGRADE_CERT_FILE** | 0 | Certificate file to be upgraded. |
| **UPLOAD_CERT_FILE** | 1 | Certificate file to be uploaded. |
| **TRIAL_CERT_FILE** | 2 | Trial license file. |
| **CONFIGURATION_FILE** | 3 | Configuration file. |
| **UPLOAD_RECORD_FILE** | 4 | Video file. |
| **SCENE_CONFIGURATION_FILE** | 5 | Scene configuration file. |
| **UPLOAD_PICTURE_FILE** | 6 | Picture file. |
| **UPLOAD_VIOLATION_FILE** | 7 | Violation dictionary file. |
| **UPLOAD_TG_FIL** | 8 | Timing generator file. |
| **UPLOAD_DATA_TO_DB** | 9 | File to be uploaded to picture and video library. |
| **UPLOAD_BACKGROUND_PIC** | 10 | Background picture. |
| **UPLOAD_CALIBRATION_FILE** | 11 | Calibration file. |
| **UPLOAD_TME_FILE** | 12 | Entrance and exiting management file. |
| **UPLOAD_VEHICLE_BLOCKALLOWLST_FILE** | 13 | Vehicle blocklist file. |
| **UPLOAD_PICTURE_TO_CLOUD** | 15 | Picture file to be uploaded to cloud storage. |
| **UPLOAD_VIDEO_FILE** | 16 | Video file. |
| **UPLOAD_SCREEN_FILE** | 17 | Screen server file. |
| **UPLOAD_PUBLISH_MATERIAL** | 18 | Local material file of information release system. |
| **UPLOAD_PUBLISH_UPGRADE_FILE** | 19 | Upgrade file of information release system. |
| **UPLOAD_RING_FILE** | 20 | Ringtone file. |
| **UPLOAD_ENCRYPT_CERT** | 21 | Encryption certificate. |

| Enumeration Type | Macro Definition Value | Description |
|---|---|---|
| **UPLOAD_THERMOMETRIC_FILE** | 22 | Calibration file for temperature measurement. |
| **UPLOAD_SUBBRAND_FILE** | 23 | Vehicle sub brand file. |
| **UPLOAD_LED_CHECK_FILE** | 24 | LED correction file. |
| **BATCH_UPLOAD_PICTURE_FILE** | 25 | Picture files for uploading in batch. |
| **UPLOAD_EDID_CFG_FILE** | 26 | EDID configuration file. |
| **UPLOAD_PANORAMIC_STITCH** | 27 | Panorama stitching configuration file. |
| **UPLOAD_BINOCULAR_COUNTING** | 28 | Binocular counting correction sheet. |
| **UPLOAD_AUDIO_FILE** | 29 | Audio file. |
| **UPLOAD_PUBLISH_THIRD_PARTY_FILE** | 30 | Third-party file. |
| **UPLOAD_DEEPEYES_BINOCULAR** | 31 | TX1 binocular correction sheet. |
| **UPLOAD_CERTIFICATE_BLOCKLIST** | 32 | ID card blocklist. |
| **UPLOAD_HD_CAMERA_CORRECT_TABLE** | 33 | HD camera correction sheet (CAL format). |
| **UPLOAD_FD_DATA** | 35 | Face data file to be imported to face picture library. |
| **UPLOAD_FACE_DATA** | 36 | Face picture file to be imported to face picture library. |
| **UPLOAD_FACE_ANALYSIS_DATA** | 37 | Picture file to be imported to picture recognition target. |
| **UPLOAD_FILEVOLUME_DATA** | 38 | File volume file |
| **IMPORT_DATA_TO_FACELIB** | 39 | Face data (face picture and picture additional information) to be imported to face picture library of device. |
| **UPLOAD_LEFTEYE_4K_CALIBFILE** | 40 | Camera calibration parameter file. |
| **UPLOAD_SECURITY_CFG_FILE** | 41 | Configuration file to be securely imported. |
| **UPLOAD_RIGHT_CONTROLLER_AUDIO** | 42 | Audio file of main controller. |

| Enumeration Type | Macro Definition Value | Description |
|---|---|---|
| **UPLOAD_MODBUS_CFG_FILE** | 43 | Configuration file of Modbus protocol. |
| **UPLOAD_NOTICE_VIDEO_DATA** | 44 | Bulletin video file. |
| **UPLOAD_RS485_PROTOCOL_DLL_FILE** | 45 | Dynamic library file of RS485 protocol. |
| **UPLOAD_PIC_BY_BUF** | 46 | Picture file for importing by picture cache. |
| **UPLOAD_CLIENT_CALIBFILE** | 47 | User calibration file (PTO format). |
| **UPLOAD_HD_CAMERA_CORRECT_TABLE_3200W** | 48 | HD camera correction sheet (CAL format). |
| **UPLOAD_DOOR_CONTENT** | 49 | Contact information of the door at the building unit. |
| **UPLOAD_ASR_CONTROL_FILE** | 50 | Speech recognition control file. |
| **UPLOAD_APP_FILE** | 51 | Application program file. |
| **UPLOAD_AI_ALGORITHM_MODEL** | 52 | Algorithm model in binary format. |
| **UPLOAD_AI_BASE_PICTURE** | 55 | Reference pictures in binary format for AI target comparison. |
| **UPLOAD_OFFLINE_CAPTURE_INFO** | 56 | User list of offline collection to be imported. |
| **IMPORT_DATA_TO_HBDLIB** | 60 | Import human body picture with linked information to library. |
| **UPLOAD_SCENE_FILE** | 61 | Scene file to be imported. |
| **UPLOAD_RATIOSTITCHING_FILE** | 62 | Ratio stitching file to be imported. |
| **UPLOAD_LENS_PARAM_FILE** | 63 | Lens parameters file to be imported. |

# Appendix A. Request URIs

| Description | URI | Method | Request and Response Message |
|---|---|---|---|
| Get device information. | /ISAPI/System/deviceInfo | GET | XML_DeviceInfo<br>XML_ResponseStatus |
| Edit device information. | /ISAPI/System/deviceInfo | PUT | - |
| Control PTZ. | /ISAPI/PTZCtrl/channels/<ID>/continuous | PUT | XML_ResponseStatus |
| Get preset list. | /ISAPI/PTZCtrl/channels/<ID>/presets | GET | XML_PTZPresetList XML_ResponseStatus |
| Manage all configured presets. | /ISAPI/PTZCtrl/channels/<ID>/presets | POST | - |
| Delete all presets. | /ISAPI/PTZCtrl/channels/<ID>/presets | DELETE | - |
| Add a preset. | /ISAPI/PTZCtrl/channels/<ID>/presets/<ID> | PUT | XML_ResponseStatus |
| Delete a preset. | /ISAPI/PTZCtrl/channels/<ID>/presets/<ID> | DELETE | XML_ResponseStatus |
| Get a preset. | /ISAPI/PTZCtrl/channels/<ID>/presets/<ID> | GET | - |
| Call a preset. | /ISAPI/PTZCtrl/channels/<ID>/presets/<ID>/goto | PUT | XML_ResponseStatus |
| Get partition status. | /ISAPI/SecurityCP/status/subSystems?format=json | GET | JSON_SubSysList<br>JSON_ResponseStatus |
| Arm a partition. | /ISAPI/SecurityCP/control/arm/<ID>?ways=<string>&format=json | PUT | JSON_ResponseStatus |
| Disarm a partition. | /ISAPI/SecurityCP/control/disarm/<ID>?format=json | PUT | JSON_ResponseStatus |
| Clear partition alarms. | /ISAPI/SecurityCP/control/clearAlarm/<ID>?format=json | PUT | JSON_ResponseStatus |
| Get zone status | /ISAPI/SecurityCP/status/zones?format=json | GET | JSON_ZoneList<br>JSON_ResponseStatus |

| | | | |
|---|---|---|---|
| Search partition status according to conditions. | /ISAPI/SecurityCP/status/zones?format=json | POST | - |
| Zone bypass. | /ISAPI/SecurityCP/control/bypass?format=json | PUT | JSON_ResponseStatus |
| Recover bypass of multiple zones. | /ISAPI/SecurityCP/control/bypassRecover?format=json | PUT | JSON_ResponseStatus |
| Get relay status by specific conditions. | /ISAPI/SecurityCP/status/outputStatus?format=json | POST | JSON_OutputSearch<br>JSON_ResponseStatus |
| Control relay in batch. | /ISAPI/SecurityCP/control/outputs?format=json | POST | JSON_ResponseStatus |
| Get the information of all I/O output ports. | /ISAPI/System/IO/outputs | GET | XML_IOOutputPortList<br>XML_ResponseStatus |
| Get status of a specific alarm output. | /ISAPI/System/IO/outputs/<ID>/status | GET | XML_IOPortStatus<br>XML_ResponseStatus |
| Manually trigger a specific alarm output. | /ISAPI/System/IO/outputs/<ID>/trigger | PUT | XML_ResponseStatus |
| Get device time zone. | /ISAPI/System/time | GET | XML_TimeData<br>XML_ResponseStatus |
| Get or set device time parameters. | /ISAPI/System/time | PUT | - |
| Operations about management of all digital channels. | /ISAPI/ContentMgmt/InputProxy/channels | GET | XML_InputProxyChannelList<br>XML_ResponseStatus |
| Configure operations about management of all digital channels. | /ISAPI/ContentMgmt/InputProxy/channels | PUT | - |
| Create digital channels | /ISAPI/ContentMgmt/InputProxy/channels | POST | - |

| | | | |
|---|---|---|---|
| Get status of all digital channels. | /ISAPI/ContentMgmt/InputProxy/channels/status | GET | XML_InputProxyChannelStatusList<br>XML_ResponseStatus |
| Refresh the video mode manually before playback. | /ISAPI/ContentMgmt/record/control/manualRefresh/channels/<ID> | PUT | XML_ResponseStatus |
| Search for access control events. | /ISAPI/AccessControl/AcsEvent?format=json | POST | JSON_AcsEvent<br>XML_ResponseStatus |
| Search for person information. | /ISAPI/AccessControl/UserInfo/Search?format=json | POST | JSON_UserInfoSearch<br>XML_ResponseStatus |

## A.1 /ISAPI/AccessControl/AcsEventTotalNum/capabilities?format=json

Get the capability of getting total number of access control events by specific conditions.

### Request URI Definition

**Table A-1 GET /ISAPI/AccessControl/AcsEventTotalNum/capabilities?format=json**

| **Method** | GET |
|---|---|
| **Description** | Get the capability of getting total number of access control events by specific conditions. |
| **Query** | **format**: determine the format of request or response message.<br>**terminalNo**: dependent, int, terminal No., starts from 1. It is required for information release system integration. The No. is generated after information release terminal is registered to the central management server, and you can call "/ISAPI/Publish/TerminalMgr/terminalSearch" by POST method to get the generated terminal No. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_Cap_AcsEventTotalNum***<br>Failed: ***JSON_ResponseStatus*** |

## A.2 /ISAPI/AccessControl/AcsEventTotalNum?format=json

Get the total number of access control events by specific conditions.

### Request URI Definition

**Table A-2 POST /ISAPI/AccessControl/AcsEventTotalNum?format=json**

| | |
|---|---|
| **Method** | POST |
| **Description** | Get the total number of access control events by specific conditions. |
| **Query** | **format**: determine the format of request or response message.<br>**terminalNo**: dependent, int, terminal No., starts from 1. It is required for information release system integration. The No. is generated after information release terminal is registered to the central management server, and you can call "/ISAPI/Publish/TerminalMgr/terminalSearch" by POST method to get the generated terminal No. |
| **Request** | ***JSON_AcsEventTotalNumCond*** |
| **Response** | Succeeded: ***JSON_AcsEventTotalNum***<br>Failed: ***JSON_ResponseStatus*** |

### Remarks

- The recommended timeout is 30s.
- This URI is not supported by integration of information release system.

## A.3 /ISAPI/AccessControl/attendanceStatusModeCfg/capabilities?format=json

Get the configuration capability of the attendance mode.

### Request URI Definition

**Table A-3 GET /ISAPI/AccessControl/attendanceStatusModeCfg/capabilities?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the configuration capability of the attendance mode. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_Cap_AttendanceStatusModeCfg***<br>Failed: ***JSON_ResponseStatus*** |

## A.4 /ISAPI/AccessControl/attendanceStatusModeCfg?format=json

Operations about the attendance mode configuration.

### Request URI Definition

**Table A-4 GET /ISAPI/AccessControl/attendanceStatusModeCfg?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the attendance mode. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_AttendanceStatusModeCfg***<br>Failed: ***JSON_ResponseStatus*** |

**Table A-5 PUT /ISAPI/AccessControl/attendanceStatusModeCfg?format=json**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Set the attendance mode. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | ***JSON_AttendanceStatusModeCfg*** |
| **Response** | ***JSON_ResponseStatus*** |

## A.5 /ISAPI/AccessControl/attendanceStatusRuleCfg/capabilities?format=json

Get the configuration capability of the attendance status and rule.

### Request URI Definition

**Table A-6 GET /ISAPI/AccessControl/attendanceStatusRuleCfg/capabilities?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the configuration capability of the attendance status and rule. |
| **Query** | **format**: determine the format of request or response message. |

| | |
|---|---|
| **Request** | None. |
| **Response** | Succeeded: ***JSON_Cap_AttendanceStatusRuleCfg***<br>Failed: ***JSON_ResponseStatus*** |

## A.6 /ISAPI/AccessControl/attendanceStatusRuleCfg?attendanceStatus=&format=json

Operations about the configuration of the attendance status and rule.

### Request URI Definition

**Table A-7 GET /ISAPI/AccessControl/attendanceStatusRuleCfg?attendanceStatus=&format=json**

| **Method** | GET |
|---|---|
| **Description** | Get the attendance status and rule. |
| **Query** | **format**: determine the format of request or response message.<br>**attendanceStatus**: attendance status, it can be set to one of the following values: "checkIn"-check in, "checkOut"-check out, "breakOut"-break out, "breakIn"-break in, "overtimeIn"-overtime in, "overtimeOut"-overtime out, e.g., ***/ISAPI/AccessControl/attendanceStatusRuleCfg?attendanceStatus=checkIn&format=json***. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_AttendanceStatusRuleCfg***<br>Failed: ***JSON_ResponseStatus*** |

**Table A-8 PUT /ISAPI/AccessControl/attendanceStatusRuleCfg?attendanceStatus=&format=json**

| **Method** | PUT |
|---|---|
| **Description** | Set the attendance status and rule. |
| **Query** | **format**: determine the format of request or response message.<br>**attendanceStatus**: attendance status, it can be set to one of the following values: "checkIn"-check in, "checkOut"-check out, "breakOut"-break out, "breakIn"-break in, "overtimeIn"-overtime in, "overtimeOut"-overtime out, e.g., ***/ISAPI/AccessControl/attendanceStatusRuleCfg?attendanceStatus=checkIn&format=json***. |

| | |
|---|---|
| **Request** | ***JSON_AttendanceStatusRuleCfg*** |
| **Response** | ***JSON_ResponseStatus*** |

## A.7 /ISAPI/AccessControl/capabilities

Get the functional capability of access control.

### Request URI Definition

**Table A-9 GET /ISAPI/AccessControl/capabilities**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the functional capability of access control. |
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_Cap_AccessControl***<br>Failed: ***XML_ResponseStatus*** |

## A.8 /ISAPI/AccessControl/CaptureCardInfo/capabilities?format=json

Get the capability of collecting card information.

### Request URI Definition

**Table A-10 GET /ISAPI/AccessControl/CaptureCardInfo/capabilities?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the capability of collecting card information. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_CardInfoCap***<br>Failed: ***JSON_ResponseStatus*** |

## A.9 /ISAPI/AccessControl/CaptureCardInfo?format=json

Collect card information.

### Request URI Definition

**Table A-11 GET /ISAPI/AccessControl/CaptureCardInfo?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Collect card information by the card reading module of the device. |
| **Query** | **format**: determine the format of request or response message.<br>**security**: the version No. of encryption scheme. When **security** does not exist, it indicates that the data is not encrypted; when **security** is 1, it indicates that the nodes of sensitive information in the message are encrypted in AES128 CBC mode; when **security** is 2, it indicates that the nodes of sensitive information in the message are encrypted in AES256 CBC mode. In the message of this request URI, the value of the field **cardNo** should be encrypted.<br>**iv**: the initialization vector, and it is required when **security** is 1 or 2. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_CardInfo_Collection***<br>Failed: ***JSON_ResponseStatus*** |

## A.10 /ISAPI/AccessControl/CaptureIDInfo/capabilities?format=json

Get the capability of collecting ID card information.

### Request URI Definition

**Table A-12 GET /ISAPI/AccessControl/CaptureIDInfo/capabilities?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the capability of collecting ID card information. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_IdentityInfoCap***<br>Failed: ***JSON_ResponseStatus*** |

## A.11 /ISAPI/AccessControl/CaptureIDInfo?format=json

Collect ID card information.

### Request URI Definition

**Table A-13 POST /ISAPI/AccessControl/CaptureIDInfo?format=json**

| | |
|---|---|
| **Method** | POST |
| **Description** | Collect ID card information. |
| **Query** | **security**: the version No. of encryption scheme. When **security** does not exist, it indicates that the data is not encrypted; when **security** is 1, it indicates that the nodes of sensitive information in the message are encrypted in AES128 CBC mode; when **security** is 2, it indicates that the nodes of sensitive information in the message are encrypted in AES256 CBC mode. In the message of this request URI, the value of the field **IDCardNo** should be encrypted.<br>**iv**: the initialization vector, and it is required when **security** is 1 or 2.<br>**format**: determine the format of request or response message. |
| **Request** | ***JSON_IdentityInfoCond*** |
| **Response** | ***JSON_IdentityInfo*** |

## A.12 /ISAPI/AccessControl/CapturePresetParam/capabilities?format=json

Get the configuration capability of online collection preset parameters.

### Request URI Definition

**Table A-14 GET /ISAPI/AccessControl/CapturePresetParam/capabilities?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the configuration capability of online collection preset parameters. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_CapturePresetCap*** |

| | |
|---|---|
| | Failed: ***JSON_ResponseStatus*** |

## A.13 /ISAPI/AccessControl/CapturePresetParam?format=json

Get or set the online collection preset parameters.

### Request URI Definition

**Table A-15 GET /ISAPI/AccessControl/CapturePresetParam?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the online collection preset parameters. |
| **Query** | **format**: determine the format of request or response message.<br>**security**: the version No. of encryption scheme. When **security** does not exist, it indicates that the data is not encrypted; when **security** is 1, it indicates that the nodes of sensitive information in the message are encrypted in AES128 CBC mode; when **security** is 2, it indicates that the nodes of sensitive information in the message are encrypted in AES256 CBC mode. In the message of this request URI, the value of the field **name** should be encrypted.<br>**iv**: the initialization vector, and it is required when **security** is 1 or 2. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_CapturePreset***<br>Failed: ***JSON_ResponseStatus*** |

**Table A-16 PUT /ISAPI/AccessControl/CapturePresetParam?format=json**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Set the online collection preset parameters. |
| **Query** | **format**: determine the format of request or response message.<br>**security**: the version No. of encryption scheme. When **security** does not exist, it indicates that the data is not encrypted; when **security** is 1, it indicates that the nodes of sensitive information in the message are encrypted in AES128 CBC mode; when **security** is 2, it indicates that the nodes of sensitive information in the message are encrypted in AES256 CBC mode. In the message of this request URI, the value of the field **name** should be encrypted.<br>**iv**: the initialization vector, and it is required when **security** is 1 or 2. |

| | |
|---|---|
| **Request** | ***JSON_CapturePreset*** |
| **Response** | ***JSON_ResponseStatus*** |

## A.14 /ISAPI/AccessControl/CaptureRule/capabilities?format=json

Get the configuration capability of online collection rules.

### Request URI Definition

**Table A-17 GET /ISAPI/AccessControl/CaptureRule/capabilities?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the configuration capability of online collection rules. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_CaptureRuleCap***<br>Failed: ***JSON_ResponseStatus*** |

## A.15 /ISAPI/AccessControl/CaptureRule?format=json

Get or set the parameters of online collection rules.

### Request URI Definition

**Table A-18 GET /ISAPI/AccessControl/CaptureRule?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the parameters of online collection rules. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_CaptureRule***<br>Failed: ***JSON_ResponseStatus*** |

**Table A-19 PUT /ISAPI/AccessControl/CaptureRule?format=json**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Set the parameters of online collection rules. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | ***JSON_CaptureRule*** |
| **Response** | ***JSON_ResponseStatus*** |

## A.16 /ISAPI/AccessControl/CardOperations/capabilities?format=json

Get card operation capability.

### Request URI Definition

**Table A-20 GET /ISAPI/AccessControl/CardOperations/capabilities?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get card operation capability. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_CardOperationsCap***<br>Failed: ***JSON_ResponseStatus*** |

## A.17 /ISAPI/AccessControl/CardOperations/cardParam?format=json

Set card parameters (only available for CPU card).

### Request URI Definition

**Table A-21 PUT /ISAPI/AccessControl/CardOperations/cardParam?format=json**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Set card parameters (only available for CPU card). |
| **Query** | **format**: determine the format of request or response message. |

| | |
|---|---|
| **Request** | ***JSON_CardParam*** |
| **Response** | ***JSON_ResponseStatus*** |

## A.18 /ISAPI/AccessControl/CardOperations/clearData?format=json

Delete data from the card.

### Request URI Definition

**Table A-22 PUT /ISAPI/AccessControl/CardOperations/clearData?format=json**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Delete data from the card. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | ***JSON_ClearData*** |
| **Response** | Succeeded: ***JSON_ClearDataRes***<br>Failed: ***JSON_ResponseStatus*** |

## A.19 /ISAPI/AccessControl/CardOperations/controlBlock?format=json

Change the control block of a specific section (only available for M1 card).

### Request URI Definition

**Table A-23 PUT /ISAPI/AccessControl/CardOperations/controlBlock?format=json**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Change the control block of a specific section (only available for M1 card). |
| **Query** | **format**: determine the format of request or response message.<br>**security**: the version No. of encryption scheme. When **security** does not exist, it indicates that the data is not encrypted; when **security** is 1, it indicates that the nodes of sensitive information in the message are encrypted in AES128 CBC mode; when **security** is 2, it indicates that the nodes of sensitive information in the message are encrypted in AES256 CBC mode.In the message of this request URI, the values of the fields **KeyA** and **KeyB** should be encrypted. |

| | |
|---|---|
| | **iv**: the initialization vector, and it is required when **security** is 1 or 2. |
| **Request** | ***JSON_ControlBlock*** |
| **Response** | ***JSON_ResponseStatus*** |

## A.20 /ISAPI/AccessControl/CardOperations/customData/searchTask?format=json

Search for custom card information.

### Request URI Definition

**Table A-24 POST /ISAPI/AccessControl/CardOperations/customData/searchTask?format=json**

| | |
|---|---|
| **Method** | POST |
| **Description** | Search for custom card information. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | ***JSON_CustomDataSearchCond*** |
| **Response** | Succeeded: ***JSON_CustomDataResult***<br>Failed: ***JSON_ResponseStatus*** |

## A.21 /ISAPI/AccessControl/CardOperations/customData?format=json

Set custom card information.

### Request URI Definition

**Table A-25 PUT /ISAPI/AccessControl/CardOperations/customData?format=json**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Set custom card information. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | ***JSON_CustomData*** |
| **Response** | Succeeded: ***JSON_CustomDataRes***<br>Failed: ***JSON_ResponseStatus*** |

## A.22 /ISAPI/AccessControl/CardOperations/dataBlock/control?format=json

Do operations (i.e., plus, minus, copy, and paste) on the data block.

### Request URI Definition

**Table A-26 PUT /ISAPI/AccessControl/CardOperations/dataBlock/control?format=json**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Do operations (i.e., plus, minus, copy, and paste) on the data block. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | ***JSON_DataBlockCtrl*** |
| **Response** | ***JSON_ResponseStatus*** |

## A.23 /ISAPI/AccessControl/CardOperations/dataBlock/<address>?format=json

Read or write data block (only available for M1 card).

### Request URI Definition

**Table A-27 GET /ISAPI/AccessControl/CardOperations/dataBlock/<address>?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Read data block (only available for M1 card). |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_DataBlock***<br>Failed: ***JSON_ResponseStatus*** |

**Table A-28 PUT /ISAPI/AccessControl/CardOperations/dataBlock/<address>?format=json**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Write data block (only available for M1 card). |
| **Query** | **format**: determine the format of request or response message. |

| | |
|---|---|
| **Request** | ***JSON_DataBlock*** |
| **Response** | ***JSON_ResponseStatus*** |

### Remarks

The <**address**> in the request URI refers to the block address, which is same as that in ***JSON_DataBlock*** .

## A.24 /ISAPI/AccessControl/CardOperations/dataTrans?format=json

Pass through the data package (only available for CPU card).

### Request URI Definition

**Table A-29 PUT /ISAPI/AccessControl/CardOperations/dataTrans?format=json**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Pass through the data package (only available for CPU card). |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | ***JSON_DataTrans*** |
| **Response** | ***JSON_ResponseStatus*** |

## A.25 /ISAPI/AccessControl/CardOperations/encryption?format=json

Set card encryption parameters (only available for CPU card).

### Request URI Definition

**Table A-30 PUT /ISAPI/AccessControl/CardOperations/encryption?format=json**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Set card encryption parameters (only available for CPU card). |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | ***JSON_CardEncryption*** |
| **Response** | ***JSON_ResponseStatus*** and **tryTimes** field (card encryption attampts) |

## A.26 /ISAPI/AccessControl/CardOperations/protocol?format=json

Set operation protocol type for the card (only available when applying card).

### Request URI Definition

**Table A-31 PUT /ISAPI/AccessControl/CardOperations/protocol?format=json**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Set operation protocol type for the card (only available when applying card). |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | ***JSON_CardProto*** |
| **Response** | ***JSON_ResponseStatus*** |

## A.27 /ISAPI/AccessControl/CardOperations/reset?format=json

Reset card parameters (only available for CPU card).

### Request URI Definition

**Table A-32 GET /ISAPI/AccessControl/CardOperations/reset?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Reset card parameters (only available for CPU card). |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_CardResetResponse***<br>Failed: ***JSON_ResponseStatus*** |

## A.28 /ISAPI/AccessControl/CardOperations/sectionEncryption?format=json

Set the encryption parameters of a specific section (only available for M1 card).

### Request URI Definition

**Table A-33 PUT /ISAPI/AccessControl/CardOperations/sectionEncryption?format=json**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Set the encryption parameters of a specific section (only available for M1 card). |
| **Query** | **format**: determine the format of request or response message.<br>**security**: the version No. of encryption scheme. When **security** does not exist, it indicates that the data is not encrypted; when **security** is 1, it indicates that the nodes of sensitive information in the message are encrypted in AES128 CBC mode; when **security** is 2, it indicates that the nodes of sensitive information in the message are encrypted in AES256 CBC mode.In the message of this request URI, the values of the fields **password**, **KeyA**, and **KeyB** should be encrypted.<br>**iv**: the initialization vector, and it is required when **security** is 1 or 2. |
| **Request** | ***JSON_SectionEncryption*** |
| **Response** | ***JSON_ResponseStatus*** |

## A.29 /ISAPI/AccessControl/CardOperations/verification?format=json

Verify the password of the encrypted section (only available for M1 card).

### Request URI Definition

**Table A-34 PUT /ISAPI/AccessControl/CardOperations/verification?format=json**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Verify the password of the encrypted section (only available for M1 card). |
| **Query** | **format**: determine the format of request or response message.<br>**security**: the version No. of encryption scheme. When **security** does not exist, it indicates that the data is not encrypted; when **security** is 1, it indicates that the nodes of sensitive information in the message are encrypted in AES128 CBC mode; when **security** is 2, it indicates that the nodes of sensitive information in the message are encrypted in AES256 CBC mode.In the message of this request URI, the value of the field **password** should be encrypted. |

| | |
|---|---|
| | **iv**: the initialization vector, and it is required when **security** is 1 or 2. |
| **Request** | ***JSON_Verification*** |
| **Response** | ***JSON_ResponseStatus*** |

## A.30 /ISAPI/AccessControl/ChannelControllerAlarmLinkage

Get or set the alarm linkage parameters of the lane controller.

### Request URI Definition

**Table A-35 GET /ISAPI/AccessControl/ChannelControllerAlarmLinkage**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the alarm linkage parameters of the lane controller. |
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_ChannelControllerAlarmLinkage***<br>Failed: ***XML_ResponseStatus*** |

**Table A-36 PUT /ISAPI/AccessControl/ChannelControllerAlarmLinkage**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Set the alarm linkage parameters of the lane controller. |
| **Query** | None. |
| **Request** | ***XML_ChannelControllerAlarmLinkage*** |
| **Response** | ***XML_ResponseStatus*** |

## A.31 /ISAPI/AccessControl/ChannelControllerAlarmLinkage/capabilities

Get the configuration capability of the alarm linkage of the lane controller.

### Request URI Definition

**Table A-37 GET /ISAPI/AccessControl/ChannelControllerAlarmLinkage/capabilities**

| **Method** | GET |
|---|---|
| **Description** | Get the configuration capability of the alarm linkage of the lane controller. |
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_Cap_ChannelControllerAlarmLinkage***<br>Failed: ***XML_ResponseStatus*** |

## A.32 /ISAPI/AccessControl/ChannelControllerAlarmOut/capabilities

Get the configuration capability of the alarm output of the lane controller.

### Request URI Definition

**Table A-38 GET /ISAPI/AccessControl/ChannelControllerAlarmOut/capabilities**

| **Method** | GET |
|---|---|
| **Description** | Get the configuration capability of the alarm output of the lane controller. |
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_Cap_ChannelControllerAlarmOut***<br>Failed: ***XML_ResponseStatus*** |

## A.33 /ISAPI/AccessControl/ChannelControllerAlarmOut?controllerType=&alarmOutNo=

Get or set the alarm output parameters of the lane controller.

### Request URI Definition

**Table A-39 GET /ISAPI/AccessControl/ChannelControllerAlarmOut?controllerType=&alarmOutNo=**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the alarm output parameters of the lane controller. |
| **Query** | **controllerType**: lane controller type, string, it can be set to "Master" (master lane controller) or "Slave" (slave lane controller).<br>**alarmOutNo**: alarm output No., integer, it is between 1 and 4. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_ChannelControllerAlarmOut***<br>Failed: ***XML_ResponseStatus*** |

**Table A-40 PUT /ISAPI/AccessControl/ChannelControllerAlarmOut?controllerType=&alarmOutNo=**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Set the alarm output parameters of the lane controller. |
| **Query** | **controllerType**: lane controller type, string, it can be set to "Master" (master lane controller) or "Slave" (slave lane controller).<br>**alarmOutNo**: alarm output No., integer, it is between 1 and 4. |
| **Request** | ***XML_ChannelControllerAlarmOut*** |
| **Response** | ***XML_ResponseStatus*** |

## A.34 /ISAPI/AccessControl/ChannelControllerAlarmOutControl

Control the alarm output of the lane controller.

### Request URI Definition

**Table A-41 PUT /ISAPI/AccessControl/ChannelControllerAlarmOutControl**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Control the alarm output of the lane controller. |
| **Query** | None. |

| | |
|---|---|
| **Request** | ***XML_ChannelControllerAlarmOutControl*** |
| **Response** | ***XML_ResponseStatus*** |

## A.35 /ISAPI/AccessControl/ChannelControllerAlarmOutControl/capabilities

Get the capability of controlling the alarm output of the lane controller.

### Request URI Definition

**Table A-42 GET /ISAPI/AccessControl/ChannelControllerAlarmOutControl/capabilities**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the capability of controlling the alarm output of the lane controller. |
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_Cap_ChannelControllerAlarmOutControl***<br>Failed: ***XML_ResponseStatus*** |

## A.36 /ISAPI/AccessControl/ChannelControllerCfg

Get or set the lane controller parameters.

### Request URI Definition

**Table A-43 GET /ISAPI/AccessControl/ChannelControllerCfg**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the lane controller parameters. |
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_ChannelControllerCfg***<br>Failed: ***XML_ResponseStatus*** |

**Table A-44 PUT /ISAPI/AccessControl/ChannelControllerCfg**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Set the lane controller parameters. |
| **Query** | None. |
| **Request** | ***XML_ChannelControllerCfg*** |
| **Response** | ***XML_ResponseStatus*** |

## A.37 /ISAPI/AccessControl/ChannelControllerCfg/capabilities

Get the lane controller configuration capability.

### Request URI Definition

**Table A-45 GET /ISAPI/AccessControl/ChannelControllerCfg/capabilities**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the lane controller configuration capability. |
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_Cap_ChannelControllerCfg***<br>Failed: ***XML_ResponseStatus*** |

## A.38 /ISAPI/AccessControl/channelControllerTypeCfg/capabilities?format=json

Get the configuration capability of the lane controller's device type.

### Request URI Definition

**Table A-46 GET /ISAPI/AccessControl/channelControllerTypeCfg/capabilities?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the configuration capability of the lane controller's device type. |
| **Query** | **format**: determine the format of request or response message. |

| | |
|---|---|
| **Request** | None. |
| **Response** | Succeeded: ***JSON_ChannelControllerTypeCfgCap***<br>Failed: ***JSON_ResponseStatus*** |

## A.39 /ISAPI/AccessControl/channelControllerTypeCfg?format=json

Get or set the device type parameters of the lane controller.

### Request URI Definition

**Table A-47 GET /ISAPI/AccessControl/channelControllerTypeCfg?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the device type parameters of the lane controller. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_ChannelControllerTypeCfg***<br>Failed: ***JSON_ResponseStatus*** |

**Table A-48 PUT /ISAPI/AccessControl/channelControllerTypeCfg?format=json**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Set the device type parameters of the lane controller. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | ***JSON_ChannelControllerTypeCfg*** |
| **Response** | ***JSON_ResponseStatus*** |

## A.40 /ISAPI/AccessControl/Configuration/IRCfg/capabilities?format=json

Get active infrared intrusion capability.

### Request URI Definition

**Table A-49 GET /ISAPI/AccessControl/Configuration/IRCfg/capabilities?format=json**

| **Method** | GET |
|---|---|
| **Description** | Get active infrared intrusion capability. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_IRCfgCap***<br>Failed: ***JSON_ResponseStatus*** |

## A.41 /ISAPI/AccessControl/Configuration/IRCfg?format=json

Get or set active infrared intrusion parameters.

### Request URI Definition

**Table A-50 GET /ISAPI/AccessControl/Configuration/IRCfg?format=json**

| **Method** | GET |
|---|---|
| **Description** | Get active infrared intrusion parameters. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_IRCfg***<br>Failed: ***JSON_ResponseStatus*** |

**Table A-51 PUT /ISAPI/AccessControl/Configuration/IRCfg?format=json**

| **Method** | PUT |
|---|---|
| **Description** | Set active infrared intrusion parameters. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | ***JSON_IRCfg*** |
| **Response** | ***JSON_ResponseStatus*** |

## A.42 /ISAPI/AccessControl/Configuration/NFCCfg/capabilities?format=json

Get the configuration capability of enabling NFC (Near-Field Communication) function.

### Request URI Definition

**Table A-52 GET /ISAPI/AccessControl/Configuration/NFCCfg/capabilities?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the configuration capability of enabling NFC (Near-Field Communication) function. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_NFCCfgCap***<br>Failed: ***JSON_ResponseStatus*** |

## A.43 /ISAPI/AccessControl/Configuration/NFCCfg?format=json

Operations about the configuration of enabling NFC (Near-Field Communication) function.

### Request URI Definition

**Table A-53 GET /ISAPI/AccessControl/Configuration/NFCCfg?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the parameters of enabling NFC (Near-Field Communication) function. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_NFCCfg***<br>Failed: ***JSON_ResponseStatus*** |

Table A-54 PUT /ISAPI/AccessControl/Configuration/NFCCfg?format=json

| | |
|---|---|
| **Method** | PUT |
| **Description** | Set the parameters of enabling NFC (Near-Field Communication) function. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | ***JSON_NFCCfg*** |
| **Response** | ***JSON_ResponseStatus*** |

## A.44 /ISAPI/AccessControl/Configuration/RFCardCfg/capabilities?format=json

Get the configuration capability of enabling RF (Radio Frequency) card recognition.

### Request URI Definition

Table A-55 GET /ISAPI/AccessControl/Configuration/RFCardCfg/capabilities?format=json

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the configuration capability of enabling RF (Radio Frequency) card recognition. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_RFCardCfgCap***<br>Failed: ***JSON_ResponseStatus*** |

## A.45 /ISAPI/AccessControl/Configuration/RFCardCfg?format=json

Operations about the configuration of enabling RF (Radio Frequency) card recognition.

### Request URI Definition

**Table A-56 GET /ISAPI/AccessControl/Configuration/RFCardCfg?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the parameters of enabling RF (Radio Frequency) card recognition. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_RFCardCfg***<br>Failed: ***JSON_ResponseStatus*** |

**Table A-57 PUT /ISAPI/AccessControl/Configuration/RFCardCfg?format=json**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Set the parameters of enabling RF (Radio Frequency) card recognition. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | ***JSON_RFCardCfg*** |
| **Response** | ***JSON_ResponseStatus*** |

## A.46 /ISAPI/AccessControl/FaceCompareCond

Get or set the condition parameters of face picture comparison.

### Request URI Definition

**Table A-58 GET /ISAPI/AccessControl/FaceCompareCond**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the condition parameters of face picture comparison. |
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_FaceCompareCond***<br>Failed: ***XML_ResponseStatus*** |

Table A-59 PUT /ISAPI/AccessControl/FaceCompareCond

| Method | PUT |
|---|---|
| **Description** | Set the condition parameters of face picture comparison. |
| **Query** | None. |
| **Request** | ***XML_FaceCompareCond*** |
| **Response** | ***XML_ResponseStatus*** |

## A.47 /ISAPI/AccessControl/FaceCompareCond/capabilities

Get condition configuration capability of face picture comparison.

### Request URI Definition

Table A-60 GET /ISAPI/AccessControl/FaceCompareCond/capabilities

| Method | GET |
|---|---|
| **Description** | Get condition configuration capability of face picture comparison. |
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_Cap_FaceCompareCond***<br>Failed: ***XML_ResponseStatus*** |

## A.48 /ISAPI/AccessControl/GateDialAndInfo

Get the local DIP (Dual In-line Package) and information of the turnstile.

### Request URI Definition

Table A-61 GET /ISAPI/AccessControl/GateDialAndInfo

| Method | GET |
|---|---|
| **Description** | Get the local DIP (Dual In-line Package) and information of the turnstile. |
| **Query** | None. |

| | |
|---|---|
| **Request** | None. |
| **Response** | Succeeded: ***XML_GateDialAndInfo***<br>Failed: ***XML_ResponseStatus*** |

## A.49 /ISAPI/AccessControl/GateDialAndInfo/capabilities

Get the capability of getting the local DIP (Dual In-line Package) and information of the turnstile.

### Request URI Definition

**Table A-62 GET /ISAPI/AccessControl/GateDialAndInfo/capabilities**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the capability of getting the local DIP (Dual In-line Package) and information of the turnstile. |
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_Cap_GateDialAndInfo***<br>Failed: ***XML_ResponseStatus*** |

## A.50 /ISAPI/AccessControl/GateIRStatus

Get the status of the active infrared intrusion detector of the turnstile.

### Request URI Definition

**Table A-63 GET /ISAPI/AccessControl/GateIRStatus**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the status of the active infrared intrusion detector of the turnstile. |
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_GateIRStatus***<br>Failed: ***XML_ResponseStatus*** |

## A.51 /ISAPI/AccessControl/GateIRStatus/capabilities

Get the capability of getting the status of the active infrared intrusion detector of the turnstile.

### Request URI Definition

**Table A-64 GET /ISAPI/AccessControl/GateIRStatus/capabilities**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the capability of getting the status of the active infrared intrusion detector of the turnstile. |
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_Cap_GateIRStatus***<br>Failed: ***XML_ResponseStatus*** |

## A.52 /ISAPI/AccessControl/GateRelatedPartsStatus

Get the related components' status of the turnstile.

### Request URI Definition

**Table A-65 GET /ISAPI/AccessControl/GateRelatedPartsStatus**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the related components' status of the turnstile. |
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_GateRelatedPartsStatus***<br>Failed: ***XML_ResponseStatus*** |

## A.53 /ISAPI/AccessControl/GateRelatedPartsStatus/capabilities

Get the capability of getting the related components' status of the turnstile.

### Request URI Definition

**Table A-66 GET /ISAPI/AccessControl/GateRelatedPartsStatus/capabilities**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the capability of getting the related components' status of the turnstile. |
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_Cap_GateRelatedPartsStatus***<br>Failed: ***XML_ResponseStatus*** |

## A.54 /ISAPI/AccessControl/GateStatus

Get the general turnstile status.

### Request URI Definition

**Table A-67 GET /ISAPI/AccessControl/GateStatus**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the general turnstile status. |
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_GateStatus***<br>Failed: ***XML_ResponseStatus*** |

## A.55 /ISAPI/AccessControl/GateStatus/capabilities

Get the capability of getting the general turnstile status.

### Request URI Definition

**Table A-68 GET /ISAPI/AccessControl/GateStatus/capabilities**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the capability of getting the general turnstile status. |

| | |
|---|---|
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_Cap_GateStatus***<br>Failed: ***XML_ResponseStatus*** |

## A.56 /ISAPI/AccessControl/GetAcsEvent/capabilities

Get capability of getting access control event.

### Request URI Definition

**Table A-69 GET /ISAPI/AccessControl/GetAcsEvent/capabilities**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get capability of getting access control event. |
| **Query** | None. |
| **Request** | None. |
| **Response** | ***XML_Cap_GetAcsEvent*** |

## A.57 /ISAPI/AccessControl/IdentityTerminal/capabilities

Get configuration capability of intelligent identity recognition terminal.

### Request URI Definition

**Table A-70 GET /ISAPI/AccessControl/IdentityTerminal/capabilities**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get configuration capability of intelligent identity recognition terminal. |
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_Cap_IdentityTerminal***<br>Failed: ***XML_ResponseStatus*** |

## A.58 /ISAPI/AccessControl/IdentityTerminal

Operations about configuration of intelligent identity recognition terminal.

### Request URI Definition

**Table A-71 GET /ISAPI/AccessControl/IdentityTerminal**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the configuration parameters of intelligent identity recognition terminal. |
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_IdentityTerminal***<br>Failed: ***XML_ResponseStatus*** |

**Table A-72 PUT /ISAPI/AccessControl/IdentityTerminal**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Set the parameters of intelligent identity recognition terminal. |
| **Query** | None. |
| **Request** | ***XML_IdentityTerminal*** |
| **Response** | ***XML_ResponseStatus*** |

## A.59 /ISAPI/AccessControl/OfflineCapture/capabilities?format=json

Get the offline collection capability.

### Request URI Definition

**Table A-73 GET /ISAPI/AccessControl/OfflineCapture/capabilities?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the offline collection capability. |
| **Query** | **format**: determine the format of request or response message. |

| | |
|---|---|
| **Request** | None. |
| **Response** | Succeeded: ***JSON_OfflineCaptureCap***<br>Failed: ***JSON_ResponseStatus*** |

## A.60 /ISAPI/AccessControl/OfflineCapture/DataCollections/<captureNo>?format=json

Deleted a specific piece of offline collected data.

### Request URI Definition

**Table A-74 DELETE /ISAPI/AccessControl/OfflineCapture/DataCollections/<captureNo>?format=json**

| | |
|---|---|
| **Method** | DELETE |
| **Description** | Deleted a specific piece of offline collected data. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | ***JSON_ResponseStatus*** |

### Remarks

The <**captureNo**> in the request URI refers to the collection No.

## A.61 /ISAPI/AccessControl/OfflineCapture/DataCollections/searchTask?format=json

Search for the collected data.

### Request URI Definition

**Table A-75 POST /ISAPI/AccessControl/OfflineCapture/DataCollections/searchTask?format=json**

| | |
|---|---|
| **Method** | POST |
| **Description** | Search for the collected data. |
| **Query** | **format**: determine the format of request or response message. |

| | |
|---|---|
| **Request** | ***JSON_SearchTaskCond*** |
| **Response** | Succeeded: ***JSON_SearchTaskResponse***<br>Failed: ***JSON_ResponseStatus*** |

## A.62 /ISAPI/AccessControl/OfflineCapture/DataCollections?format=json

Delete all offline collected data.

### Request URI Definition

**Table A-76 DELETE /ISAPI/AccessControl/OfflineCapture/DataCollections?format=json**

| | |
|---|---|
| **Method** | DELETE |
| **Description** | Delete all offline collected data. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | ***JSON_ResponseStatus*** |

## A.63 /ISAPI/AccessControl/OfflineCapture/dataOutput/progress?format=json

Get the progress of exporting the offline collected data.

### Request URI Definition

**Table A-77 GET /ISAPI/AccessControl/OfflineCapture/dataOutput/progress?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the progress of exporting the offline collected data. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_DataOutputProgress***<br>Failed: ***JSON_ResponseStatus*** |

## A.64 /ISAPI/AccessControl/OfflineCapture/dataOutput?format=json

Export the offline collected data.

### Request URI Definition

**Table A-78 PUT /ISAPI/AccessControl/OfflineCapture/dataOutput?format=json**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Export the offline collected data. |
| **Query** | **format**: determine the format of request or response message.<br>**security**: the version No. of encryption scheme. When **security** does not exist, it indicates that the data is not encrypted; when **security** is 1, it indicates that the nodes of sensitive information in the message are encrypted in AES128 CBC mode; when **security** is 2, it indicates that the nodes of sensitive information in the message are encrypted in AES256 CBC mode. In the message of this request URI, the value of the field **password** should be encrypted.<br>**iv**: the initialization vector, and it is required when **security** is 1 or 2. |
| **Request** | ***JSON_DataOutputCfg*** |
| **Response** | ***JSON_ResponseStatus*** |

## A.65 /ISAPI/AccessControl/OfflineCapture/progress?format=json

Get the offline collection progress.

### Request URI Definition

**Table A-79 GET /ISAPI/AccessControl/OfflineCapture/progress?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the offline collection progress. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_CaptureProgress***<br>Failed: ***JSON_ResponseStatus*** |

## A.66 /ISAPI/AccessControl/OfflineCapture/ruleInfo?format=json

Get or set the parameters of offline collection rules.

### Request URI Definition

**Table A-80 GET /ISAPI/AccessControl/OfflineCapture/ruleInfo?format=json**

| **Method** | GET |
|---|---|
| **Description** | Get the parameters of offline collection rules. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_RuleInfo***<br>Failed: ***JSON_ResponseStatus*** |

**Table A-81 PUT /ISAPI/AccessControl/OfflineCapture/ruleInfo?format=json**

| **Method** | PUT |
|---|---|
| **Description** | Set the parameters of offline collection rules. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | ***JSON_RuleInfo*** |
| **Response** | ***JSON_ResponseStatus*** |

## A.67 /ISAPI/AccessControl/OfflineCapture/uploadFailedDetails?format=json

Get the details of failing to upload the user list of offline collection.

### Request URI Definition

**Table A-82 GET /ISAPI/AccessControl/OfflineCapture/uploadFailedDetails?format=json**

| **Method** | GET |
|---|---|
| **Description** | Get the details of failing to upload the user list of offline collection. |
| **Query** | **format**: determine the format of request or response message. |

| | |
|---|---|
| **Request** | None. |
| **Response** | Succeeded: ***JSON_UploadFailedDetails***<br>Failed: ***JSON_ResponseStatus*** |

## A.68 /ISAPI/AccessControl/remoteCtrllerModeCfg/capabilities?format=json

Get the configuration capability of the keyfob control mode.

### Request URI Definition

**Table A-83 GET /ISAPI/AccessControl/remoteCtrllerModeCfg/capabilities?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the configuration capability of the keyfob control mode. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_RemoteCtrllerModeCfgCap***<br>Failed: ***JSON_ResponseStatus*** |

## A.69 /ISAPI/AccessControl/remoteCtrllerModeCfg?format=json

Get or set the parameters of the keyfob control mode.

### Request URI Definition

**Table A-84 GET /ISAPI/AccessControl/remoteCtrllerModeCfg?format=json**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the parameters of the keyfob control mode. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | None. |
| **Response** | Succeeded: ***JSON_RemoteCtrllerModeCfg***<br>Failed: ***JSON_ResponseStatus*** |

**Table A-85 PUT /ISAPI/AccessControl/remoteCtrllerModeCfg?format=json**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Set the parameters of the keyfob control mode. |
| **Query** | **format**: determine the format of request or response message. |
| **Request** | ***JSON_RemoteCtrllerModeCfg*** |
| **Response** | ***JSON_ResponseStatus*** |

## A.70 /ISAPI/AccessControl/RightControllerAudio/capabilities

Get the configuration capability of the audio file of the main controller.

### Request URI Definition

**Table A-86 GET /ISAPI/AccessControl/RightControllerAudio/capabilities**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the configuration capability of the audio file of the main controller. |
| **Query** | None. |
| **Request** | None. |
| **Response** | Succeeded: ***XML_Cap_RightControllerAudio***<br>Failed: ***XML_ResponseStatus*** |

## A.71 /ISAPI/AccessControl/RightControllerAudio/<ID>

Get or set the audio file parameters of the main controller, or delete the audio file of the main controller.

### Request URI Definition

**Table A-87 GET /ISAPI/AccessControl/RightControllerAudio/<ID>**

| | |
|---|---|
| **Method** | GET |
| **Description** | Get the audio file parameters of the main controller. |
| **Query** | None. |

| | |
|---|---|
| **Request** | None. |
| **Response** | Succeeded: ***XML_RightControllerAudio***<br>Failed: ***XML_ResponseStatus*** |

**Table A-88 PUT /ISAPI/AccessControl/RightControllerAudio/<ID>**

| | |
|---|---|
| **Method** | PUT |
| **Description** | Set the audio file parameters of the main controller. |
| **Query** | None. |
| **Request** | ***XML_RightControllerAudio*** |
| **Response** | ***XML_ResponseStatus*** |

**Table A-89 DELETE /ISAPI/AccessControl/RightControllerAudio/<ID>**

| | |
|---|---|
| **Method** | DELETE |
| **Description** | Delete the audio file of the main controller. |
| **Query** | None. |
| **Request** | None. |
| **Response** | ***XML_ResponseStatus*** |

## Remarks

- The <**ID**> in the request URI refers to the audio ID, and its range depends on the device capability.
- The timeout of deleting the audio file should be set to 20s.

# Appendix B. Request and Response Messages

## B.1 JSON_AcsEventTotalNum

AcsEventTotalNum message in JSON format

```
{
  "AcsEventTotalNum":{
    "totalNum":
/*required, integer, total number of events that match the search conditions*/
  }
}
```

## B.2 JSON_AcsEventTotalNumCond

AcsEventTotalNumCond message in JSON format

```
{
  "AcsEventTotalNumCond":{
    "major": ,
/*required, integer, major type (the type value should be transformed to the decimal number), refer to Access Control
Event Types for details*/
    "minor": ,
/*required, integer, minor type (the type value should be transformed to the decimal number), refer to Access Control
Event Types for details*/
    "startTime":"",
/*optional, string, start time (UTC time), e.g., "2016-12-12T17:30:08+08:00"*/
    "endTime":"",
/*optional, string, end time (UTC time), e.g., "2017-12-12T17:30:08+08:00"*/
    "cardNo":"",
/*optional, string, card No.*/
    "name":"",
/*optional, string, cardholder name*/
    "picEnable": ,
/*optional, boolean, whether to contain pictures: "true"-yes, "false"-no*/
    "beginSerialNo": ,
/*optional, integer, start serial No.*/
    "endSerialNo": ,
/*optional, integer, end serial No.*/
    "employeeNoString":"",
/*optional, string, employee No. (person ID)*/
    "eventAttribute":""
/*optional, string, event attribute: "attendance"-valid authentication, "other"*/
  }
}
```

## See Also

***Access Control Event Types***

## B.3 JSON_AttendanceStatusModeCfg

AttendanceStatusModeCfg message in JSON format

```
{
  "AttendanceStatusModeCfg":{
    "mode":"",
/*optional, string type, attendance mode: "disable", "manual", "auto"-automatic, "manualAndAuto"-manual and
automatic*/
    "manualStatusTime": ,
/*optional, integer type, duration of manual attendance status, unit: second. This node is valid when mode is
"manual" or "manualAndAuto"*/
    "attendanceStatusEnable":
/*optional, boolean type, whether to enable attendance status: "true"-yes (if the device has not been configured with
start time and end time of the automatic attendance mode, the user will be prompted to select the attendance
status), "false"-no (if the device has not been configured with start time and end time of the automatic attendance
mode, there will be no prompt)*/
  }
}
```

## B.4 JSON_AttendanceStatusRuleCfg

AttendanceStatusRuleCfg message in JSON format

```
{
  "AttendanceStatusRuleCfg":{
    "statusKey":"",
/*optional, string type, status shortcut key: "Up", "Down", "Left", "Right", "ESC", "OK", "notConfig". If this node is not
configured, this node will be set to "notConfig" by default*/
    "statusValue": ,
/*optional, integer type, status value*/
    "WeekPlanCfg":[{
/*optional, schedule*/
      "week":"",
/*optional, string type, day of the week: "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday",
"Sunday"*/
      "enable": ,
/*optional, boolean type, whether to enable: "true"-yes, "false"-no*/
      "beginTime":""
/*optional, start time*/
    }]
  }
}
```

## B.5 JSON_Cap_AcsEventTotalNum

AcsEventTotalNum capability message in JSON format

```
{
  "AcsEvent":{
    "AcsEventTotalNumCond":{
/*optional, search conditions*/
      "major":{
/*required, integer type, major type (the type value should be transformed to the decimal number): 0-all, 1-major
alarm type, 2-major exception type, 3-major operation type, 5-major event type, refer to
                              Access Control Event Types
                             for details*/
        "@opt":"0,1,2,3,5"
      },
      "minorAlarm":{
/*required, integer, minor alarm type (the type value should be transformed to the decimal number), refer to Access
Control Event Types for details*/
        "@opt":"1024,1025,1026,1027…"
      },
      "minorException":{
/*required, integer, minor exception type (the type value should be transformed to the decimal number), refer to
Access Control Event Types for details*/
        "@opt":"39,58,59,1024…"
      },
      "minorOperation":{
/*required, integer, minor operation type (the type value should be transformed to the decimal number), refer to
Access Control Event Types for details*/
        "@opt":"80,90,112,113…"
      },
      "minorEvent":{
/*required, integer, minor event type (the type value should be transformed to the decimal number), refer to Access
Control Event Types for details*/
        "@opt":"1,2,3,4…"
      },
      "startTime":{
/*optional, string, start time (UTC time)*/
        "@min": ,
        "@max":
      },
      "endTime":{
/*optional, string, end time (UTC time)*/
        "@min": ,
        "@max":
      },
      "cardNo":{
/*optional, string, card No.*/
        "@min": ,
        "@max":
      },
      "name":{
```

```
/*optional, string, cardholder name*/
       "@min": ,
       "@max":
     },
     "picEnable":"true,false",
/*optional, boolean, whether to contain pictures: "false"-no, "true"-yes*/
     "beginSerialNo":{
/*optional, integer, start serial No.*/
       "@min": ,
       "@max":
     },
     "endSerialNo":{
/*optional, integer, end serial No.*/
       "@min": ,
       "@max":
     },
     "employeeNoString":{
/*optional, string, employee No. (person ID)*/
       "@min": ,
       "@max":
     },
     "eventAttribute":{
/*optional, string, event attribute: "attendance"-valid authentication, "other"*/
       "@opt":"attendance,other"
     }
   },
   "totalNum":{
/*required, integer, total number of events that match the search conditions*/
     "@min": ,
     "@max":
   }
  }
}
```

## See Also

***Access Control Event Types***

## B.6 JSON_Cap_AttendanceStatusModeCfg

AttendanceStatusModeCfg capability message in JSON format

```
{
  "AttendanceStatusModeCfg":{
    "mode":{
/*optional, string type, attendance mode: "disable", "manual", "auto"-automatic, "manualAndAuto"-manual and
automatic*/
     "@opt":"disable,manual,auto,manualAndAuto"
    },
    "manualStatusTime":{
/*optional, integer type, duration of manual attendance status, unit: second. This node is valid when mode is
"manual" or "manualAndAuto"*/
      "@min":5,
      "@max":999
    },
    "attendanceStatusEnable":"true,false"
/*optional, boolean type, whether to enable attendance status: "true"-yes (if the device has not been configured with start time and end time of the automatic attendance mode, the user will be prompted to select the attendance status), "false"-no (if the device has not been configured with start time and end time of the automatic attendance mode, there will be no prompt)*/
  }
}
```

## B.7 JSON_Cap_AttendanceStatusRuleCfg

AttendanceStatusRuleCfg capability message in JSON format

```
{
  "AttendanceStatusRuleCfg":{
    "statusKey":{
/*optional, string type, status shortcut key: "Up", "Down", "Left", "Right", "ESC", "OK", "notConfig". If this node is not configured, this node will be set to "notConfig" by default*/
      "@opt":"Up,Down,Left,Right,ESC,OK"
    },
    "attendanceStatus":{
/*optional, string type, attendance status: "undefined", "checkIn"-check in, "checkOut"-check out, "breakOut"-break out, "breakIn"-break in, "overtimeIn"-overtime in, "overTimeOut"-overtime out*/
      "@opt":"undefined,checkIn,checkOut,breakOut,breakIn,overtimeIn,overtimeOut"
    },
    "statusValue":{
/*optional, integer type, status value*/
      "@min":0,
      "@max":255
    },
    "WeekPlanCfg":{
/*schedule*/
      "maxSize":7,
      "week":{
        "@opt":"Monday,Tuesday,Wednesday,Thursday,Friday,Saturday,Sunday"
      },
      "beginTime":"",
/*start time*/
      "timeValid":"minute"
/*time accuracy: "day", "hour", "minute", "second"*/
    }
  }
}
```

## B.8 JSON_CapturePreset

CapturePreset message in JSON format

```
{
  "CapturePreset":{
    "name":""
/*optional, string, name, the maximum size is 128 bytes by default. This field is NULL by default*/
  }
}
```

## B.9 JSON_CapturePresetCap

CapturePresetCap capability message in JSON format

```
{
  "CapturePresetCap":{
    "name":{
/*optional, string, name*/
      "@min":0,
      "@max":0
    }
  }
}
```

## B.10 JSON_CaptureProgress

CaptureProgress message in JSON format

```
{
  "CaptureProgress":{
    "reqCaptureNum": ,
/*optional, integer, total number of person to be collected*/
    "completelyCaptureNum": ,
/*optional, integer, number of completely collected persons*/
    "partiallyCaptureNum": ,
/*optional, integer, number of partially collected persons*/
    "reqFaceNum": ,
/*optional, integer, number of faces to be collected*/
    "faceNum": ,
/*optional, integer, number of collected faces*/
    "reqFingerprintNum": ,
/*optional, integer, number of fingerprints to be collected*/
    "fingerprintNum": ,
/*optional, integer, number of collected fingerprints*/
    "reqCardNum": ,
/*optional, integer, number of cards to be collected*/
```

```
    "cardNum": ,
/*optional, integer, number of collected cards*/
    "reqIDCardNum": ,
/*optional, integer, number of ID cards to be collected*/
    "IDCardNum": ,
/*optional, integer, number of collected ID cards*/
    "reqIssueNum": ,
/*optional, int, number of persons to be issued with smart cards*/
    "IssuedNum":
/*optional, int, number of persons that have been issued with smart cards*/
  }
}
```

## B.11 JSON_CaptureRule

CaptureRule message in JSON format

```
{
  "CaptureRule":{
    "enableCardNoLenAuto": ,
/*optional, boolean, whether to enable length self-adaption of the card serial No.*/
    "cardNoLen": ,
/*dependency, integer, length of the card serial No.: 3, 4, 7, 10, unit: byte. This field is valid when
enableCardNoLenAuto is "false". If this field is set to 3, it refers to Wiegand 26*/
    "cardTimeout":
/*optional, integer, card collection timeout, unit: ms*/
  }
}
```

## B.12 JSON_CaptureRuleCap

CaptureRuleCap capability message in JSON format

```
{
  "CaptureRuleCap":{
    "enableCardNoLenAuto":[true,flase],
/*optional, boolean, whether to enable length self-adaption of the card serial No.*/
    "cardNoLen":{
/*dependency, integer, length of the card serial No.: 3, 4, 7, 10*/
      "@opt":[3,4,7,10]
    },
    "cardTimeout":{
/*optional, integer, card collection timeout, unit: ms*/
      "@min":0,
      "@max":0
    }
  }
}
```

## B.13 JSON_CardEncryption

JSON message about card encryption parameters

```
{
  "CardEncryption": {
    "cardType": "",
/*required, string type, card types: "blank"-blank card, "private"-private CPU card, encrypted-other encrypted cards*/
    "keyLen":,
/*depend, integer, size of key for external authentication, this field is valid only when cardType is set to "encrypted"*/
    "key": ""
/*required, hexadecimal string, a 16-byte key content for external authentication*/
  }
}
```

## B.14 JSON_CardInfo_Collection

CardInfo message in JSON format

```
{
  "CardInfo":{
    "cardNo":"",
/*required, string, card No.*/
    "cardType":""
/*optional, string, card type: "TypeA_M1", "TypeA_CPU", "TypeB", "ID_125K", "FelicaCard"-FeliCa card, "DesfireCard"-
DESFire card*/
  }
}
```

## B.15 JSON_CardInfoCap

CardInfoCap capability message in JSON format

```
{
  "CardInfoCap":{
    "cardNo":{
/*required, string, card No.*/
      "@min":1,
      "@max":32
    },
    "cardType": ["TypeA_M1","TypeA_CPU","TypeB","ID_125K","FelicaCard","DesfireCard"]
/*optional, string, card type: "TypeA_M1", "TypeA_CPU", "TypeB", "ID_125K", "FelicaCard"-FeliCa card, "DesfireCard"-
DESFire card*/
  }
}
```

## B.16 JSON_CardOperationsCap

JSON message about card operation capability

```
{
 "CardOperationsCap":{
  "SectionEncryption":{
   "supportFunction":{
/*required, string, supported methods*/
    "@opt": ["put", "get", "delete", "post"]
   },
   "sectionNo":{
/*required, integer, section No.*/
    "@min": 0,
    "@max": 0
   },
   "keyType":{
/*required, string, verification key types: "private"-private key, "normal"-other valid keys*/
    "@opt": ["private", "normal"]
   },
   "password":{
/*optional, string, a hexadecimal verification key, this field is valid only when keyType is set to "nomal"*/
    "@min": 0,
    "@max": 0
   },
   "newKeyType":{
/*required, string, new key types: "private"-private key, "normal"-other valid keys*/
    "@opt": ["private", "normal"]
   },
   "KeyA":{
/*optional, string, a hexadecimal key A password*/
    "@min": 0,
    "@max": 0
   },
   "KeyB":{
/*optional, string, a hexadecimal key B password*/
    "@min": 0,
    "@max": 0
   },
   "controlBits":{
/*optional, string, a hexadecimal control bit*/
    "@min": 0,
    "@max": 0
   }
  },
  "Verification":{
   "supportFunction":{
/*required, string, supported methods*/
    "@opt": ["put", "get", "delete", "post"]
   },
   "sectionNo":{
```

```
/*required, integer, section No.*/
      "@min": 0,
      "@max": 0
    },
    "passwordType":{
/*optional, password types: "KeyA" (default), "KeyB"*/
      "@opt": ["KeyA", "KeyB"]
    },
    "password":{
/*optional, string, a hexadecimal password*/
      "@min": 0,
      "@max": 0
    }
  },
  "DataBlock":{
    "supportFunction":{
/*required, string, supported methods*/
      "@opt":["put", "get", "delete", "post"]
    },
    "addressOfBlock":{
/*optional, integer, block address*/
      "@min": 0,
      "@max": 0
    },
    "data":{
/*required, a hexBinary string, e.g., "f2345678abf2345678abf2345678abf2"*/
      "@min": 0,
      "@max": 0
    },
  },
  "DataBlockCtrl":{
    "supportFunction":{
/*required, string, supported methods*/
      "@opt":["put", "get", "delete", "post"]
    },
    "addressOfBlock":{
/*required, integer, block address*/
      "@min": 0,
      "@max": 0
    },
    "command":{
/*required, string, control commands*/
      "@opt": ["add", "minus", "copy", "paste"]
    },
    "value":{
/*depend, integer, relative value to be changed, this field is valid only when the command is set to "add" or "minus"*/
      "@min": 0,
      "@max": 0
    },
  },
  "ControlBlock":{
    "supportFunction":{
```

```
/*required, string, supported methods*/
    "@opt": ["put", "get", "delete", "post"]
   },
   "sectionNo":{
/*required, integer, section No.*/
    "@min": 0,
    "@max": 0
   },
   "KeyA":{
/*optional, string, a hexadecimal key A*/
    "@min": 0,
    "@max": 0
   },
   "KeyB":{
/*optional, string, a hexadecimal key B*/
    "@min": 0,
    "@max": 0
   },
   "controlBits":{
/*optional, string, a hexadecimal control bit*/
    "@min": 0,
    "@max": 0
   }
  },
  "CardProto":{
   "supportFunction":{
/*required, string, supported methods*/
    "@opt":["put", "get", "delete", "post"]
   },
   "protocol":{
/*required, string, operation protocol types*/
    "@opt": ["TypeA", "TypeB", "TypeAB", "125K", "all"]
   }
  },
  "CardEncryption":{
   "supportFunction":{
/*required, string, supported methods*/
    "@opt": ["put", "get", "delete", "post"]
   },
   "cardType":{
/*required, string, card types: "blank"-blank card, "private"-private CPU card, "encrypted"-other encrypted card*/
    "@opt":[ "blank","private","encrypted"]
   }
   "keyLen":{
/*depend, integer, size of key for external authentication, this field is valid only when cardType is set to "encrypted"*/
    "@min": 0,
    "@max": 0
   },
   "key": {
/*required, hexadecimal string, a 16-byte key content for external authentication*/
    "@min": 0,
    "@max": 0
    }
  },
  "CardParam":{
    "supportFunction":{
/*required, string, supported methods*/
      "@opt": ["put", "get", "delete", "post"]
    },
    "type":{
/*required, string, card types*/
      "@opt": ["CPU1356", "PSAM1", "PSAM2","PSAM3","PSAM4"]
    },
    "protocol":{
/*required, string, card protocol types*/
      "@opt": ["T0", "T1"]
    }
  },
  "CardResetResponse":{
    "supportFunction":{
/*required, string, supported methods*/
      "@opt": ["put", "get", "delete", "post"]
    },
    "data":{
/*required, string, resetting response information (usually, it is manufacturer, which is encoded by Base64 and
specified by device*/
      "@min": 0,
      "@max": 0
    }
  },
  "DataTrans":{
    "supportFunction":{
/*required, string, supported methods*/
      "@opt": ["put", "get", "delete", "post"]
    },
    "content":{
/*required, string, data to be passed through, which is encoded in Base64*/
      "@min": 0,
      "@max": 0
    }
  },
  "Issue":{
/*capability of sending a request for card issuing and getting the current card issuing status and real-time card issuing
results, related URIs: /ISAPI/AccessControl/CardOperations/localIssueRequest?format=json and /ISAPI/AccessControl/
CardOperations/localIssueStatus?format=json*/
    "supportFunction":{
/*required, string, supported methods. The actually supported methods will be returned*/
      "@opt":["put", "get", "delete", "post"]
    },
    "LocalIssueRequest":{
      "operation":{
/*required, string, operation type: "face"-issue card to be enrolled with face picture, "fingerprint"-issue card to be
enrolled with fingerprint*/
        "@opt":["face", "fingerprint"]
      },
      "FPIndex":{
/*optional, int, fingerprint storage index (card storage area). This field is valid when operation is "fingerprint"*/
        "@min":0,
        "@max":0
      },
      "facePic":{
/*optional, string, face picture type: "visible"-visible light picture, "infrared"-IR light picture. This field is valid when
operation is "face"*/
        "@opt":["visible", "infrared"]
      }
    },
    "LocalIssueRes":{
      "status":{
/*required, string, card issuing status: "ok"-succeeded, "failed"-card operation failed, "timeout"-timed out,
"verifiyFailure"-authentication failed, "noCard"-no card detected, "processing"-processing*/
        "@opt":["ok", "failed", "processing", "timeout", "verifiyFailure", "noCard"]
      },
      "cardNo":{
/*optional, string, issued card No.*/
        "@min":0
      },
      "cardErrorCode":{
/*dependent, string, internal error code of card operation returned by the device*/
        "@opt":
      }
    }
  },
  "localIssueCfg":{
/*capability of configuring rule parameters for issuing smart cards, related URI: /ISAPI/AccessControl/CardOperations/
localIssueCfg?format=json*/
    "validFP":{
/*optional, array of int, valid fingerprint ID. This field is valid for applying fingerprint to the card*/
      "@size":2,
      "@min":1,
      "@max":10
    },
    "validFacePicture":{
/*optional, string, valid face picture type: "visible"-visible light picture, "infrared"-IR light picture. This field is valid for
applying face picture to the card*/
      "@opt":["visible", "infrared"]
    }
  },
  "ClearData":{
/*capability of deleting data from the card, related URI: /ISAPI/AccessControl/CardOperations/clearData?
format=json*/
    "supportFunction":{
/*required, string, supported methods. The actually supported methods will be returned*/
      "@opt":["put", "get", "delete", "post"]
    },
    "checkAll":{
/*optional, boolean, whether to delete all data*/
```

```
    "@opt":[true, false]
   },
   "checkFingerprint":{
/*optional, boolean, whether to delete fingerprint data. This field is valid when checkAll is false or does not exist*/
    "@opt":[true, false]
   },
   "fingerprints":{
/*optional, array of int, list of addresses whether the fingerprints to be deleted are stored. This field is valid when
checkFingerprint exists. If this field does not exist, it indicates deleting all fingerprints*/
    "@size":2,
    "@min":0,
    "@max":0
   },
   "checkFacePicture":{
/*optional, boolean, whether to delete face data. This field is valid when checkAll is false or does not exist*/
    "@opt":[true, false]
   },
   "checkCustom":{
/*optional, boolean, whether to delete custom data. This field is valid when checkAll is false or does not exist*/
    "@opt":[true, false]
   },
   "ClearDataRes":{
    "status":{
/*required, string, card issuing status: "ok"-succeeded, "failed"-operation failed, "timeout"-timed out, "verifiyFailure"-
authentication failed, "noCard"-no card detected, "processing"-processing*/
     "@opt":["ok", "failed", "processing", "timeout", "verifiyFailure", "noCard"]
    },
    "cardErrorCode":{
/*dependent, int, internal error code of card operation*/
     "@opt":
    }
   }
  },
  "CustomData":{
/*capability of setting custom card information, related URI: /ISAPI/AccessControl/CardOperations/customData?
format=json*/
   "supportFunction":{
/*required, string, supported methods. The actually supported methods will be returned*/
    "@opt":["put", "get", "delete", "post"]
   },
   "address":{
/*optional, int, start address for writing. By default the data will be written from the start address*/
    "@min":0,
    "@max":0
   },
   "length":{
/*optional, int, length of source data to be written, it is 0 by default, unit: byte*/
    "@min":0,
    "@max":0
   },
   "data":{
/*required, string, custom information encoded by Base64*/
```

```
    "@min":0,
    "@max":0
   },
   "CustomDataRes":{
    "status":{
/*required, string, card issuing status: "ok"-succeeded, "failed"-operation failed, "timeout"-timed out, "verifiyFailure"-
authentication failed, "noCard"-no card detected, "processing"-processing*/
     "@opt":["ok", "failed", "processing", "timeout", "verifiyFailure", "noCard"]
    },
    "cardErrorCode":{
/*dependent, int, internal error code of card operation*/
     "@opt":
    }
   }
  },
  "CustomDataSearchCond":{
/*condition configuration capability of searching for custom card information, related URI: /ISAPI/AccessControl/
CardOperations/customData/searchTask?format=json*/
   "address":{
/*optional, int, start address for reading. By default the data will be read from the start address*/
    "@min":0,
    "@max":0
   },
   "length":{
/*optional, int, length of data to be read, it is 0 by default, unit: byte*/
    "@min":0,
    "@max":0
   }
  },
  "CustomDataResult":{
/*result capability of searching for custom card information, related URI: /ISAPI/AccessControl/CardOperations/
customData/searchTask?format=json*/
   "length":{
/*required, int, length of data that has been read, unit: byte*/
    "@min":0,
    "@max":0
   },
   "data":{
/*required, string, card information encoded by Base64*/
    "@min":0,
    "@max":0
   },
   "status":{
/*required, string, card issuing status: "ok"-succeeded, "failed"-operation failed, "timeout"-timed out, "verifiyFailure"-
authentication failed, "noCard"-no card detected, "processing"-processing*/
    "@opt":["ok", "failed", "processing", "timeout", "verifiyFailure", "noCard"]
   },
   "cardErrorCode":{
/*required, int, internal error code of card operation*/
    "@opt":
   }
  },
   "CardIssueStatus":{
/*capability of getting the smart card issuing status, related URI: /ISAPI/AccessControl/CardOperations/
cardIssueStatus?format=json*/
     "status":{
/*required, string, card issuing status: "ok"-succeeded, "failed"-operation failed, "timeout"-timed out, "verifiyFailure"-
authentication failed, "noCard"-no card detected, "processing"-processing*/
       "@opt":["ok", "failed", "processing", "timeout", "verifiyFailure", "noCard"]
     },
     "cardNo":{
/*optional, string, issued card No.*/
       "@min":0,
       "@max":0
     },
     "cardErrorCode":{
/*dependent, int, internal error code of card operation*/
       "@opt":
     },
     "face":{
/*optional, boolean, issuing status of the card containing the face picture: true-issued, false-not issued*/
       "@opt":[true, false]
     },
     "fingprint1":{
/*optional, boolean, issuing status of the card containing fingerprint 1: true-issued, false-not issued*/
       "@opt":[true, false]
     },
     "fingprint2":{
/*optional, boolean, issuing status of the card containing fingerprint 2: true-issued, false-not issued*/
       "@opt":[true, false]
     },
     "customData":{
/*optional, boolean, issuing status of the card containing custom information: true-issued, false-not issued*/
       "@opt":[true, false]
     }
   }
  }
}
```

## B.17 JSON_CardParam

JSON message about card parameters

```
{
  "CardParam": {
    "type": ""
/*required, string, card types: " CPU1356,PSAM1,PSAM2,PSAM3,PSAM4"*/
    "protocol": ""
/*required, string, card protocol types: "T0,T1"*/
  }
}
```

## B.18 JSON_CardProto

JSON message about operation protocol types of card

```
{
  "CardProto": {
    "protocol": "TypeA"
/*required, string, operation protocol types: "TypeA,TypeB,TypeAB,125K,all"*/
  }
}
```

## B.19 JSON_CardResetResponse

JSON message about card resetting response

```
{
  "CardResetResponse": {
    "data": ""
/*required, string, resetting response information (usually, it is manufacturer, which is encoded by Base64 and
specified by device*/
  }
}
```

## B.20 JSON_ChannelControllerTypeCfg

JSON message about the device type parameters of the lane controller

```
{
  "ChannelControllerTypeCfg":{
    "deviceModel":""
/*required, string, device type: "K3Y501-A"-DS-K3Y501 series flap barrier, "K3B501S-A"-DS-K3B501S series swing
barrier, "K3B601S-A"-DS-K3B601S series swing barrier, "K3G501"-DS-K3G501 series tripod turnstile*/
  }
}
```

## B.21 JSON_ChannelControllerTypeCfgCap

JSON message about the configuration capability of the lane controller's device type

```
{
  "ChannelControllerTypeCfgCap":{
    "deviceModel":{
/*required, device type: "K3Y501-A"-DS-K3Y501 series flap barrier, "K3B501S-A"-DS-K3B501S series swing barrier,
"K3B601S-A"-DS-K3B601S series swing barrier, "K3G501"-DS-K3G501 series tripod turnstile*/
      "@opt":["K3Y501-A","K3B501S-A","K3B601S-A","K3G501"]
    }
  }
}
```

## B.22 JSON_ClearData

JSON message about the conditions of deleting data from the card

```
{
  "ClearData":{
    "checkAll":true,
/*optional, boolean, whether to delete all data*/
    "checkFingerprint":true,
/*optional, boolean, whether to delete fingerprint data. This node is valid when the value of checkAll is false or the
node checkAll does not exist*/
    "fingerprints":[1, 2],
/*optional, array of int, address list of storage areas where the fingerprints to be deleted are stored. This node is valid
when the node checkFingerprint exists. If this node does not exist, it indicates deleting all fingerprints*/
    "checkFacePicture":true,
/*optional, boolean, whether to delete face data. This node is valid when the value of checkAll is false or the node
checkAll does not exist*/
    "checkCustom":true
/*optional, boolean, whether to delete custom data. This node is valid when the value of checkAll is false or the node
checkAll does not exist*/
  }
}
```

## B.23 JSON_ClearDataRes

JSON message about the result parameters of deleting data from the card

```
{
  "ClearDataRes":{
    "status":"ok",
/*required, string, card issuing status: "ok"-succeeded, "failed"-operation failed, "timeout"-timed out, "verifiyFailure"-
authentication failed, "noCard"-no card detected, "processing"-processing*/
    "cardErrorCode":
/*dependent, int, internal error code of card operation*/
  }
}
```

## B.24 JSON_ControlBlock

JSON message about the control block parameters of a specific section.

```
{
  "ControlBlock": {
    "sectionNo": ,
/*required, integer, section No.*/
    "KeyA": "",
/*optional, string type, a hexadecimal key A password*/
    "KeyB": "",
/*optional, string type, a hexadecimal key B password*/
    "controlBits":""
/*optional, string type, a hexadecimal control bit*/
  }
}
```

## B.25 JSON_CustomData

JSON message about the conditions of setting custom card information

```
{
  "CustomData":{
    "address":1,
/*optional, int, start address for writing. By default the data will be written from the start address*/
    "length":1,
/*optional, int, length of the source data to be written, it is 0 by default, unit: byte*/
    "data":""
/*required, string, custom information encoded by Base64*/
  }
}
```

## B.26 JSON_CustomDataRes

JSON message about the result parameters of setting custom card information

```
{
  "CustomDataRes":{
    "status":"ok",
/*required, string, card issuing status: "ok"-succeeded, "failed"-operation failed, "timeout"-timed out, "verifiyFailure"-
authentication failed, "noCard"-no card detected, "processing"-processing*/
    "cardErrorCode":
/*dependent, int, internal error code of card operation*/
  }
}
```

## B.27 JSON_CustomDataResult

JSON message about the results of searching for custom card information

```
{
  "CustomDataResult":{
    "status":"ok",
/*required, string, card issuing status: "ok"-succeeded, "failed"-operation failed, "timeout"-timed out, "verifiyFailure"-
authentication failed, "noCard"-no card detected, "processing"-processing*/
    "cardErrorCode":0,
/*dependent, int, internal error code of card operation. This node is valid when the value of status is "failed"*/
    "length":1,
/*dependent, int, length of the source data that has been read, unit: byte. This node is valid when the value of status
is "ok"*/
    "data":""
/*dependent, string, card information encoded by Base64. This node is valid when the value of status is "ok"*/
  }
}
```

## B.28 JSON_CustomDataSearchCond

JSON message about condition parameters of searching for custom card information

```
{
  "CustomDataSearchCond":{
    "address":1,
/*optional, int, start address for reading data. By default the data will be read from the start address*/
    "length":1
/*optional, int, length of the data that can be read, it is 0 by default, unit: byte*/
  }
}
```

## B.29 JSON_DataBlock

JSON message about data block details

```
{
  "DataBlock": {
    "addressOfBlock": ,
/*optional, integer, block address*/
    "data": "",
/*required, string, a hexBinary character string, i.e., "f2345678abf2345678abf2345678abf2"*/
  }
}
```

## B.30 JSON_DataBlockCtrl

JSON message about operation parameters of data block

```
{
  "DataBlockCtrl": {
    "addressOfBlock": ,
/*required, integer, block address*/
    "command":"",
/*required, string, control commands: "add, minus, copy, paste"*/
    "value":,
/*depend, integer, relative value to be changed, this field is value only when the command is set to "add" or "minus"*/
  }
}
```

## B.31 JSON_DataOutputCfg

DataOutputCfg message in JSON format

```
{
  "DataOutputCfg":{
    "password":"",
/*required, string, password for exporting*/
    "type":""
/*optional, string, exporting type: "UsbDisk"-exporting via USB flash drive, "UsbPrivate"-exporting via private USB,
"ISAPI"-exporting via ISAPI*/
  }
}
```

## B.32 JSON_DataOutputProgress

DataOutputProgress message in JSON format

```
{
  "DataOutputProgress":{
    "progress":
/*required, integer, exporting progress*/
  }
}
```

## B.33 JSON_DataTrans

JSON message about data package to be passed through

```
{
  "DataTrans": {
    "content": ""
/*required, string, data to be passed through, which is encoded by Base64*/
  }
}
```

## B.34 JSON_EventNotificationAlert_Alarm/EventInfo

EventNotificationAlert message with alarm or event information in JSON format.

```
{
  "ipAddress": "",
/*required, device IPv4 address , string, the maximum size is 32 bytes*/
  "ipv6Address": "",
/*optional, device IPv6 address, string, the maximum size is 128 bytes*/
  "portNo": ,
/*optional, device port No., integer32*/
  "protocol": "",
/*optional, protocol type, "HTTP, HTTPS", string, the maximum size is 32 bytes*/
  "macAddress": "",
/*optional, MAC address, string, the maximum size is 32 bytes, e.g., 01:17:24:45:D9:F4*/
  "channelID": "",
/*optional, device channel No., integer32*/
  "dateTime": "",
/*optional, string, alarm/event triggered or occurred time based on ISO8601, the maximum size is 32 bytes, e.g., 2009-11-14T15:27Z*/
  "activePostCount": "",
/*required, alarm/event frequency, integer32*/
  "eventType": "",
/*required, alarm/event type, "captureResult, faceCapture,...", string, the maximum size is 128 bytes*/
  "eventState": "",
/*required, string, the maximum size is 32 bytes, durative alarm/event status: "active"-valid, "inactive"-invalid*/
  "eventDescription": "",
/*required, event description, string, the maximum size is 128 bytes*/
  "deviceID":"",
/*string type, device ID*/
  "uuid":"",
/*string type, event UUID, which is used to uniquely identify an event, the standard UUID format is xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx*/
  ...
/*optional, for different alarm/event types, the nodes are different, see the message examples in different applications*/
}
```

## B.35 JSON_IdentityInfo

IdentityInfo message in JSON format

```
{
  "IdentityInfo":{
    "chnName":"",
/*optional, string, reserved*/
    "enName":"",
/*optional, string, English name*/
    "sex":"",
/*optional, string, gender: "male", "female"*/
    "birth":"",
/*optional, string, date of birth, e.g., 1990-02-24*/
    "addr":"",
/*optional, string, address*/
    "IDCardNo":"",
/*optional, string, ID card No., it is the sensitive information that should be encrypted*/
    "issuingAuthority":"",
/*optional, string, authority*/
    "startDate":"",
/*optional, string, start time of the validity period*/
    "endDate":"",
/*optional, string, end time of the validity period*/
    "passNo":"",
/*optional, string, entry-exit permit No.*/
    "issueNumber":"",
/*optional, string, issuing times*/
    "certificateType":"",
/*optional, string, certificate type*/
    "permanentResidenceCardNo":"",
/*optional, string, permanent resident card No.*/
    "nationalityOrAreaCode":"",
/*optional, string, country or region code*/
    "version":"",
/*optional, string, certificate version No.*/
    "receivingAuthorityCode":"",
/*optional, string, acceptance authority code*/
    "FingerprintList":[{
      "fingerprint":""
/*optional, string, fingerprint information, it is encoded using base64*/
    }],
    "pic":""
/*optional, string, ID photo information, it is encoded using base64. The encrypted data should be decrypted using
the specific decryption library*/
  }
}
```

## B.36 JSON_IdentityInfoCap

IdentityInfoCap capability message in JSON format

```
{
  "IdentityInfoCap":{
    "IdentityInfoCond":{ },
/*optional, conditions of collecting ID card information*/
    "chnName":{
/*optional, string, reserved*/
      "@min":0,
      "@max":0
    },
   "enName":{
/*optional, string, English name*/
    "@min":0,
    "@max":0
   },
   "sex":{
/*optional, string, gender: "male", "female"*/
    "@opt":["male", "female"]
   },
   "birth":{
/*optional, string, date of birth, e.g., 1990-02-24*/
    "@min":0,
    "@max":0
   },
   "addr":{
/*optional, string, address*/
    "@min":0,
    "@max":0
   },
   "IDCardNo":{
/*optional, string, ID card No.*/
    "@min":0,
    "@max":0
   },
   "issuingAuthority":{
/*optional, string, authority*/
    "@min":0,
    "@max":0
   },
   "startDate":{
/*optional, string, start time of the validity period*/
    "@min":0,
    "@max":0
   },
   "endDate":{
/*optional, string, end time of the validity period*/
    "@min":0,
    "@max":0
   },
   "passNo":{
/*optional, string, entry-exit permit No.*/
    "@min":0,
    "@max":0
   },
   "issueNumber":{
/*optional, string, issuing times*/
    "@min":0,
    "@max":0
   },
   "certificateType":{
/*optional, string, certificate type*/
    "@min":0,
    "@max":0
   },
   "permanentResidenceCardNo":{
/*optional, string, permanent resident card No.*/
    "@min":0,
    "@max":0
   },
   "nationalityOrAreaCode":{
/*optional, string, country or region code*/
    "@min":0,
    "@max":0
   },
   "version":{
/*optional, string, certificate version No.*/
    "@min":0,
    "@max":0
   },
   "receivingAuthorityCode":{
/*optional, string, acceptance authority code*/
    "@min":0,
    "@max":0
   },
   "FingerprintList":{
    "maxSize":0,
    "fingerprint":{
/*optional, string, fingerprint information, it is encoded using base64. This field is the data size capability*/
    "@min":0,
    "@max":0
    }
   },
   "pic":{
/*optional, string, ID photo information, it is encoded using base64. This field is the data size capability*/
    "@min":0,
    "@max":0
   }
  }
}
```

## B.37 JSON_IdentityInfoCond

IdentityInfoCond message in JSON format

```
{
 "IdentityInfoCond":{ }
/*currently there are no condition parameters, so this field can be set to NULL*/
}
```

## B.38 JSON_IRCfg

JSON message about active infrared intrusion parameters

```
{
  "IRCfg": {
    "enable": ,
/*required, boolean, whether to enable: true (yes), false (no)*/
    "distance":
/*optional, float, distance, unit: m*/
  }
}
```

## B.39 JSON_IRCfgCap

JSON message about active infrared intrusion capability

```
{
  "IRCfgCap": {
    "enable":[true,false],
/*required, boolean, whether to enable*/
    "distance":{
      "@opt":[0.5,1,1.5]
    }
  }
}
```

## B.40 JSON_NFCCfg

NFCCfg message in JSON format

```
{
  "NFCCfg":{
    "enable":
/*required, boolean, whether to enable NFC function: "true"-yes, "false"-no*/
  }
}
```

## B.41 JSON_NFCCfgCap

NFCCfgCap capability message in JSON format

```
{
  "NFCCfgCap":{
    "enable":"true, false"
/*required, whether to enable NFC function: "true"-yes, "false"-no (default)*/
  }
}
```

## B.42 JSON_OfflineCaptureCap

OfflineCaptureCap capability message in JSON format

```
{
  "OfflineCaptureCap":{
    "isSuportDownloadOfflineCaptureInfoTemplate":true,
/*optional, whether it supports downloading template of offline user list: "true"-yes, this node is not returned-no*/
    "isSuportUploadOfflineCaptureInfo":true,
/*optional, whether it supports uploading offline user list: "true"-yes, this node is not returned-no*/
    "isSupportDownloadCaptureData":true,
/*optional, whether it supports downloading collected data: "true"-yes, this node is not returned-no*/
    "isSupportDeleteAllData":true,
/*optional, whether it supports deleting all collected data: "true"-yes, this node is not returned-no*/
    "isSupportDeleteTheData":true,
/*optional, whether it supports deleting specific collected data: "true"-yes, this node is not returned-no*/
    "SearchTask":{
      "supportFunction":{
/*required, string, supported methods, actually supported methods will be returned*/
        "@opt":["put", "get", "delete", "post"]
      },
      "searchID":{
/*required, string, search ID which is used to check whether the upper-layer clients are the same one*/
        "@min":0,
        "@max":0
      },
      "maxResults":{
        "@min":0,
        "@max":0
      },
      "captureNoList":{
        "maxSize":0,
        "@min":0,
        "@max":0
      },
      "searchType":{
        "@opt":["new", "modified"]
      },
      "DataCollections":{
/*optional, array, matched data information that has been searched*/
        "maxSize":0,
        "captureNo":{
/*optional, integer, collection No.*/
          "@min":0,
          "@max":0
        },
    "name":{
/*optional, string, name*/
     "@min":0,
     "@max":0
    },
    "employeeNo":{
/*optional, string, employee No.*/
     "@min":0,
     "@max":0
    },
    "CardNoList":{
/*optional, string, card No. list*/
     "maxSize":0,
     "cardNo":{
      "@min": 0,
      "@max": 0
     },
     "cardType": {
/*optional, string, card type: "TypeA_M1", "TypeA_CPU", "TypeB", "ID_125K", "FelicaCard", "DesfireCard"*/
      "@opt":["TypeA_M1","TypeA_CPU","TypeB","ID_125K","FelicaCard","DesfireCard"]
     }
    },
    "IDCardNo":{
/*optional, string, ID card No.*/
     "@min":0,
     "@max":0
    },
    "FingerprintList":{
     "fingerprintID":{
      "@min":0,
      "@max":0
     },
     "fingerprint":{
/*optional, fingerprint information, it is encoded using base64*/
      "@min":0,
      "@max":0
     }
    },
    "FaceFeature":{
/*optional, string, facial feature information*/
     "isSupportFaceRegion":true,
/*optional, whether it supports facial feature area*/
     "isSupportCommonPoint":true
/*optional, whether it supports feature point coordinates (e.g., left eye, right eye, left mouth corner, right mouth
corner, nose)*/
    },
    "isSupportRiskMark":true,
/*optional, whether it supports risk data mark*/
    "dataType":{
/*optional, data type*/
     "@opt":["new", "modified","normal"]
    },
      "IdentityInfo":{
/*identity information*/
        "chnName":{
/*optional, string, Chinese name*/
          "@min":0,
          "@max":0
        },
        "enName":{
/*optional, string, English name*/
          "@min":0,
          "@max":0
        },
        "sex":{
/*optional, string, gender: "male", "female"*/
          "@opt":["male", "female"]
        },
        "birth":{
/*optional, string, data of birth, e.g., "1990-02-24"*/
          "@min":0,
          "@max":0
        },
        "addr":{
/*optional, string, address*/
          "@min":0,
          "@max":0
        },
        "IDCardNo":{
/*optional, string, ID card No.*/
          "@min":0,
          "@max":0
        },
        "issuingAuthority":{
/*optional, string, issuing authority*/
          "@min":0,
          "@max":0
        },
        "startDate":{
/*optional, string, start date of validity period*/
          "@min":0,
          "@max":0
        },
        "endDate":{
/*optional, string, end date of validity period*/
          "@min":0,
          "@max":0
        },
        "passNo":{
/*optional, string, entry-exit permit No.*/
          "@min":0,
          "@max":0
        },
        "issueNumber":{
```

```
/*optional, string, issued times*/
      "@min":0,
      "@max":0
     },
     "certificateType":{
/*optional, string, certificate type*/
      "@min":0,
      "@max":0
     },
     "permanentResidenceCardNo":{
/*optional, string, permanent resident visa No.*/
      "@min":0,
      "@max":0
     },
     "nationalityOrAreaCode":{
/*optional, string, country/region code*/
      "@min":0,
      "@max":0
     },
     "version":{
/*optional, string, certificate version No.*/
      "@min":0,
      "@max":0
     },
     "receivingAuthorityCode":{
/*optional, string, acceptance authority code*/
      "@min":0,
      "@max":0
     },
     "FingerprintList":{
      "maxSize":0,
      "fingerprint":{
/*optional, string, fingerprint information, which should be encoded by Base64*/
       "@min":0,
       "@max":0
      }
     },
     "pic":{
/*optional, string, certificate picture information, which should be encoded by Base64, encrypted and decrypted by a
specific decryption library*/
      "@min":0,
      "@max":0
     }
    },
    "CardIssueStatus":{
/*optional, issuing status list of cards containing face pictures and fingerprints*/
     "@size":0,
/*optional, capability of number of elements in the array*/
     "face":{
/*optional, boolean, card issuing status of the face picture: true-with card issued, false-without card issued*/
      "@opt":[true, false]
     },
        "fingprint1":{
/*optional, boolean, card issuing status of the fingerprint 1: true-with card issued, false-without card issued*/
          "@opt":[true, false]
        },
        "fingprint2":{
/*optional, boolean, card issuing status of the fingerprint 2: true-with card issued, false-without card issued*/
          "@opt":[true, false]
        }
      }
    }
  },
  "RuleInfo":{
/*rule list, which lists rules for collecting different types of data*/
     "reqAdminRights":[true,false],
/*required, boolean, whether the administrator permission is required: "true"-yes, "false"-no*/
     "enableCardNoLenAuto":[true,false],
/*optional, boolean, whether to enable length self-adaption of the card serial No.*/
     "maxSize":0,
     "supportFunction":{
/*required, string, supported methods, actually supported methods will be returned*/
       "@opt":["put", "get", "delete", "post"]
     },
     "dataType":{
/*required, string, data type: "name", "employeeNo"-employee No., "IDCardNo"-ID card No., "IDCardSerialNo"-ID
card serial No.,  "IDCardDetails"-ID card details, "card", "fingprint"-fingerprint, "face"*/
       "@opt":["name","employeeNo","IDCardNo","IDCardSerialNo", "IDCardDetails","card", "fingprint", "face"]
     },
     "enable":[true, false],
/*required, string, whether to collect and display: "true"-collect and display, "false"-not collect and display*/
     "uniqueCheck":[true, false],
/*dependency, boolean, whether to enable uniqueness verification: "true"-yes, "false" (default) or this node is not
returned-no. This field is valid when dataType is "name". For other data types, the field is the read-only optional
parameter*/
     "len":[{
/*dependency, integer, data length. If dataType is "name", it refers to the name length and the default value is 128.
For other data types, this field is the read-only optional parameter. This node will not be returned if it is not
supported. The capability list will be returned according to the data type*/
       "dataType":"",
       "@min":0,
       "@max":0
     }],
     "num":[{
/*dependency, integer, number of collected data, this field is valid when dataType is "fingerprint" or "card". The
capability list will be returned according to the data type*/
       "dataType":"",
       "@min":0,
       "@max":0
     }],
     "fingerprintIDs":{
/*dependency, integer, No. list of collected fingerprints, this field is valid when dataType is "fingerprint"*/
       "maxSize":0,
       "@min":0,
    "@max":0
   },
   "enableLocalIssueCard":  {
/*optional, boolean, whether to enable issuing smart cards locally*/
    "@opt": [true,false]
   },
   "isLocalStorage":  {
/*optional, boolean, whether to store face picture and fingerprint information in the device locally*/
    "@opt": [true,false]
   }
  },
  "CaptureProgress":{
   "supportFunction":{
/*required, string, supported methods, actually supported methods will be returned*/
    "@opt":["put", "get", "delete", "post"]
   },
   "reqCaptureNum":{
/*optional, integer, total number of persons to be collected*/
    "@min":0,
    "@max":0
   },
   "completelyCaptureNum":{
/*optional, integer, number of completely collected persons*/
    "@min":0,
    "@max":0
   },
   "partiallyCaptureNum":{
/*optional, integer, number of partially collected persons*/
    "@min":0,
    "@max":0
   },
   "reqFaceNum":{
/*optional, integer, number of faces to be collected*/
    "@min":0,
    "@max":0
   },
   "faceNum":{
/*optional, integer, number of collected faces*/
    "@min":0,
    "@max":0
   },
   "reqFingerprintNum":{
/*optional, integer, number of fingerprints to be collected*/
    "@min":0,
    "@max":0
   },
   "fingerprintNum":{
/*optional, integer, number of collected fingerprints*/
    "@min":0,
    "@max":0
   },
   "reqCardNum":{
```

```
/*optional, integer, number of cards to be collected*/
    "@min":0,
    "@max":0
   },
   "cardNum":{
/*optional, integer, number of collected cards*/
    "@min":0,
    "@max":0
   },
   "reqIDCardNum":{
/*optional, integer, number of ID cards to be collected*/
    "@min":0,
    "@max":0
   },
   "IDCardNum":{
/*optional, integer, number of collected ID cards*/
    "@min":0,
    "@max":0
   },
   "reqIssueNum":{
/*optional, int, number of persons to be issued with smart cards*/
    "@min": 0,
    "@max": 0
   },
   "IssuedNum":{
/*optional, int, number of persons that have been issued with smart cards*/
    "@min": 0,
    "@max": 0
   }
  },
  "DataOutput":{
   "supportFunction":{
/*required, string, supported methods, actually supported methods will be returned*/
    "@opt":["put", "get", "delete", "post"]
   },
   "password":{
/*required, string, password for exporting*/
    "@min":0,
    "@max":0
   },
   "type":{
/*optional, string, exporting method, the default method is "USB"*/
    "@opt":"USB"
   },
   "progress":{
/*required, integer, exporting progress*/
    "@min":0,
    "@max":0
   }
  }
 }
}
```

## B.43 JSON_RemoteCtrllerModeCfg

JSON message about the parameters of the keyfob control mode.

```
{
  "RemoteCtrllerModeCfg":{
    "mode": ""
/*required, string, keyfob control mode: "oneToOne"-one-to-one mode (default, the keyfob can only control one device), "oneToMany"-one-to-many mode (the keyfob can control multiple devices)*/
  }
}
```

## B.44 JSON_RemoteCtrllerModeCfgCap

JSON message about the configuration capability of the keyfob control mode

```
{
  "RemoteCtrllerModeCfgCap":{
    "mode": {
/*required, keyfob control mode: "oneToOne"-one-to-one mode (default, the keyfob can only control one device), "oneToMany"-one-to-many mode (the keyfob can control multiple devices)*/
      "@opt": ["oneToOne","oneToMany"]
    }
  }
}
```

## B.45 JSON_ResponseStatus

JSON message about response status

```
{
  "requestURL":"",
/*optional, string, request URL*/
  "statusCode": ,
/*optional, int, status code*/
  "statusString":"",
/*optional, string, status description*/
  "subStatusCode":"",
/*optional, string, sub status code*/
  "errorCode": ,
/*required, int, error code, which corresponds to subStatusCode, this field is required when statusCode is not 1. The returned value is the transformed decimal number*/
  "errorMsg":"",
/*required, string, error details, this field is required when statusCode is not 1*/
  "MErrCode": "0xFFFFFFFF",
/*optional, string, error code categorized by functional modules*/
  "MErrDevSelfEx": "0xFFFFFFFF"
/*optional, string, extension of MErrCode. It is used to define the custom error code, which is categorized by
functional modules*/
}
```

## B.46 JSON_RFCardCfg

RFCardCfg message in JSON format

```
{
  "RFCardCfg":[{
    "cardType":"",
/*required, string, card type: "EMCard"-EM card, "M1Card"-M1 card, "CPUCard"-CPU card, "IDCard"-ID card,
"DesfireCard"-DESFire card, "FelicaCard"-FeliCa card*/
    "enabled":
/*required, boolean, whether to enable RF card recognition: "true"-yes, "false"-no*/
  }]
}
```

## B.47 JSON_RFCardCfgCap

RFCardCfgCap capability message in JSON format

```
{
  "RFCardCfgCap":{
    "cardType":{
/*required, string, card type: "EMCard"-EM card, "M1Card"-M1 card, "CPUCard"-CPU card, "IDCard"-ID card,
"DesfireCard"-DESFire card, "FelicaCard"-FeliCa card*/
      "@opt":["EMCard","M1Card","CPUCard","IDCard"]
    },
    "enabled":{
/*required, boolean, whether to enable RF card recognition: "true"-yes, "false"-no*/
      "@opt":[true,false]
    }
  }
}
```

## B.48 JSON_RuleInfo

RuleInfo message in JSON format

```
{
  "RuleInfo":{
    "reqAdminRights": ,
/*required, boolean, whether the administrator permission is required: "true"-yes, "false"-no*/
    "enableCardNoLenAuto": ,
/*optional, boolean, whether to enable length self-adaption of the card serial No. The priority of this field is higher
than len*/
```

```
   "RuleList":[{
/*rule list, which contains rules for collecting different types of data*/
     "dataType":"",
/*required, string, data type: "name", "employeeNo"-employee No., "IDCardNo"-ID card No., "IDCardSerialNo"-ID
card serial No.,  "IDCardDetails"-ID card details, "card", "fingprint"-fingerprint, "face"*/
     "enable": ,
/*required, boolean, whether to collect and display: "true"-collect and display, "false"-not collect and display*/
     "uniqueCheck": ,
/*dependency, boolean, whether to enable uniqueness verification: "true"-yes, "false" (default) or this field is not
returned-no. This field is valid when dataType is "name". For other data types, this field is the read-only optional
parameter*/
     "len": ,
/*dependency, integer, data length, this field is valid when dataType is "name", "enployeeNo" or "card". The default
data length of name is 128. For other data types, this field is the read-only optional parameter. If it is not supported,
this field will not be returned*/
     "num": ,
/*dependency, integer, number of collected data, this field is valid when dataType is "fingerprint" or "card"*/
     "fingerprintIDs":
/*dependency, integer, ID list of fingerprints that need to be collected, this field is valid when dataType is
"fingerprint"*/
   }],
   "enableLocalIssueCard": true,
/*optional, boolean, whether to enable issuing smart cards locally*/
   "isLocalStorage": false
/*optional, boolean, whether to store face picture and fingerprint information in the device locally*/
  }
}
```

## B.49 JSON_SearchTaskCond

SearchTaskCond message in JSON format

```
{
  "SearchTaskCond":{
   "searchID":"",
/*required, string, search ID which is used to check whether the upper-layer clients are the same one*/
   "searchResultPosition": ,
/*required, integer32, the start position of the search result in the result list. When there are multiple records and you
cannot get all search results at a time, you can search for the records after the specified position next time. If the
device returns the picture along with the response message, this field should be between 0 and totalMatches*/
   "maxResults": ,
/*required, integer32, the maximum number of results that can be obtained by calling the URL at a time. If the device
returns the picture along with the response message, this field can only be set to 1*/
   "captureNoList": ,
/*optional, integer, collection No. list. If the collection No. is not configured, it will search all data according to
searchResultPosition*/
   "searchType":""
/*optional, search type: "new"-search and only return newly added data, "modified"-search and only return edited
data. By default all data will be searched*/
```

```
  }
}
```

## B.50 JSON_SearchTaskResponse

SearchTaskResponse message in JSON format

```
{
  "SearchTaskResponse":{
    "searchID":"",
/*required, string, search ID which is used to check whether the upper-layer clients are the same one*/
    "responseStatusStrg":"",
/*optional, string, searching status: "OK"-searching completed, "NO MATCH"-no matched results, "MORE"-searching
for more results*/
    "numOfMatches": ,
/*optional, integer32, number of returned results this time*/
    "totalMatches": ,
/*optional, integer32, total number of matched results*/
    "DataCollections":[{
/*optional, array, searched matched data information*/
      "lastCaptureNo": ,
/*required, integer, last collection No., it is used to check whether there is data lost*/
      "captureNo": ,
/*required, integer, current collection No.*/
      "name":"",
/*optional, string, name*/
      "employeeNo":"",
/*optional, string, employee No.*/
      "IDCardNo":"",
/*optional, string, ID card No.*/
      "CardNoList":[{
/*optional, string, card No. list*/
        "cardNo":"",
        "cardType": "TypeA_M1"
/*optional, string, card type: "TypeA_M1", "TypeA_CPU", "TypeB", "ID_125K", "FelicaCard", "DesfireCard"*/
      }],
      "FingerprintList":[{
        "fingerprintID": ,
/*optional, integer, fingerprint No.*/
        "fingerprint":""
/*optional, string, fingerprint information which is encoded using base64*/
      }],
      "FaceFeature":{
/*optional, feature information of face picture matting*/
        "Region":{
/*required, area coordinates of face picture matting, it is a rectangle*/
          "height": ,
/*required, float, height*/
          "width": ,
/*required, float, width*/
```

```
          "x": ,
/*required, float, X-coordinate of the left corner*/
          "y":
/*required, float, Y-coordinate of the left corner*/
        },
        "LeftEyePoint":{
/*optional, coordinates of the left eye*/
          "x": ,
/*required, float, X-coordinate, it is between 0.000 and 1*/
          "y":
/*required, float, Y-coordinate, it is between 0.000 and 1*/
        },
        "RightEyePoint":{
/*optional, coordiantes of the right eye*/
          "x": ,
/*required, float, X-coordinate, it is between 0.000 and 1*/
          "y":
/*required, float, Y-coordinate, it is between 0.000 and 1*/
        },
        "LeftMouthPoint":{
/*optional, coordinates of the left mouth corner*/
          "x": ,
/*required, float, X-coordinate, it is between 0.000 and 1*/
          "y":
/*required, float, Y-coordinate, it is between 0.000 and 1*/
        },
        "RightMouthPoint":{
/*optional, coordinates of the right mouth corner*/
          "x": ,
/*required, float, X-coordinate, it is between 0.000 and 1*/
          "y":
/*required, float, Y-coordinate, it is between 0.000 and 1*/
        },
        "NoseTipPoint":{
/*optional, coordinates of the nose*/
          "x": ,
/*required, float, X-coordinate, it is between 0.000 and 1*/
          "y":
/*required, float, Y-coordinate, it is between 0.000 and 1*/
        }
      },
      "riskDataMark": ,
/*optional, boolean, whether to mark risk data: "true"-mark the data as the risk data and person and ID comparison
failed, "false" or this field is not returned-the data is normal*/
      "dataType":"",
/*optional, string, data type and status: "new"-newly added data, "modified"-edited data, "normal"-unchanged data*/
      "IdentityInfo":{
/*identity information*/
        "chnName":"",
/*optional, string, Chinese name*/
        "enName":"",
/*optional, string, English name*/
```

```
    "sex":"",
/*optional, string, gender: "male", "female"*/
    "birth":"",
/*optional, string, data of birth, e.g., "1990-02-24"*/
    "addr":"",
/*optional, string, address*/
    "IDCardNo":"",
/*optional, string, ID card No.*/
    "issuingAuthority":"",
/*optional, string, issuing authority*/
    "startDate":"",
/*optional, string, start date of validity period*/
    "endDate":"",
/*optional, string, end date of validity period*/
    "passNo":"",
/*optional, string, entry-exit permit No.*/
    "issueNumber":"",
/*optional, string, issued times*/
    "certificateType":"",
/*optional, string, certificate type*/
    "permanentResidenceCardNo":"",
/*optional, string, permanent resident visa No.*/
    "nationalityOrAreaCode":"",
/*optional, string, country/region code*/
    "version":"",
/*optional, string, certificate version No.*/
    "receivingAuthorityCode":"",
/*optional, string, acceptance authority code*/
    "FingerprintList":[{
     "fingerprint":""
/*optional, string, fingerprint information, which should be encoded by Base64*/
    }],
    "pic":""
/*optional, string, certificate picture information, which should be encoded by Base64, encrypted and decrypted by a
specific decryption library*/
   },
   "CardIssueStatus":[{
/*optional, issuing status list of cards containing face pictures and fingerprints*/
    "cardNo":"",
/*optional, string, card information*/
    "face":true,
/*optional, boolean, card issuing status of the face picture: true-with card issued, false-without card issued*/
    "fingprint1":true,
/*optional, boolean, card issuing status of the fingerprint 1: true-with card issued, false-without card issued*/
    "fingprint2":true
/*optional, boolean, card issuing status of the fingerprint 2: true-with card issued, false-without card issued*/
   }]
  }]
 }
}
```

## B.51 JSON_SectionEncryption

JSON message about section encryption parameters

```
{
  "SectionEncryption": {
    "sectionNo": ,
/*required, integer, section No.*/
    "keyType": "",
/*required, string, key types: "private"-private key, "normal"-other valid keys*/
    "password": ""
/*depend, string, a hexadecimal verification key, this field is valid only when the keyType is "normal"*/
    "newKeyType": "",
/*required, string, new key types: "private"-private key, "normal"-other valid keys*/
    "KeyA": "",
/*depend, string, a hexadecimal password of key A, this field is valid only when the keyType is "normal"*/
    "KeyB": "",
/*depend, string, a hexadecimal password of key B, this field is valid only when the keyType is "normal"*/
    "controlBits":
/*depend, a hexadecimal control bit, this field is valid only when the keyType is "normal"*/
  }
}
```

## B.52 JSON_UploadFailedDetails

JSON message about the details of failing to upload the user list of offline collection

```
{
  "UploadFailedDetails ":{
    "description":""
/*required, string, details of failing to uploading the user list of offline collection, including detailed error descriptions
and reports*/
  }
}
```

## B.53 JSON_Verification

JSON message about verification parameters of section password.

```
{
  "Verification": {
    "sectionNo": ,
/*requried, integer, section No.*/
    "passwordType": "",
/*optional, string, password types: "KeyA" (default), "KeyB"*/
    "password": ""
/*optional, string, a hexadecimal key, which depends on the password type*/
```

```
  }
}
```

## B.54 XML_AcsAbility

AcsAbility capability message in XML format

```
<AcsAbility version="2.0">
  <channelControllerNo min="" max=""/>
  <!--required, lane controller range-->
  <doorNo min="" max=""/>
  <!--req , door No. rang  or  floor No. range-->
  <cardReaderNo min="" max=""/>
  <!--required, card reader No. range-->
  <maxCardNum></maxCardNum>
  <!--required, supported card number-->
  <caseSensorNo min="" max=""/>
  <!--required, event trigger No.-->
  <gateOpenDirectionNum opt="1,2"/>
  <!--required, the number of door opening directions (e.g., for the flap barrier which has only one direction, the attribute "opt" should be set to 1; for the swing barrier and the tripod turnstile which have two directions, the attribute "opt" should be set to 2)-->
  <DoorRelateCardReaderList>
    <!--optional, card reader No., which is linked with the door No. (it will be returned only when the card reader has linked with card reader, otherwise it will not be returned)-->
    <Action>
      <doorNo>1</doorNo>
      <cardReaderNo>1,2</cardReaderNo>
    </Action>
  </DoorRelateCardReaderList>
  <DoorStatusPlan>
    <!--required, door status schedule capability -->
    <WeekPlan>
      <!--required, weekly schedule capability -->
      <weekPlanNo min="" max=""/>
      <!--required, weekly schedule No. range -->
      <maxDaySegment>8</maxDaySegment>
      <!--required, supported daily time segment number -->
      <status opt="invalid,sleep,alwaysopen,alwaysclose,normal"/>
      <!--required, status value range -->
      <verifyType
opt="invalid,sleep,swipecard,swipecardandpassword,swipecardorpasswd,fingerPrint,fingerPrintAndPasswd,fingerPrintorCard,fingerPrintAndCard,fingerPrintAndCardAndPasswd,faceOrFpOrCardOrPw,faceAndFingerPrint,faceAndPassword,faceAndCard,face,employeeNoAndPassword,fingerPrintOrPassword,employeeNoAndFp,employeeNoAndFpAndPw,faceAndFpAndCard,faceAndPwAndFp,employeeNoAndface,cardOrFace,cardOrFaceOrFp,cardOrFpOrPw"/>
      <!--required, authentication method range -->
      <TimeAccuracy>
        <!--required, time accuracy -->
        <hour>enable</hour>
        <minute>enable</minute>
    <second>enable</second>
   </TimeAccuracy>
  </WeekPlan>
  <HolidayPlan>
   <!--required, holiday schedule -->
   <holidayPlanNo min="" max=""/>
   <!--required,  holiday schedule No. range -->
   <maxDaySegment>8</maxDaySegment>
   <!--required, supported daily time segment number -->
   <TimeAccuracy>
    <!--required, time accuracy-->
    <hour>enable</hour>
    <minute>enable</minute>
    <second>enable</second>
   </TimeAccuracy>
  </HolidayPlan>
  <HolidayGroup>
   <!--required, holiday group capability-->
   <holidayGroupNo min="" max=""/>
   <!--required, holiday group No. range -->
   <holidayGroupName min="" max=""/>
   <!--required, holiday group name length -->
   <maxHolidayPlanNum></maxHolidayPlanNum>
   <!--required, max. holiday schedule number for the holiday group -->
  </HolidayGroup>
  <PlanTemplate>
   <!--required, schedule template capability -->
   <templateNo min="" max=""/>
    <!--optional, range of schedule template No.-->
   <templateName min="" max=""/>
   <!--required, schedule template name length -->
   <maxHolidayGroupNum></maxHolidayGroupNum>
   <!--required, max. holiday group number for the schedule template -->
  </PlanTemplate>
  <supportLocalController>enable</supportLocalController>
  <!--required, support distributed access controller-->
 </DoorStatusPlan>
 <CardReaderVerifyTypePlan>
  <!--required, card reader authentication schedule capability -->
  <WeekPlan>
   <!--required, weekly schedule capability -->
   <weekPlanNo min="" max=""/>
   <!--required, weekly schedule No. range -->
   <maxDaySegment>8</maxDaySegment>
   <!--required, supported daily time segment number -->
   <status opt="invalid,sleep,alwaysopen,alwaysclose,normal"/>
   <!--required, status value range -->
   <verifyType
opt="invalid,sleep,swipecard,swipecardandpassword,swipecardorpasswd,fingerPrint,fingerPrintAndPasswd,fingerPrint
OrCard,fingerPrintAndCard,fingerPrintAndCardAndPasswd,fingerPrintorCard,fingerPrintAndCard,fingerPrintAndCardAn
dPasswd,faceOrFpOrCardOrPw,faceAndFingerPrint,
faceAndPassword,faceAndCard,face,employeeNoAndPassword,fingerPrintOrPassword,employeeNoAndFp,employeeN
```

```
oAndFpAndPw,faceAndFpAndCard,faceAndPwAndFp,employeeNoAndface,employeeNoAndFpAndPw,faceAndFpAndCa
rd,faceAndPwAndFp,employeeNoAndface,faceOrFaceAndCard,fingerPrintOrFace,swipecardOrFaceOrPw,cardOrFace,ca
rdOrFaceOrFp,cardOrFpOrPw"/>
    <!--required,verification mode range: invalid, sleep, card, card and password, card or password, fingerprint,
fingerprint and password, fingerprint or card, fingerprint and card, fingerprint and card and password (no order), face
or fingerprint or card or password, face and fingerprint, face and password, face and card, face, employee No. and
password, fingerprint or password, employee No.and fingerprint, employee No. and fingerprint and password, face
and fingerprint and card, face and password and fingerprint, employee No. and face, employee No. and fingerprint
and password, face and fingerprint and card, face and password and fingerprint, employee No. and face, face or face
and card, fingerprint or face, card or face or password, card or face, card or face or fingerprint-->
    <purePwdVerifyEnable><!--optional, boolean, whether the device supports opening the door only by password:
true-yes, this node is not returned-no--></purePwdVerifyEnable>
    <!--For opening the door only by password: 1. The password in "XXX or password" in the authentication mode
refers to the person's password (the value of the node password in JSON_UserInfo); 2. The device will not check the
duplication of the password, and the upper platform should ensure that the password is unique; 3. The password
cannot be added, deleted, edited, or searched for on the device locally-->
    <TimeAccuracy>
      <!--required, time accuracy -->
      <hour>enable</hour>
      <minute>enable</minute>
      <second>enable</second>
    </TimeAccuracy>
  </WeekPlan>
  <HolidayPlan>
    <!--required, holiday schedule -->
    <holidayPlanNo min="" max=""/>
    <!--required, holiday schedule No. range -->
    <maxDaySegment>8</maxDaySegment>
    <!--required, supported daily time segment number -->
    <TimeAccuracy>
      <!--required,  time accuracy -->
      <hour>enable</hour>
      <minute>enable</minute>
      <second>enable</second>
    </TimeAccuracy>
  </HolidayPlan>
  <HolidayGroup>
    <!--required, holiday group capability -->
    <holidayGroupNo min="" max=""/>
    <!--required, holiday group No. range -->
    <holidayGroupName min="" max=""/>
    <!--required, holiday group name length -->
    <maxHolidayPlanNum></maxHolidayPlanNum>
    <!--required, max. holiday schedule number for holiday group -->
  </HolidayGroup>
  <PlanTemplate>
    <!--required, schedule template capability -->
    <templateNo min="" max=""/>
    <!--optional, range of schedule template No.-->
    <templateName min="" max=""/>
    <!--required, schedule template name lenghth -->
    <maxHolidayGroupNum></maxHolidayGroupNum>
```

```
      <!--required, max. holiday group number for schedule template -->
    </PlanTemplate>
    <supportLocalController>enable</supportLocalController>
    <!--required, support distributed access controller-->
  </CardReaderVerifyTypePlan>
  <CardRightPlan>
    <!--required, card permission schedule capability -->
    <WeekPlan>
      <!--required, weekly schedule capability -->
      <weekPlanNo min="" max=""/>
      <!--required, weekly schedule No. range -->
      <maxDaySegment>8</maxDaySegment>
      <!--required, supported daily time segment number -->
      <status opt="invalid,sleep,alwaysopen,alwaysclose,normal"/>
      <!--required, status value range -->
      <verifyType opt="invalid,sleep,swipecard,swipecardandpassword"/>
      <!--required, authentication method range -->
      <TimeAccuracy>
        <!--required, time accuracy -->
        <hour>enable</hour>
        <minute>enable</minute>
        <second>enable</second>
      </TimeAccuracy>
    </WeekPlan>
    <HolidayPlan><!--required, holiday schedule -->
      <holidayPlanNo min="" max=""/><!--required, holiday schedule No. range -->
      <maxDaySegment>8</maxDaySegment>
      <!--required, supported daily time segment number -->
      <TimeAccuracy>
        <!--required, time accuracy -->
        <hour>enable</hour>
        <minute>enable</minute>
        <second>enable</second>
      </TimeAccuracy>
    </HolidayPlan>
    <HolidayGroup>
      <!--required, holiday group capability-->
      <holidayGroupNo min="" max=""/>
      <!--required, holiday group No. range -->
      <holidayGroupName min="" max=""/>
      <!--required, holiday group name length -->
      <maxHolidayPlanNum></maxHolidayPlanNum>
      <!--required, max. holiday schedule number for holiday group -->
    </HolidayGroup>
    <PlanTemplate>
      <!--required, schedule template capability -->
      <templateNo min="" max=""/>
        <!--optional, range of schedule template No.-->
      <templateName min="" max=""/>
      <!--required, schedule template name length -->
      <maxHolidayGroupNum></maxHolidayGroupNum>
      <!--required, max. holiday group number for schedule template -->
    </PlanTemplate>
    <supportLocalController>enable</supportLocalController>
    <!--required, support distributed access controller-->
   </CardRightPlan>
   <Door>
    <!--required, door parameters capaility -->
    <doorName min="" max=""/>
    <!--required, door name length -->
    <magneticMode opt="alwaysclose,alwaysopen"/>
    <!--required, door magnetic type -->
    <openButtonMode opt="alwaysclose,alwaysopen"/>
    <!--required, exit button type-->
    <openDuration min="" max=""/>
    <!--required, door opening duration range, unit: second -->
    <disabledOpenDuration min="" max=""/>
    <!--required, disabled card opening door duration range, unit: second)-->
    <magneticAlarmTimeout min="" max=""/>
    <!--required, magnetic detection overtime alarm time, unit: second,  0 indicates not to alarm. -->
    <doorLock>enable</doorLock>
    <!--required,whether support locking door when door closed. -->
    <leaderCard>enable</leaderCard>
    <!--required, whether to enable first card opening door -->
    <stressPassword min="" max=""/>
    <!--required, duress password length -->
    <superPassword min="" max=""/>
    <!--required, super password length -->
    <unlockPassword min="" max=""/>
    <!--optional, unlocking password length -->
    <leaderCardMode opt="close,alwaysopen,authorized"/>
    <!--required, first card mode-->
    <useLocalController>enable</useLocalController>
    <!--required, whether the door is connected to distributed access controller-->
    <localControllerID min="" max=""/>
    <!--required, distributed access controller No.-->
    <localControllerDoorNumber min="" max=""/>
    <!--required, distributed access controller door No.-->
    <localControllerStatus opt="offline,netOnline,authorized"/>
    <!--required, distributed access controller online status-->
    <lockInputCheck>enable</lockInputCheck>
    <!--required, whether to enable door lock input check (1 byte, 0- disbale, 1- enable, default to disable)-->
    <lockInputType opt="NormallyClose,NormallyOpen"/>
    <!--required, door lock input type (1 byte, 0- normally closed, 1- normally open, default to normally closed)-->
    <doorTerminalMode opt="PreventCutShort,Normal"/>
    <!--required, door related terminal operating mode (1 byte, 0- anti-cut & short-circuit, 1- normal, default to anti-cut
  & short-circuit)-->
    <openButton>enable</openButton>
    <!--required, whether to enable door button (1 byte, 0- yes, 1- no, default to yes)-->
   </Door>
   <DoorStatusPlan>
    <!--required, door status schedule parameters -->
    <enable>true</enable>
   </DoorStatusPlan>
   <Group>
    <!--required, group parameters capability -->
    <ValidCfg>
     <!--required, validate capability -->
     <TimeAccuracy>
      <!--required, time accuracy -->
      <year>enable</year>
      <month>enable</month>
      <day>enable</day>
      <hour>enable</hour>
      <minute>enable</minute>
      <second>enable</second>
     </TimeAccuracy>
     <timeType opt="local,UTC"/>
      <!--optional, time type: "local"-device local time (default), "UTC"-UTC time>
    </ValidCfg>
    <groupName min="" max=""/>
    <!--required, group name length -->
    <groupNo min="" max=""/><!--required, group No. range. If this node cannot be parsed or is not returned, it will be
  set to the default value-->
   </Group>
   <MultiCard>
    <!--required, multi-card capability -->
    <swipeIntervalTimeout min="" max=""/>
    <!--required, multi-card swiping interval overtime, unit: second -->
    <maxMultiCardGroupNum, min="1", max="20"></maxMultiCardGroupNum>
    <!--required, max. multi-card group number >
    <maxGroupCombinationNum></maxGroupCombinationNum>
    <!--required, max. group number for a multi-card group -->
    <remoteOpenDoor>enable</remoteOpenDoor>
    <!--required, supports remote door opening authentication method -->
    <offlineVerifyMode>enable</offlineVerifyMode>
    <!--required, supported offline control panel authentication mode (super password replaces remote door opening
  control) -->
   </MultiCard>
   <Card>
    <!--required, card parameters capability -->
    <cardNo min="" max=""/>
    <!--required, card No. length -->
    <modifyParamType opt="cardvalid,validtime,cardtype,doorright,leadercard,swipenum,group,password, rightplan,
  swipednum, employeeno, name, departmentNo, schedulePlanNo,
  schedulePlanType,roomNo,simNo,floorNo,userType"/>
    <!--required,edit separately --> opt="cardvalid- card valid or not, validtime- expiry date, cardtype- card type,
  doorright- door permission,
    leadercard- first card, swipenum- max. card swiping times, group- group, password- card password,,rightplan- card
  permission schedule,
    swipednum- card swiped times,  employeeno- employee No., name-Name, departmentNo-Apartment No.,
  schedulePlanNo-Schedule No., schedulePlanType-Schedule Type-->
    <cardValid>enable</cardValid>
    <timeRangeBegin>
     <!--optional, start time that can be configured by beginTime and endTime. If this node is not returned by the
  capability, the start time that can be configured is 1970-01-01T00:00:00 by default-->
```

```
   </timeRangeBegin>
   <timeRangeEnd>
     <!--optional, end time that can be configured by beginTime and endTime. If this node is not returned by the
capability, the end time that can be configured is 2037-12-31T23:59:59 by default-->
   </timeRangeEnd>
   <cardValidUnit opt="day,hour,minute,second">
     <!--required, accuracy of card expiry date (if device supports correcting to minute, opt="minute"), if this node is
not returned, the default accuracy is day (opt="day")>
   </cardValidUnit>
   <!--required, whether the card is valid-->
   <!--required, card type-->
   <cardType opt="normalcard,disabledcard,blacklistcard,nightwatchcard,
stresscard,supercard,guestcard,mastercard,staffcard,normalopencard,cleancard,standbycard, unlockcard"/>
   <doorRight>enable</doorRight>
   <!--required, door permission-->
   <leaderCard>enable</leaderCard>
   <!--required, whether to enable the first card? -->
   <swipeNum min="" max=""/>
   <!--required, max. card swiping number, o indicates no limit-->
   <maxBelongGroup></maxBelongGroup>
   <!--required, max. group number belonged to -->
   <cardPassword min="" max=""/>
   <!--required, card password-->
   <doorRightPlanNum></doorRightPlanNum>
   <!--required, max. schedule template number for a single door -->
   <swipeTime>enabled</swipeTime>
   <!--required, swiping times -->
   <onlyPasswdOpen opt="true,false"/>
   <!--optional, whether to support password opening door, invalid currently -->
    <roomNumber min="" max=""/>
    <!--optional, Room No.-->
    <floorNumber min="" max=""/>
    <!--optional,Floor number-->
    <employeeNo min="" max=""/>
    <!--optional, employee No.-->
   <name min="" max=""></name>
   <!--required, name (if device returns this node, you can get and set the linked user name of the card by calling card
parameter API directly, so there is no need to API NET_DVR_SET_CARD_USERINFO_CFG and
NET_DVR_GET_CARD_USERINFO_CFG)-->
   <departmentNo min="" max=""/>
   <!--optional, department No.-->
   <schedulePlanNo min="" max=""/>
   <!--optional, shift schedule-->
   <schedulePlanType opt="personal,department"/>
   <!--optional, shift schedule type-->
   <lockID min="" max=""/>
   <!--required, lock ID-->
   <roomCode min="" max=""/>
   <!--required, room code-->
   <cardRight opt="lowPowerAlarm,openDoorSound,customCardLimit,normalOpen,openLockedDoor,keepWatch"/>
   <!--required, card permission-->
   <supportLocalController>enable</supportLocalController>
```

```
    <!--required, support distributed access controller-->
    <roomNumber min="" max=""></roomNumber><!--required, room No.>
    <floorNumber min="" max=""></floorNumber><!--required, floor No.>
    <SIMNum min="" max=""></SIMNum><!--required, mobile phone number>
    <isSupportCardModify>true</isSupportCardModify>
      <!--required, support downloading when card parameters changed (for video intercom device only, by default, this function is supported by all access control devices)>
  </Card>
  <AntiSneak>
    <!--required, anti-passback capability-->
    <startCardReaderNo>enable</startCardReaderNo>
    <!--required, anti-passback card reader No. configuration -->
    <maxSneakPath></maxSneakPath>
    <!--required, max. anti-passback follow-up card reader number-->
  </AntiSneak>
  <MultiDoorInterlock>
    <!--required, multi-door interlocking parameters -->
    <maxMultiDoorInterlockGroup></maxMultiDoorInterlockGroup>
    <!--required, max. multi-door interlocking group number -->
    <maxInterlockDoorNum></maxInterlockDoorNum>
    <!--required, max. interlocked door number for one multi-door interlocking group -->
  </MultiDoorInterlock>
  <AcsWorkStatus>
    <!--required, access controller working status parameters -->
    <doorLogicalStatus>enable</doorLogicalStatus>
    <!--required, door logic status -->
    <doorStatus opt="alwaysopen,alwaysclose,normal"/>
    <!--required, door status parameters -->
    <magneticStatus>enable</magneticStatus>
    <!--required, door magnetic status parameters -->
    <relayStatus>enable</relayStatus>
    <!--required, relay status-->
    <caseSensorStatus>enable</caseSensorStatus>
    <!--required, case trigger status-->
    <BatteryVoltage>enable</BatteryVoltage>
    <!--required, battery voltage value -->
    <BatteryLowVoltage>enable</BatteryLowVoltage>
    <!--required, battery low voltage detection -->
    <PowerSupplyStatus>enable</PowerSupplyStatus>
    <!--required, device power supply status-->
    <multiDoorInterlockStatus>enable</multiDoorInterlockStatus>
    <!--required, multi-door interlocking status parameters-->
    <antiSneakStatus>enable</antiSneakStatus>
    <!--required, anti-passback status parameters-->
    <hostAntiDismantleStatus>enable</hostAntiDismantleStatus>
    <!--required, control ler tamper ?proof status-->
<indicatorLightStatus>enable</indicatorLightStatus>
    <!--required, Supports indicator status-->
    <cardReaderOnlineStatus>enable</cardReaderOnlineStatus>
    <!--required, card reader connection status -->
    <cardReaderAntiDismantleStatus>enable</cardReaderAntiDismantleStatus>
    <!--required, card reader tamper-proof status -->
    <cardReaderVerifyMode opt="invalid,sleep,swipecard,swipecardandpassword, swipecardorpasswd,
fingerPrint,fingerPrintAndPasswd,fingerPrintor
Card,fingerPrintAndCard,fingerPrintAndCardAndPasswd,faceOrFpOrCardOrPw,
faceAndFingerPrint,faceAndPassword,faceAndCard,face,employeeNoAndPassword,fingerPrintOrPassword,employeeN
oAndFp,
employeeNoAndFpAndPw,faceAndFpAndCard,faceAndPwAndFp,employeeNoAndface,faceOrfaceAndCard,fingerPrintO
rFace,swipecardOrFaceOrPw,"/>
    <!--required, supported card reader authentication modes: 0-invalid, 1-card, 2-card+password, 3-card, 4-card/
password, 5-fingerprint, 6-fingerprint+password, 7-fingerprint/card, 8-fingerprint_card, 9-fingerprint_card+password,
10-face/fingerprint/card/password, 11-face+fingerprint, 12-face+password, 13-face+card, 14-face, 15-employee No.
+password, 16-fingerprint/password, 17-employee No.+fingerprint, 18-employee No.+fingerprint+password, 19-face
+fingerprint+card, 20-face+password+fingerprint, 21-employee No.+face, 22-face/face+card, 23-fingerprint/face, 24-
card/face/password-->
    <setupAlarmStatus>enable</setupAlarmStatus>
    <!--required, zone arming status -->
    <alarmInStatus>enable</alarmInStatus>
    <!--required, alarm input status -->
    <alarmOutStatus>enable</alarmOutStatus>
    <!--required, alarm output status -->
    <cardNum>enable</cardNum>
    <!--required, added card number -->
<fireAlarmStatus opt="normal,shortCircuit,break"/>
    <!--required, support fire alarm status-->
    <supportLocalController>enable</supportLocalController>
    <!--required, support distributed access controller-->
    <batteryChargeStatus opt="InCharge,NotCharge"/>
    <!--required, battery status: InCharge-Charging, NotCharge-Uncharged>
    <masterChannelControllerStatus>enable</masterChannelControllerStatus>
    <!--required, supports online status of main lane controller-->
    <slaveChannelControllerStatus>enable</slaveChannelControllerStatus>
    <!--required, supports online status of sub-lane controller-->
    <antiSneakServerStatus opt="disable,normal,disconnect"/>
     <!--optional, anti-passing back server status: "disable"-disabled, "normal"-normal, "disconnect"-disconnected-->
    <whiteFaceNum>enable</whitefaceNum>
    <!--required, supports the parameters of face picture quantity in allowlist-->
    <blackFaceNum>enable</blackfaceNum>
    <!--required, supports the parameters of face picture quantity in blocklist-->
  </AcsWorkStatus>
  <CaseSensor>
    <!--required, event trigger parameters capability -->
    <triggerHostBuzzer>enable</triggerHostBuzzer>
    <!--required, trigger controller buzzer -->
    <triggerCardReaderBuzzer>enable</triggerCardReaderBuzzer>
    <!--required, trigger card reader buzzer -->
    <triggerAlarmOut>enable</triggerAlarmOut>
    <!--required, trigger alarm output -->
<triggerDoorOpen>enable</triggerDoorOpen>
    <!--required, support triggered open door by ID-->
    <triggerAlarmOutClose>enable</triggerAlarmOutClose>
    <!--required, support disable triggered alarm input-->
    <triggerAlarmInSetup>enable</triggerAlarmInSetup>
    <!--required, support triggered arming region arming-->
    <triggerAlarmInClose>enable</triggerAlarmInClose>
    <!--required, support triggered arming region disarming-->
  </CaseSensor>
  <CardReaderCfg>
    <!--required, reader parameters capability-->
    <!--required, supported reader type-->
    <cardReaderType opt="DS-K110XM/MK/C/CK,DS-K192AM/AMP,DS-K192BM/BMP,DS-K182AM/AMP,DS-K182BM/
BMP,DS-K182AMF/ACF,
                Wiegand or RS485 offline,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, DS-K1T200EF/EF-C/MF-MF-C/CF/CF-C,
                DS-K1T300EF/EF-C/MF-MF-C/CF/CF-C"/>
    <okLedPolarity op="cathode,anode"/>
    <!--required,OK LED polarity-->
    <errorLedPolarity op="cathode,anode"/>
    <!--required,ERROR LED polarity-->
    <buzzerLedPolarity op="cathode,anode"/>
    <!--required, buzzer polarity -->
    <swipeInterval min="" max=""/>
    <!--required, time interval of duplicated authentication, unit: second -->
    <pressTimeout min="" max=""/>
    <!--required, key pressing overtime, unit: second -->
    <enableFailAlarm>enable</enableFailAlarm>
    <!--required, whether to enable authentication failure over times alarm configuration-->
    <maxReadCardFailNum min="" max=""/>
    <!--required, max. times of authentication failure -->
    <enableTamperCheck>enable</enableTamperCheck>
    <!--optional, whether to support anti-tamper check-->
    <offlineCheckTime min="" max=""/>
    <!--optional, offline check time, unit:s-->
    <fingerPrintCheckLevel
opt="1/10,1/100,1/1000,1/10000,1/100000,1/1000000,1/10000000,1/100000000,3/100,3/1000,
                      3/10000,3/100000,3/1000000,3/10000000,3/100000000,Automatic Normal,Automatic
Secure,Automatic More Secure"/>
    <!--optional, fingerprint recognition level-->
    <useLocalController>enable</useLocalController>
    <!--required, whether door is connected to distributed access controller-->
    <localControllerID min="" max=""/>
    <!--optional, distributed access controller No.-->
    <localControllerReaderID min="" max=""/>
    <!--optional, ID of distributed access controller card reader-->
    <cardReaderChannel opt="Wiegand/Offline,RS485A,RS485B"/>
    <!--opt  card reader communication channel No.-->
    <fingerPrintImageQuality min="1" max="8"/>
    <!--optional,fingerprint picture quality-->
    <fingerPrintContrastTimeOut min="0" max="20"/>
    <!--optional,fingerprint comparison overtime, 0 - infinite, that is 0xff-->
    <fingerPrintRecogizeInterval min="0" max="10"/>
    <!--optional,time interval of fingerprint continuous recognition, 0- no delay, that is 0xff-->
    <fingerPrintMatchFastMode min="0" max="5"/>
    <!--optional,fingerprint fast matching mode, 0- auto, that is 0xff-->
    <fingerPrintModuleSensitive min="1" max="8"/>
    <!--optional,fingerprint module sensitivity-->
    <fingerPrintModuleLightCondition opt="outdoor,indoor"/>
    <!--optional,light condition of fingerprint module-->
    <faceMatchThresholdN min="0" max="100"/>
    <!--optional,face 1:N matching threshold-->
    <faceQuality min="0" max="100"/>
    <!--optional,face picture quality-->
    <faceRecogizeTimeOut min="0" max="20"/>
    <!--optional,face recognition overtime, 0 - infinite, that is 0xff-->
    <faceRecogizeInterval min="0" max="10"/>
    <!--optional,time interval of face continuous recognition, 0- no delay, that is 0xff-->
    <cardReaderFunction opt="fingerPrint,face,fingerVein"/>
    <!--optional,card reader types-->
    <cardReaderDescription min="1" max="32"/>
    <!--optional,card reader description-->
    <faceImageSensitometryl min="0" max="65535"/>
    <!--optional, face picture exposure-->
    <livingBodyDetect opt="disable,enable"/>
    <!--optional, face detection-->
    <faceMatchThreshold1 min="0" max="100"/>
    <!--optional,Face 1:1 matching threshold-->
    <buzzerTime  min="0" max="5999"/>
    <!--optional, buzzing time-->
    <faceMatch1SecurityLevel opt="normal, more secure, extremely secure"/>
    <!--optional, face picture 1:1 security level: 1-normal, 2-high, 3-higher-->
    <faceMatchNSecurityLevel opt="0,1,2"/>
    <!--optional, face picture 1:N security level: 1-normal, 2-high, 3-higher-->
    <envirMode opt="normal, more secure, extremely secure"/>
    <!--optional, face recognition environment mode: 0-invalid, 1-indoor, 2-other -->
    <liveDetLevelSet opt="0,1,2,3"/>
    <!--optional, set live face detection threshold level: 0-invalid, 1-low, 2-medium, 3-high-->
    <liveDetAntiAttackCntLimit min="0"max="255"/>
    <!--optional, max. live face detection failed attempts-->
    <enableLiveDetAntiAttack opt="0,1,2"/>
    <!--optional, whether enable locking face: 0-invalid, 1-disabled, 2-enabled-->
    <fingerPrintCapacity min="" max=""/>
    <!--ro, optional, xs:integer, fingerprint capacity-->
    <fingerPrintNum min="" max=""/>
    <!--ro, optional, xs:integer, the number of existed fingerprints-->
    <enableFingerPrintNum opt="true"/>
    <!--ro, optional, xs:boolean, enable fingerprint capacity or not (when it is "true", fingerPrintCapacity and
fingerPrintNum are valid)-->
    <envirMode opt="0,1,2"></envirMode>
    <!--optional, environment mode of face recognition, 0-invalid, 1-indoor, 2-other->
    <liveDetLevelSet opt="0,1,2,3"></liveDetLevelSet>
    <!--optional, set live face detection security level, 0-invalid, 1-normal, 2-high, 3-higher-->
    <liveDetAntiAttackCntLimit min="0"max="255">/liveDetAntiAttackCntLimit>
    <!--optional, maximum failed attempts-->
    <enableLiveDetAntiAttack opt="0,1,2">
```

```
    <!--optional, enable locking face, 0-invalid, 1-disable, 2-enable-->
   </enableLiveDetAntiAttack>
   <faceContrastMotionDetLevel opt="low,middle,high"/><!--optional, motion detection level during face picture comparison: low, middle, high-->
   <dayFaceMatchThresholdN min="0" max="100"/><!--optional, 1:N face picture comparison threshold in day-->
   <nightFaceMatchThresholdN min="0" max="100"/><!--optional, 1:N face picture comparison threshold at night-->
   <faceRecogizeEnable opt="true,false,multi"/><!-optional, whether to enable facial recognition: "true"-yes (one face),"false"-no, "multi"-yes (multiple faces)-->
   <supportDelFPByID opt="true"/>
    <!--ro, optional, xs:boolean, whether the fingerprint module supports deleting fingerprint by fingerprint ID: "true"-yes, "false"-no-->
   <defaultVerifyMode opt="1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27"/>
   <!--ro, optional, card reader authentication mode (factory default): 1-sleeping mode, 2-card swiping + password, 3-card swiping, 4-card swiping or password, 5-fingerprint, 6-fingerprint + password, 7-fingerprint or card swiping, 8-fingerprint + card swiping, 9-fingerprint + card swiping + password, 10-face or fingerprint or card swiping or password, 11-face + fingerprint, 12-face + password, 13-face + card swiping, 14-face, 15-employee ID + password, 16-fingerprint or password, 17-employee ID + fingerprint, 18-employee ID + fingerprint + password, 19-face + fingerprint + card swiping, 20-face + password + fingerprint, 21-employee ID + face, 22-face or face + card swiping, 23-fingerprint or face, 24-card swiping or face or password, 25-card or face, 26-card or face or fingerprint, 27-card or fingerprint or password-->
   <fingerPrintCapacity min="" max=""/><!--ro, optional, xs: integer, fingerprint capacity, this node is valid only when enableFingerPrintNum is "true"-->
   <fingerPrintNum min="" max=""/><!--ro, optional, xs: integer, number of exiting fingerprints, this node is valid only when enableFingerPrintNum is "true"-->
   <enableFingerPrintNum opt="true"/><!--ro, optional, xs: boolean, whether to enable fingerprint capacity-->
   <blackFaceMatchThreshold min="0" max="100"/><!--optional, face picture comparison threshold in blocklist-->
  </CardReaderCfg>
  <AcsUpgrade><!--required, upgrade capability of access control device-->
   <hostUpgrade>
    <!--required, whether to support upgrading main module-->
   </hostUpgrade>
   <cardReaderUpgrade>
    <!--required, whether to support upgrading card reader-->
   </cardReaderUpgrade>
   <localControllerUpgrade>
    <!--required, whether to support upgrading distributed access controller-->
   </localControllerUpgrade>
   <channelControllerUpgrade>
    <!--required,whether to support upgrading lane access controller-- >
   </channelControllerUpgrade>
   <extensionModuleUpgrade>
    <!--required, whether to support upgrading extension module-->
   </extensionModuleUpgrade>
   <smartLockUpgrade>
    <!--required, whether to support upgrading smart lock-->
   </smartLockUpgrade>
   <cardReaderFPAlgorithmUpgrade>
    <!--required, whether to support upgrading fingerprint algorithm program of fingerprint module-->
   </cardReaderFPAlgorithmUpgrade>
   <outdoorModules>
    <!--optional, whether to support upgrading the modules of door station, if not support, this node will not be returned-->
    </outdoorModules>
    <modules opt="keybord,display,button,card,signal"/>
    <!--opt， supported module type, “keybord”-keypad module, “display”-display module,”button”-nametag module.
“card”-card reader, “signal”-indicator module, if not support, this node will not be returned-->
  </AcsUpgrade>
  <clearAcsParam
opt="doorstatusweekplan,cardreaderverifyweek,cardrightweekplan,doorstatusholidayplan,cardreaderverifyholidaypla
n,cardrightholidayplan,doorstatusholidayplan,doorstatusholidaygroup,cardreaderverifyholidaygroup,cardrightplantem
plate,doorstatusplantemolate,cardreaderverifyplantemplate,card,group,antisneak,eventandCardLinkage,cardPasswdO
pendoor,personStatistics, blackListPicture, IDBlackList"/>
  <!--required, supported parameters clearing option -->
  <ACSClearParam>
    <!--required, extend clear access control host parameter node-->
    <localControllerID min="" max=""/>
    <!--optional, distributed access controller No.-->
  </ACSClearParam>
  <MultiHostAntiSneak>
    <!--required, over-controllers anti-passback -->
    <startAntiSnealHost opt="true,false"/>
    <!--optional, whether to enable anti-passback controller -->
    <antiSnealHostNum min="" max=""/>
    <!--required, controller number for anti-passback controller group -->
    <ReadersCfg>
      <!--required, over-controllers anti-passback card reader parameters -->
      <maxRouteGroupNum></maxRouteGroupNum>
      <!--required, max. path number -->
      <oneRouteReadersNum min="" max=""/>
      <!--required, follow-up card reader number for each path -->
    </ReadersCfg>
  </MultiHostAntiSneak>

  <AcsHostCfg>
    <!--optional, access control settings capability -->
    <enableRS485Backup opt="true,false"/>
    <!--required, whether to support downstream RS485 communication backup -->
      <showCapPic opt="true,false"/>
      <!--optional,whether to support displaying captured picture on LCD screen-->
      <showCardNo opt="true,false"/>
      <!--optional,whether to support displaying card No. on LCD screen-->
      <showUserInfo opt="true,false"/>
      <!--optional,whether to support displaying user information on LCD screen-->
      <overlayUserInfo opt="true,false"/>
      <!--optional,Whether to overlay user information on the captured picture-->
      <voicePrompt opt="true,false"/>
      <!--optional,Whether to support sound prompt-->
      <uploadCapPic opt="true,false"/>
      <!--optional, Whether to support uploading picture after capturing-->
      <saveCapPic opt="true,false"/>
      <!--optional,Whether to support saving captured picture-->
    <inputCardNo opt="true,false"/>
    <!--optional, whether supports inputting card No. by button-->
    <wifiDetect opt="true,false"/>
    <!--optional, whether supports enabling Wi-Fi probe-->
    <enable3G4G opt="true,false"/>
    <!--optional, enable 3G/4G-->
    <protocol opt="Private,OSDP"/>
    <!--optional, card reader communication protocol type: "Private"-private protocol (default), "OSDP"-OSDP protocol--
>
  </AcsHostCfg>

  <EventLinkage>
    <!--required, event card linkage-->
    <maxEventNum></maxEventNum>
    <!--required, max. event linkage number supported by the device -->
    <supportMode opt="Event,CardNo,MAC,EmployeeNo"/>
    <!--required, supported linkage method, "Event"-event linkage, "CardNo"-Card No. linkage, "MAC"-MAC address
linkage, "EmployeeNo"-Employee No. (person ID)-->
    <isSupportRecordVideo opt="true,false"/>
    <!--required, whether supports recording linkage-->
    <supportLocalController>enable</supportLocalController>
    <!--required, support distributed access controller-->
    <isSupportAlarmOutClose opt="true,false"/>
    <!--required, whether supports disabling linked alarm output-->
    <isSupportAlarmInSetup opt="true,false"/>
    <!--required, whether supports arming linked zone-->
    <isSupportAlarmInClose opt="true,false"/>
    <!--required, whether supports disarming linked zone-->
    <isSupportMainDevStopBuzzer opt="true,false"/>
    <!--required, whether supports stopping buzzing by access controller-->
    <isSupportReaderStopBuzzer opt="true,false"/>
    <!--required, whether supports stopping buzzing by linked card reader-->
    <audioDisplayMode opt="Close,SinglePlay,CyclePlay"/>
    <!--required, linked audio prompt mode: "Close"-disable, "SinglePlay"-play once, "CyclePlay"-loop playing-->
    <audioDisplayID min="1" max="32"/>
    <!--required, linked audio prompt mode: "Close"-disable, "SinglePlay"-play once, "CyclePlay"-loop playing-->
    <isNotSupportOpenDoor>
      <!--optional, whether the opening door linkage is not supported-->
    </isNotSupportOpenDoor>
    <isNotSupportCloseDoor>
      <!--optional, whether the closing door linkage is not supported-->
    </isNotSupportCloseDoor>
    <isNotSupportNormalOpen>
      <!--optional, whether the remaining door open is not supported-->
    </isNotSupportNormalOpen>
    <isNotSupportNormalClose>
      <!--optional, whether the remaining door closed is not supported-->
    </isNotSupportNormalClose>
    <isNotSupportAlarmout>
      <!--optional, whether the alarm output linkage is not supported-->
    </isNotSupportAlarmout>
    <isNotSupportCapturePic>
      <!--optional, whether the capture linkage is not supported-->
    </isNotSupportCapturePic>
    <isNotSupportMainDevBuzzer>
```

```
   <!--optional, whether not supports buzzing linkage of access controller, if supports, this node will not return-->
  </isNotSupportMainDevBuzzer>
  <isNotSupportReaderBuzzer>
   <!--optional, whether not supports buzzing linkage of card reader, if supports, this node will not returned-->
  </isNotSupportReaderBuzzer>
  <purePwdVerifyEnable><!--optional, boolean, whether the device supports opening the door only by password:
true-yes, this node is not returned-no--></purePwdVerifyEnable>
  <!--For opening the door only by password: 1. The password in "XXX or password" in the authentication mode refers
to the person's password (the value of the node password in JSON_UserInfo); 2. The device will not check the
duplication of the password, and the upper platform should ensure that the password is unique; 3. The password
cannot be added, deleted, edited, or searched for on the device locally-->
  <EventList>
   <EventEntry>
    <Index>0</Index>
    <mainEventName>DevEvent</mainEventName>
    <SubEventNameList>
     <subEventName>hostAntiDismantle</subEventName>
     <!--required, controller tampering alarm -->
     <subEventName>OfflineEcentNearlyFull</subEventName>
     <!--required, alarm for offline event exceeding 90% -->
     <subEventName>NetBroken</subEventName>
     <!--required, network disconnected -->
     <subEventName>NetRume</subEventName>
     <!--required,  network recovery -->
     <subEventName>LowBattery</subEventName>
     <!--required, battery low voltage -->
     <subEventName>BatteryReume</subEventName>
     <!--required, battery voltage recovered -->
     <subEventName>ACOff</subEventName>
     <!--required, AC power off -->
     <subEventName>ACResume</subEventName>
     <!--required, AC power recovery-->
     <subEventName>SDCardFull</subEventName>
     <!--required,SD card full alarm-->
     <subEventName>LinkageCapturePic</subEventName>
     <!--required,Linked capture event alarm-->
     <subEventName>ImageQualityLow</subEventName>
     <!--required,low face picture quality-->
     <subEventName>FingerPrintQualityLow</subEventName>
     <!--required,low fingerprint picture quality-->
     <subEventName>BatteryElectricLow</subEventName>
     <!--required,low battery voltage (for face device only)-->
     <subEventName>BatteryElectricResume</subEventName>
     <!--required,battery voltage recovery (for face device only)-->
     <subEventName>FireImportShortCircuit</subEventName>
     <!--req fire input short-circuit alarm-->
     <subEventName>FireImportBrokenCircuit</subEventName>
     <!--req fire input broken-circuit alarm-->
     <subEventName>FireImportResume</subEventName>
     <!--req fire input recovery-->
     <subEventName>MasterRS485LoopnodeBroken</subEventName>
     <!--req main controller RS485 loop node disconnection-->
```

```
          <subEventName>MasterRS485LoopnodeResume</subEventName>
          <!--req main controller RS485 loop node connection recovery-->
          <subEventName>DistractControllerOnLine</subEventName>
          <!--required,Distributed controller online-->
          <subEventName>DistractControllerOffLine</subEventName>
          <!--required,Distributed controller offline-->
          <subEventName>FireButtonTrigger</subEventName>
          <!--required,Fire button triggered-->
          <subEventName>FireButtonResume</subEventName>
          <!--required,Fire button recovered-->
          <subEventName>MaintenanceButtonTrigger</subEventName>
          <!--required,Maintenance button triggered-->
          <subEventName>MaintenanceButtonResume</subEventName>
          <!--required,Maintenance button recovered-->
          <subEventName>EmergencyButtonTrigger</subEventName>
          <!--required,Emergency button triggered-->
          <subEventName>EmergencyButtonResume</subEventName>
          <!--required,Emergency button recovered-->
          <subEventName>LocalControlOffline</subEventName>
          <!--req distributed access controller offline-->
          <subEventName>LocalControlResume</subEventName>
          <!--required, distributed access controller connection recovered-->
          <subEventName>LocalDownsideRS485LoopNodeBroken</subEventName>
          <!--required, distributed access controller downlink RS485 loop disconnection-->
          <subEventName>LocalDownsideRS485LoopNodeResume</subEventName>
          <!--required, distributed access controller downlink RS485 loop connection recovered-->
          <subEventName>SubmarinebackCommBreak</subEventName>
          <!--required, disconnected with anti-passing back server-->
          <subEventName>SubmarinebackCommResume</subEventName>
          <!--required, resume connection with anti-passing back server-->
          <subEventName>RemoteActualGuard</subEventName>
          <!--required, remote real-time arming-->
          <subEventName>RemoteActualUnguard</subEventName>
          <!--required, remote real-time disarming-->
          <subEventName>MotorSensorException</subEventName>
          <!--required, motor or sensor exception-->
          <subEventName>CanBusException</subEventName>
          <!--required, CAN bus exception-->
          <subEventName>CanBusResume</subEventName>
          <!--required, CAN bus restored-->
          <subEventName>GateTemperatureOverrun</subEventName>
          <!--required, too high pedestal temperature-->
          <subEventName>IREmitterException</subEventName>
          <!--required, active infrared intrusion detector exception-->
          <subEventName>IREmitterResume</subEventName>
          <!--required, active infrared intrusion detector restorted-->
          <subEventName>LampBoardCommException</subEventName>
          <!--required, communication with light board failed-->
          <subEventName>LampBoardCommResume</subEventName>
          <!--required, communication with light board restored-->
          <subEventName>IRAdaptorBoardCommException</subEventName>
          <!--required, communicated with IR adaptor exception-->
```

```
    <subEventName>IRAdaptorBoardCommResume</subEventName>
    <!--required, communication with IR adaptor restored-->
    <subEventName>ChannelControllerDesmantleAlarm</subEventName>
    <!--required, lane controller tampering alarm-->
    <subEventName>ChannelControllerDesmantleResume</subEventName>
    <!--required, lane controller tampering alarm restored-->
    <subEventName>ChannelControllerFireImportAlarm</subEventName>
    <!--required, lane controller fire input alarm-->
    <subEventName>ChannelControllerFireImportResume</subEventName>
    <!--required, lane controller fire input alarm restored-->
    <subEventName>StayEvent</subEventName>
    <!--optional, loitering event-->
    <subEventName>LegalEventNearlyFull</subEventName>
    <!--optional, alarm of no memory for legal offline event storage-->
   </SubEventNameList>
  </EventEntry>
  <EventEntry>
   <Index>1</Index>
   <mainEventName>AlarmEvent</mainEventName>
   <SubEventNameList>
    <subEventName>AlarminShortCircuit</subEventName>
    <!--required,  zone short circuit alarm-->
    <subEventName>AlarminBrokenCircuit</subEventName>
    <!--required, zone open circuit alarm -->
    <subEventName>AlarminException</subEventName>
    <!--required, zone exception alarm -->
    <subEventName>AlarmResume</subEventName>
    <!--required, zone alarm recovery -->
    <subEventName>CaseSensorAlarm</subEventName>
    <!--required, event input alarm -->
    <subEventName>CaseSensorResume</subEventName>
    <!--required, event input recovery -->
   </SubEventNameList>
  </EventEntry>
  <EventEntry>
   <Index>2</Index>
   <mainEventName>DoorEvent</mainEventName>
   <SubEventNameList>
    <subEventName>LeaderCardOpenBegin</subEventName>
    <!--required, first card opening door starts -->
    <subEventName>LeaderCardOpenStop</subEventName>
    <!--required, first card open status door ends -->
    <subEventName>AlwaysOpenBegin</subEventName>
    <!--required, remained open status starts -->
    <subEventName>AlwaysOpenStop</subEventName>
    <!--required, remained open status ends -->
    <subEventName>AlwaysCloseBegin</subEventName>
    <!--required, remained closed status ends -->
    <subEventName>AlwaysCloseStop</subEventName>
    <!--required, remaining closed status ends-->
    <subEventName>LockOpen</subEventName>
    <!--required, open the door lock -->
```

```
<subEventName>LockClose</subEventName>
<!--required, close the lock -->
<subEventName>DoorButtonPress</subEventName>
<!--required, exit button pressed -->
<subEventName>DoorButtonRelease</subEventName>
<!--required, exit button released -->
<subEventName>DoorOpenNormal</subEventName>
<!--required, normally open the door (door magnetic) -->
<subEventName>DoorCloseNormal</subEventName>
<!--required, normally close the door (door magnetic) -->
<subEventName>DoorOpenAbnormal</subEventName>
<!--required, door opening exception (door magnetic )-->
<subEventName>DoorOpenTimeout</subEventName>
<!--required, door opening timeout (door magnetic )-->
<subEventName>RemoteOpenDoor</subEventName>
<!--required, remotely open the door-->
<subEventName>RemoteCloseDoor</subEventName>
<!--required, remotely closed the door-->
<subEventName>RemoteAlwaysOpen</subEventName>
<!--required, remotely remain open -->
<subEventName>RemoteAlwaysClose</subEventName>
<!--required, remotely remain closed -->
<subEventName>NotBelongMultiCard</subEventName>
<!--required, the card is not associated to the multi-authentication group-->
<subEventName>InvalidMultiVerifyPeriod</subEventName>
<!--required, the card is not in the multi-authentication time period -->
<subEventName>MultiVerifySuperRightFail</subEventName>
<!--required, super password authentication failed -->
<subEventName>MultiVerifyRemoteRightFail</subEventName>
<!--required, remote authentication failed -->
<subEventName>MultiVerifySuccess</subEventName>
<!--required, successfully multi -authentication -->
<subEventName>MultiVerifyNeedRemoteOpen</subEventName>
<!--required, multi-authentication needs remote opening door -->
<subEventName>MultiVerifySuperRightSuccess</subEventName>
<!--required, successfully super password -->
<subEventName>MultiVerifyRepeatFail</subEventName>
<!--required, repeat authentication failed -->
<subEventName>MultiVerifyTimeout</subEventName>
<!--required, multi-authentication timeout -->
<subEventName>RemoteCapturePic</subEventName>
<!--required,remote capture-->
<subEventName>DoorBellRing</subEventName>
<!--required,door bell ringing-->
<subEventName>CallCenter</subEventName>
<!--required, call center-->
<subEventName>FirstCardAuthorizeBegin</subEventName>
<!--required,first card authorization started-->
<subEventName>FirstCardAuthorizeEnd</subEventName>
<!--required,first card authorization ended-->
<subEventName>FirstCardOpenWithoutAuthorize</subEventName>
<!--required,open door with unauthorized first card failed.-->
```

```
<subEventName>SecurityMoudleDesmantleAlarm</subEventName>
    <!--required, door control security module anti-tamper alarm-->
    <subEventName>FirstCardAuthorizeBegin</subEventName>
    <!--req first card authorization start-->
    <subEventName>FirstCardAuthorizeEnd</subEventName>
    <!--req first card authorization end-->
    <subEventName>DoorLockInputShortCircuit</subEventName>
    <!--req door lock input short-circuit alarm-->
    <subEventName>DoorLockInputBrokenCircuit</subEventName>
    <!--req door lock input broken-circuit alarm-->
    <subEventName>DoorLockInputException</subEventName>
    <!--req door lock input exception alarm-->
    <subEventName>DoorContactInputShortCircuit</subEventName>
    <!--req magnet input short-circuit alarm-->
    <subEventName>DoorContactInputBrokenCircuit</subEventName>
    <!--req magnet input broken-circuit alarm-->
    <subEventName>DoorContactInputException</subEventName>
    <!--req magnet input exception alarm-->
    <subEventName>OpenButtonInputShortCircuit</subEventName>
    <!--req door button input short-circuit alarm-->
    <subEventName>OpenButtonInputBrokenCircuit</subEventName>
    <!--req door button input broken-circuit alarm-->
    <subEventName>OpenButtonInputException</subEventName>
    <!--req door button input exception alarm-->
    <subEventName>DoorLockOpenException</subEventName>
    <!--req door lock open exception-->
    <subEventName>DoorLockOpenTimeout</subEventName>
    <!--req door lock open timeout-->
    <subEventName>FirstCardOpenWithoutAuthorize</subEventName>
    <!--req first card failed to open door without authorization-->
    <subEventName>CallLadderRelayBreak</subEventName>
    <!--required,Elevator relay disconnected-->
    <subEventName>CallLadderRelayClose</subEventName>
    <!--required,Elevator relay connected-->
    <subEventName>AutoKeyRelayBreak</subEventName>
    <!--required,Auto-button relay disconnected-->
    <subEventName>AutoKeyRelayClose</subEventName>
    <!--required,Auto-button relay connected-->
    <subEventName>KeyControlRelayBreak</subEventName>
    <!--required,Button relay disconnected-->
    <subEventName>KeyControlRelayClose</subEventName>
    <!--required,Button relay connected-->
    <subEventName>RemoteVisitorCallLadder</subEventName>
    <!--required,Visitor called elevator-->
    <subEventName>RemoteHouseholdCallLadder</subEventName>
    <!--required,Resident called elevator-->
    <subEventName>LegalMessage</subEventName>
    <!--required, valid message-->
    <subEventName>IllegalMessage</subEventName>
    <!--required, invalid message-->
    <subEventName>Trailing</subEventName>
    <!--required, tailgating-->
```

```
      <subEventName>ReverseAccess</subEventName>
      <!--required, reserve passing-->
      <subEventName>ForceAccess</subEventName>
      <!--required, force accessing-->
      <subEventName>ClimbingOverGate</subEventName>
      <!--required, climbing over barrier-->
      <subEventName>PassingTimeout</subEventName>
      <!--required, passing timed out-->
      <subEventName>IntrusionAlarm</subEventName>
      <!--required, intrusion alarm-->
      <subEventName>FreeGatePassNotAuth</subEventName>
      <!--required, authentication failed when free passing the turnstile-->
      <subEventName>DropArmBlock</subEventName>
      <!--required, barrier obstructed-->
      <subEventName>DropArmBlockResume</subEventName>
      <!--required, barrier obstruction restored-->
      <subEventName>RemoteControlCloseDoor</subEventName>
      <!--required, close door via keyfob-->
      <subEventName>RemoteControlOpenDoor</subEventName>
      <!--required, open door via keyfob-->
      <subEventName>RemoteControlAlwaysOpenDoor</subEventName>
      <!--required, remain door open via keyfob-->
    </SubEventNameList>
  </EventEntry>
  <EventEntry>
    <Index>3</Index>
    <mainEventName>ReaderEvent</mainEventName>
    <SubEventNameList>
      <subEventName>StressAlarm</subEventName>
      <!--required, duress alarm-->
      <subEventName>ReaderDesmantleAlarm</subEventName>
      <!--required, card reader tamper-proof alarm-->
      <subEventName>LegalCardPass</subEventName>
      <!--required, valid card successfully authenticated -->
      <subEventName>CardAndPasswdPass</subEventName>
      <!--required, card and password successfully authenticated -->
      <subEventName>CardAndPasswdFail</subEventName>
      <!--required, card and password authentication failed -->
      <subEventName>CardAndPasswdTimeout</subEventName>
      <!--required, card and password authentication timeout -->
      <subEventName>CardMaxAuthenticateFail</subEventName>
      <!--required, card reader authentication over times -->
      <subEventName>CardNoRight</subEventName>
      <!--required, no permission for the card -->
      <subEventName>CardInvalidPeriod</subEventName>
      <!--required, invalid time segment -->
      <subEventName>CardOutofDate</subEventName>
      <!--required, card exceeds the validate -->
      <subEventName>InvalidCard</subEventName>
      <!--required, invalid card No. -->
      <subEventName>AntiSneakFail</subEventName>
      <!--required, anti-passback authentication failed -->
```

```
<subEventName>InterlockDoorNotClose</subEventName>
<!--required, interlocking door not closed -->
<subEventName>FingerprintComparePass</subEventName>
<!--required, Fingerprint Recognition Passed-->
<subEventName>FingerprintCompareFail</subEventName>
<!--required, Fingerprint Recognition Failed-->
<subEventName>CardFingerprintVerifyPass</subEventName>
<!--required, Card + Fingerprint Authentication Passed-->
<subEventName>CardFingerprintVerifyFail</subEventName>
<!--required,Card + Fingerprint Authentication Failed-->
<subEventName>CardFingerprintVerifyTimeout</subEventName>
<!--required,Card + Fingerprint Authentication Timeout-->
<subEventName>CardFingerprintPasswdVerifyPass</subEventName>
<!--required, Card + Fingerprint + Password Authentication Passed-->
<subEventName>CardFingerprintPasswdVerifyFail</subEventName>
<!--required, Card + Fingerprint + Password Authentication Failed-->
<subEventName>CardFingerprintPasswdVerifyTimeout</subEventName>
<!--required, Card + Fingerprint + Password Authentication Timeout-->
<subEventName>FingerprintPasswdVerifyPass</subEventName>
<!--required, Fingerprint + Password Authentication Passed-->
<subEventName>FingerprintPasswdVerifyFail</subEventName>
<!--required, Fingerprint + Password Authentication Failed-->
<subEventName>FingerprintPasswdVerifyTimeout</subEventName>
<!--required, Fingerprint + Password Authentication Timeout-->
<subEventName>FingerprintInexistence</subEventName>
<!--required, No Fingerprint-->
<subEventName>FaceVerifyPass</subEventName>
<!--required, Face Authentication Passed-->
<subEventName>FaceVerifyFail</subEventName>
<!--required, Face Authentication Failed-->
<subEventName>FaceAndFpVerifyPass</subEventName>
<!--required, Face + Fingerprint Authentication Passed-->
<subEventName>FaceAndFpVerifyFail</subEventName>
<!--required,Face + Fingerprint Authentication Failed-->
<subEventName>FaceAndFpVerifyTimeout</subEventName>
<!--required, Face + Fingerprint Authentication Timeout-->
<subEventName>FaceAndPwVerifyPass</subEventName>
<!--required, Face + Password Authentication Passed-->
<subEventName>FaceAndPwVerifyFail</subEventName>
<!--required, Face + Password Authentication Failed-->
<subEventName>FaceAndPwVerifyTimeout</subEventName>
<!--required, Face + Password Authentication Timeout-->
<subEventName>FaceAndCardVerifyPass</subEventName>
<!--required, Face + Card Authentication Passed-->
<subEventName>FaceAndCardVerifyFail</subEventName>
<!--required, Face + Card Authentication Failed-->
<subEventName>FaceAndCardVerifyTimeout</subEventName>
<!--required,Face + Card Authentication Timeout-->
<subEventName>FaceAndPwAndFpVerifyPass</subEventName>
<!--required,Face + Password + Fingerprint Authentication Passed-->
<subEventName>FaceAndPwAndFpVerifyFail</subEventName>
<!--required,Face + Password + Fingerprint Authentication Failed-->
```

```
<subEventName>FaceAndPwAndFpVerifyTimeout</subEventName>
<!--required,Face + Password + Fingerprint Authentication Timeout-->
<subEventName>FaceAndCardAndFpVerifyPass</subEventName>
<!--required,Face + Card + Fingerprint Authentication Passed-->
<subEventName>FaceAndCardAndFpVerifyFail</subEventName>
<!--required,Face + Card + Fingerprint Authentication Failed-->
<subEventName>FaceAndCardAndFpVerifyTimeout</subEventName>
<!--required,Face + Card + Fingerprint Authentication Timeout-->
<subEventName>EmployeeAndFpVerifyPass</subEventName>
<!--required,Employee No. + Fingerprint Authentication Passed-->
<subEventName>EmployeeAndFpVerifyFail</subEventName>
<!--required,Employee No. + Fingerprint Authentication Failed-->
<subEventName>EmployeeAndFpVerifyTimeout</subEventName>
<!--required,Employee No. + Fingerprint Authentication Timeout-->
<subEventName>EmployeeAndFpAndPwVerifyPass</subEventName>
<!--required,Employee No. + Fingerprint + Password Authentication Passed-->
<subEventName>EmployeeAndFpAndPwVerifyFail</subEventName>
<!--required,Employee No. + Fingerprint + Password Authentication Failed-->
<subEventName>EmployeeAndFpAndPwVerifyTimeout</subEventName>
<!--required,Employee No. + Fingerprint + Password Authentication Timeout-->
<subEventName>EmployeeAndFaceVerifyPass</subEventName>
<!--required,Employee No. + Face Authentication Passed-->
<subEventName>EmployeeAndFaceVerifyFail</subEventName>
<!--required,Employee No. + Face Authentication Failed-->
<subEventName>EmployeeAndFaceVerifyTimeout</subEventName>
<!--required,Employee No. + Face Authentication Timeout-->
<subEventName>FaceRecognizeFail</subEventName>
<!--required, Face picture recognization failed-->
<subEventName>EmployeeAndPwVerifyPass</subEventName>
<!--required,Employee No. + Password Authentication Passed-->
<subEventName>EmployeeAndPwVerifyFail</subEventName>
<!--required,Employee No. + Password Authentication Failed-->
<subEventName>EmployeeAndPwVerifyTimeout</subEventName>
<!--required,Employee No. + Password Authentication Timeout-->
<subEventName>DoorOpenOrDormantFail</subEventName>
<!--required,door remains closed or sleepy status authentication failed.-->
<subEventName>AuthPlanDormantFail</subEventName>
<!--required,authentication of sleepy mode in the schedule failed.-->
<subEventName>CardEncryptVerifyFail</subEventName>
<!--required,authentication of card encryption failed.-->
<subEventName>SubmarinebackReplyFail</subEventName>
<!--required,response of anti-passing back server failed.-->
<subEventName>PasswordMismatch</subEventName>
<!--optional, password mismatched.-->
<subEventName>EmployeeNoNotExist</subEventName>
<!--required, the employee ID does not exist.-->
<subEventName>CombinedVerifyPass</subEventName>
<!--required, authenticated .-->
<subEventName>CombinedVerifyTimeout</subEventName>
<!--required, authentication timed out.-->
<subEventName>VerifyModeMismatch</subEventName>
<!--required, authentication mode mismatched.-->
```

```
     <subEventName>PasswordVerifyPass</subEventName>
     <!--optional, password authenticated-->
     <subEventName>HumanDetectFail</subEventName>
     <!--required,human detection failed.-->
     <subEventName>PeopleAndIdCardComparePass</subEventName>
     <!--required, face and ID card authenticated-->
     <subEventName>PeopleAndIdCardCompareFail</subEventName>
     <!--required, face and ID card authentication failed-->
     <subEventName>CPUCardEncryptVerifyFail</subEventName>
     <!--optional, verifying CPU card encryption failed-->
     <subEventName>NFCDisableVerifyFail</subEventName>
     <!--optional, disabling NFC verification failed-->
     <subEventName>EMCardRecognizeNotEnabled</subEventName>
     <!--optional, EM card recognition is disabled-->
     <subEventName>M1CardRecognizeNotEnabled</subEventName>
     <!--optional, M1 card recognition is disabled-->
     <subEventName>CPUCardRecognizeNotEnabled</subEventName>
     <!--optional, CPU card recognition is disabled-->
     <subEventName>IDCardRecognizeNotEnabled</subEventName>
     <!--optional, ID card recognition is disabled-->
     <subEventName>CardSetSecretKeyFail</subEventName>
     <!--optional, importing key to the card failed-->
    </SubEventNameList>
   </EventEntry>
  </EventList>
 </EventLinkage>

 <FingerPrint>
  <!--required, fingerprint parameters -->
  <enable opt="true,false"/>
  <!--required, whether to support fingerprint settings -->
  <cardNo min="" max=""/>
  <!--required, card No. length -->
  <fingerPrintLen min="" max=""/>
  <!--required, fingerprint data length-->
  <EnableCardReader min="" max=""/>
  <!--required, supported card reader No.-->
  <fingerType opt="Normal,Stress,patrolFP,superFP,dismissingFP"/>
  <!--required, "Normal"-normal fingerprint, "Stress"-duress fingerprint, "patrolFP"-patrol fingerprint, "superFP"-
super fingerprint, "dismissingFP"-dismiss fingerprint-->
  <fingerPrintID min="" max=""/>
  <!--required, finger ID-->
  <callbackMode opt="allRetrun,partReturn"/>
  <!--required, callback mode, allRetrun-block (return after all the card readers are offline), partReturn-non-block
(return after a part of card readers are offline)-->
  <isSupportFingerNo/>
  <!--optional, boolean, whether the device supports setting finger ID: "true"-yes-->
  <recvStatus opt="0,1,2,3,4,5,6,7,8,9,10"/>
  <!--optional, error status: 0-success, 1-incorrect finger ID, 2-incorrect fingerprint type, 3-invalid card No. (the card
No. does not meet the device requirements), 4-the fingerprint is not linked with employee No. or card No. (the
employee No. or the card No. is NULL), 5-the employee No. does not exist, 6-the fingerprint data length is 0, 7-invalid
card reader No., 8-invalid employee No., 9-invalid first-time authentication value, 10-other parameters error-->
```

```
  <employeeNo min="" max=""/>
  <!--optional, employee No. (person ID)-->
  <leaderFP opt="true"/>
  <!--optional, whether the fingerprint supports first-time authentication: "true"-yes, "false" or this node is not
returned-no-->
  <isSupportFingerCover>
   <!--optional, xs:boolean, whether to overwrite the old fingerprint information when applying a new fingerprint
information linked to the same employee No. (person ID): "true"-yes, this node is not returned-no-->
  </isSupportFingerCover>
 </FingerPrint>
 <DelFingerPrint>
  <!--required, delete fingerprint parameter, which corresponds to the command
NET_DVR_DEL_FINGERPRINT_CFG_V50. This node will not be returned if device does not support this function. After
calling the API NET_DVR_StartRemoteConfig with command NET_DVR_DEL_FINGERPRINT_CFG_V50, if this node is
returned, you should wait for the return of callback function to get the actual deleting result; if this node is not
returned, the return of API NET_DVR_StartRemoteConfig already indicates the deleting result-->
  <delFingerPrintMode opt="byCard,byReader"/>
  <!--required, deleting fingerprint mode: byCard-by card No., byReader- by card reader-->
  <FingerPrintStatus>
   <!--required, delete fingerprint status-->
   <cardReaderNo min="" max=""/>
   <!--required, fingerprint recorder No.-->
   <status min="0" max="3"/>
   <!--required, status: 0-invalid, 1-handling, 2-deleting failed, 3-completed-->
  </FingerPrintStatus>
  <employeeNo min="" max=""/>
  <!--required, employee No. (person ID)-->
 </DelFingerPrint>

  <SMS>
  <enable opt="true,false"/>
  <!--required, whether to support SMS funtion -->
  <PhoneLinkageDoor>
   <!--required, mobile phone links with door -->
   <openRight opt="true,false"/>
   <!--required, door opening permission -->
   <closeRight opt="true,false"/>
   <!--required, door closing permission -->
   <NormalOpenRight opt="true,false"/>
   <!--required, door remained opening permission -->
   <NormalCloseRight opt="true,false"/>
   <!--required, door remained closing permission -->
   <armRight opt="true,false"/>
   <!--required, arming permission -->
   <DisarmRight opt="true,false"/>
   <!--required, disarming permission -->
  </PhoneLinkageDoor>
  <whiteListNum min="1" max="32"/>
  <!--required, allowlist number-->
 </SMS>
  <RealteUserInfo>
    <!--required, NET_DVR_CARD_CFG_SEND_DATA and NET_DVR_CARD_USER_INFO_CFG-->
    <enabled opt="true,false"/>
    <!--required, whether to support card No. being linked to user information-->
    <userNameLen min="" max=""/>
    <!--required, user name length-->
  </RealteUserInfo>
  <ContinuousShootCfg>
    <!--required,NET_DVR_SNAPCFG-->
    <enabled opt="true,false"/>
    <!--required,whether to support triggering capture parameters configuration-->
    <relatedDriveWay min="" max=""/>
    <!--required, IO related vehicle lane No.-->
    <snapTimes min="" max=""/>
    <!--required, coil capture times:, 0-5-->
    <snapWaitTime min="" max=""/>
    <!--required, capture waiting time, unit:ms, value range[0,60000]-->
    <IntervalTimeList size="4">
      <intervalTime min="" max=""/>
      <!--required,interval of continuous capture, unit:ms-->
    </IntervalTimeList>
    <JpegParam>
   <picSize
opt="CIF,QCIF,D1,UXGA,SVGA,HD720P,VGA,XVGA,HD900p,HD1080,2560*1920,1600*304,2048*1536,2448*2048,2448
*1200,
       2448*800,XGA,SXGA,WD1,1080i,
576*576,1536*1536,1920*1920,320*240,720*720,1024*768,1280*1280,1600*600,

2048*768,160*120,336*256,384*256,384*216,320*256,320*192,512*384,480*272,512*272,288*320,144*176,
       480*640,240*320,120*160,576*720,720*1280,576*960, 180*240, 360*480, 540*720, 720*960, 960*1280,
1080*1440, Auto"/>
    <!-- optional,image size-->
    <picQuality opt="best,good,general" />
    <!-- optional,image quality: 0-Best, 1- Better, 2- Good-->
  </JpegParam>
  </ContinuousShootCfg>
  <PictureCfg>
    <!--required,reuse some fields of NET_DVR_PICTURECFG-->
    <enableUp opt="true,false"/>
    <!--required, whether to support background picture uploading-->
    <enableDel opt="true,false"/>
    <!--required, whether to support deleting background picture-->
    <useType min="" max=""/>
    <!--required,picture type, 1- background picture, 2-GIF picture, 3-CAD picture-->
    <sequence min="" max=""/>
    <!--required, sequence No.-->
    <BasemapCfg>
      <sourWidth min="" max=""/>
      <!--required, initial picture width-->
      <sourHeight min="" max=""/>
      <!--required, initial picture height-->
    </BasemapCfg>
  </PictureCfg>
 <ExternalDevCfg>
```

```
    <!--required,NET_DVR_ACS_EXTERNAL_DEV_CFG-->
    <IDCardUpMode opt="number,all"/>
    <!--required, ID information report, number: upload 18-digit ID number; all: upload all information-->
    <cardVerifyMode opt="remoteCenter,clientPlatform"/>
    <!--required, card verification mode, remoteCenter: remote center verification; clientPlatform: client platform
verification-->
    <ACSDevType
opt="IDCardReader,ICReader,QRCodeReader,fingerPrintReader,QRCodeReaderandScreen,recycleCard,screen,fingerPrin
tModule,voiceModule,peopleAndIdCard"/>
    <!--required, device model: 1- ID card reader, 2- IC card reader, 3- QR code reader, 4- Fingerprint reader, 5- Screen +
QR code reader, 6- Card collector, 7- Screen, 8- Fingerprint scanner, 9- Voice module, 10-person and ID card device-->
    <doorMode opt="inDoor,outDoor"/>
    <!--required, door in/out type, inDoor: enter, outDoor: exit-->
    <DevDetailType>
      <IDCardReaderType opt="iDR210,IDM10,HikIDCardReader"/>
      <!--required, ID card reader model-->
      <screenType opt="DC48270RS043_01T,DC80480B070_03T"/>
      <!--required,LCD model-->
    </DevDetailType>
  </ExternalDevCfg>
  <PersonnelChannelCfg>
    <!--required,NET_DVR_PERSONNEL_CHANNEL_CFG-->
    <inMode opt="controlled,forbid,freedom"/>
    <!--required, enter mode, 0- controlled; 1- denied; 2- free-->
    <outMode opt="controlled,forbid,freedom"/>
    <!--required, exit mode, 0- controlled; 1- denied; 2- free-->
    <workMode opt="urgent,repair,normalClose,normalOpen"/>
    <!--required, operating mode, 0- emergency, 1- maintenance, 2- normally closed, 3- normally open-->
  </PersonnelChannelCfg>
  <PlatformVerifyCfg>
    <!--required,NET_DVR_PLATFORM_VERIFY_CFG-->
    <doorNo min="" max=""/>
    <!--required, door No.-->
    <resultType opt="legal,illegal"/>
    <!--required, verification result type, legal: illegal, illegal: legal-->
    <screenDisplay min="" max=""/>
    <!--required,LED display character length-->
  </PlatformVerifyCfg>
  <PersonStatisticsCfg>
    <!--required,NET_DVR_PERSON_STATISTICS_CFG-->
    <enableStatistics opt="true,false"/>
    <!--required, whether to enable people counting-->
    <enableOfflineStatistics opt="true,false"/>
    <!--required, whether to enable offline people counting-->
    <countSignalStatisticalStandard opt="IRDetectPass,AuthQuantity"/>
    <!--required, people counting type: IRDetectPass- by IR detection, AuthQuantity- by authentication number-->
  </PersonStatisticsCfg>
  <ScreenDisplayCfg>
    <!--required,NET_DVR_SCREEN_DISPLAY_CFG-->
    <FontSize min="" max=""/>
    <!--required, font size-->
    <rowSpacing min="" max=""/>
    <!--required, row space-->
    <columnSpacing min="" max=""/>
    <!--required, column space-->
    <firstRowPosition opt="0,1/8,2/8,3/8,4/8,5/8,6/8,7/8"/>
    <!--required, first row position-->
    <degree opt="0,90"/>
    <!--required, character display direction abgle, unit: degree-->
    <screenType opt="DC48270RS043_01T,DC80480B070_03T"/>
    <!--required, screen type-->
  </ScreenDisplayCfg>
  <GateTimeCfg>
    <!--required,NET_DVR_GATE_TIME_CFG-->
    <holdOnALarmTime min="" max=""/>
    <!--required, extend alarm buzzer time, unit: ms -->
    <holdOnGateOpenTime min="" max=""/>
    <!--required, door open time before receiving close command, unit: ms-->
    <postponeIntrusionAlarmTime min="" max=""/>
    <!--required, delay trigger intrusion alarm time, unit: ms-->
    <noLaneAccessTimeLimitTime min="" max=""/>
    <!--required, timeout alarm time for no people passing after channel received valid passing verification signal, unit: s-->
    <safetyZoneStayTime min="" max=""/>
    <!--required, timeout alarm time for people staying in the channel when reached safety region after the channel received valid passing verification signal, unit:s-->
    <IRTriggerTimeoutTime min="0" max="255"/>
    <!--required, IR triggering timeout, unit: s-->
  </GateTimeCfg>
  <LocalControllerStatus>enable</LocalControllerStatus>
  <!--required, support getting distributed access controller status-->
  <searchLocalController>enable</searchLocalController>
  <!--required, support searching distributed access controller-->
  <showDeviceType opt="Floor"/>
  <!--optional,Display device type (by default, display the door parameters if there is no this field),Floor- Displayed floor-->

  <FaceParam>
    <!--required,Face parameter-->
    <enable opt="true"/>
    <!--required,whether to support face parameter configuration-->
    <cardNo min="" max=""/>
    <!--required,Card No. length-->
    <faceLen min="" max=""/>
    <!--required,Face data length-->
    <enableCardReader min="" max=""/>
    <!--required,Supported card reader No.-->
    <faceID min="" max=""/>
    <!--required,Face No.-->
    <faceDataType opt="module,picture"/>
    <!--required,Face data type (the default type is template if there is no this node)-->
    <isSupportFaceCover>
      <!--optional, whether supports covering existed data when applying face picture data-->
    </isSupportFaceCover>
  </FaceParam>
  <isSupportGetDeviceEvent opt="true,false"/>
  <!--optional, whether to support getting device event: "true"-yes, "false" or this node is not returned-no-->
  <isSupportDeployType min="0" max="1"/>
    <!--optional, supported arming type: 0-arm via client software, 1-real-time arming>
  <UploadRightControllerAudio>
    <!--required, uploading audio file of main controller-->
    <audioID min="2" max="32"/>
      <!--required, audio file ID. 0xffffffff indicates uploading all audio files, and currently the device only supports
uploading all audio files instead of uploading a single audio file by ID-->
  </UploadRightControllerAudio>
  <DownloadRightControllerAudio>
    <!--required, downloading audio file of main controller-->
    <audioID min="2" max="32"/>
      <!--required, audio file ID. 0xffffffff indicates downloading all audio files, and currently the device only supports
downloading all audio files instead of downloading a single audio file by ID-->
  </DownloadRightControllerAudio>
  <BlackListPictureParam>
    <!--required, parameter of picture in blocklist (NET_DVR_BULK_UPLOAD_BLOCKLIST_PICTURE)-->
    <BlackListPictureCond>
      <!--required, blocklist picture condition-->
      <pictureNum min="" max=""/>
      <!--required, picture quantity-->
    </BlackListPictureCond>
    <cardNo min="" max=""/>
    <!--required, card No.-->
    <name min="" max=""/>
    <!--required, name-->
    <sex opt="male,female"/>
    <!--required, gender: male- Male, female- Female-->
    <pictureValid opt="invalid,valid"/>
    <!--required, whether blocklist picture is valid: invalid- Invalid,valid? Valid-->
    <pictureLen min="" max=""/>
    <!--required, blocklist picture size-->
    <BlackListPictureStatus>
      <!--required, blocklist picture status-->
      <cardNo min="" max=""/>
      <!--required, card No.-->
      <status opt=" processing,failed,success"/>
      <!--required, status: processing- Processing, failed- Failed,success- Succeeded-->
    </BlackListPictureStatus>
  </BlackListPictureParam>
  <IDBlackListParam>
  <!--ID blocklist parameter (NET_DVR_BULK_UPLOAD_ID_BLOCKLIST)-->
    <IDBlackListCond>
      <!--required, ID blocklist condition-->
      <blackListNum min="" max=""/>
      <!--required, blocklist quantity-->
    </IDBlackListCond>
    <blackListValid opt="invalid,valid"/>
    <!--required, whether ID card blocklist is valid or not-->
    <IDBlackListStatus>
      <!--required, ID card blocklist status-->
      <IDNum min="" max=""/>
      <!--required, ID number-->
      <status opt=" processing,failed,success"/>
      <!--required, status: processing- Processing, failed- Failed, success- Succeeded-->
    </IDBlackListStatus>
  </IDBlackListParam>
  <CaptureFingerPrint>
    <!--optional, xs:boolean, collect fingerprint information-->
    <pictureType opt="full,quarter">
      <!--required, xs:string, fingerprint picture type-->
    </pictureType>
    <fingerNo min="1" max="10">
      <!--required, xs:integer, fingerprint No.-->
    </fingerNo>
    <isSupportFingerData opt="true,false">
      <!--required, xs:boolen, fingerprint data-->
    </isSupportFingerData>
    <isSupportFingerPicture opt="true,false">
      <!--required, xs:boolen, fingerprint picture-->
    </isSupportFingerPicture>
    <fingerPrintQuality min="1" max="100">
      <!--required, xs:integer, fingerprint quality-->
    </fingerPrintQuality>
  </CaptureFingerPrint>
  <CaptureFace>
    <!--optional, xs:boolean, collect face information-->
    <isSupportFaceTemplate1 opt="true,false">
      <!--required, xs:boolen, face template data 1-->
    </isSupportFaceTemplate1>
    <isSupportFaceTemplate2 opt="true,false">
      <!--required, xs:boolen, face template data 2-->
    </isSupportFaceTemplate2>
    <isSupportFacePic opt="true,false">
      <!--required, xs:boolen, face picture data-->
    </isSupportFacePic>
    <faceQuality min="1" max="100">
      <!--required, xs:integer, face quality-->
    </faceQuality>
    <captureProgress opt="0,100">
      <!--required, xs:integer, collection progress-->
    </captureProgress>
    <isSupportInfraredFacePic opt="true,false"><!--required, xs:boolen, whether to support infrared face picture
data></isSupportInfraredFacePic>
  </CaptureFace>
  <isSupportUploadCertificateBlackList>
    <!--optional, xs:boolean, Whether to support uploading ID Card blocklist-->
  </isSupportUploadCertificateBlackList>
  <isSupportGetRegisterInfo>
    <!--optional, xs:boolean, Whether supports getting registered information-->
  </isSupportGetRegisterInfo>
```

```
<isSupportDownloadCertificateBlackListTemplet>
 <!--optional, xs:boolean, Whether to support downloading template of ID card blocklist-->
</isSupportDownloadCertificateBlackListTemplet>
<ScheduleInfo>
 <!-- optional, xs:boolean, shift schedule information-->
 <command opt="personal,everyone">
  <!--required, xs:string, Search condition-->
 </command>
 <employeeNo min="" max="">
  <!--required, xs:integer, Employee No.-->
 </employeeNo>
 <name min="1" max="32">
  <!--required, xs:string, Name-->
 </name>
 <departmentName min="1" max="32">
  <!--required, xs:string, Department name-->
 </departmentName>
 <schedulePlanNo min="" max="">
  <!--required, xs:integer, Shift schedule No.-->
 </schedulePlanNo>
 <schedulePlanType opt="personal,department">
  <!--required, xs:string, Shift schedule type-->
 </schedulePlanType>
 <enabled opt="true,false">
  <!--required, xs:boolen, Enable-->
 </enabled>
 <scheduleType opt="noSchedule,ordinaryClass,workingClass">
  <!--required, xs:string, Shift type-->
 </scheduleType>
 <scheduleNo min="" max="">
  <!--required, xs:integer, Shift No.-->
 </scheduleNo>
 <scheduleStartTime>
  <!--required, xs:time, ISO8601 time, "2016-01-01", Start time-->
 </scheduleStartTime>
 <scheduleEndTime>
  <!--required, xs:time, ISO8601 time, "2016-02-17", End time-->
 </scheduleEndTime>
 <holidayNo min="" max="">
  <!--required, xs:integer, Holiday group No.-->
 </holidayNo>
</ScheduleInfo>
<AttendanceSummaryInfo>
 <!-- optional, xs:boolean, Time and attendance information overview-->
 <command opt="personal,everyone">
  <!--required, xs:string, Search condition-->
 </command>
 <employeeNo min="" max="">
  <!--required, xs:integer, Employee No.-->
 </employeeNo>
 <name min="1" max="32">
  <!--required, xs:string, Name-->
 </name>
 <departmentName min="1" max="32">
  <!--required, xs:string, Department name-->
 </departmentName>
 <workStandard min="" max="">
  <!--required, xs:integer, Standard working time (minutes)-->
 </workStandard>
 <workActual min="" max="">
  <!--required, xs:integer, Actual working time (minutes)-->
 </workActual>
 <lateTimes min="" max="">
  <!--required, xs:integer, Late times-->
 </lateTimes>
 <lateMinutes min="" max="">
  <!--required, xs:integer, Total late time (minutes)-->
 </lateMinutes>
 <leaveEarlyTimes min="" max="">
  <!--required, xs:integer, Early Leave Times-->
 </leaveEarlyTimes>
 <leaveEarlyMinutes min="" max="">
  <!--required, xs:integer, Total eearly leave time (minutes)-->
 </leaveEarlyMinutes>
 <overtimeStandard min="" max="">
  <!--required, xs:integer, Standard Overtime (minutes)-->
 </overtimeStandard>
 <overtimeActual min="" max="">
  <!--required, xs:integer, Actual Overtime (minutes)-->
 </overtimeActual>
 <attendanceStandard min="" max="">
  <!--required, xs:integer, Standard Attendance (day)-->
 </attendanceStandard>
 <attendanceActual min="" max="">
  <!--required, xs:integer, Actual Attendance (Day)-->
 </attendanceActual>
 <absentDays min="" max="">
  <!--required, xs:integer, Absent (Day)-->
 </absentDays>
</AttendanceSummaryInfo>
<AttendanceRecordInfo>
 <!--optional, xs:boolean, Time and Attendance Records-->
 <command opt="personal,everyone">
  <!--required, xs:string, Search Condition-->
 </command>
 <employeeNo min="" max="">
  <!--required, xs:integer, Employee No.-->
 </employeeNo>
 <name min="1" max="32">
  <!--required, xs:string, Name-->
 </name>
 <departmentName min="1" max="32">
  <!--required, xs:string, Department Name-->
 </departmentName>
 <attendanceTime>
  <!--required, xs:time, ISO8601 time, "2016-02-17T17:30:08+08:00", Attendance Time-->
 </attendanceTime>
</AttendanceRecordInfo>
<AbnormalInfo>
 <!-- optional, xs:boolean, Exception Statistics Information-->
 <command opt="personal,everyone">
  <!--required, xs:string, Search Condition-->
 </command>
 <employeeNo min="" max="">
  <!--required, xs:integer, Employee No.-->
 </employeeNo>
 <name min="1" max="32">
  <!--required, xs:string, Name-->
 </name>
 <departmentName min="1" max="32">
  <!--required, xs:string, Department Name-->
 </departmentName>
 <onDutyTime>
  <!--required, xs:time, ISO8601 time, "2016-02-17T08:30:08+08:00", Start-Work Time-->
 </onDutyTime>
 <offDutyTime>
  <!--required, xs:time, ISO8601 time, "2016-02-17T17:30:08+08:00", End-Work Time-->
 </offDutyTime>
 <lateMinutes min="" max="">
  <!--required, xs:integer, Late Duration (minutes)-->
 </lateMinutes>
 <leaveEarlyMinutes min="" max="">
  <!--required, xs:integer, Early Leave Duration (minutes)-->
 </leaveEarlyMinutes>
 <absenceMinutes min="" max="">
  <!--required, xs:integer, Absent Duration (minutes)-->
 </absenceMinutes>
 <totalMinutes min="" max="">
  <!--required, xs:integer, Total Duration (minutes)-->
 </totalMinutes>
</AbnormalInfo>
<CheckFacePicture>
<!-- optional, xs:boolean, authenticate identity via 1:N face picture matching-->
 <pictureNum min="" max="">
  <!--required, xs:integer, picture number>
 </pictureNum>
 <checkStatus opt="1,2,3,4,5,6,7,8,9,10,11">
  <!--required, xs:integer, matching result: 1-modeling completed, 2-modeling failed, 3-the communication with the
face picture module failed, 4-no face in the picture, 5-the face is too close to the top picture border, 6-the face is too
close to the bottom picture border, 7-the face is too close to the left picture border, 8-the face is too close to the right
picture border, 9-the face picture is clockwise, 10-the face picture is anticlockwise, 11-the proportion of the pupillary
distance is small, 12-face picture matches the template, 13-face picture mismatches the template>
 </checkStatus>
 <checkTemplate opt="0,1">
  <!--optional, xs:integer, 0-picture verification, 1-picture and modeling data matching verification>
 </checkTemplate>
</CheckFacePicture>
<supplementLightNo min="" max=""/>
<!--optional, supplement light No.-->
<maxWhiteFaceNum/>
<!--optional, the maximum number of face picture in allowlist>
<maxBlackFaceNum/>
<!--optional, the maximum number of face picture in blocklist>
<isSupportGetFailedFaceInfo>
 <!--optional, xs:boolean, whether supports getting the information of face modeling failure after upgrading-->
</isSupportGetFailedFaceInfo>
<FailedFaceInfoParam>
 <!--optional, xs:boolean, get the information of face modeling failure after upgrading-->
 <FailedFaceInfoCond/>
 <FailedFaceInfo>
  <!--required, face modeling failure information-->
  <cardNo min="" max=""/>
  <!--required, card number-->
  <errorCode min="" max=""/>
  <!--required, face modeling failure error code-->
 </FailedFaceInfo>
<isSupportFaceAndTemplate>
<!--optional, xs:boolean, whether supports configuring face picture and modeling data-->
</isSupportFaceAndTemplate>
<FaceAndTemplateParam>
 <!--optional, face picture and modeling data configuration-->
 <cardNo min="" max=""/>
 <!--required, card number-->
 <faceLen min="" max=""/>
 <!--required, face picture size-->
 <faceTemplateLen min="" max=""/>
 <!--required, face picture template data size-->
</FaceAndTemplateParam>
</AcsAbility>
```

## B.55 XML_Cap_AccessControl

AccessControl capability message in XML format

```
<AccessControl version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <isSupportWiegandCfg>
    <!--optional, xs:boolean, whether it supports Wiegand configuration-->
  </isSupportWiegandCfg>
  <isSupportModuleStatus>
    <!--optional, xs:boolean, whether it supports getting the status of secure door control unit-->
  </isSupportModuleStatus>
  <isSupportSNAPConfig>
    <!--optional, xs:boolean, whether it supports getting capture linkage parameters-->
  </isSupportSNAPConfig>
  <LocalController><!--opt-->
    <isSupportLocalControllerManage>
  <!--optional, xs:boolean, whether it supports distributed access controller management-->
 </isSupportLocalControllerManage>
 <isSupportLocalControllerControl>
  <!--optional, xs:boolean, whether it supports distributed access controller control-->
 </isSupportLocalControllerControl>
</LocalController>
<isSupportUSBManage>
 <!--optional, xs:boolean, whether it supports USB management of access control device-->
</isSupportUSBManage>
<isSupportIdentityTerminal>
 <!--optional, xs:boolean, whether it supports face recognition terminal configuration-->
</isSupportIdentityTerminal>
<isSupportDepartmentParam>
 <!--optional, xs:boolean, whether it supports setting department parameters-->
</isSupportDepartmentParam>
<isSupportSchedulePlan>
 <!--optional, xs:boolean, whether it supports setting shift schedule-->
</isSupportSchedulePlan>
<isSupportAttendanceRule>
 <!--optional, xs:boolean, whether it supports setting time and attendance rule-->
</isSupportAttendanceRule>
<isSupportOrdinaryClass>
 <!--optional, xs:boolean, whether it supports setting normal shift parameters-->
</isSupportOrdinaryClass>
<isSupportWorkingClass>
 <!--optional, xs:boolean, whether it supports setting man-hour shift parameters-->
</isSupportWorkingClass>
<isSupportAttendanceHolidayGroup>
 <!--optional, xs:boolean, whether it supports setting holiday group for time and attendance-->
</isSupportAttendanceHolidayGroup>
<isSupportAttendanceHolidayPlan>
 <!--optional, xs:boolean, whether it supports setting holiday schedule for time and attendance-->
</isSupportAttendanceHolidayPlan>
<isSupportLadderControlRelay>
 <!--optional, xs:boolean, whether it supports setting elevator controller relay-->
</isSupportLadderControlRelay>
<isSupportWiegandRuleCfg>
 <!--optional, xs:boolean, whether it supports setting Wiegand rule-->
</isSupportWiegandRuleCfg>
<isSupportM1CardEncryptCfg>
 <!--optional, xs:boolean, whether it supports M1 card encryption authentication-->
</isSupportM1CardEncryptCfg>
<isSupportDeployInfo>
 <!--optional, xs:boolean, whether it supports getting arming information-->
</isSupportDeployInfo>
<isSupportSubmarineBack>
 <!--optional, xs:boolean, whether it supports specifying anti-passing back server-->
</isSupportSubmarineBack>
<isSupportSubmarineBackHostInfo>
 <!--optional, xs:boolean, whether it supports setting access controllers with anti-passing back enabled-->
</isSupportSubmarineBackHostInfo>
<isSupportStartReaderInfo>
```

```
    <!--optional, xs:boolean, whether it supports setting first card reader-->
  </isSupportStartReaderInfo>
  <isSupportSubmarineBackReader>
    <!--optional, xs:boolean, whether it supports setting anti-passing back card reader-->
  </isSupportSubmarineBackReader>
  <isSupportServerDevice>
    <!--optional, xs:boolean, whether it supports setting anti-passing back server information-->
  </isSupportServerDevice>
  <isSupportReaderAcrossHost>
    <!--optional, xs:boolean, whether it supports enabling cross-controller anti-passing back function of card reader-->
  </isSupportReaderAcrossHost>
  <isSupportClearCardRecord>
    <!--optional, xs:boolean, whether it supports clearing card swiping records in anti-passing back server-->
  </isSupportClearCardRecord>
  <isSupportSubmarineBackMode>
    <!--optional, xs:boolean, whether it supports setting anti-passing back mode-->
  </isSupportSubmarineBackMode>
  <isSupportClearSubmarineBack>
    <!--optional, xs:boolean, whether it supports clearing cross-controller anti-passing back information-->
  </isSupportClearSubmarineBack>
  <isSupportFaceCompareCond><!--optional, xs:boolean, whether it supports configuring restriction condition
parameters of face picture comparison--></isSupportFaceCompareCond>
  <isSupportRemoteControlDoor>
    <!--optional, xs:boolean, whether it supports remote door, elevator, and lock control: "true"-yes, this node is not
returned-no-->
  </isSupportRemoteControlDoor>
  <isSupportUserInfo><!--optional, xs:boolean, whether it supports person management based on person--></
isSupportUserInfo>
  <EmployeeNoInfo><!--dep, employee No. (person ID) information, this node is valid only when the
isSupportUserInfo is "true"-->
    <employeeNo min="" max=""><!--optional, employee No. (person ID)--></employeeNo>
    <characterType opt="any,number">
      <!--optional, employee No. (person) ID type: "any"-any characters (default), "number"-digits (from 0 to 9), only one
value can be returned-->
    </characterType>
    <isSupportCompress>
      <!--optional, xs:boolean, whether it supports compressing employee No. (person ID) for storage: "true"-yes, this
node is not returned-no-->
    </isSupportCompress>
  </EmployeeNoInfo>
  <isSupportCardInfo><!--optional, xs:boolean, whether it supports card management based on person: "true"-yes,
this node is not returned-no--></isSupportCardInfo>
  <isSupportFDLib><!--optional, xs:boolean, whether it supports face picture library management--></isSupportFDLib>
  <isSupportUserInfoDetailDelete><!--optional, xs:boolean, whether it supports deleting person information and
permission: "true"-yes, this node is not returned-no--></isSupportUserInfoDetailDelete>
  <isSupportAuthCodeInfo>
    <!--optional, xs:boolean, whether it supports authentication password management: "true"-yes, this node is not
returned-no-->
  </isSupportAuthCodeInfo>
  <isSupportFingerPrintCfg>
    <!--optional, xs:boolean, whether it supports configuring fingerprint parameters: "true"-yes, this node is not
returned-no-->
```

```
  </isSupportFingerPrintCfg>
  <isSupportFingerPrintDelete>
    <!--optional, xs:boolean, whether it supports deleting fingerprint: "true"-yes, this node is not returned-no-->
  </isSupportFingerPrintDelete>
  <isSupportCaptureFingerPrint>
    <!--optional, xs:boolean, whether it supports collecting fingerprint information: "true"-yes, this node is not returned-
no-->
  </isSupportCaptureFingerPrint>
  <isSupportDoorStatusWeekPlanCfg>
    <!--optional, xs:boolean, whether it supports configuring door control week schedule: "true"-yes, this node is not
returned-no-->
  </isSupportDoorStatusWeekPlanCfg>
  <isSupportVerifyWeekPlanCfg>
    <!--optional, xs:boolean, whether it supports configuring week schedule of the card reader authentication mode:
"true"-yes, this node is not returned-no-->
  </isSupportVerifyWeekPlanCfg>
  <isSupportCardRightWeekPlanCfg>
    <!--optional, xs:boolean, whether it supports configuring week schedule of the access permission control: "true"-
yes, this node is not returned-no-->
  </isSupportCardRightWeekPlanCfg>
  <isSupportDoorStatusHolidayPlanCfg>
    <!--optional, xs:boolean, whether it supports configuring door control holiday schedule: "true"-yes, this node is not
returned-no-->
  </isSupportDoorStatusHolidayPlanCfg>
  <isSupportVerifyHolidayPlanCfg>
    <!--optional, xs:boolean, whether it supports configuring holiday schedule of the card reader authentication mode:
"true"-yes, this node is not returned-no-->
  </isSupportVerifyHolidayPlanCfg>
  <isSupportCardRightHolidayPlanCfg>
    <!--optional, xs:boolean, whether it supports configuring holiday schedule of the access permission control: "true"-
yes, this node is not returned-no-->
  </isSupportCardRightHolidayPlanCfg>
  <isSupportDoorStatusHolidayGroupCfg>
    <!--optional, xs:boolean, whether it supports configuring holiday group of the door control schedule: "true"-yes, this
node is not returned-no-->
  </isSupportDoorStatusHolidayGroupCfg>
  <isSupportVerifyHolidayGroupCfg>
    <!--optional, xs:boolean, whether it supports configuring holiday group of the control schedule of the card reader
authentication mode: "true"-yes, this node is not returned-no-->
  </isSupportVerifyHolidayGroupCfg>
  <isSupportUserRightHolidayGroupCfg>
    <!--optional, xs:boolean, whether it supports configuring holiday group of the access permission control schedule:
"true"-yes, this node is not returned-no-->
  </isSupportUserRightHolidayGroupCfg>
  <isSupportDoorStatusPlanTemplate>
    <!--optional, xs:boolean, whether it supports configuring door control schedule template: "true"-yes, this node is
not returned-no-->
  </isSupportDoorStatusPlanTemplate>
  <isSupportVerifyPlanTemplate>
    <!--optional, xs:boolean, whether it supports configuring schedule template of the card reader authentication
mode: "true"-yes, this node is not returned-no-->
  </isSupportVerifyPlanTemplate>
  <isSupportUserRightPlanTemplate>
    <!--optional, xs:boolean, whether it supports configuring schedule template of the access permission control: "true"-yes, this node is not returned-no-->
  </isSupportUserRightPlanTemplate>
  <isSupportDoorStatusPlan>
    <!--optional, xs:boolean, whether it supports configuring door control schedule: "true"-yes, this node is not returned-no-->
  </isSupportDoorStatusPlan>
  <isSupportCardReaderPlan>
    <!--optional, xs:boolean, whether it supports configuring control schedule of the card reader authentication mode: "true"-yes, this node is not returned-no-->
  </isSupportCardReaderPlan>
  <isSupportClearPlansCfg>
    <!--optional, xs:boolean, whether it supports clearing the access control schedule parameters: "true"-yes, this node is not returned-no-->
  </isSupportClearPlansCfg>
  <isSupportRemoteControlBuzzer>
    <!--optional, xs:boolean, whether it supports remotely controlling the buzzer of the card reader: "true"-yes, this node is not returned-no-->
  </isSupportRemoteControlBuzzer>
  <isSupportEventCardNoList>
    <!--optional, xs:boolean, whether it supports getting the list of event and card linkage ID: "true"-yes, this node is not returned-no-->
  </isSupportEventCardNoList>
  <isSupportEventCardLinkageCfg>
    <!--optional, xs:boolean, whether it supports configuring event and card linkage parameters: "true"-yes, this node is not returned-no-->
  </isSupportEventCardLinkageCfg>
  <isSupportClearEventCardLinkageCfg>
    <!--optional, xs:boolean, whether it supports clearing event and card linkage parameters: "true"-yes, this node is not returned-no-->
  </isSupportClearEventCardLinkageCfg>
  <isSupportAcsEvent>
    <!--optional, xs:boolean, whether it supports searching for access control events: "true"-yes, this node is not returned-no-->
  </isSupportAcsEvent>
  <isSupportAcsEventTotalNum>
    <!--optional, xs:boolean, whether it supports getting total number of access control events by specific conditions: "true"-yes, this node is not returned-no-->
  </isSupportAcsEventTotalNum>
  <isSupportDeployInfo>
    <!--optional, xs:boolean, whether it supports getting the arming information: "true"-yes, this node is not returned-no-->
  </isSupportDeployInfo>
  <isSupportEventOptimizationCfg>
    <!--optional, xs:boolean, whether it supports configuring event optimization: "true"-yes, this node is not returned-no-->
  </isSupportEventOptimizationCfg>
  <isSupportAcsWorkStatus>
    <!--optional, xs:boolean, whether it supports getting working status of the access control device: "true"-yes, this node is not returned-no-->
  </isSupportAcsWorkStatus>
  <isSupportDoorCfg>
    <!--optional, xs:boolean, whether it supports configuring door parameters: "true"-yes, this node is not returned-no--
>
  </isSupportDoorCfg>
  <isSupportCardReaderCfg>
    <!--optional, xs:boolean, whether it supports configuring card reader parameters: "true"-yes, this node is not
returned-no-->
  </isSupportCardReaderCfg>
  <isSupportAcsCfg>
    <!--optional, xs:boolean, whether it supports configuring parameters of access control device: "true"-yes, this node
is not returned-no-->
  </isSupportAcsCfg>
  <isSupportRemoteCheck>
    <!--optional, xs:boolean, whether it supports verifying access control events remotely: true-yes, this field is not
returned-no-->
  </isSupportRemoteCheck>
  <isSupportMaskDetection>
    <!--optional, xs:boolean, whether it supports mask detection: true-yes, this field is not returned-no-->
  </isSupportMaskDetection>
  <isSupportGroupCfg>
    <!--optional, xs:boolean, whether it supports configuring group parameters: "true"-yes, this node is not returned-
no-->
  </isSupportGroupCfg>
  <isSupportClearGroupCfg>
    <!--optional, xs:boolean, whether it supports clearing group parameters: "true"-yes, this node is not returned-no-->
  </isSupportClearGroupCfg>
  <isSupportMultiCardCfg>
    <!--optional, xs:boolean, whether it supports configuring multiple authentication mode: "true"-yes, this node is not
returned-no-->
  </isSupportMultiCardCfg>
  <isSupportMultiDoorInterLockCfg>
    <!--optional, xs:boolean, whether it supports configuring multi-door interlocking parameters: "true"-yes, this node
is not returned-no-->
  </isSupportMultiDoorInterLockCfg>
  <isSupportAntiSneakCfg>
    <!--optional, xs:boolean, whether it supports configuring anti-passing back parameters in the device: "true"-yes, this
node is not returned-no-->
  </isSupportAntiSneakCfg>
  <isSupportCardReaderAntiSneakCfg>
    <!--optional, xs:boolean, whether it supports configuring anti-passing back parameters for the card reader in the
device: "true"-yes, this node is not returned-no-->
  </isSupportCardReaderAntiSneakCfg>
  <isSupportClearAntiSneakCfg>
    <!--optional, xs:boolean, whether it supports clearing anti-passing back parameters: "true"-yes, this node is not
returned-no-->
  </isSupportClearAntiSneakCfg>
  <isSupportClearAntiSneak>
    <!--optional, xs:boolean, whether it supports clearing anti-passing back records in the device: "true"-yes, this node
is not returned-no-->
  </isSupportClearAntiSneak>
  <isSupportSmsRelativeParam>
    <!--optional, xs:boolean, whether it supports configuring message function: "true"-yes, this node is not returned-
```

```
no-->
  </isSupportSmsRelativeParam>
  <isSupportPhoneDoorRightCfg>
    <!--optional, xs:boolean, whether it supports configuring the door permission linked to the mobile phone number:
"true"-yes, this node is not returned-no-->
  </isSupportPhoneDoorRightCfg>
  <isSupportOSDPStatus>
    <!--optional, xs:boolean, whether it supports searching for OSDP card reader status: "true"-yes, this node is not
returned-no-->
  </isSupportOSDPStatus>
  <isSupportOSDPModify>
    <!--optional, xs:boolean, whether it supports editing OSDP card reader ID: "true"-yes, this node is not returned-no-->
  </isSupportOSDPModify>
  <isSupportLogModeCfg>
    <!--optional, xs:boolean, whether it supports configuring log mode: "true"-yes, this node is not returned-no-->
  </isSupportLogModeCfg>
  <FactoryReset>
    <isSupportFactoryReset><!--optional, xs: boolean, whether it supports restoring to default settings by condition--></
isSupportFactoryReset>
    <mode opt="full,basic,part"><!--optional, xs: string, conditions for restoring to default settings--></mode>
  </FactoryReset>
  <isSupportNFCCfg><!--optional, xs:boolean，  whether it supports enabling or disabling NFC function: "true"-yes, this
node is not returned-no--></isSupportNFCCfg>
  <isSupportRFCardCfg><!--optional, xs:boolean，  whether it supports enabling or disabling RF card recognition: "true"-
yes, this node is not returned-no--></isSupportRFCardCfg>
  <isSupportCaptureFace>
    <!--optional, xs:boolean, whether it supports collecting face pictures: "true"-yes, this node is not returned-no-->
  </isSupportCaptureFace>
  <isSupportCaptureInfraredFace>
    <!--optional, xs:boolean, whether it supports collecting infrared face pictures: "true"-yes, this node is not returned-
no-->
  </isSupportCaptureInfraredFace>
  <isSupportFaceRecognizeMode>
    <!--optional, xs:boolean, whether it supports configuring facial recognition mode: "true"-yes, this node is not
returned-no-->
  </isSupportFaceRecognizeMode>
  <isSupportRemoteControlPWChcek>
    <!--optional, xs:boolean, whether it supports verifying the password for remote door control: "true"-yes, this node
is not returned-no-->
  </isSupportRemoteControlPWChcek>
  <isSupportRemoteControlPWCfg>
    <!--optional, xs:boolean, whether it supports configuring the password for remote door control: "true"-yes, this
node is not returned-no-->
  </isSupportRemoteControlPWCfg>
  <isSupportAttendanceStatusModeCfg>
    <!--optional, xs:boolean, whether it supports configuring attendance mode: "true"-yes, this node is not returned-
no-->
  </isSupportAttendanceStatusModeCfg>
  <isSupportAttendanceStatusRuleCfg>
    <!--optional, xs:boolean, whether it supports configuring attendance status and rule: "true"-yes, this node is not
returned-no-->
  </isSupportAttendanceStatusRuleCfg>
  <isSupportCaptureCardInfo>
    <!--optional, xs:boolean, whether it supports collecting card information: "true"-yes, this node is not returned-no-->
  </isSupportCaptureCardInfo>
  <isSupportCaptureIDInfo>
    <!--optional, xs:boolean, whether it supports collecting ID card information: "true"-yes, this node is not returned-
no-->
  </isSupportCaptureIDInfo>
  <isSupportCaptureRule>
    <!--optional, xs:boolean, whether it supports configuring online collection rules: "true"-yes, this node is not
returned-no-->
  </isSupportCaptureRule>
  <isSupportCapturePresetParam>
    <!--optional, xs:boolean, whether it supports configuring preset parameters of online collection: "true"-yes, this
node is not returned-no-->
  </isSupportCapturePresetParam>
  <isSupportOfflineCapture>
    <!--optional, xs:boolean, whether it supports offline collection: "true"-yes, this node is not returned-no-->
  </isSupportOfflineCapture>
  <isSupportCardOperations>
    <!--optional, xs:boolean, whether it supports card operation: "true"-yes, this node is not returned-no-->
  </isSupportCardOperations>
  <isSupportRightControllerAudio>
    <!--optional, xs:boolean, whether it supports configuring audio file parameters of the main controller-->
  </isSupportRightControllerAudio>
  <isSupportChannelControllerCfg>
    <!--optional, xs:boolean, whether it supports configuring lane controller-->
  </isSupportChannelControllerCfg>
  <isSupportGateDialAndInfo>
    <!--optional, xs:boolean, whether it supports getting local DIP and information of the turnstile-->
  </isSupportGateDialAndInfo>
  <isSupportGateStatus>
    <!--optional, xs:boolean, whether it supports getting turnstile status-->
  </isSupportGateStatus>
  <isSupportGateIRStatus>
    <!--optional, xs:boolean, whether it supports getting  the status of the active infrared intrusion detector of the
turnstile-->
  </isSupportGateIRStatus>
  <isSupportGateRelatedPartsStatus>
    <!--optional, xs:boolean, whether it supports getting related components' status of the turnstile-->
  </isSupportGateRelatedPartsStatus>
  <isSupportChannelControllerAlarmLinkage>
    <!--optional, xs:boolean, whether it supports configuring alarm linkage of the lane controller-->
  </isSupportChannelControllerAlarmLinkage>
  <isSupportChannelControllerAlarmOut>
    <!--optional, xs:boolean, whether it supports configuring alarm output of the lane controller-->
  </isSupportChannelControllerAlarmOut>
  <isSupportChannelControllerAlarmOutControl>
    <!--optional, xs:boolean, whether it supports controlling alarm output of the lane controller-->
  </isSupportChannelControllerAlarmOutControl>
  <isSupportChannelControllerTypeCfg>
    <!--optional, xs:boolean, whether it supports configuring device type of the lane controller-->
  </isSupportChannelControllerTypeCfg>
  <isSupportRemoteCtrllerModeCfg>
    <!--optional, xs:boolean, whether it supports configuring parameters of the keyfob control mode-->
  </isSupportRemoteCtrllerModeCfg>
  <isSupportTTSText><!--optional, xs:boolean, whether it supports configuring the text of the audio prompt: true-yes. If this function is not supported, this node will be not returned--></isSupportTTSText>
  <isSupportIDBlackListCfg><!--optional, xs:boolean, whether it supports applying ID card blocklist: true-yes. If this function is not supported, this node will be not returned--></isSupportIDBlackListCfg>
  <isSupportUserDataImport><!--optional, xs:boolean, whether it supports importing person permission data: true-yes. If this function is not supported, this node will be not returned--></isSupportUserDataImport>
  <isSupportUserDataExport><!--optional, xs:boolean, whether it supports exporting person permission data: true-yes. If this function is not supported, this node will be not returned--></isSupportUserDataExport>
  <isSupportMaintenanceDataExport><!--optional, xs:boolean, whether it supports exporting maintenance data: true-yes. If this function is not supported, this node will be not returned--></isSupportMaintenanceDataExport>
  <isSupportLockTypeCfg><!--optional, xs:boolean, whether it supports configuring door lock status when the device is powered off: true-yes. If this function is not supported, this node will be not returned--></isSupportLockTypeCfg>
  <isSupportSafetyHelmetDetection><!--optional, xs:boolean, whether it supports configuring hard hat detection: true-yes, this node is not returned-no--></isSupportSafetyHelmetDetection>
  <isSupportKeyCfgAttendance><!--optional, xs:boolean, whether it supports configuring parameters of attendance check by pressing the key: true-yes, this node is not returned-no--></isSupportKeyCfgAttendance>
  <isSupportIDBlackListTemplate><!--optional, xs:boolean, whether it supports downloading the ID card blocklist template: true-yes, this node is not returned-no--></isSupportIDBlackListTemplate>
  <isSupportAttendanceWeekPlan><!--optional, xs:boolean, whether it supports configuring parameters of the week attendance schedule: true-yes, this node is not returned-no--></isSupportAttendanceWeekPlan>
  <isSupportClearAttendancePlan><!--optional, xs:boolean, whether it supports clearing the week attendance schedule: true-yes, this node is not returned-no--></isSupportClearAttendancePlan>
  <isSupportAttendanceMode><!--optional, xs:boolean, whether it supports configuring the attendance mode: true-yes, this node is not returned-no--></isSupportAttendanceMode>
  <isSupportAttendancePlanTemplate><!--whether it supports configuring the attendance schedule template: true-yes, this node is not returned-no--></isSupportAttendancePlanTemplate>
  <isSupportAttendancePlanTemplateList><!--optional, xs:boolean, whether it supports getting the list of attendance schedule templates: true-yes, this node is not returned-no--></isSupportAttendancePlanTemplateList>
  <isSupportCardVerificationRule><!--optional, xs:boolean, whether it supports configuring card No. authentication mode: true-yes, this node is not returned-no--></isSupportCardVerificationRule>
    <isSupportTemperatureMeasureCfg><!--optional, xs:boolean, whether it supports configuring temperature measurement parameters: true (support), this node is not returned (not support)--></isSupportTemperatureMeasureCfg>
  <isSupportTemperatureMeasureAreaCfg><!--optional, xs:boolean, whether it supports configuring parameters of the temperature measurement area: true (support), this node is not returned (not support)--></isSupportTemperatureMeasureAreaCfg>
  <isSupportTemperatureMeasureAreaCalibrationCfg><!--optional, xs:boolean, whether it supports configuring calibration parameters of the temperature measurement area: true (support), this node is not returned (not support)--></isSupportTemperatureMeasureAreaCalibrationCfg>
  <isSupportBlackObjectCfg><!--optional, xs:boolean, whether it supports configuring black body parameters: true (support), this node is not returned (not support)--></isSupportBlackObjectCfg>
  <isSupportHealthCodeCfg><!--optional, xs:boolean, whether it supports configuring health code parameters: true (support), this node is not returned (not support)--></isSupportHealthCodeCfg>
  <isSupportShowHealthCodeCfg><!--optional, xs:boolean, whether it supports configuring display parameters of the health code: true (support), this node is not returned (not support)--></isSupportShowHealthCodeCfg>
  <isSupportAddCustomAudio><!--optional, boolean, whether it supports importing custom audio, related URI: /ISAPI/AccessControl/customAudio/addCustomAudio?format=json--></isSupportAddCustomAudio>
  <isSupportDeleteCustomAudio><!--optional, boolean, whether it supports deleting custom audio, related URI: /ISAPI/AccessControl/customAudio/deleteCustomAudio?format=json--></isSupportDeleteCustomAudio>
  <isSupportSearchCustomAudio><!--optional, boolean, whether it supports searching for custom audio, related URI: /ISAPI/AccessControl/customAudio/searchCustomAudioStatus?format=json--></isSupportSearchCustomAudio>
  <isSupportBluetoothEncryptionInfo><!--optional, xs:boolean, whether it supports configuring bluetooth encryption information: true (support). If this function is not supported, this node will not be returned--></isSupportBluetoothEncryptionInfo>
  <isSupportBluetoothEncryptionVersion><!--optional, xs:boolean, whether it supports configuring bluetooth encryption version: true (support). If this function is not supported, this node will not be returned--></isSupportBluetoothEncryptionVersion>
  <isSupportBluetooth><!--optional, xs:boolean, whether it supports bluetooth configuration--></isSupportBluetooth>
</AccessControl>
```

## B.56 XML_Cap_ChannelControllerAlarmLinkage

ChannelControllerAlarmLinkage capability message in XML format

```
<ChannelControllerAlarmLinkage version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <Trailing><!--required, tailgating-->
    <masterAlarmOut min="1" max="4" ><!--required, xs:string, local alarm output ID linked to the master lane controller, it is between 1 and 4--></masterAlarmOut>
    <slaveAlarmOut min="1" max="4" ><!--required, xs:string, local alarm output ID linked to the slave lane controller, it is between 1 and 4--></slaveAlarmOut>
  </Trailing>
  <ReverseAccess><!--required, reverse passing-->
    <masterAlarmOut min="1" max="4" ><!--required, xs:string, local alarm output ID linked to the master lane controller, it is between 1 and 4--></masterAlarmOut>
    <slaveAlarmOut min="1" max="4" ><!--required, xs:string, local alarm output ID linked to the slave lane controller, it is between 1 and 4--></slaveAlarmOut>
  </ReverseAccess>
  <ForceAccess><!--required, force accessing-->
    <masterAlarmOut min="1" max="4" ><!--required, xs:string, local alarm output ID linked to the master lane controller, it is between 1 and 4--></masterAlarmOut>
    <slaveAlarmOut min="1" max="4" ><!--required, xs:string, local alarm output ID linked to the slave lane controller, it is between 1 and 4--></slaveAlarmOut>
  </ForceAccess>
  <ClimbingOverGate><!--required, climbing over barrier-->
    <masterAlarmOut min="1" max="4" ><!--required, xs:string, local alarm output ID linked to the master lane controller, it is between 1 and 4--></masterAlarmOut>
    <slaveAlarmOut min="1" max="4" ><!--required, xs:string, local alarm output ID linked to the slave lane controller, it is between 1 and 4--></slaveAlarmOut>
  </ClimbingOverGate>
  <PassingTimeout><!--required, passing timeout-->
    <masterAlarmOut min="1" max="4" ><!--required, xs:string, local alarm output ID linked to the master lane controller, it is between 1 and 4--></masterAlarmOut>
    <slaveAlarmOut min="1" max="4" ><!--required, xs:string, local alarm output ID linked to the slave lane controller, it is between 1 and 4--></slaveAlarmOut>
  </PassingTimeout>
  <IntrusionAlarm><!--required, intrusion alarm-->
    <masterAlarmOut min="1" max="4" ><!--required, xs:string, local alarm output ID linked to the master lane controller, it is between 1 and 4--></masterAlarmOut>
    <slaveAlarmOut min="1" max="4" ><!--required, xs:string, local alarm output ID linked to the slave lane controller, it
```

```
is between 1 and 4--></slaveAlarmOut>
  </IntrusionAlarm>
</ChannelControllerAlarmLinkage>
```

## B.57 XML_Cap_ChannelControllerAlarmOut

ChannelControllerAlarmOut capability message in XML format

```
<ChannelControllerAlarmOut version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <controllerType opt="Master,Slave"><!--required, xs:string, lane controller type: "Master"-main lane controller,
"Slave"-sub-lane controller--></controllerType>
  <alarmOutNo min="1" max="4"><!--required, xs:integer, alarm output No., it is between 1 and 4--></alarmOutNo>
  <delay min="0" max="5999"><!--required, xs:integer, alarm output duration, it is between 0 and 5999, and 0 refers
to continuous output, unit: second--></delay>
</ChannelControllerAlarmOut>
```

## B.58 XML_Cap_ChannelControllerAlarmOutControl

ChannelControllerAlarmOutControl capability message in XML format

```
<ChannelControllerAlarmOutControl version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <controllerType opt="Master,Slave" >
    <!--required, xs:string, lane controller type: "Master"-master lane controller, "Slave"-slave lane controller-->
  </controllerType>
  <alarmOutNo min="1" max="4" >
    <!--optional, xs:integer, alarm output No., which is between 1 and 4. If this node is not transmitted, it refers to all
alarm outputs-->
  </alarmOutNo>
  <alarmOutControl opt="Start,Stop" >
    <!--optional, xs:string, control alarm output: "Start"-start alarm output, "Stop"-stop alarm output-->
  </alarmOutControl>
</ChannelControllerAlarmOutControl>
```

## B.59 XML_Cap_ChannelControllerCfg

XML message about the configuration capability of lane controller

```
<ChannelControllerCfg version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <gatePassingMode opt="ByChannelController,ByRightController"><!--required, xs:string, turnstile passing mode:
"ByChannelController"-based on the lane controller's local DIP settings, "ByRightController"-based on the main
controller's settings--></gatePassingMode>
  <freePassAuthEnabled opt="enable,disable"><!--required, xs:string, whether the authentication is required for free
passing: "enable"-yes, "disable"-no--></freePassAuthEnabled>
  <openAndCloseSpeed min="1" max="10"><!--required, xs:integer, barrier's opening and closing speed, it is between
1 and 10, which represents the speed from 10% to 100%--></openAndCloseSpeed>
  <alarmSoundTime min="0" max="599"><!--required, xs:integer, alarm prompt sound duration, unit: second. The
value is between 0 and 599, and 0 refers to continuously playing alarm prompt sound--></alarmSoundTime>
```

```
  <tempUnit opt="Centigrade,Fahrenheit"><!--required, xs:string, temperature unit to be displayed: "Centigrade"-Celsius (°C), "Fahrenheit"-Fahrenheit (°F)--></tempUnit>
  <alarmAreaNoAuth opt="true,false"><!--optional, xs:boolean, whether opening door is prohibited in the alarm area--></alarmAreaNoAuth>
  <gateWingMaterial opt="Acrylic,StellPipe,SinglePUGate,DoublePUGate"><!--optional, xs:string, barrier material: "Acrylic"-acrylic, "StellPipe"-steel pipe, "SinglePUGate"-single PU gate, "DoublePUGate"-two PU gates--></gateWingMaterial>
  <channelLength min="550" max="1400"><!--optional, xs:integer, barrier length, unit: mm--></channelLength>
  <motorDirection opt="Clockwise,AntiClockwise"><!--optional, xs:string, motor rotation direction: "Clockwise", "AntiClockwise"--></motorDirection>
  <lampBoardLight min="" max=""><!--optional, xs:integer, light board brightness, it is between 0 and 100--></lampBoardLight>
  <openSpeed min="" max=""><!--optional, xs:string, barrier's opening speed, it is between 1 and 10 which represents the speed from 10% to 100%, and the default speed is 50%. If openAndCloseSpeed and openSpeed are both configured, the barrier's opening speed is determined by openSpeed--></openSpeed>
  <closeSpeed min="1" max="10"><!--optional, xs:integer, barrier's closing speed, it is between 1 and 10 which represents the speed from 10% to 100%, and the default speed is 40%. If openAndCloseSpeed and closeSpeed are both configured, the barrier's closing speed is determined by closeSpeed--></closeSpeed>
  <runMode><!--optional, xs:string, running mode: "doubleGateWing"-two barriers mode (default), "singleGateWing"-single barrier mode--></runMode>
</ChannelControllerCfg>
```

## B.60 XML_Cap_FaceCompareCond

XML message about condition configuration capability of face picture comparison

```
<FaceCompareCond version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <faceWidthLowerLimit min="" max=""><!--optional, xs:integer, face width threshold with highest priority, value range: [0, 100], when the detected face width is larger than this threshold, the following conditions will be ignored and the face comparison will be executed--></faceWidthLowerLimit>
  <pitch min="" max=""><!--optional, xs:integer, face raising or bowing angle, value range: [0, 90], unit: degree, the smaller the better--></pitch>
  <yaw min="" max=""><!--optional, xs:integer, face siding left or right angle, value range: [0, 90], unit: degree, the smaller the better--></yaw>
  <width min="" max=""><!--optional, xs:integer, face width, value range: [0, 100]--></width>
  <height min="" max=""><!--optional, xs:integer, face height, value range: [0, 100]--></height>
  <leftBorder min="" max=""><!--optional, xs:integer, left border of face, value range: [0, 100]--></leftBorder>
  <rightBorder min="" max=""><!--optional, xs:integer, right border of face, value range: [0, 100]--></rightBorder>
  <upBorder min="" max=""><!--optional, xs:integer, top border of face, value range: [0, 100]--></upBorder>
  <bottomBorder min="" max=""><!--optional, xs:integer, bottom border of face, value range: [0, 100]--></bottomBorder>
  <interorbitalDistance min="" max=""><!--optional, xs:integer, pupil distance, value range: [0, 100]--></interorbitalDistance>
  <faceScore min="" max=""><!--optional, xs:integer, face score, value range: [0, 100], the valid face score must be larger than this score--></faceScore>
  <maxDistance opt="0.5,1,1.5,2,auto"><!--optional, xs:string, maximum recognition distance: "0.5,1,1.5,2,auto", unit: m. This node has higher priority over <interorbitalDistance>--></maxDistance>
  <similarity min="0.0" max="1.0"><!--optional, xs:float, face comparison similarity--></similarity>
</FaceCompareCond>
```

## B.61 XML_Cap_GateDialAndInfo

GateDialAndInfo capability message in XML format

```
<GateDialAndInfo version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <workMode opt="Normal,Origin,Debug" ><!--required, xs:string, working mode: "Normal"-normal mode, "Origin"-
closed position setting mode, "Debug"-test mode--></workMode>
  <memoryModeEnabled opt="enable,disable" ><!--required, xs:string, whether to enable memory mode: "enable",
"disable"--></memoryModeEnabled>
  <alarmAreaNoAuth opt="true,false" ><!--required, xs:boolean, whether opening barrier is prohibited in the alarm
area--></alarmAreaNoAuth>
  <deviceType opt="DropGate,WingGate,ThreeRollerGate" ><!--required, xs:string, device type: "DropGate"-swing
barrier, "WingGate"-flap barrier, "ThreeRollerGate"-tripod turnstile--></deviceType>
  <DialMode><!--local DIP communication mode-->
    <InDoor opt="Controlled,Forbid,Free" ><!--required, xs:string, entrance: "Controlled"-controlled, "Forbid"-
prohibited, "Free"-free--></InDoor>
    <OutDoor opt="Controlled,Forbid,Free" ><!--required, xs:string, exit: "Controlled"-controlled, "Forbid"-prohibited,
"Free"-free--></OutDoor>
  </DialMode>
</GateDialAndInfo>
```

## B.62 XML_Cap_GateIRStatus

GateIRStatus capability message in XML format

```
<GateIRStatus version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <IREmitter><!--status of active infrared intrusion detector-->
    <triggered min="1" max="48" ><!--required, xs:string, triggering IR ID of the active infrared intrusion detector, it is
between 1 and 48--></triggered>
    <triggeredTimeout min="1" max="48" ><!--required, xs:string, triggering timeout IR ID of the active infrared
intrusion detector, it is between 1 and 48--></triggeredTimeout>
    <receiveBoardAbnormal min="1" max="48" ><!--required, xs:string, communication exception IR ID of the receiving
board, it is between 1 and 48--></receiveBoardAbnormal>
    <sendBoardAbnormal min="1" max="48" ><!--required, xs:string, communication exception IR ID of the sending
board, it is between 1 and 48--></sendBoardAbnormal>
    <sendAndReceiveLocateAbnormal min="1" max="48" ><!--required, xs:string, sending and receiving position
exception ID, it is between 1 and 48--></sendAndReceiveLocateAbnormal>
  </IREmitter>
  <masterIRAdaptorCommFailed min="1" max="2" ><!--required, xs:string, ID of communication with IR adapter of the
main lane controller failed, it can be set to 1 or 2--></masterIRAdaptorCommFailed>
  <slaveIRAdaptorCommFailed min="1" max="2" ><!--required, xs:string, ID of communication with IR adapter of the
sub-lane controller failed, it can be set to 1 or 2--></slaveIRAdaptorCommFailed>
</GateIRStatus>
```

## B.63 XML_Cap_GateRelatedPartsStatus

GateRelatedPartsStatus message in XML format

```
<GateRelatedPartsStatus version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <MasterChannelController><!--related components' status of master lane controller-->
    <motorSensor opt="Normal,Abnormal"><!--required, xs:string, whether the motor or the sensor is normal: "Normal"-normal, "Abnormal"-exception. This is used to detect the consistency between the motor encoder and the hall sensor--></motorSensor>
    <dropArmSensorAbnormal min="1" max="4"><!--required, xs:string, ID of barrier position sensor exception, it is between 1 and 4. This is used to detect barrier open position switch--></dropArmSensorAbnormal>
    <dropArm opt="Normal,Abnormal"><!--required, xs:string, barrier status: "Normal"-normal, "Abnormal"-exception (obstructed or not rotate)--></dropArm>
    <fireInput opt="Normal,Alarm"><!--required, xs:string, fire input status: "Normal"-normal, "Alarm"-alarm--></fireInput>
    <caseTemp min="-2000.0" max="3000.0"><!--required, xs:float, pedestal temperature, it is between -2000.0 and 3000.0 and it is accurate to one decimal place--></caseTemp>
    <alarmInTriggered min="1" max="8"><!--required, xs:string, alarm input triggering ID, it is between 1 and 8--></alarmInTriggered>
    <alarmOutTriggered min="1" max="4"><!--required, xs:string, alarm output triggering ID, it is between 1 and 4--></alarmOutTriggered>
    <brakeStatus opt="NotBrake,Brake"><!--required, xs:string, brake status: "NotBrake"-disable, "Brake"-enable--></brakeStatus>
    <fanStatus opt="NotStart,Start"><!--required, xs:string, fan status: "NotStart"-disable, "Start"-enable--></fanStatus>
    <lampBoardCommFailed min="1" max="4"><!--required, xs:string, ID of communication with light board failed, it is between 1 and 4--></lampBoardCommFailed>
  </MasterChannelController>
  <SlaveChannelController><!--related components' status of slave lane controller-->
    <motorSensor opt="Normal,Abnormal"><!--required, xs:string, whether the motor or the sensor is normal: "Normal"-normal, "Abnormal"-exception. This is used to detect the consistency between the motor encoder and the hall sensor--></motorSensor>
    <dropArmSensorAbnormal min="1" max="4"><!--required, xs:string, ID of barrier position sensor exception, it is between 1 and 4. This is used to detect barrier open position switch--></dropArmSensorAbnormal>
    <dropArm opt="Normal,Abnormal"><!--required, xs:string, barrier status: "Normal"-normal, "Abnormal"-exception (obstructed or not rotate)--></dropArm>
    <fireInput opt="Normal,Alarm"><!--required, xs:string, fire input status: "Normal"-normal, "Alarm"-alarm--></fireInput>
    <caseTemp min="-2000.0" max="3000.0"><!--required, xs:float, pedestal temperature, it is between -2000.0 and 3000.0 and it is accurate to one decimal place--></caseTemp>
    <alarmInTriggered min="1" max="8"><!--required, xs:string, alarm input triggering ID, it is between 1 and 8--></alarmInTriggered>
    <alarmOutTriggered min="1" max="4"><!--required, xs:string, alarm output triggering ID, it is between 1 and 4--></alarmOutTriggered>
    <brakeStatus opt="NotBrake,Brake"><!--required, xs:string, brake status: "NotBrake"-disable, "Brake"-enable--></brakeStatus>
    <fanStatus opt="NotStart,Start"><!--required, xs:string, fan status: "NotStart"-disable, "Start"-enable--></fanStatus>
    <lampBoardCommFailed min="1" max="4"><!--required, xs:string, ID of communication with light board failed, it is between 1 and 4--></lampBoardCommFailed>
  </SlaveChannelController>
</GateRelatedPartsStatus>
```

## B.64 XML_Cap_GateStatus

GateStatus capability message in XML format

```
<GateStatus version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <busSync opt="Normal,DropArmPoorSyn,BusCommFailed" ><!--required, xs:string, BUS synchronization status:
"Normal"-normal, "DropArmPoorSyn"-poor barriers synchronization, "BusCommFailed"-BUS communication failed--
></busSync>
  <inDoorPassCount min="0" max="0xffffffff" ><!--required, xs:integer, IR people counting (entrance), it is between 0
and 0xffffffff. If reaching 0xffffffff, it will count from 0 again--></inDoorPassCount>
  <inDoorAuthCount min="0" max="0xffffffff" ><!--required, xs:integer, people counting by authenticated times
(entrance), it is between 0 and 0xffffffff. If reaching 0xffffffff, it will count from 0 again--></inDoorAuthCount>
  <outDoorPassCount min="0" max="0xffffffff" ><!--required, xs:integer, IR people counting (exit), it is between 0 and
0xffffffff. If reaching 0xffffffff, it will count from 0 again--></outDoorPassCount>
  <outDoorAuthCount min="0" max="0xffffffff" ><!--required, xs:integer, people counting by authenticated times (exit),
it is between 0 and 0xffffffff. If reaching 0xffffffff, it will count from 0 again--></outDoorAuthCount>
  <remoteControlRecvModule opt="Normal,Abnormal" ><!--required, xs:string, keyfob receiving module status:
"Normal"-normal, "Abnormal"-communication failed or the module is not installed--></remoteControlRecvModule>
  <caseTempUnit opt="Centigrade,Fahrenheit" ><!--required, xs:string, pedestal temperature unit to be displayed:
"Centigrade"-Centigrade (°C), "Fahrenheit"-Fahrenheit (°F)--></caseTempUnit>
  <currentInDoorMode opt="Controlled,Forbid,Free" ><!--required, xs:string, current passing mode (entrance):
"Controlled"-controlled, "Forbid"-prohibited, "Free"-free--></currentInDoorMode>
  <currentOutDoorMode opt="Controlled,Forbid,Free" ><!--required, xs:string, current passing mode (exit):
"Controlled"-controlled, "Forbid"-prohibited, "Free"-free--></currentOutDoorMode>
  <powerSupplyMode opt="ACPower,Battery" ><!--required, xs:string, device power supply mode: "ACPower"-by AC
power supply, "Battery"-by storage battery power supply--></powerSupplyMode>
</GateStatus>
```

## B.65 XML_Cap_GetAcsEvent

GetAcsEvent capability message in XML format

```
<GetAcsEvent version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <EventCond><!--req, event search conditions-->
    <majorType></majorType><!--req, event major type, see details in Access Control Event Types-->
    <minorType></minorType><!--req, event minor type, see details in Access Control Event Types-->
    <startTime></startTime><!--req, start time-->
    <endTime></endTime><!--req, end time-->
    <localOrUTC></localOrUTC><!--opt, time type: "Local"-device local time (default), "UTC"-UTC time. If this node is
not returned, the startTime and endTime will be used as the local time by default-->
    <cardNo min="" max=""></cardNo><!--req, card No.-->
    <name min="" max=""></name><!--req, cardholder name-->
    <picEnable opt="true,false"></picEnable><!--req, whether contains picture-->
    <beginSerialNo min="" max=""></beginSerialNo><!--req, start serial No.-->
    <endSerialNo min="" max=""></endSerialNo><!--req, end serial No.-->
    <employeeNo min="" max=""></employeeNo><!--opt, employee No. (person ID)-->
  </EventCond>
  <EventLog>
    <majorType>0x1</majorType><!--req, alarm event-->
```

```
   <MinorTypeList>
    <minorType>0x400</minorType><!--req, Zone short circuit attempts alarm-->
    <minorType>0x401</minorType><!--req, Zone open circuit attempts alarm-->
    <!--See more minor types of alarm event in Access Control Event Types-->
   <MinorTypeList>
  </EventLog>
  <EventLog>
   <majorType>0x2</majorType><!--req, exception alarm-->
   <MinorTypeList>
    <minorType>0x27</minorType><!--req, Network disconnected-->
    <minorType>0x3a</minorType><!--req, Connection exception-->
    <!--See more minor types of exception event in Access Control Event Types-->
   <MinorTypeList>
  </EventLog>
  <EventLog>
   <majorType>0x3</majorType><!--req, operation event-->
   <MinorTypeList>
    <minorType>0x400</minorType><!--req, Remotely opened door-->
    <minorType>0x401</minorType><!--req, remotely closed door-->
    <!--See more minor types of operation event in Access Control Event Types-->
   <MinorTypeList>
  </EventLog>
  <EventLog>
   <majorType>0x5</majorType><!--req, other event-->
   <MinorTypeList>
    <minorType>0x01</minorType><!--req, Authenticated by valid card-->
    <minorType>0x02</minorType><!--req, Authenticated by card and password-->
    <!--See more minor types of other event in Access Control Event Types-->
   <MinorTypeList>
  </EventLog>
</GetAcsEvent>
```

## See Also

***Access Control Event Types***

## B.66 XML_Cap_IdentityTerminal

IdentityTerminal capability message in XML format

```
<IdentityTerminal version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
 <terminalMode opt="authMode,registerMode">
  <!--req, xs: string, terminal mode: "authMode"-authentication mode, "registerMode"-registration mode-->
 </terminalMode>
 <idCardReader opt="iDR210,DS-K1F110-I,DS-K1F1110-B，  DS-K1F1110-AB, none">
  <!--req, xs: string,ID card reader model-->
 </idCardReader>
 <camera opt="C270,DS-2CS5432B-S"><!--req, xs: string, camera--></camera>
 <fingerPrintModule opt="ALIWARD,HikModule"><!--req, xs: string, fingerprint module--></fingerPrintModule>
 <videoStorageTime min="0" max="10"><!--req, xs: integer, time for saving video (unit: second)--></
videoStorageTime>
```

```
  <faceContrastThreshold min="0" max="100"><!--req, xs: integer, face picture comparison threshold--></faceContrastThreshold>
  <twoDimensionCode opt="enable,disable"><!--req, xs: string, whether to enable QR code recognition--></twoDimensionCode>
  <blackListCheck opt="enable,disable"><!--req, xs: string, whether to enable blocklist verification--></blackListCheck>
  <idCardCheckCenter opt="local,server">
    <!--req, xs: string, ID card comparison mode: local-compare with ID card of local storage, server-compare with ID card of remote server storage-->
  </idCardCheckCenter>
  <faceAlgorithm opt="HIK-Z,HIK-H">
    <!--req, xs: string, face picture algorithm: HIK-Z-Private algorithm, HIK-third-party algorithm-->
  </faceAlgorithm>
  <comNo min="1" max="9"><!--req, xs: integer, COM No.--></comNo>
  <memoryLearning opt="enable,disable"><!--req, xs: string, whether to enable learning and memory function--></memoryLearning>
  <saveCertifiedImage opt="enable,disable"><!--req, xs: string, whether to enable saving authenticated picture--></saveCertifiedImage>
  <MCUVersion min="" max=""><!--opt, xs: string, MCU version information--></MCUVersion>
  <usbOutput opt="enable,disable"><!--req, xs: string, whether to enable USB output of ID card reader--></usbOutput>
  <serialOutput opt="enable,disable"><!--req, xs: string, whether to enable serial port output of ID card reader--></serialOutput>
  <readInfoOfCard opt="serialNo,file"><!--opt, xs: string, set content to be read from CPU card--></readInfoOfCard>
  <workMode opt="passMode,accessControlMode"><!--opt, xs: string, authentication mode--></workMode>
  <ecoMode>
    <eco opt="enable,disable"><!--opt, xs: string, whether to enable ECO mode--></eco>
    <faceMatchThreshold1 min="" max=""><!--req, xs: integer, 1V1 face picture comparison threshold of ECO mode, which is between 0 and 100--></faceMatchThreshold1>
    <faceMatchThresholdN min="" max=""><!--req, xs: integer, 1:N face picture comparison threshold of ECO mode, which is between 0 and 100--></faceMatchThresholdN>
    <changeThreshold min="" max=""><!--opt, xs: string, switching threshold of ECO mode, which is between 0 and 8--></changeThreshold>
    <maskFaceMatchThresholdN min="0" max="100"><!--req, xs:integer, 1:N face picture (face with mask and normal background picture) comparison threshold of ECO mode, value range: [0,100]--></maskFaceMatchThresholdN>
    <maskFaceMatchThreshold1 min="0" max="100"><!--req, xs:integer, 1:1 face picture (face with mask and normal background picture) comparison threshold of ECO mode, value range: [0,100]--></maskFaceMatchThreshold1>
  </ecoMode>
  <readCardRule opt="wiegand26,wiegand34"><!--opt, xs: string, card No. setting rule: "wiegand26", "wiegand34"--></readCardRule>
  <enableScreenOff opt="true,false"><!--optional, xs:boolean, whether the device enters the sleep mode when there is no operation after the configured sleep time--></enableScreenOff>
  <screenOffTimeout min="" max=""><!--dependent, xs:integer, sleep time, unit: second--></screenOffTimeout>
  <enableScreensaver opt="true,false"><!--optional, xs:boolean, whether to enable the screen saver function--></enableScreensaver>
  <showMode opt="concise,normal"><!--optional, xs:string, display mode: "concise" (simple mode, only the authentication result will be displayed), "normal" (normal mode). The default mode is normal mode. If this node does not exist, the default mode is normal mode--></showMode>
  <menuTimeout min="" max=""><!--dependent, xs:integer, timeout period to exit, unit: second--></menuTimeout>
</IdentityTerminal>
```

## B.67 XML_Cap_RightControllerAudio

RightControllerAudio capability message in XML format

```
<RightControllerAudio version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <id min="2" max="32" ><!--required, xs:string, audio ID--></id>
  <audioName min="1" max="32" ><!--required, xs:string, audio name--></audioName>
  <playCondition opt="NotPlay,CompleteAuth,AuthFail,Alarm" >
    <!--required, xs:string, playing condition: "NotPlay"-not play, "CompleteAuth"-completely authenticated, "AuthFail"-
authentication failed, "Alarm"-alarm-->
  </playCondition>
</RightControllerAudio>
```

## B.68 XML_ChannelControllerAlarmLinkage

ChannelControllerAlarmLinkage message in XML format

```
<ChannelControllerAlarmLinkage version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <Trailing><!--required, tailgating-->
    <masterAlarmOut><!--required, xs:string, local alarm output ID linked to the master lane controller, it is between 1
and 4--></masterAlarmOut>
    <slaveAlarmOut><!--required, xs:string, local alarm output ID linked to the slave lane controller, it is between 1 and
4--></slaveAlarmOut>
  </Trailing>
  <ReverseAccess><!--required, reverse passing-->
    <masterAlarmOut><!--required, xs:string, local alarm output ID linked to the master lane controller, it is between 1
and 4--></masterAlarmOut>
    <slaveAlarmOut><!--required, xs:string, local alarm output ID linked to the slave lane controller, it is between 1 and
4--></slaveAlarmOut>
  </ReverseAccess>
  <ForceAccess><!--required, force accessing-->
    <masterAlarmOut><!--required, xs:string, local alarm output ID linked to the master lane controller, it is between 1
and 4--></masterAlarmOut>
    <slaveAlarmOut><!--required, xs:string, local alarm output ID linked to the slave lane controller, it is between 1 and
4--></slaveAlarmOut>
  </ForceAccess>
  <ClimbingOverGate><!--required, climbing over barrier-->
    <masterAlarmOut><!--required, xs:string, local alarm output ID linked to the master lane controller, it is between 1
and 4--></masterAlarmOut>
    <slaveAlarmOut><!--required, xs:string, local alarm output ID linked to the slave lane controller, it is between 1 and
4--></slaveAlarmOut>
  </ClimbingOverGate>
  <PassingTimeout><!--required, passing timeout-->
    <masterAlarmOut><!--required, xs:string, local alarm output ID linked to the master lane controller, it is between 1
and 4--></masterAlarmOut>
    <slaveAlarmOut><!--required, xs:string, local alarm output ID linked to the slave lane controller, it is between 1 and
4--></slaveAlarmOut>
  </PassingTimeout>
  <IntrusionAlarm><!--required, intrusion alarm-->
```

```
    <masterAlarmOut><!--required, xs:string, local alarm output ID linked to the master lane controller, it is between 1 and 4--></masterAlarmOut>
    <slaveAlarmOut><!--required, xs:string, local alarm output ID linked to the slave lane controller, it is between 1 and 4--></slaveAlarmOut>
  </IntrusionAlarm>
</ChannelControllerAlarmLinkage>
```

## B.69 XML_ChannelControllerAlarmOut

ChannelControllerAlarmOut message in XML format

```
<ChannelControllerAlarmOut version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <delay><!--required, xs:integer, alarm output duration, it is between 0 and 5999, and 0 refers to continuous output, unit: second--></delay>
</ChannelControllerAlarmOut>
```

## B.70 XML_ChannelControllerAlarmOutControl

ChannelControllerAlarmOutControl message in XML format

```
<ChannelControllerAlarmOutControl version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <controllerType>
    <!--required, xs:string, lane controller type: "Master"-master lane controller, "Slave"-slave lane controller-->
  </controllerType>
  <alarmOutNo>
    <!--optional, xs:integer, alarm output No., which is between 1 and 4. If this node is not transmitted, it refers to all alarm outputs-->
  </alarmOutNo>
  <alarmOutControl>
    <!--optional, xs:string, control alarm output: "Start"-start alarm output, "Stop"-stop alarm output-->
  </alarmOutControl>
</ChannelControllerAlarmOutControl>
```

## B.71 XML_ChannelControllerCfg

XML message about lane controller parameters

```
<ChannelControllerCfg version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <gatePassingMode><!--required, xs:string, turnstile passing mode: "ByChannelController"-based on the lane controller's local DIP settings, "ByRightController"-based on the main controller's settings--></gatePassingMode>
  <freePassAuthEnabled><!--required, xs:string, whether the authentication is required for free passing: "enable"-yes, "disable"-no--></freePassAuthEnabled>
  <openAndCloseSpeed><!--required, xs:integer, barrier's opening and closing speed, it is between 1 and 10, which represents the speed from 10% to 100%--></openAndCloseSpeed>
  <alarmSoundTime><!--required, xs:integer, alarm prompt sound duration, unit: second. The value is between 0 and 599, and 0 refers to continuously playing alarm prompt sound--></alarmSoundTime>
  <tempUnit><!--required, xs:string, temperature unit to be displayed: "Centigrade"-Celsius (°C), "Fahrenheit"-
```

```
Fahrenheit (°F)--></tempUnit>
  <alarmAreaNoAuth><!--optional, xs:boolean, whether opening door is prohibited in the alarm area--></
alarmAreaNoAuth>
  <gateWingMaterial><!--optional, xs:string, barrier material: "Acrylic"-acrylic, "StellPipe"-steel pipe, "SinglePUGate"-
single PU gate, "DoublePUGate"-two PU gates--></gateWingMaterial>
  <channelLength><!--optional, xs:integer, barrier length, unit: mm--></channelLength>
  <motorDirection><!--optional, xs:string, motor rotation direction: "Clockwise", "AntiClockwise"--></motorDirection>
  <lampBoardLight><!--optional, xs:integer, light board brightness, it is between 0 and 100--></lampBoardLight>
  <openSpeed><!--optional, xs:string, barrier's opening speed, it is between 1 and 10 which represents the speed from
10% to 100%, and the default speed is 50%. If openAndCloseSpeed and openSpeed are both configured, the barrier's
opening speed is determined by openSpeed--></openSpeed>
  <closeSpeed><!--optional, xs:integer, barrier's closing speed, it is between 1 and 10 which represents the speed from
10% to 100%, and the default speed is 40%. If openAndCloseSpeed and closeSpeed are both configured, the barrier's
closing speed is determined by closeSpeed--></closeSpeed>
  <runMode><!--optional, xs:string, running mode: "doubleGateWing"-two barriers mode (default), "singleGateWing"-
single barrier mode--></runMode>
</ChannelControllerCfg>
```

## B.72 XML_Desc_AcsAbility

Input description message for getting access control capability.

```
<AcsAbility version="2.0">
  <!--opt, specify child nodes about access control capabilities to be returned-->
</AcsAbility>
```

.

## B.73 XML_EventNotificationAlert_AlarmEventInfo

EventNotificationAlert message with alarm/event information in XML format.

```
<EventNotificationAlert version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <ipAddress><!--dep, xs:string, device IPv4 address--></ipAddress>
  <ipv6Address><!--dep, xs:string, device IPv6 address--></ipv6Address>
  <portNo><!--opt, xs:integer, device port number--></portNo>
  <protocol><!--opt, xs:string, protocol type for uploading alarm/event information, "HTTP,HTTPS"--></protocol>
  <macAddress><!--opt, xs:string, MAC address--></macAddress>
  <channelID><!--dep, xs:string, device channel No., starts from 1--></channelID>
  <dateTime><!--req, alarm/event triggered or occurred time, format: 2017-07-19T10:06:41+08:00--></dateTime>
  <activePostCount><!--req, xs:integer, alarm/event frequency, starts from 1--></activePostCount>
  <eventType><!--req, xs:string, alarm/event type, "peopleCounting, ANPR,..."--></eventType>
  <eventState>
    <!--req, xs:string, durative alarm/event status: "active"-valid, "inactive"-invalid, e.g., when a moving target is
detected,
     the alarm/event information will be uploaded continuously unit the status is set to "inactive"-->
  </eventState>
  <eventDescription><!--req, xs:string, alarm/event description--></eventDescription>
  <...><!--opt, for different alarm/event types, the nodes are different, see the message examples in different
```

```
applications--></...>
</EventNotificationAlert>
```

## B.74 XML_FaceCompareCond

XML message about condition parameters of face picture comparison

```
<FaceCompareCond version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <faceWidthLowerLimit><!--optional, xs:integer, face width threshold with highest priority, value range: [0, 100],
when the detected face width is larger than this threshold, the following conditions will be ignored and the face
comparison will be executed--></faceWidthLowerLimit>
  <pitch><!--optional, xs:integer, face raising or bowing angle, value range: [0, 90], unit: degree, the smaller the better--
></pitch>
  <yaw><!--optional, xs:integer, face siding left or right angle, value range: [0, 90], unit: degree, the smaller the better--
></yaw>
  <width><!--optional, xs:integer, face width, value range: [0, 100]--></width>
  <height><!--optional, xs:integer, face height, value range: [0, 100]--></height>
  <leftBorder><!--optional, xs:integer, left border of face, value range: [0, 100]--></leftBorder>
  <rightBorder><!--optional, xs:integer, right border of face, value range: [0, 100]--></rightBorder>
  <upBorder><!--optional, xs:integer, top border of face, value range: [0, 100]--></upBorder>
  <bottomBorder><!--optional, xs:integer, bottom border of face, value range: [0, 100]--></bottomBorder>
  <interorbitalDistance><!--optional, xs:integer, pupil distance, value range: [0, 100]--></interorbitalDistance>
  <faceScore><!--optional, xs:integer, face score, value range: [0, 100], the valid face score must be larger than this
score--></faceScore>
  <maxDistance><!--optional, xs:string, maximum recognition distance: "0.5,1,1.5,2,auto", unit: m. This node has
higher priority over <interorbitalDistance>--></maxDistance>
  <similarity><!--optional, xs:float, face comparison similarity, value range: [0.0,1.0]--></similarity>
</FaceCompareCond>
```

## B.75 XML_GateDialAndInfo

GateDialAndInfo message in XML format

```
<GateDialAndInfo version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <workMode><!--required, xs:string, working mode: "Normal"-normal mode, "Origin"-closed position setting mode,
"Debug"-test mode--></workMode>
  <memoryModeEnabled><!--required, xs:string, whether to enable memory mode: "enable", "disable"--></
memoryModeEnabled>
  <alarmAreaNoAuth><!--required, xs:boolean, whether opening barrier is prohibited in the alarm area--></
alarmAreaNoAuth>
  <deviceType><!--required, xs:string, device type: "DropGate"-swing barrier, "WingGate"-flap barrier,
"ThreeRollerGate"-tripod turnstile--></deviceType>
  <DialMode><!--local DIP communication mode-->
    <InDoor><!--required, xs:string, entrance: "Controlled"-controlled, "Forbid"-prohibited, "Free"-free--></InDoor>
    <OutDoor><!--required, xs:string, exit: "Controlled"-controlled, "Forbid"-prohibited, "Free"-free--></OutDoor>
  </DialMode>
</GateDialAndInfo>
```

## B.76 XML_GateIRStatus

GateIRStatus message in XML format

```
<GateIRStatus version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <IREmitter><!--status of active infrared intrusion detector-->
    <triggered><!--required, xs:string, triggering IR ID of the active infrared intrusion detector, it is between 1 and 48--
></triggered>
    <triggeredTimeout><!--required, xs:string, triggering timeout IR ID of the active infrared intrusion detector, it is
between 1 and 48--></triggeredTimeout>
    <receiveBoardAbnormal><!--required, xs:string, communication exception IR ID of the receiving board, it is between
1 and 48--></receiveBoardAbnormal>
    <sendBoardAbnormal><!--required, xs:string, communication exception IR ID of the sending board, it is between 1
and 48--></sendBoardAbnormal>
    <sendAndReceiveLocateAbnormal><!--required, xs:string, sending and receiving position exception ID, it is between
1 and 48--></sendAndReceiveLocateAbnormal>
  </IREmitter>
  <masterIRAdaptorCommFailed><!--required, xs:string, ID of communication with IR adapter of the main lane
controller failed, it can be set to 1 or 2--></masterIRAdaptorCommFailed>
  <slaveIRAdaptorCommFailed><!--required, xs:string, ID of communication with IR adapter of the sub-lane controller
failed, it can be set to 1 or 2--></slaveIRAdaptorCommFailed>
</GateIRStatus>
```

## B.77 XML_GateRelatedPartsStatus

GateRelatedPartsStatus message in XML format

```
<GateRelatedPartsStatus version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <MasterChannelController><!--related components' status of main lane controller-->
    <motorSensor><!--required, xs:string, whether the motor or the sensor is normal: "Normal"-normal, "Abnormal"-
exception. This is used to detect the consistency between the motor encoder and the hall sensor--></motorSensor>
    <dropArmSensorAbnormal><!--required, xs:string, ID of barrier position sensor exception, it is between 1 and 4.
This is used to detect barrier open position switch--></dropArmSensorAbnormal>
    <dropArm><!--required, xs:string, barrier status: "Normal"-normal, "Abnormal"-exception (obstructed or not
rotate)--></dropArm>
    <fireInput><!--required, xs:string, fire input status: "Normal"-normal, "Alarm"-alarm--></fireInput>
    <caseTemp><!--required, xs:float, pedestal temperature, it is between -2000.0 and 3000.0 and it is accurate to one
decimal place--></caseTemp>
    <alarmInTriggered><!--required, xs:string, alarm input triggering ID, it is between 1 and 8--></alarmInTriggered>
    <alarmOutTriggered><!--required, xs:string, alarm output triggering ID, it is between 1 and 4--></
alarmOutTriggered>
    <brakeStatus><!--required, xs:string, brake status: "NotBrake"-disable, "Brake"-enable--></brakeStatus>
    <fanStatus><!--required, xs:string, fan status: "NotStart"-disable, "Start"-enable--></fanStatus>
    <lampBoardCommFailed><!--required, xs:string, ID of communication with light board failed, it is between 1 and 4--
></lampBoardCommFailed>
  </MasterChannelController>
  <SlaveChannelController><!--related components' status of sub-lane controller-->
    <motorSensor><!--required, xs:string, whether the motor or the sensor is normal: "Normal"-normal, "Abnormal"-
exception. This is used to detect the consistency between the motor encoder and the hall sensor--></motorSensor>
```

```
   <dropArmSensorAbnormal><!--required, xs:string, ID of barrier position sensor exception, it is between 1 and 4. This is used to detect barrier open position switch--></dropArmSensorAbnormal>
   <dropArm><!--required, xs:string, barrier status: "Normal"-normal, "Abnormal"-exception (obstructed or not rotate)--></dropArm>
   <fireInput><!--required, xs:string, fire input status: "Normal"-normal, "Alarm"-alarm--></fireInput>
   <caseTemp><!--required, xs:float, pedestal temperature, it is between -2000.0 and 3000.0 and it is accurate to one decimal place--></caseTemp>
   <alarmInTriggered><!--required, xs:string, alarm input triggering ID, it is between 1 and 8--></alarmInTriggered>
   <alarmOutTriggered><!--required, xs:string, alarm output triggering ID, it is between 1 and 4--></alarmOutTriggered>
   <brakeStatus><!--required, xs:string, brake status: "NotBrake"-disable, "Brake"-enable--></brakeStatus>
   <fanStatus><!--required, xs:string, fan status: "NotStart"-disable, "Start"-enable--></fanStatus>
   <lampBoardCommFailed><!--required, xs:string, ID of communication with light board failed, it is between 1 and 4--></lampBoardCommFailed>
  </SlaveChannelController>
</GateRelatedPartsStatus>
```

## B.78 XML_GateStatus

GateStatus message in XML format

```
<GateStatus version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <busSync><!--required, xs:string, BUS synchronization status: "Normal"-normal, "DropArmPoorSyn"-poor barriers synchronization, "BusCommFailed"-BUS communication failed--></busSync>
  <inDoorPassCount><!--required, xs:integer, IR people counting (entrance), it is between 0 and 0xffffffff. If reaching 0xffffffff, it will count from 0 again--></inDoorPassCount>
  <inDoorAuthCount><!--required, xs:integer, people counting by authenticated times (entrance), it is between 0 and 0xffffffff. If reaching 0xffffffff, it will count from 0 again--></inDoorAuthCount>
  <outDoorPassCount><!--required, xs:integer, IR people counting (exit), it is between 0 and 0xffffffff. If reaching 0xffffffff, it will count from 0 again--></outDoorPassCount>
  <outDoorAuthCount><!--required, xs:integer, people counting by authenticated times (exit), it is between 0 and 0xffffffff. If reaching 0xffffffff, it will count from 0 again--></outDoorAuthCount>
  <remoteControlRecvModule><!--required, xs:string, keyfob receiving module status: "Normal"-normal, "Abnormal"-communication failed or the module is not installed--></remoteControlRecvModule>
  <caseTempUnit><!--required, xs:string, pedestal temperature unit to be displayed: "Centigrade"-Centigrade (°C), "Fahrenheit"-Fahrenheit (°F)--></caseTempUnit>
  <currentInDoorMode><!--required, xs:string, current passing mode (entrance): "Controlled"-controlled, "Forbid"-prohibited, "Free"-free--></currentInDoorMode>
  <currentOutDoorMode><!--required, xs:string, current passing mode (exit): "Controlled"-controlled, "Forbid"-prohibited, "Free"-free--></currentOutDoorMode>
  <powerSupplyMode><!--required, xs:string, device power supply mode: "ACPower"-by AC power supply, "Battery"-by storage battery power supply--></powerSupplyMode>
</GateStatus>
```

## B.79 XML_IdentityTerminal

IdentityTerminal message in XML format

```
<IdentityTerminal version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <terminalMode>
    <!--req, xs: string, terminal mode: "authMode"-authentication mode, "registerMode"-registration mode-->
  </terminalMode>
  <idCardReader>
    <!--req, xs: string, ID card reader model: iDR210, DS-K1F110-I, DS-K1F1110-B, DS-K1F1110-AB, none, DS-K1F1001-
I(USB), DS-K1F1002-I(USB), none-->
  </idCardReader>
  <camera><!--req, xs: string, camera model: C270, DS-2CS5432B-S--></camera>
  <fingerPrintModule><!--req, xs: string, fingerprint module type: ALIWARD, HikModule--></fingerPrintModule>
  <videoStorageTime><!--req, xs: integer, time for saving video (unit: second), which is between 0 and 10--></
videoStorageTime>
  <faceContrastThreshold><!--req, xs: integer, face picture comparison threshold, which is between 0 and 100--></
faceContrastThreshold>
  <twoDimensionCode><!--req, xs: string, whether to enable QR code recognition: enable, disable--></
twoDimensionCode>
  <blackListCheck><!--req, xs: string, whether to enable blocklist verification: enable, disable--></blackListCheck>
  <idCardCheckCenter>
    <!--req, xs: string, ID card comparison mode: local-compare with ID card of local storage, server-compare with ID
card of remote server storage-->
  </idCardCheckCenter>
  <faceAlgorithm>
    <!--req, xs: string, face picture algorithm: HIK-Z-Private algorithm, HIK-third-party algorithm-->
  </faceAlgorithm>
  <comNo><!--req, xs: integer, COM No., which is between 1 and 9--></comNo>
  <memoryLearning><!--req, xs: string, whether to enable learning and memory function: enable, disable--></
memoryLearning>
  <saveCertifiedImage><!--req, xs: string, whether to enable saving authenticated picture: enable, disable--></
saveCertifiedImage>
  <MCUVersion><!--opt, xs: string, MCU version information, read-only--></MCUVersion>
  <usbOutput><!--opt, xs: string, whether to enable USB output of ID card reader: enable, disable--></usbOutput>
  <serialOutput><!--opt, xs: string, whether to enable serial port output of ID card reader: enable, disable--></
serialOutput>
  <readInfoOfCard><!--opt, xs: string, set content to be read from CPU card: serialNo-read serial No., file-read file--></
readInfoOfCard>
  <workMode><!--opt, xs: string, authentication mode: passMode, accessControlMode--></workMode>
  <ecoMode>
    <eco><!--opt, xs: string, whether to enable ECO mode: enable, disable--></eco>
    <faceMatchThreshold1><!--req, xs: integer, 1V1 face picture comparison threshold of ECO mode, which is between
0 and 100--></faceMatchThreshold1>
    <faceMatchThresholdN><!--req, xs: integer, 1:N face picture comparison threshold of ECO mode, which is between
0 and 100--></faceMatchThresholdN>
    <changeThreshold><!--opt, xs: string, switching threshold of ECO mode, which is between 0 and 8--></
changeThreshold>
    <maskFaceMatchThresholdN><!--req, xs:integer, 1:N face picture (face with mask and normal background picture)
comparison threshold of ECO mode, value range: [0,100]--></maskFaceMatchThresholdN>
    <maskFaceMatchThreshold1><!--req, xs:integer, 1:1 face picture (face with mask and normal background picture)
comparison threshold of ECO mode, value range: [0,100]--></maskFaceMatchThreshold1>
  </ecoMode>
  <readCardRule><!--opt, xs: string, card No. setting rule: "wiegand26", "wiegand34"--></readCardRule>
  <enableScreenOff><!--optional, xs:boolean, whether the device enters the sleep mode when there is no operation
after the configured sleep time--></enableScreenOff>
```

```
  <screenOffTimeout><!--dependent, xs:integer, sleep time, unit: second--></screenOffTimeout>
  <enableScreensaver><!--optional, xs:boolean, whether to enable the screen saver function--></enableScreensaver>
  <showMode><!--optional, xs:string, display mode: "concise" (simple mode, only the authentication result will be displayed), "normal" (normal mode). The default mode is normal mode. If this node does not exist, the default mode is normal mode--></showMode>
  <menuTimeout><!--dependent, xs:integer, timeout period to exit, unit: second--></menuTimeout>
</IdentityTerminal>
```

## B.80 XML_ResponseStatus

XML message about response status

```
<?xml version="1.0" encoding="utf-8"?>
<ResponseStatus version="2.0" xmlns="http://www.std-cgi.org/ver20/XMLSchema">
  <requestURL>
    <!--required, read-only, xs:string, request URL-->
  </requestURL>
  <statusCode>
    <!--required, read-only, xs:integer, status code: 0,1-OK, 2-Device Busy, 3-Device Error, 4-Invalid Operation, 5-Invalid XML Format, 6-Invalid XML Content, 7-Reboot Required, 9-Additional Error-->
  </statusCode>
  <statusString>
    <!--required, read-only, xs:string, status description: OK, Device Busy, Device Error, Invalid Operation, Invalid XML Format, Invalid XML Content, Reboot, Additional Error-->
  </statusString>
  <subStatusCode>
    <!--required, read-only, xs:string, describe the error reason in detail-->
  </subStatusCode>
  <MErrCode>
    <!--optional, xs:string, error code categorized by functional modules, e.g., 0x12345678-->
  </MErrCode>
  <MErrDevSelfEx>
    <!--optional, xs:string, extension field of MErrCode. It is used to define the custom error code, which is categorized by functional modules-->
  </MErrDevSelfEx>
</ResponseStatus>
```

## B.81 XML_RightControllerAudio

RightControllerAudio message in XML format

```
<RightControllerAudio version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <audioName><!--required, xs:string, audio name--></audioName>
  <playCondition>
    <!--required, xs:string, playing condition: "NotPlay"-not play, "CompleteAuth"-completely authenticated, "AuthFail"-authentication failed, "Alarm"-alarm-->
  </playCondition>
</RightControllerAudio>
```

# Appendix C. Appendixes

## C.1 Access Control Event Types

The access control events are classified as four major types, i.e., alarm events (MAJOR_ALARM-0x1), exception events (MAJOR_EXCEPTION-0x2), operation events (MAJOR_OPERATION-0x3), and other events (MAJOR_EVENT-0x5). Each major type corresponds to multiple minor types, see details below.

### MAJOR_ALARM

| Event Minor Type | Value | Description |
|---|---|---|
| MINOR_ALARMIN_SHORT_ CIRCUIT | 0x400 | Zone Short Circuit Attempts Alarm |
| MINOR_ALARMIN_BROKEN_ CIRCUIT | 0x401 | Zone Disconnected Alarm |
| MINOR_ALARMIN_EXCEPTION | 0x402 | Zone Exception Alarm |
| MINOR_ALARMIN_RESUME | 0x403 | Zone Restored |
| MINOR_HOST_DESMANTLE_ ALARM | 0x404 | Zone Tampering Alarm |
| MINOR_HOST_DESMANTLE_ RESUME | 0x405 | Zone Tampering Restored |
| MINOR_CARD_READER_ DESMANTLE_ALARM | 0x406 | Card Reader Tampering Alarm |
| MINOR_CARD_READER_ DESMANTLE_RESUME | 0x407 | Card Reader Tampering Restored |
| MINOR_CASE_SENSOR_ALARM | 0x408 | Alarm Input Alarm Triggered |
| MINOR_CASE_SENSOR_ RESUME | 0x409 | Alarm Input Restored |
| MINOR_STRESS_ALARM | 0x40a | Duress Alarm |
| MINOR_OFFLINE_ECENT_ NEARLY_FULL | 0x40b | No Memory Alarm for Offline Events |
| MINOR_CARD_MAX_ AUTHENTICATE_FAIL | 0x40c | Maximum Failed Card Authentications Alarm |

| Event Minor Type | Value | Description |
|---|---|---|
| MINOR_SD_CARD_FULL | 0x40d | SD Card Full Alarm |
| MINOR_LINKAGE_CAPTURE_PIC | 0x40e | Capture Linkage Alarm |
| MINOR_SECURITY_MODULE_DESMANTLE_ALARM | 0x40f | Secure Door Control Unit Tampering Alarm |
| MINOR_SECURITY_MODULE_DESMANTLE_RESUME | 0x410 | Secure Door Control Unit Tampering Restored |
| MINOR_FIRE_IMPORT_SHORT_CIRCUIT | 0x415 | Fire Input Short Circuit Attempts Alarm |
| MINOR_FIRE_IMPORT_BROKEN_CIRCUIT | 0x416 | Fire Input Open Circuit Attempts Alarm |
| MINOR_FIRE_IMPORT_RESUME | 0x417 | Fire Input Restored |
| MINOR_FIRE_BUTTON_TRIGGER | 0x418 | Fire Button Triggered |
| MINOR_FIRE_BUTTON_RESUME | 0x419 | Fire Button Resumed |
| MINOR_MAINTENANCE_BUTTON_TRIGGER | 0x41a | Maintenance Button Triggered |
| MINOR_MAINTENANCE_BUTTON_RESUME | 0x41b | Maintenance Button Resumed |
| MINOR_EMERGENCY_BUTTON_TRIGGER | 0x41c | Panic Button Triggered |
| MINOR_EMERGENCY_BUTTON_RESUME | 0x41d | Panic Button Resumed |
| MINOR_DISTRACT_CONTROLLER_ALARM | 0x41e | Distributed Elevator Controller Tampering Alarm |
| MINOR_DISTRACT_CONTROLLER_RESUME | 0x41f | Distributed Elevator Controller Tampering Restored |
| MINOR_CHANNEL_CONTROLLER_DESMANTLE_ALARM | 0x422 | Lane Controller Tampering Alarm |

| Event Minor Type | Value | Description |
|---|---|---|
| MINOR_CHANNEL_ CONTROLLER_DESMANTLE_ RESUME | 0x423 | Lane Controller Tampering Alarm Restored |
| MINOR_CHANNEL_ CONTROLLER_FIRE_IMPORT_ ALARM | 0x424 | Lane Controller Fire Input Alarm |
| MINOR_CHANNEL_ CONTROLLER_FIRE_IMPORT_ RESUME | 0x425 | Lane Controller Fire Input Alarm Restored |
| MINOR_PRINTER_OUT_OF_ PAPER | 0x440 | No Paper in Printer Alarm |
| MINOR_LEGAL_EVENT_ NEARLY_FULL | 0x442 | No Memory Alarm for Valid Offline Events |
| MINOR_ALARM_CUSTOM1 to MINOR_ALARM_CUSTOM64 | 0x900 to 0x93f | Access Control: Custom Alarm Event 1 to Custom Alarm Event 64 |

## MAJOR_EXCEPTION

| Event Minor Type | Value | Description |
|---|---|---|
| MINOR_NET_BROKEN | 0x27 | Network Disconnected |
| MINOR_RS485_DEVICE_ ABNORMAL | 0x3a | RS485 Connection Exception |
| MINOR_RS485_DEVICE_ REVERT | 0x3b | RS485 Connection Restored |
| MINOR_DEV_POWER_ON | 0x400 | Power on |
| MINOR_DEV_POWER_OFF | 0x401 | Power off |
| MINOR_WATCH_DOG_RESET | 0x402 | Watchdog Reset |
| MINOR_LOW_BATTERY | 0x403 | Low Battery Voltage |
| MINOR_BATTERY_RESUME | 0x404 | Battery Voltage Restored |
| MINOR_AC_OFF | 0x405 | AC Power Disconnected |
| MINOR_AC_RESUME | 0x406 | AC Power Restored |
| MINOR_NET_RESUME | 0x407 | Network Restored |

| Event Minor Type | Value | Description |
|---|---|---|
| MINOR_FLASH_ABNORMAL | 0x408 | Flash Reading and Writing Exception |
| MINOR_CARD_READER_ OFFLINE | 0x409 | Card Reader Offline |
| MINOR_CAED_READER_ RESUME | 0x40a | Card Reader Online |
| MINOR_INDICATOR_LIGHT_ OFF | 0x40b | Indicator Turns off |
| MINOR_INDICATOR_LIGHT_ RESUME | 0x40c | Indicator Resumed |
| MINOR_CHANNEL_ CONTROLLER_OFF | 0x40d | Lane Controller Offline |
| MINOR_CHANNEL_ CONTROLLER_RESUME | 0x40e | Lane Controller Online |
| MINOR_SECURITY_MODULE_ OFF | 0x40f | Secure Door Control Unit Offline |
| MINOR_SECURITY_MODULE_ RESUME | 0x410 | Secure Door Control Unit Online |
| MINOR_BATTERY_ELECTRIC_ LOW | 0x411 | Low Battery Voltage (Only for Face Recognition Terminal) |
| MINOR_BATTERY_ELECTRIC_ RESUME | 0x412 | Battery Voltage Recovered (Only for Face Recognition Terminal) |
| MINOR_LOCAL_CONTROL_ NET_BROKEN | 0x413 | Network of Distributed Access Controller Disconnected |
| MINOR_LOCAL_CONTROL_ NET_RSUME | 0x414 | Network of Distributed Access Controller Restored |
| MINOR_MASTER_RS485_ LOOPNODE_BROKEN | 0x415 | RS485 Loop of Main Access Controller Disconnected |
| MINOR_MASTER_RS485_ LOOPNODE_RESUME | 0x416 | RS485 Loop of Main Access Controller Connected |
| MINOR_LOCAL_CONTROL_ OFFLINE | 0x417 | Distributed Access Controller Offline |

| Event Minor Type | Value | Description |
|---|---|---|
| MINOR_LOCAL_CONTROL_RESUME | 0x418 | Distributed Access Controller Online |
| MINOR_LOCAL_DOWNSIDE_RS485_LOOPNODE_BROKEN | 0x419 | Downstream RS485 Loop of Distributed Access Control Disconnected |
| MINOR_LOCAL_DOWNSIDE_RS485_LOOPNODE_RESUME | 0x41a | Downstream RS485 Loop of Distributed Access Control Connected |
| MINOR_DISTRACT_CONTROLLER_ONLINE | 0x41b | Distributed Elevator Controller Online |
| MINOR_DISTRACT_CONTROLLER_OFFLINE | 0x41c | Distributed Elevator Controller Offline |
| MINOR_ID_CARD_READER_NOT_CONNECT | 0x41d | ID Card Reader Disconnected |
| MINOR_ID_CARD_READER_RESUME | 0x41e | ID Card Reader Connected |
| MINOR_FINGER_PRINT_MODULE_NOT_CONNECT | 0x41f | Fingerprint Module Disconnected |
| MINOR_FINGER_PRINT_MODULE_RESUME | 0x420 | Fingerprint Module Connected |
| MINOR_CAMERA_NOT_CONNECT | 0x421 | Camera Disconnected |
| MINOR_CAMERA_RESUME | 0x422 | Camera Connected |
| MINOR_COM_NOT_CONNECT | 0x423 | COM Port Disconnected |
| MINOR_COM_RESUME | 0x424 | COM Port Connected |
| MINOR_DEVICE_NOT_AUTHORIZE | 0x425 | Device Unauthorized |
| MINOR_PEOPLE_AND_ID_CARD_DEVICE_ONLINE | 0x426 | Face Recognition Terminal Online |
| MINOR_PEOPLE_AND_ID_CARD_DEVICE_OFFLINE | 0x427 | Face Recognition Terminal Offline |
| MINOR_LOCAL_LOGIN_LOCK | 0x428 | Local Login Lock |

| Event Minor Type | Value | Description |
|---|---|---|
| MINOR_LOCAL_LOGIN_UNLOCK | 0x429 | Local Login Unlock |
| MINOR_SUBMARINEBACK_COMM_BREAK | 0x42a | Communication with Anti-passing Back Server Failed |
| MINOR_SUBMARINEBACK_COMM_RESUME | 0x42b | Communication with Anti-passing Back Server Restored |
| MINOR_MOTOR_SENSOR_EXCEPTION | 0x42c | Motor or Sensor Exception |
| MINOR_CAN_BUS_EXCEPTION | 0x42d | CAN Bus Exception |
| MINOR_CAN_BUS_RESUME | 0x42e | CAN Bus Exception Restored |
| MINOR_GATE_TEMPERATURE_OVERRUN | 0x42f | Too High Pedestal Temperature |
| MINOR_IR_EMITTER_EXCEPTION | 0x430 | Active Infrared Intrusion Detector Exception |
| MINOR_IR_EMITTER_RESUME | 0x431 | Active Infrared Intrusion Detector Restored |
| MINOR_LAMP_BOARD_COMM_EXCEPTION | 0x432 | Communication with Light Board Failed |
| MINOR_LAMP_BOARD_COMM_RESUME | 0x433 | Communication with Light Board Restored |
| MINOR_IR_ADAPTOR_COMM_EXCEPTION | 0x434 | Communication with IR Adaptor Failed |
| MINOR_IR_ADAPTOR_COMM_RESUME | 0x435 | Communication with IR Adaptor Restored |
| MINOR_PRINTER_ONLINE | 0x436 | Printer Online |
| MINOR_PRINTER_OFFLINE | 0x437 | Printer Offline |
| MINOR_4G_MOUDLE_ONLINE | 0x438 | 4G Module Online |
| MINOR_4G_MOUDLE_OFFLINE | 0x439 | 4G Module Offline |
| MINOR_AUXILIARY_BOARD_OFFLINE | 0x43c | Auxiliary Board Disconnected |
| MINOR_AUXILIARY_BOARD_RESUME | 0x43d | Auxiliary Board Connected |

| Event Minor Type | Value | Description |
|---|---|---|
| MINOR_IDCARD_SECURITY_ MOUDLE_EXCEPTION | 0x43e | Secure ID Card Unit Exception |
| MINOR_IDCARD_SECURITY_ MOUDLE_RESUME | 0x43f | Secure ID Card Unit Restored |
| MINOR_FP_PERIPHERAL_ EXCEPTION | 0x440 | Fingerprint Collection Peripheral Exception |
| MINOR_FP_PERIPHERAL_ RESUME | 0x441 | Fingerprint Collection Peripheral Restored |
| MINOR_EXTEND_MODULE_ ONLINE | 0x44d | Extension Module Online |
| MINOR_EXTEND_MODULE_ OFFLINE | 0x44e | Extension Module Offline |
| MINOR_EXCEPTION_CUSTOM1 to MINOR_EXCEPTION_ CUSTOM64 | 0x900 to 0x93f | Access Control: Custom Exception Event 1 to Custom Exception Event 64 |

## MAJOR_OPERATION

| Alarm Minor Types | Value | Description |
|---|---|---|
| MINOR_LOCAL_LOGIN | 0x50 | Local Login |
| MINOR_LOCAL_LOGOUT | 0x51 | Local Logout |
| MINOR_LOCAL_UPGRADE | 0x5a | Local Upgrade |
| MINOR_REMOTE_LOGIN | 0x70 | Remote Login |
| MINOR_REMOTE_LOGOUT | 0x71 | Remote Logout |
| MINOR_REMOTE_ARM | 0x79 | Remote Arming |
| MINOR_REMOTE_DISARM | 0x7a | Remote Disarming |
| MINOR_REMOTE_REBOOT | 0x7b | Remote Reboot |
| MINOR_REMOTE_UPGRADE | 0x7e | Remote Upgrade |
| MINOR_REMOTE_CFGFILE_ OUTPUT | 0x86 | Remote Operation: Export Configuration File |
| MINOR_REMOTE_CFGFILE_ INTPUT | 0x87 | Remote Operation: Import Configuration File |

| Alarm Minor Types | Value | Description |
|---|---|---|
| MINOR_REMOTE_ALARMOUT_OPEN_MAN | 0xd6 | Remote Operation: Enable Alarm Output Manually |
| MINOR_REMOTE_ALARMOUT_CLOSE_MAN | 0xd7 | Remote Operation: Disable Alarm Output Manually |
| MINOR_REMOTE_OPEN_DOOR | 0x400 | Door Remotely Open |
| MINOR_REMOTE_CLOSE_DOOR | 0x401 | Door Remotely Closed |
| MINOR_REMOTE_ALWAYS_OPEN | 0x402 | Remain Open Remotely |
| MINOR_REMOTE_ALWAYS_CLOSE | 0x403 | Remain Closed Remotely |
| MINOR_REMOTE_CHECK_TIME | 0x404 | Remote: Manual Time Sync |
| MINOR_NTP_CHECK_TIME | 0x405 | Network Time Protocol Synchronization |
| MINOR_REMOTE_CLEAR_CARD | 0x406 | Remote Operation: Clear All Card No. |
| MINOR_REMOTE_RESTORE_CFG | 0x407 | Remote Operation: Restore Defaults |
| MINOR_ALARMIN_ARM | 0x408 | Zone Arming |
| MINOR_ALARMIN_DISARM | 0x409 | Zone Disarming |
| MINOR_LOCAL_RESTORE_CFG | 0x40a | Local Operation: Restore Defaults |
| MINOR_REMOTE_CAPTURE_PIC | 0x40b | Remote Operation: Capture |
| MINOR_MOD_NET_REPORT_CFG | 0x40c | Edit Network Parameters |
| MINOR_MOD_GPRS_REPORT_PARAM | 0x40d | Edit GPRS Parameters |
| MINOR_MOD_REPORT_GROUP_PARAM | 0x40e | Edit Control Center Parameters |
| MINOR_UNLOCK_PASSWORD_OPEN_DOOR | 0x40f | Enter Dismiss Code |

| Alarm Minor Types | Value | Description |
|---|---|---|
| MINOR_AUTO_RENUMBER | 0x410 | Auto Renumber |
| MINOR_AUTO_COMPLEMENT_NUMBER | 0x411 | Auto Supplement Number |
| MINOR_NORMAL_CFGFILE_INPUT | 0x412 | Import Configuration File |
| MINOR_NORMAL_CFGFILE_OUTTPUT | 0x413 | Export Configuration File |
| MINOR_CARD_RIGHT_INPUT | 0x414 | Import Card Permission Parameters |
| MINOR_CARD_RIGHT_OUTTPUT | 0x415 | Export Card Permission Parameters |
| MINOR_LOCAL_USB_UPGRADE | 0x416 | Upgrade Device via USB flash Drive |
| MINOR_REMOTE_VISITOR_CALL_LADDER | 0x417 | Visitor Calling Elevator |
| MINOR_REMOTE_HOUSEHOLD_CALL_LADDER | 0x418 | Resident Calling Elevator |
| MINOR_REMOTE_ACTUAL_GUARD | 0x419 | Remotely Arming |
| MINOR_REMOTE_ACTUAL_UNGUARD | 0x41a | Remotely Disarming |
| MINOR_REMOTE_CONTROL_NOT_CODE_OPER_FAILED | 0x41b | Operation Failed: Keyfob Not Pairing |
| MINOR_REMOTE_CONTROL_CLOSE_DOOR | 0x41c | Keyfob Operation: Close Door |
| MINOR_REMOTE_CONTROL_OPEN_DOOR | 0x41d | Keyfob Operation: Open Door |
| MINOR_REMOTE_CONTROL_ALWAYS_OPEN_DOOR | 0x41e | Keyfob Operation: Remain Door Open |
| MINOR_M1_CARD_ENCRYPT_VERIFY_OPEN | 0x41f | M1 Card Encryption Verification Enabled |
| MINOR_M1_CARD_ENCRYPT_VERIFY_CLOSE | 0x420 | M1 Card Encryption Verification Disabled |

| Alarm Minor Types | Value | Description |
|---|---|---|
| MINOR_NFC_FUNCTION_OPEN | 0X421 | Opening Door with NFC Card Enabled |
| MINOR_NFC_FUNCTION_CLOSE | 0X422 | Opening Door with NFC Card Disabled |
| MINOR_OFFLINE_DATA_OUTPUT | 0x423 | Export Offline Collected Data |
| MINOR_CREATE_SSH_LINK | 0x42d | Establish SSH Connection |
| MINOR_CLOSE_SSH_LINK | 0x42e | Disconnect SSH Connection |
| MINOR_BLUETOOTH_KEY_MODIFY | / | Bluetooth Key Modified |
| MINOR_OPERATION_CUSTOM1 to MINOR_OPERATION_CUSTOM64 | 0x900-0x93f | Access Control: Custom Operation Event 1 to Custom Operation Event 64 |

### MAJOR_EVENT

| Event Minor Types | Value | Description |
|---|---|---|
| MINOR_LEGAL_CARD_PASS | 0x01 | Valid Card Authentication Completed |
| MINOR_CARD_AND_PSW_PASS | 0x02 | Card and Password Authentication Completed |
| MINOR_CARD_AND_PSW_FAIL | 0x03 | Card and Password Authentication Failed |
| MINOR_CARD_AND_PSW_TIMEOUT | 0x04 | Card and Password Authentication Timed Out |
| MINOR_CARD_AND_PSW_OVER_TIME | 0x05 | Card and Password Authentication Timed Out |
| MINOR_CARD_NO_RIGHT | 0x06 | No Permission |
| MINOR_CARD_INVALID_PERIOD | 0x07 | Invalid Card Swiping Time Period |
| MINOR_CARD_OUT_OF_DATE | 0x08 | Expired Card |
| MINOR_INVALID_CARD | 0x09 | Card No. Not Exist |

| Event Minor Types | Value | Description |
|---|---|---|
| MINOR_ANTI_SNEAK_FAIL | 0x0a | Anti-passing Back Authentication Failed |
| MINOR_INTERLOCK_DOOR_NOT_CLOSE | 0x0b | Interlocking Door Not Closed |
| MINOR_NOT_BELONG_MULTI_GROUP | 0x0c | Card Not in Multiple Authentication Group |
| MINOR_INVALID_MULTI_VERIFY_PERIOD | 0x0d | Card Not in Multiple Authentication Duration |
| MINOR_MULTI_VERIFY_SUPER_RIGHT_FAIL | 0x0e | Multiple Authentications: Super Password Authentication Failed |
| MINOR_MULTI_VERIFY_REMOTE_RIGHT_FAIL | 0x0f | Multiple Authentication Completed |
| MINOR_MULTI_VERIFY_SUCCESS | 0x10 | Multiple Authenticated |
| MINOR_LEADER_CARD_OPEN_BEGIN | 0x11 | Open Door with First Card Started |
| MINOR_LEADER_CARD_OPEN_END | 0x12 | Open Door with First Card Stopped |
| MINOR_ALWAYS_OPEN_BEGIN | 0x13 | Remain Open Started |
| MINOR_ALWAYS_OPEN_END | 0x14 | Remain Open Stopped |
| MINOR_LOCK_OPEN | 0x15 | Door Unlocked |
| MINOR_LOCK_CLOSE | 0x16 | Door Locked |
| MINOR_DOOR_BUTTON_PRESS | 0x17 | Exit Button Pressed |
| MINOR_DOOR_BUTTON_RELEASE | 0x18 | Exit Button Released |
| MINOR_DOOR_OPEN_NORMAL | 0x19 | Door Open (Contact) |
| MINOR_DOOR_CLOSE_NORMAL | 0x1a | Door Closed (Contact) |
| MINOR_DOOR_OPEN_ABNORMAL | 0x1b | Door Abnormally Open (Contact) |

| Event Minor Types | Value | Description |
|---|---|---|
| MINOR_DOOR_OPEN_TIMEOUT | 0x1c | Door Open Timed Out (Contact) |
| MINOR_ALARMOUT_ON | 0x1d | Alarm Output Enabled |
| MINOR_ALARMOUT_OFF | 0x1e | Alarm Output Disabled |
| MINOR_ALWAYS_CLOSE_BEGIN | 0x1f | Remain Closed Started |
| MINOR_ALWAYS_CLOSE_END | 0x20 | Remain Closed Stopped |
| MINOR_MULTI_VERIFY_NEED_REMOTE_OPEN | 0x21 | Multiple Authentications: Remotely Open Door |
| MINOR_MULTI_VERIFY_SUPERPASSWD_VERIFY_SUCCESS | 0x22 | Multiple Authentications: Super Password Authentication Completed |
| MINOR_MULTI_VERIFY_REPEAT_VERIFY | 0x23 | Multiple Authentications: Repeated Authentication |
| MINOR_MULTI_VERIFY_TIMEOUT | 0x24 | Multiple Authentications Timed Out |
| MINOR_DOORBELL_RINGING | 0x25 | Doorbell Ring |
| MINOR_FINGERPRINT_COMPARE_PASS | 0x26 | Fingerprint Matched |
| MINOR_FINGERPRINT_COMPARE_FAIL | 0x27 | Fingerprint Mismatched |
| MINOR_CARD_FINGERPRINT_VERIFY_PASS | 0x28 | Card and Fingerprint Authentication Completed |
| MINOR_CARD_FINGERPRINT_VERIFY_FAIL | 0x29 | Card and Fingerprint Authentication Failed |
| MINOR_CARD_FINGERPRINT_VERIFY_TIMEOUT | 0x2a | Card and Fingerprint Authentication Timed Out |
| MINOR_CARD_FINGERPRINT_PASSWD_VERIFY_PASS | 0x2b | Card and Fingerprint and Password Authentication Completed |
| MINOR_CARD_FINGERPRINT_PASSWD_VERIFY_FAIL | 0x2c | Card and Fingerprint and Password Authentication Failed |

| Event Minor Types | Value | Description |
|---|---|---|
| MINOR_CARD_FINGERPRINT_PASSWD_VERIFY_TIMEOUT | 0x2d | Card and Fingerprint and Password Authentication Timed Out |
| MINOR_FINGERPRINT_PASSWD_VERIFY_PASS | 0x2e | Fingerprint and Password Authentication Completed |
| MINOR_FINGERPRINT_PASSWD_VERIFY_FAIL | 0x2f | Fingerprint and Password Authentication Failed |
| MINOR_FINGERPRINT_PASSWD_VERIFY_TIMEOUT | 0x30 | Fingerprint and Password Authentication Timed Out |
| MINOR_FINGERPRINT_INEXISTENCE | 0x31 | Fingerprint Not Exists |
| MINOR_CARD_PLATFORM_VERIFY | 0x32 | Card Platform Authentication |
| MINOR_CALL_CENTER | 0x33 | Call Center |
| MINOR_FIRE_RELAY_TURN_ON_DOOR_ALWAYS_OPEN | 0x34 | Fire Relay Closed: Door Remains Open |
| MINOR_FIRE_RELAY_RECOVER_DOOR_RECOVER_NORMAL | 0x35 | Fire Relay Opened: Door Remains Closed |
| MINOR_EMPLOYEENO_AND_FP_VERIFY_PASS | 0x45 | Employee ID and Fingerprint Authentication Completed |
| MINOR_EMPLOYEENO_AND_FP_VERIFY_FAIL | 0x46 | Employee ID and Fingerprint Authentication Failed |
| MINOR_EMPLOYEENO_AND_FP_VERIFY_TIMEOUT | 0x47 | Employee ID and Fingerprint Authentication Timed Out |
| MINOR_EMPLOYEENO_AND_FP_AND_PW_VERIFY_PASS | 0x48 | Employee ID and Fingerprint and Password Authentication Completed |
| MINOR_EMPLOYEENO_AND_FP_AND_PW_VERIFY_FAIL | 0x49 | Employee ID and Fingerprint and Password Authentication Failed |
| MINOR_EMPLOYEENO_AND_FP_AND_PW_VERIFY_TIMEOUT | 0x4a | Employee ID and Fingerprint and Password Authentication Timed Out |

| Event Minor Types | Value | Description |
|---|---|---|
| MINOR_FACE_VERIFY_PASS | 0x4b | Face Authentication Completed |
| MINOR_FACE_VERIFY_FAIL | 0x4c | Face Authentication Failed |
| MINOR_EMPLOYEENO_AND_FACE_VERIFY_PASS | 0x4d | Employee ID and Face Authentication Completed |
| MINOR_EMPLOYEENO_AND_FACE_VERIFY_FAIL | 0x4e | Employee ID and Face Authentication Failed |
| MINOR_EMPLOYEENO_AND_FACE_VERIFY_TIMEOUT | 0x4f | Employee ID and Face Authentication Timed Out |
| MINOR_FACE_RECOGNIZE_FAIL | 0x50 | Face Recognition Failed |
| MINOR_FIRSTCARD_AUTHORIZE_BEGIN | 0x51 | First Card Authorization Started |
| MINOR_FIRSTCARD_AUTHORIZE_END | 0x52 | First Card Authorization Ended |
| MINOR_DOORLOCK_INPUT_SHORT_CIRCUIT | 0x53 | Lock Input Short Circuit Attempts Alarm |
| MINOR_DOORLOCK_INPUT_BROKEN_CIRCUIT | 0x54 | Lock Input Open Circuit Attempts Alarm |
| MINOR_DOORLOCK_INPUT_EXCEPTION | 0x55 | Lock Input Exception Alarm |
| MINOR_DOORCONTACT_INPUT_SHORT_CIRCUIT | 0x56 | Contact Input Short Circuit Attempts Alarm |
| MINOR_DOORCONTACT_INPUT_BROKEN_CIRCUIT | 0x57 | Contact Input Open Circuit Attempts Alarm |
| MINOR_DOORCONTACT_INPUT_EXCEPTION | 0x58 | Contact Input Exception Alarm |
| MINOR_OPENBUTTON_INPUT_SHORT_CIRCUIT | 0x59 | Exit Button Input Short Circuit Attempts Alarm |
| MINOR_OPENBUTTON_INPUT_BROKEN_CIRCUIT | 0x5a | Exit Button Input Open Circuit Attempts Alarm |
| MINOR_OPENBUTTON_INPUT_EXCEPTION | 0x5b | Exit Button Input Exception Alarm |
| MINOR_DOORLOCK_OPEN_EXCEPTION | 0x5c | Unlocking Exception |

| Event Minor Types | Value | Description |
|---|---|---|
| MINOR_DOORLOCK_OPEN_TIMEOUT | 0x5d | Unlocking Timed Out |
| MINOR_FIRSTCARD_OPEN_WITHOUT_AUTHORIZE | 0x5e | Unauthorized First Card Opening Failed |
| MINOR_CALL_LADDER_RELAY_BREAK | 0x5f | Call Elevator Relay Open |
| MINOR_CALL_LADDER_RELAY_CLOSE | 0x60 | Call Elevator Relay Closed |
| MINOR_AUTO_KEY_RELAY_BREAK | 0x61 | Auto Button Relay Open |
| MINOR_AUTO_KEY_RELAY_CLOSE | 0x62 | Auto Button Relay Closed |
| MINOR_KEY_CONTROL_RELAY_BREAK | 0x63 | Button Relay Open |
| MINOR_KEY_CONTROL_RELAY_CLOSE | 0x64 | Button Relay Closed |
| MINOR_EMPLOYEENO_AND_PW_PASS | 0x65 | Employee ID and Password Authentication Completed |
| MINOR_EMPLOYEENO_AND_PW_FAIL | 0x66 | Employee ID and Password Authentication Failed |
| MINOR_EMPLOYEENO_AND_PW_TIMEOUT | 0x67 | Employee ID and Password Authentication Timed Out |
| MINOR_HUMAN_DETECT_FAIL | 0x68 | Human Detection Failed |
| MINOR_PEOPLE_AND_ID_CARD_COMPARE_PASS | 0x69 | Person and ID Card Matched |
| MINOR_PEOPLE_AND_ID_CARD_COMPARE_FAIL | 0x70 | Person and ID Card Mismatched |
| MINOR_CERTIFICATE_BLOCKLIST | 0x71 | Blocklist Event |
| MINOR_LEGAL_MESSAGE | 0x72 | Valid Message |
| MINOR_ILLEGAL_MESSAGE | 0x73 | Invalid Message |

| Event Minor Types | Value | Description |
|---|---|---|
| MINOR_DOOR_OPEN_OR_ DORMANT_FAIL | 0x75 | Authentication Failed: Door Remain Closed or Door in Sleeping Mode |
| MINOR_AUTH_PLAN_ DORMANT_FAIL | 0x76 | Authentication Failed: Authentication Schedule in Sleeping Mode |
| MINOR_CARD_ENCRYPT_ VERIFY_FAIL | 0x77 | Card Encryption Verification Failed |
| MINOR_SUBMARINEBACK_ REPLY_FAIL | 0x78 | Anti-passing Back Server Response Failed |
| MINOR_DOOR_OPEN_OR_ DORMANT_OPEN_FAIL | 0x82 | Open Door via Exit Button Failed When Door Remain Closed or in Sleeping Mode |
| MINOR_DOOR_OPEN_OR_ DORMANT_LINKAGE_OPEN_ FAIL | 0x84 | Door Linkage Open Failed During Door Remain Close or Sleeping |
| MINOR_TRAILING | 0x85 | Tailgating |
| MINOR_REVERSE_ACCESS | 0x86 | Reverse Passing |
| MINOR_FORCE_ACCESS | 0x87 | Force Accessing |
| MINOR_CLIMBING_OVER_ GATE | 0x88 | Climb Over |
| MINOR_PASSING_TIMEOUT | 0x89 | Passing Timed Out |
| MINOR_INTRUSION_ALARM | 0x8a | Intrusion Alarm |
| MINOR_FREE_GATE_PASS_ NOT_AUTH | 0x8b | Authentication Failed When Free Passing |
| MINOR_DROP_ARM_BLOCK | 0x8c | Barrier Obstructed |
| MINOR_DROP_ARM_BLOCK_ RESUME | 0x8d | Barrier Restored |
| MINOR_PASSWORD_ MISMATCH | 0x97 | Passwords Mismatched |
| MINOR_EMPLOYEE_NO_NOT_ EXIST | 0x98 | Employee ID Not Exists |

| Event Minor Types | Value | Description |
|---|---|---|
| MINOR_COMBINED_VERIFY_PASS | 0x99 | Combined Authentication Completed |
| MINOR_COMBINED_VERIFY_TIMEOUT | 0x9a | Combined Authentication Timed Out |
| MINOR_VERIFY_MODE_MISMATCH | 0x9b | Authentication Type Mismatched |
| MINOR_BLUETOOTH_VERIFY_PASS | 0x9f | Authenticated via Bluetooth |
| MINOR_BLUETOOTH_VERIFY_FAIL | 0xa0 | Authentication via Bluetooth Failed |
| MINOR_INFORMAL_M1_CARD_VERIFY_FAIL | 0xa2 | Authentication Failed: Invalid M1 Card |
| MINOR_CPU_CARD_ENCRYPT_VERIFY_FAIL | 0xa3 | Verifying CPU Card Encryption Failed |
| MINOR_NFC_DISABLE_VERIFY_FAIL | 0xa4 | Disabling NFC Verification Failed |
| MINOR_EM_CARD_RECOGNIZE_NOT_ENABLED | 0xa8 | EM Card Recognition Disabled |
| MINOR_M1_CARD_RECOGNIZE_NOT_ENABLED | 0xa9 | M1 Card Recognition Disabled |
| MINOR_CPU_CARD_RECOGNIZE_NOT_ENABLED | 0xaa | CPU Card Recognition Disabled |
| MINOR_ID_CARD_RECOGNIZE_NOT_ENABLED | 0xab | ID Card Recognition Disabled |
| MINOR_CARD_SET_SECRET_KEY_FAIL | 0xac | Importing Key to Card Failed |
| MINOR_LOCAL_UPGRADE_FAIL | 0xad | Local Upgrade Failed |
| MINOR_REMOTE_UPGRADE_FAIL | 0xae | Remote Upgrade Failed |
| MINOR_REMOTE_EXTEND_MODULE_UPGRADE_SUCC | 0xaf | Extension Module is Remotely Upgraded |
| MINOR_REMOTE_EXTEND_MODULE_UPGRADE_FAIL | 0xb0 | Upgrading Extension Module Remotely Failed |

| Event Minor Types | Value | Description |
|---|---|---|
| MINOR_REMOTE_FINGER_ PRINT_MODULE_UPGRADE_ SUCC | 0xb1 | Fingerprint Module is Remotely Upgraded |
| MINOR_REMOTE_FINGER_ PRINT_MODULE_UPGRADE_ FAIL | 0xb2 | Upgrading Fingerprint Module Remotely Failed |
| MINOR_DYNAMICCODE_ VERIFY_PASS | 0xb3 | Dynamic Verification Code Authenticated |
| MINOR_DYNAMICCODE_ VERIFY_FAIL | 0xb4 | Authentication with Verification Code Failed |
| MINOR_PASSWD_VERIFY_PASS | 0xb5 | Password Authenticated |
| MINOR_FULL_STAFF | 0xc1 | Number of People Exceeds 90% of Capacity |
| MINOR_BLUETOOTH_KEY_ VERIFY_FAIL | / | Verifying Bluetooth Key Failed |
| MINOR_EVENT_CUSTOM1 to MINOR_EVENT_CUSTOM64 | 0x500 to 0x53f | Access Control: Custom Event 1 to Custom Event 64 |

## C.2 Event Linkage Types

For event card linkages, if the linkage type is event, four major event linkage types are available: 0-device event linkage, 1-alarm input event linkage, 2-access control point (e.g., doors, elevators, etc.) event linkage, and 3-authentication unit (e.g., card reader, fingerprint module, etc.) event linkage. Each major event linkage type corresponds multiple minor types of event linkage, see details in the following content.

### Device Event Linkage

| Minor Type | Value | Description |
|---|---|---|
| EVENT_ACS_HOST_ANTI_ DISMANTLE | 0 | Access Controller Tampering Alarm |
| EVENT_ACS_OFFLINE_ECENT_ NEARLY_FULL | 1 | No Memory Alarm |
| EVENT_ACS_NET_BROKEN | 2 | Network Disconnected |

| Minor Type | Value | Description |
|---|---|---|
| EVENT_ACS_NET_RESUME | 3 | Network Connected |
| EVENT_ACS_LOW_BATTERY | 4 | Low Battery Voltage |
| EVENT_ACS_BATTERY_RESUME | 5 | Battery Fully Charged |
| EVENT_ACS_AC_OFF | 6 | AC Power Off |
| EVENT_ACS_AC_RESUME | 7 | AC Power On |
| EVENT_ACS_SD_CARD_FULL | 8 | SD Card Full Alarm |
| EVENT_ACS_LINKAGE_ CAPTURE_PIC | 9 | Capture Linkage Event Alarm |
| EVENT_ACS_IMAGE_QUALITY_ LOW | 10 | Low Face Picture Quality |
| EVENT_ACS_FINGER_PRINT_ QUALITY_LOW | 11 | Low Fingerprint Picture Quality |
| EVENT_ACS_BATTERY_ ELECTRIC_LOW | 12 | Low Battery Voltage |
| EVENT_ACS_BATTERY_ ELECTRIC_RESUME | 13 | Battery Fully Charged |
| EVENT_ACS_FIRE_IMPORT_ SHORT_CIRCUIT | 14 | Fire Input Short Circuit Attempts Alarm |
| EVENT_ACS_FIRE_IMPORT_ BROKEN_CIRCUIT | 15 | Fire Input Open Circuit Attempts Alarm |
| EVENT_ACS_FIRE_IMPORT_ RESUME | 16 | Fire Input Alarm Restored |
| EVENT_ACS_MASTER_RS485_ LOOPNODE_BROKEN | 17 | RS485 Loop of Main Access Controller Disconnected |
| EVENT_ACS_MASTER_RS485_ LOOPNODE_RESUME | 18 | RS485 Loop of Main Access Controller Connected |
| EVENT_ACS_LOCAL_CONTROL_ OFFLINE | 19 | Distributed Access Controller Offline |
| EVENT_ACS_LOCAL_CONTROL_ RESUME | 20 | Distributed Access Controller Online |

| Minor Type | Value | Description |
|---|---|---|
| EVENT_ACS_LOCAL_ DOWNSIDE_RS485_ LOOPNODE_BROKEN | 21 | Downstream RS485 Loop of Distributed Access Control Disconnected |
| EVENT_ACS_LOCAL_ DOWNSIDE_RS485_ LOOPNODE_RESUME | 22 | Downstream RS485 Loop of Distributed Access Control Connected |
| EVENT_ACS_DISTRACT_ CONTROLLER_ONLINE | 23 | Distributed Elevator Controller Online |
| EVENT_ACS_DISTRACT_ CONTROLLER_OFFLINE | 24 | Distributed Elevator Controller Offline |
| EVENT_ACS_FIRE_BUTTON_ TRIGGER | 25 | Fire Button Pressed |
| EVENT_ACS_FIRE_BUTTON_ RESUME | 26 | Fire Button Released |
| EVENT_ACS_MAINTENANCE_ BUTTON_TRIGGER | 27 | Maintenance Button Pressed |
| EVENT_ACS_MAINTENANCE_ BUTTON_RESUME | 28 | Maintenance Button Released |
| EVENT_ACS_EMERGENCY_ BUTTON_TRIGGER | 29 | Panic Button Pressed |
| EVENT_ACS_EMERGENCY_ BUTTON_RESUME | 30 | Panic Button Released |
| EVENT_ACS_ SUBMARINEBACK_COMM_ BREAK | 32 | Communication with Anti-passing Back Server Failed |
| EVENT_ACS_ SUBMARINEBACK_COMM_ RESUME | 33 | Communication with Anti-passing Back Server Restored |
| EVENT_ACS_REMOTE_ ACTUAL_GUARD | 34 | Remotely Armed |
| EVENT_ACS_REMOTE_ ACTUAL_UNGUARD | 35 | Remotely Disarmed |
| EVENT_ACS_MOTOR_SENSOR_ EXCEPTION | 36 | Motor or Sensor Exception |

| Minor Type | Value | Description |
|---|---|---|
| EVENT_ACS_CAN_BUS_EXCEPTION | 37 | CAN Bus Exception |
| EVENT_ACS_CAN_BUS_RESUME | 38 | CAN Bus Restored |
| EVENT_ACS_GATE_TEMPERATURE_OVERRUN | 39 | Too High Pedestal Temperature |
| EVENT_ACS_IR_EMITTER_EXCEPTION | 40 | Active Infrared Intrusion Detector Exception |
| EVENT_ACS_IR_EMITTER_RESUME | 41 | Active Infrared Intrusion Detector Restored |
| EVENT_ACS_LAMP_BOARD_COMM_EXCEPTION | 42 | Communication with Light Board Failed |
| EVENT_ACS_LAMP_BOARD_COMM_RESUME | 43 | Communication with Light Board Restored |
| EVENT_ACS_IR_ADAPTOR_BOARD_COMM_EXCEPTION | 44 | Communication with IR Adaptor Failed |
| EVENT_ACS_IR_ADAPTOR_BOARD_COMM_RESUME | 45 | Communication with IR Adaptor Restored |
| EVENT_ACS_CHANNEL_CONTROLLER_DESMANTLE_ALARM | 46 | Lane Controller Tampering Alarm |
| EVENT_ACS_CHANNEL_CONTROLLER_DESMANTLE_RESUME | 47 | Lane Controller Tampering Alarm Restored |
| EVENT_ACS_CHANNEL_CONTROLLER_FIRE_IMPORT_ALARM | 48 | Lane Controller Fire Input Alarm |
| EVENT_ACS_CHANNEL_CONTROLLER_FIRE_IMPORT_RESUME | 49 | Lane Controller Fire Input Alarm Restored |
| EVENT_ACS_STAY_EVENT | / | Staying Event |
| EVENT_ACS_LEGAL_EVENT_NEARLY_FULL | / | No Memory Alarm for Valid Offline Event Storage |

## Alarm Input Event Linkage

| Minor Type | Value | Description |
|---|---|---|
| EVENT_ACS_ALARMIN_ SHORT_CIRCUIT | 0 | Zone Short Circuit Attempts Alarm |
| EVENT_ACS_ALARMIN_ BROKEN_CIRCUIT | 1 | Zone Open Circuit Attempts Alarm |
| EVENT_ACS_ALARMIN_ EXCEPTION | 2 | Zone Exception Alarm |
| EVENT_ACS_ALARMIN_ RESUME | 3 | Zone Alarm Restored |
| EVENT_ACS_CASE_SENSOR_ ALARM | 4 | Alarm Input Alarm |
| EVENT_ACS_CASE_SENSOR_ RESUME | 5 | Alarm Input Alarm Restored |

## Access Control Point Event Linkage

| Minor Type | Value | Description |
|---|---|---|
| EVENT_ACS_LEADER_CARD_ OPEN_BEGIN | 0 | Open Door with First Card Started |
| EVENT_ACS_LEADER_CARD_ OPEN_END | 1 | Open Door with First Card Ended |
| EVENT_ACS_ALWAYS_OPEN_ BEGIN | 2 | Remain Open Started |
| EVENT_ACS_ALWAYS_OPEN_ END | 3 | Remain Open Ended |
| EVENT_ACS_ALWAYS_CLOSE_ BEGIN | 4 | Remain Closed Started |
| EVENT_ACS_ALWAYS_CLOSE_ END | 5 | Remain Closed Ended |
| EVENT_ACS_LOCK_OPEN | 6 | Door Unlocked |
| EVENT_ACS_LOCK_CLOSE | 7 | Door Locked |
| EVENT_ACS_DOOR_BUTTON_ PRESS | 8 | Exit Button Pressed |

| Minor Type | Value | Description |
|---|---|---|
| EVENT_ACS_DOOR_BUTTON_ RELEASE | 9 | Exit Button Released |
| EVENT_ACS_DOOR_OPEN_ NORMAL | 10 | Door Open (Contact) |
| EVENT_ACS_DOOR_CLOSE_ NORMAL | 11 | Door Closed (Contact) |
| EVENT_ACS_DOOR_OPEN_ ABNORMAL | 12 | Door Abnormally Open (Contact) |
| EVENT_ACS_DOOR_OPEN_ TIMEOUT | 13 | Door Open Timed Out (Contact) |
| EVENT_ACS_REMOTE_OPEN_ DOOR | 14 | Door Remotely Open |
| EVENT_ACS_REMOTE_CLOSE_ DOOR | 15 | Door Remotely Closed |
| EVENT_ACS_REMOTE_ ALWAYS_OPEN | 16 | Remain Open Remotely |
| EVENT_ACS_REMOTE_ ALWAYS_CLOSE | 17 | Remain Closed Remotely |
| EVENT_ACS_NOT_BELONG_ MULTI_GROUP | 18 | Card Not in Multiple Authentication Group |
| EVENT_ACS_INVALID_MULTI_ VERIFY_PERIOD | 19 | Card Not in Multiple Authentication Duration |
| EVENT_ACS_MULTI_VERIFY_ SUPER_RIGHT_FAIL | 20 | Multiple Authentication Mode: Super Password Authentication Failed |
| EVENT_ACS_MULTI_VERIFY_ REMOTE_RIGHT_FAIL | 21 | Multiple Authentication Mode: Remote Authentication Failed |
| EVENT_ACS_MULTI_VERIFY_ SUCCESS | 22 | Multiple Authentication Completed |
| EVENT_ACS_MULTI_VERIFY_ NEED_REMOTE_OPEN | 23 | Multiple Authentication: Remotely Open Door |
| EVENT_ACS_MULTI_VERIFY_ SUPERPASSWD_VERIFY_ SUCCESS | 24 | Multiple Authentication: Super Password Authentication Completed |

| Minor Type | Value | Description |
|---|---|---|
| EVENT_ACS_MULTI_VERIFY_ REPEAT_VERIFY_FAIL | 25 | Multiple Authentication: Repeated Authentication Failed |
| EVENT_ACS_MULTI_VERIFY_ TIMEOUT | 26 | Multiple Authentication Timed Out |
| EVENT_ACS_REMOTE_ CAPTURE_PIC | 27 | Remote Capture |
| EVENT_ACS_DOORBELL_ RINGING | 28 | Doorbell Ring |
| EVENT_ACS_SECURITY_ MODULE_DESMANTLE_ALARM | 29 | Secure Door Control Unit Tampering Alarm |
| EVENT_ACS_CALL_CENTER | 30 | Center Event |
| EVENT_ACS_FIRSTCARD_ AUTHORIZE_BEGIN | 31 | First Card Authentication Started |
| EVENT_ACS_FIRSTCARD_ AUTHORIZE_END | 32 | First Card Authentication End |
| EVENT_ACS_DOORLOCK_ INPUT_SHORT_CIRCUIT | 33 | Lock Input Short Circuit Attempts Alarm |
| EVENT_ACS_DOORLOCK_ INPUT_BROKEN_CIRCUIT | 34 | Lock Input Open Circuit Attempts Alarm |
| EVENT_ACS_DOORLOCK_ INPUT_EXCEPTION | 35 | Lock Input Exception Alarm |
| EVENT_ACS_DOORCONTACT_ INPUT_SHORT_CIRCUIT | 36 | Contact Input Short Circuit Attempts Alarm |
| EVENT_ACS_DOORCONTACT_ INPUT_BROKEN_CIRCUIT | 37 | Contact Input Open Circuit Attempts Alarm |
| EVENT_ACS_DOORCONTACT_ INPUT_EXCEPTION | 38 | Contact Input Exception Alarm |
| EVENT_ACS_OPENBUTTON_ INPUT_SHORT_CIRCUIT | 39 | Exit Button Input Short Circuit Attempts Alarm |
| EVENT_ACS_OPENBUTTON_ INPUT_BROKEN_CIRCUIT | 40 | Exit Button Input Open Circuit Attempts Alarm |
| EVENT_ACS_OPENBUTTON_ INPUT_EXCEPTION | 41 | Exit Button Input Exception Alarm |

| Minor Type | Value | Description |
|---|---|---|
| EVENT_ACS_DOORLOCK_ OPEN_EXCEPTION | 42 | Unlocking Exception |
| EVENT_ACS_DOORLOCK_ OPEN_TIMEOUT | 43 | Unlocking Timed Out |
| EVENT_ACS_FIRSTCARD_ OPEN_WITHOUT_AUTHORIZE | 44 | Unauthorized First Card Opening Failed |
| EVENT_ACS_CALL_LADDER_ RELAY_BREAK | 45 | Call Elevator Relay Open |
| EVENT_ACS_CALL_LADDER_ RELAY_CLOSE | 46 | Call Elevator Relay Closed |
| EVENT_ACS_AUTO_KEY_ RELAY_BREAK | 47 | Auto Button Relay Open |
| EVENT_ACS_AUTO_KEY_ RELAY_CLOSE | 48 | Auto Button Relay Closed |
| EVENT_ACS_KEY_CONTROL_ RELAY_BREAK | 49 | Button Relay Open |
| EVENT_ACS_KEY_CONTROL_ RELAY_CLOSE | 50 | Button Relay Closed |
| EVENT_ACS_REMOTE_ VISITOR_CALL_LADDER | 51 | Visitor Calling Elevator |
| EVENT_ACS_REMOTE_ HOUSEHOLD_CALL_LADDER | 52 | Resident Calling Elevator |
| EVENT_ACS_LEGAL_MESSAGE | 52 | Valid Message |
| EVENT_ACS_ILLEGAL_ MESSAGE | 53 | Invalid Message |
| EVENT_ACS_TRAILING | 54 | Tailgating |
| EVENT_ACS_REVERSE_ACCESS | 55 | Reverse Passing |
| EVENT_ACS_FORCE_ACCESS | 56 | Force Collision |
| EVENT_ACS_CLIMBING_OVER_ GATE | 57 | Climbing Over |
| EVENT_ACS_PASSING_ TIMEOUT | 58 | Passing Timed Out |

| Minor Type | Value | Description |
|---|---|---|
| EVENT_ACS_INTRUSION_ALARM | 59 | Intrusion Alarm |
| EVENT_ACS_FREE_GATE_PASS_NOT_AUTH | 60 | Authentication Failed When Free Passing |
| EVENT_ACS_DROP_ARM_BLOCK | 61 | Barrier Obstructed |
| EVENT_ACS_DROP_ARM_BLOCK_RESUME | 62 | Barrier Restored |
| EVENT_ACS_REMOTE_CONTROL_CLOSE_DOOR | 63 | Door Closed via Keyfob |
| EVENT_ACS_REMOTE_CONTROL_OPEN_DOOR | 64 | Door Opened via Keyfob |
| EVENT_ACS_REMOTE_CONTROL_ALWAYS_OPEN_DOOR | 65 | Remain Open via Keyfob |

### Authentication Unit Event Linkage

| Minor Type | Value | Description |
|---|---|---|
| EVENT_ACS_STRESS_ALARM | 0 | Duress Alarm |
| EVENT_ACS_CARD_READER_DESMANTLE_ALARM | 1 | Card Reader Tampering Alarm |
| EVENT_ACS_LEGAL_CARD_PASS | 2 | Valid Card Authentication Completed |
| EVENT_ACS_CARD_AND_PSW_PASS | 3 | Card and Password Authentication Completed |
| EVENT_ACS_CARD_AND_PSW_FAIL | 4 | Card and Password Authentication Failed |
| EVENT_ACS_CARD_AND_PSW_TIMEOUT | 5 | Card and Password Authentication Timed Out |
| EVENT_ACS_CARD_MAX_AUTHENTICATE_FAIL | 6 | Card Authentication Attempts Reach Limit |
| EVENT_ACS_CARD_NO_RIGHT | 7 | No Permission for Card |

| Minor Type | Value | Description |
|---|---|---|
| EVENT_ACS_CARD_INVALID_PERIOD | 8 | Invalid Card Swiping Time Period |
| EVENT_ACS_CARD_OUT_OF_DATE | 9 | Expired Card |
| EVENT_ACS_INVALID_CARD | 10 | Card No. Not Exist |
| EVENT_ACS_ANTI_SNEAK_FAIL | 11 | Anti-passing Back Authentication Failed |
| EEVENT_ACS_INTERLOCK_DOOR_NOT_CLOSE | 12 | Interlocking Door Not Closed |
| EVENT_ACS_FINGERPRINT_COMPARE_PASS | 13 | Fingerprint Matched |
| EVENT_ACS_FINGERPRINT_COMPARE_FAIL | 14 | Fingerprint Mismatched |
| EVENT_ACS_CARD_FINGERPRINT_VERIFY_PASS | 15 | Card and Fingerprint Authentication Completed |
| EVENT_ACS_CARD_FINGERPRINT_VERIFY_FAIL | 16 | Card and Fingerprint Authentication Failed |
| EVENT_ACS_CARD_FINGERPRINT_VERIFY_TIMEOUT | 17 | Card and Fingerprint Authentication Timed Out |
| EVENT_ACS_CARD_FINGERPRINT_PASSWD_VERIFY_PASS | 18 | Card, Fingerprint, and Password Authentication Completed |
| EVENT_ACS_CARD_FINGERPRINT_PASSWD_VERIFY_FAIL | 19 | Card and Fingerprint Authentication Failed |
| EVENT_ACS_CARD_FINGERPRINT_PASSWD_VERIFY_TIMEOUT | 20 | Card and Fingerprint Authentication Timed Out |
| EVENT_ACS_FINGERPRINT_PASSWD_VERIFY_PASS | 21 | Fingerprint and Password Authentication Completed |
| EVENT_ACS_FINGERPRINT_PASSWD_VERIFY_FAIL | 22 | Fingerprint and Password Authentication Failed |

| Minor Type | Value | Description |
|---|---|---|
| EVENT_ACS_FINGERPRINT_PASSWD_VERIFY_TIMEOUT | 23 | Fingerprint and Password Authentication Timed Out |
| EVENT_ACS_FINGERPRINT_INEXISTENCE | 24 | Fingerprint Not Exist |
| EVENT_ACS_EMPLOYEENO_AND_FP_VERIFY_PASS | 42 | Employee ID and Fingerprint Authentication Completed |
| EVENT_ACS_EMPLOYEENO_AND_FP_VERIFY_FAIL | 43 | Employee ID and Fingerprint Authentication Failed |
| EVENT_ACS_EMPLOYEENO_AND_FP_VERIFY_TIMEOUT | 44 | Employee ID and Fingerprint Authentication Timed Out |
| EVENT_ACS_EMPLOYEENO_AND_FP_AND_PW_VERIFY_PASS | 45 | Employee ID, Fingerprint, and Password Authentication Completed |
| EVENT_ACS_EMPLOYEENO_AND_FP_AND_PW_VERIFY_FAIL | 46 | Employee ID, Fingerprint, and Password Authentication Failed |
| EVENT_ACS_EMPLOYEENO_AND_FP_AND_PW_VERIFY_TIMEOUT | 47 | Employee ID, Fingerprint, and Password Authentication Timed Out |
| EVENT_ACS_EMPLOYEENO_AND_PW_PASS | 52 | Employee ID and Password Authentication Completed |
| EVENT_ACS_EMPLOYEENO_AND_PW_FAIL | 52 | Employee ID and Password Authentication Failed |
| EVENT_ACS_EMPLOYEENO_AND_PW_TIMEOUT | 53 | Employee ID and Password Authentication Timed Out |
| EVENT_ACS_DOOR_OPEN_OR_DORMANT_FAIL | 57 | Authentication Failed When Door Remain Closed or Door in Sleeping Mode |
| EVENT_ACS_AUTH_PLAN_DORMANT_FAIL | 58 | Authentication Failed When Authentication Schedule in Sleeping Mode |
| EVENT_ACS_CARD_ENCRYPT_VERIFY_FAIL | 59 | Card Encryption Verification Failed |

| Minor Type | Value | Description |
|---|---|---|
| EVENT_ACS_ SUBMARINEBACK_REPLY_FAIL | 60 | Anti-passing Back Server Response Failed |
| EVENT_ACS_PASSWORD_ MISMATCH | 61 | Password Mismatched |
| EVENT_ACS_EMPLOYEE_NO_ NOT_EXIST | 62 | Employee ID Not Exist |
| EVENT_ACS_COMBINED_ VERIFY_PASS | 63 | Combined Authentication Completed |
| EVENT_ACS_COMBINED_ VERIFY_TIMEOUT | 64 | Combined Authentication Timed Out |
| EVENT_ACS_VERIFY_MODE_ MISMATCH | 65 | Authentication Type Mismatched |
| EVENT_ACS_PSW_ERROR_ OVER_TIMES | 67 | Maximum Password Authentication Failure Attempts |
| EVENT_ACS_PSW_VERIFY_PASS | 68 | Password Authenticated |
| EVENT_ACS_PSW_VERIFY_FAIL | 69 | Password Authentication Failed |
| EVENT_ACS_ORCODE_VERIFY_ PASS | 70 | QR Code Authenticated |
| EVENT_ACS_ORCODE_VERIFY_ FAIL | 71 | QR Code Authentication Failed |
| EVENT_ACS_HOUSEHOLDER_ AUTHORIZE_PASS | 72 | Resident Authorization Authenticated |
| EVENT_ACS_BLUETOOTH_ VERIFY_PASS | 73 | Bluetooth Authenticated |
| EVENT_ACS_BLUETOOTH_ VERIFY_FAIL | 74 | Bluetooth Authentication Failed |
| EVENT_ACS_INFORMAL_M1_ CARD_VERIFY_FAIL | / | Authentication Failed: Invalid M1 Card |
| EVENT_ACS_CPU_CARD_ ENCRYPT_VERIFY_FAIL | / | Verifying CPU Card Encryption Failed |
| EVENT_ACS_NFC_DISABLE_ VERIFY_FAIL | / | Disabling NFC Verification Failed |

| Minor Type | Value | Description |
|---|---|---|
| EVENT_ACS_EM_CARD_ RECOGNIZE_NOT_ENABLED | / | EM Card Recognition Disabled |
| EVENT_ACS_M1_CARD_ RECOGNIZE_NOT_ENABLED | / | M1 Card Recognition Disabled |
| EVENT_ACS_CPU_CARD_ RECOGNIZE_NOT_ENABLED | / | CPU Card Recognition Disabled |
| EVENT_ACS_ID_CARD_ RECOGNIZE_NOT_ENABLED | / | ID Card Recognition Disabled |
| EVENT_ACS_CARD_SET_ SECRET_KEY_FAIL | / | Importing Key to Card Failed |

## C.3 Device Network SDK Errors

The errors that may occur during the device network SDK integration are listed here for reference. You can search for the error descriptions according to the error codes or names returned by a specific API (NET_DVR_GetLastError or NET_DVR_GetErrorMsg).

### General Errors

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_NOERROR | 0 | No error. |
| NET_DVR_PASSWORD_ERROR | 1 | Incorrect user name or password. |
| NET_DVR_NOENOUGHPRI | 2 | No permission. |
| NET_DVR_NOINIT | 3 | Uninitialized. |
| NET_DVR_CHANNEL_ERROR | 4 | Incorrect channel No. |
| NET_DVR_OVER_MAXLINK | 5 | No more device can be connected. |
| NET_DVR_VERSIONNOMATCH | 6 | Version mismatches. |
| NET_DVR_NETWORK_FAIL_CONNECT | 7 | Connecting to device failed. The device is offline or network connection timed out. |
| NET_DVR_NETWORK_SEND_ERROR | 8 | Sending data to device failed. |
| NET_DVR_NETWORK_RECV_ERROR | 9 | Receiving data from device failed. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_NETWORK_RECV_TIMEOUT | 10 | Receiving data from device timed out. |
| NET_DVR_NETWORK_ERRORDATA | 11 | The data sent to the device is illegal, or the data received from the device error. E.g. The input data is not supported by the device for remote configuration. |
| NET_DVR_ORDER_ERROR | 12 | API calling order error. |
| NET_DVR_OPERNOPERMIT | 13 | No permission for this operation. |
| NET_DVR_COMMANDTIMEOUT | 14 | Executing device command timed out. |
| NET_DVR_ERRORSERIALPORT | 15 | Incorrect serial port No. The specified serial port does not exist. |
| NET_DVR_ERRORALARMPORT | 16 | Alarm port No. error. The alarm input or output port of the specified device does not exist. |
| NET_DVR_PARAMETER_ERROR | 17 | Incorrect parameter. The input or output parameters of the SDK API is empty, or the parameter value or format is invalid. |
| NET_DVR_CHAN_EXCEPTION | 18 | Device channel is in exception status. |
| NET_DVR_NODISK | 19 | No HDD in the device. |
| NET_DVR_ERRORDISKNUM | 20 | Incorrect HDD No. |
| NET_DVR_DISK_FULL | 21 | HDD full. |
| NET_DVR_DISK_ERROR | 22 | HDD error. |
| NET_DVR_NOSUPPORT | 23 | Device does not support this function. |
| NET_DVR_BUSY | 24 | Device is busy. |
| NET_DVR_MODIFY_FAIL | 25 | Failed to edit device parameters. |
| NET_DVR_PASSWORD_FORMAT_ ERROR | 26 | Invalid password format. |
| NET_DVR_DISK_FORMATING | 27 | HDD is formatting. Failed to startup. |
| NET_DVR_DVRNORESOURCE | 28 | Insufficient device resources. |
| NET_DVR_DVROPRATEFAILED | 29 | Device operation failed. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_OPENHOSTSOUND_FAIL | 30 | Failed to collect local audio data or open audio output during two-way audio and broadcast. |
| NET_DVR_DVRVOICEOPENED | 31 | Two-way audio channel is occupied. |
| NET_DVR_TIMEINPUTERROR | 32 | Incorrect time input. |
| NET_DVR_NOSPECFILE | 33 | No video file for playback. |
| NET_DVR_CREATEFILE_ERROR | 34 | Failed to create a file during local recording, saving picture, getting configuration file or downloading video file remotely. |
| NET_DVR_FILEOPENFAIL | 35 | Failed to open a file. The file does not exist or directory error. |
| NET_DVR_OPERNOTFINISH | 36 | Operation conflicted. |
| NET_DVR_GETPLAYTIMEFAIL | 37 | Failed to get the current played time. |
| NET_DVR_PLAYFAIL | 38 | Failed to play. |
| NET_DVR_FILEFORMAT_ERROR | 39 | Invalid file format. |
| NET_DVR_DIR_ERROR | 40 | File directory error. |
| NET_DVR_ALLOC_RESOURCE_ERROR | 41 | Allocating resources failed. |
| NET_DVR_AUDIO_MODE_ERROR | 42 | Invalid sound card mode error. The opened sound play mode and configured mode mismatched. |
| NET_DVR_NOENOUGH_BUF | 43 | Insufficient buffer for receiving data or saving picture. |
| NET_DVR_CREATESOCKET_ERROR | 44 | Failed to create SOCKET. |
| NET_DVR_SETSOCKET_ERROR | 45 | Failed to set SOCKET. |
| NET_DVR_MAX_NUM | 46 | No more registrations and live views can be connected. |
| NET_DVR_USERNOTEXIST | 47 | The user doest not exist. The user ID is logged out or unavailable. |
| NET_DVR_WRITEFLASHERROR | 48 | Writing FLASH error during device upgrade. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_UPGRADEFAIL | 49 | Failed to upgrade device. Network problem or language mismatches. |
| NET_DVR_CARDHAVEINIT | 50 | The decoding card is already initialized. |
| NET_DVR_PLAYERFAILED | 51 | Failed to call the function of player SDK. |
| NET_DVR_MAX_USERNUM | 52 | No more users can log in to. |
| NET_DVR_GETLOCALIPANDMACFAIL | 53 | Failed to get the IP address or physical address of local PC. |
| NET_DVR_NOENCODEING | 54 | The decoding function of this channel is not enabled. |
| NET_DVR_IPMISMATCH | 55 | IP address mismatches. |
| NET_DVR_MACMISMATCH | 56 | MAC address mismatches. |
| NET_DVR_UPGRADELANGMISMATCH | 57 | The language of upgrade file mismatches. |
| NET_DVR_MAX_PLAYERPORT | 58 | No more channels can be started to play. |
| NET_DVR_NOSPACEBACKUP | 59 | Insufficient space to back up file. |
| NET_DVR_NODEVICEBACKUP | 60 | No backup device found. |
| NET_DVR_PICTURE_BITS_ERROR | 61 | Picture pixel bit mismatches. Only 24 bits are allowed. |
| NET_DVR_PICTURE_DIMENSION_ERROR | 62 | Too large picture. The height*width should be less than 128x256. |
| NET_DVR_PICTURE_SIZ_ERROR | 63 | Too large picture. The picture size should be smaller than 100K. |
| NET_DVR_LOADPLAYERSDKFAILED | 64 | Failed to load the player(PlayCtrl.dll, SuperRender.dll, AudioRender.dll) to the current directory. |
| NET_DVR_LOADPLAYERSDKPROC_ERROR | 65 | Failed to find the function in player SDK. |
| NET_DVR_LOADDSSDKFAILED | 66 | Failed to load the DS SDK to the current directory. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_LOADDSSDKPROC_ERROR | 67 | Failed to find the function in the DS SDK. |
| NET_DVR_DSSDK_ERROR | 68 | Failed to call the API in the hardware decoding library. |
| NET_DVR_VOICEMONOPOLIZE | 69 | The sound card is exclusive. |
| NET_DVR_JOINMULTICASTFAILED | 70 | Failed to join to multicast group. |
| NET_DVR_CREATEDIR_ERROR | 71 | Failed to create log file directory. |
| NET_DVR_BINDSOCKET_ERROR | 72 | Failed to bind socket. |
| NET_DVR_SOCKETCLOSE_ERROR | 73 | Socket disconnected. Network disconnected or the destination is unreachable. |
| NET_DVR_USERID_ISUSING | 74 | Operation is executing. Failed to log out. |
| NET_DVR_SOCKETLISTEN_ERROR | 75 | Failed to listen. |
| NET_DVR_PROGRAM_EXCEPTION | 76 | Program exception. |
| NET_DVR_WRITEFILE_FAILED | 77 | Failed to write file during local recording, downloading file remotely or saving picture. |
| NET_DVR_FORMAT_READONLY | 78 | The HDD is read-only. Formatting is forbidden. |
| NET_DVR_WITHSAMEUSERNAME | 79 | The user name already exists. |
| NET_DVR_DEVICETYPE_ERROR | 80 | Device model mismatches when importing parameters. |
| NET_DVR_LANGUAGE_ERROR | 81 | Language mismatches when importing parameters. |
| NET_DVR_PARAVERSION_ERROR | 82 | Software version mismatches when importing parameters. |
| NET_DVR_IPCHAN_NOTALIVE | 83 | The external IP channel is offline live view. |
| NET_DVR_RTSP_SDK_ERROR | 84 | Failed to load StreamTransClient.dll. |
| NET_DVR_CONVERT_SDK_ERROR | 85 | Failed to load SystemTransform.dll. |
| NET_DVR_IPC_COUNT_OVERFLOW | 86 | No more IP channels can access to. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_MAX_ADD_NUM | 87 | No more video tags can be added. |
| NET_DVR_PARAMMODE_ERROR | 88 | Invalid parameter mode of image enhancement. |
| NET_DVR_CODESPITTER_OFFLINE | 89 | Code distributer is offline. |
| NET_DVR_BACKUP_COPYING | 90 | Device is backing up. |
| NET_DVR_CHAN_NOTSUPPORT | 91 | This operation is not supported by the channel. |
| NET_DVR_CALLINEINVALID | 92 | The height line is too concentrated, or the length line is not inclined enough. |
| NET_DVR_CALCANCELCONFLICT | 93 | Cancel calibration conflict, if the rule and global actual size filter are configured. |
| NET_DVR_CALPOINTOUTRANGE | 94 | The calibration point is out of limitation. |
| NET_DVR_FILTERRECTINVALID | 95 | The size filter does not meet the requirement. |
| NET_DVR_DDNS_DEVOFFLINE | 96 | Device has not registered to DDNS. |
| NET_DVR_DDNS_INTER_ERROR | 97 | DDNS internal error. |
| NET_DVR_FUNCTION_NOT_SUPPORT_OS | 98 | This function is not supported by this Operating system. |
| NET_DVR_DEC_CHAN_REBIND | 99 | Decoding channel binding display output is limited. |
| NET_DVR_INTERCOM_SDK_ERROR | 100 | Failed to load the two-way audio SDK of the current directory. |
| NET_DVR_NO_CURRENT_UPDATEFILE | 101 | No correct upgrade packet. |
| NET_DVR_USER_NOT_SUCC_LOGIN | 102 | Login failed. |
| NET_DVR_USE_LOG_SWITCH_FILE | 103 | The log switch file is under using. |
| NET_DVR_POOL_PORT_EXHAUST | 104 | No port can be bound in the port pool. |
| NET_DVR_PACKET_TYPE_NOT_SUPPORT | 105 | Incorrect stream packaging format. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_IPPARA_IPID_ERROR | 106 | Incorrect IPID for IP access configuration. |
| NET_DVR_LOAD_HCPREVIEW_SDK_ERROR | 107 | Failed to load the live view component. |
| NET_DVR_LOAD_HCVOICETALK_SDK_ERROR | 108 | Failed to load the audio component. |
| NET_DVR_LOAD_HCALARM_SDK_ERROR | 109 | Failed to load the alarm component. |
| NET_DVR_LOAD_HCPLAYBACK_SDK_ERROR | 110 | Failed to load the playback component. |
| NET_DVR_LOAD_HCDISPLAY_SDK_ERROR | 111 | Failed to load the display component. |
| NET_DVR_LOAD_HCINDUSTRY_SDK_ERROR | 112 | Failed to load application component. |
| NET_DVR_LOAD_HCGENERALCFGMGR_SDK_ERROR | 113 | Failed to load the general configuration management component. |
| NET_DVR_CORE_VER_MISMATCH | 121 | Component version and core version mismatched when loading the component singly. |
| NET_DVR_CORE_VER_MISMATCH_HCPREVIEW | 122 | Live view component version and core version mismatched. |
| NET_DVR_CORE_VER_MISMATCH_HCVOICETALK | 123 | Audio component version and the core version mismatched. |
| NET_DVR_CORE_VER_MISMATCH_HCALARM | 124 | Alarm component version and the core version mismatched. |
| NET_DVR_CORE_VER_MISMATCH_HCPLAYBACK | 125 | Playback component version and the core version mismatched. |
| NET_DVR_CORE_VER_MISMATCH_HCDISPLAY | 126 | Display component version and the core version mismatched. |
| NET_DVR_CORE_VER_MISMATCH_HCINDUSTRY | 127 | Application component version and the core version mismatched. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_CORE_VER_MISMATCH_HCGENERALCFGMGR | 128 | General configuration management component version and the core version mismatched. |
| NET_DVR_COM_VER_MISMATCH_HCPREVIEW | 136 | Live view component version and SDK version mismatched. |
| NET_DVR_COM_VER_MISMATCH_HCVOICETALKy | 137 | Audio component version and SDK version mismatched. |
| NET_DVR_COM_VER_MISMATCH_HCALARM | 138 | Alarm component version and SDK version mismatched. |
| NET_DVR_COM_VER_MISMATCH_HCPLAYBACK | 139 | Playback component version and SDK version mismatched. |
| NET_DVR_COM_VER_MISMATCH_HCDISPLAY | 140 | Display component version and SDK version mismatched. |
| NET_DVR_COM_VER_MISMATCH_HCINDUSTRY | 141 | Application component version and SDK version mismatched. |
| NET_DVR_COM_VER_MISMATCH_HCGENERALCFGMGR | 142 | General configuration management component version and SDK version mismatched. |
| NET_DVR_ALIAS_DUPLICATE | 150 | Duplicated alias(for HiDDNS configuration). |
| NET_DVR_USERNAME_NOT_EXIST | 152 | User name does not exist (error code of network camera and network speed dome with version from 5.1.7 to 5.3.1). |
| NET_ERR_USERNAME_LOCKED | 153 | The user name is locked. |
| NET_DVR_INVALID_USERID | 154 | Invalid user ID. |
| NET_DVR_LOW_LOGIN_VERSION | 155 | The version is too low. |
| NET_DVR_LOAD_LIBEAY32_DLL_ERROR | 156 | Failed to load libeay32.dl.l |
| NET_DVR_LOAD_SSLEAY32_DLL_ERROR | 157 | Failed to load ssleay32.dll. |
| NET_ERR_LOAD_LIBICONV | 158 | Failed to load libiconv.dll. |
| NET_ERR_SSL_CONNECT_FAILED | 159 | Connecting to SSL failed. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_TEST_SERVER_FAIL_CONNECT | 165 | Failed to connect to test server. |
| NET_DVR_NAS_SERVER_INVALID_DIR | 166 | Failed to load NAS server to the directory, Invalid directory, or incorrect user name and password. |
| NET_DVR_NAS_SERVER_NOENOUGH_PRI | 167 | Failed to load NAS server th the directory. No permission. |
| NET_DVR_EMAIL_SERVER_NOT_CONFIG_DNS | 168 | The server uses domain name without configuring DNS, the domain name may be invalid. |
| NET_DVR_EMAIL_SERVER_NOT_CONFIG_GATEWAY | 169 | No gateway configured. Sending email may be failed. |
| NET_DVR_TEST_SERVER_PASSWORD_ERROR | 170 | Incorrect user name or password of test server. |
| NET_DVR_EMAIL_SERVER_CONNECT_EXCEPTION_WITH_SMTP | 171 | Interaction exception between device and SMTP server. |
| NET_DVR_FTP_SERVER_FAIL_CREATE_DIR | 172 | FTP server creating directory failed. |
| NET_DVR_FTP_SERVER_NO_WRITE_PIR | 173 | FTP server has no wirting permission. |
| NET_DVR_IP_CONFLICT | 174 | IP conflicted. |
| NET_DVR_INSUFFICIENT_STORAGEPOOL_SPACE | 175 | Storage pool space is full. |
| NET_DVR_STORAGEPOOL_INVALID | 176 | Invalid cloud storage pool. No storage pool configured or incorrect storage pool ID. |
| NET_DVR_EFFECTIVENESS_REBOOT | 177 | Restart to take effect. |
| NET_ERR_ANR_ARMING_EXIST | 178 | The ANR arming connection already exists( the error will be returned when arming with ANR function if the private SDK protocol arming connection is established). |
| NET_ERR_UPLOADLINK_EXIST | 179 | The ANR uploading connection already exists( the error will be |

| Error Name | Error Code | Error Description |
|---|---|---|
| | | returned when EHome protocol and private SDK protocol do not support ANR at the same time). |
| NET_ERR_INCORRECT_FILE_FORMAT | 180 | The imported file format is incorrect. |
| NET_ERR_INCORRECT_FILE_CONTENT | 181 | The imported file content is incorrect. |
| NET_ERR_MAX_HRUDP_LINK | 182 | No more HRUDP can be connected to device. |
| NET_ERR_MAX_PORT_MULTIPLEX | 183 | Maximum number of multiplexed ports reaches. |
| NET_ERR_CREATE_PORT_MULTIPLEX | 184 | Creating port multiplier failed. |
| NET_DVR_NONBLOCKING_CAPTURE_NOTSUPPORT | 185 | Non-blocking picture capture is not supported. |
| NET_SDK_ERR_FUNCTION_INVALID | 186 | Invalid function. The asynchronous mode is enabled. |
| NET_SDK_ERR_MAX_PORT_MULTIPLEX | 187 | Maximum number of multiplex ports reached. |
| NET_DVR_INVALID_LINK | 188 | Link has not been created or the link is invalid. |
| NET_DVR_NAME_NOT_ONLY | 200 | This name already exists. |
| NET_DVR_OVER_MAX_ARRAY | 201 | The number of RAID reaches the upper-limit. |
| NET_DVR_OVER_MAX_VD | 202 | The number of virtual disk reaches the upper-limit. |
| NET_DVR_VD_SLOT_EXCEED | 203 | The virtual disk slots are full. |
| NET_DVR_PD_STATUS_INVALID | 204 | The physical disk for rebuilding RAID is error. |
| NET_DVR_PD_BE_DEDICATE_SPARE | 205 | The physical disk for rebuilding RAID is specified as hot spare. |
| NET_DVR_PD_NOT_FREE | 206 | The physical disk for rebuilding RAID is busy. |
| NET_DVR_CANNOT_MIG2NEWMODE | 207 | Failed to migrate the current RAID type to the new type. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_MIG_PAUSE | 208 | Migration is paused. |
| NET_DVR_MIG_ABOUTED | 209 | Migration is cancelled. |
| NET_DVR_EXIST_VD | 210 | Failed to delete RAID. Virtual disk exists in the RAID. |
| NET_DVR_TARGET_IN_LD_ FUNCTIONAL | 211 | Target physical disk is a part of the virtual disk and it is working normally. |
| NET_DVR_HD_IS_ASSIGNED_ ALREADY | 212 | The specified physical disk is allocated as virtual disk. |
| NET_DVR_INVALID_HD_COUNT | 213 | The number of physical disks and specified RAID level mismatched. |
| NET_DVR_LD_IS_FUNCTIONAL | 214 | The RAID is normal. Failed to rebuild. |
| NET_DVR_BGA_RUNNING | 215 | Background task is executing. |
| NET_DVR_LD_NO_ATAPI | 216 | Failed to create virtual disk by ATAPI disk. |
| NET_DVR_MIGRATION_NOT_NEED | 217 | There is no need to migrate the RAID. |
| NET_DVR_HD_TYPE_MISMATCH | 218 | The physical disk type is not allowed. |
| NET_DVR_NO_LD_IN_DG | 219 | No virtual disk. Operation failed. |
| NET_DVR_NO_ROOM_FOR_SPARE | 220 | Insufficient disk space. Failed to allocate the disk as hot spare. |
| NET_DVR_SPARE_IS_IN_MULTI_DG | 221 | The disk is already allocated as the hot spare of one RAID. |
| NET_DVR_DG_HAS_MISSING_PD | 222 | No disk in the RAID. |
| NET_DVR_NAME_EMPTY | 223 | The name is empty. |
| NET_DVR_INPUT_PARAM | 224 | Incorrect input parameters. |
| NET_DVR_PD_NOT_AVAILABLE | 225 | The physical disk is not available. |
| NET_DVR_ARRAY_NOT_AVAILABLE | 226 | The RAID is not available. |
| NET_DVR_PD_COUNT | 227 | Incorrect number of physical disks. |
| NET_DVR_VD_SMALL | 228 | Insufficient virtual disk space. |
| NET_DVR_NO_EXIST | 229 | Not exist. |
| NET_DVR_NOT_SUPPORT | 230 | This operation is not supported. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_NOT_FUNCTIONAL | 231 | The RAID status is exception. |
| NET_DVR_DEV_NODE_NOT_FOUND | 232 | The device node of virtual disk does not exist. |
| NET_DVR_SLOT_EXCEED | 233 | No more slots are allowed. |
| NET_DVR_NO_VD_IN_ARRAY | 234 | No virtual disk exists in the RAID. |
| NET_DVR_VD_SLOT_INVALID | 235 | Invalid virtual disk slot. |
| NET_DVR_PD_NO_ENOUGH_SPACE | 236 | Insufficient physical disk space. |
| NET_DVR_ARRAY_NONFUNCTION | 237 | Only the RAID in normal status supports to be migrated. |
| NET_DVR_ARRAY_NO_ENOUGH_SPACE | 238 | Insufficient RAID space. |
| NET_DVR_STOPPING_SCANNING_ARRAY | 239 | Pulling disk out safely or rescanning. |
| NET_DVR_NOT_SUPPORT_16T | 240 | Creating RAID with size larger than 16T is not supported. |
| NET_DVR_ERROR_DEVICE_NOT_ACTIVATED | 250 | The device is not activated (login failed.) |
| NET_DVR_ERROR_RISK_PASSWORD | 251 | Risky password. |
| NET_DVR_ERROR_DEVICE_HAS_ACTIVATED | 252 | The device is already activated. |
| NET_DVR_ID_ERROR | 300 | The configured ID is invalid. |
| NET_DVR_POLYGON_ERROR | 301 | Invalid polygon shape. |
| NET_DVR_RULE_PARAM_ERROR | 302 | Invalid rule parameters. |
| NET_DVR_RULE_CFG_CONFLICT | 303 | Configured information conflicted. |
| NET_DVR_CALIBRATE_NOT_READY | 304 | No calibration information. |
| NET_DVR_CAMERA_DATA_ERROR | 305 | Invalid camera parameters. |
| NET_DVR_CALIBRATE_DATA_UNFIT | 306 | Invalid inclination angle for calibration. |
| NET_DVR_CALIBRATE_DATA_CONFILICT | 307 | Calibration error. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_CALIBRATE_CALC_FAIL | 308 | Failed to calculate calibration parameter values of camera. |
| NET_DVR_CALIBRATE_LINE_OUT_RECT | 309 | The inputted calibration line exceeds the external sample rectangle. |
| NET_DVR_ENTER_RULE_NOT_READY | 310 | No region entrance is configured. |
| NET_DVR_AID_RULE_NO_INCLUDE_LANE | 311 | No lane configured in the traffic event rull(especially for traffic jam or driving against the traffic). |
| NET_DVR_LANE_NOT_READY | 312 | Lane not configured. |
| NET_DVR_RULE_INCLUDE_TWO_WAY | 313 | Two different directions are contained in event rule. |
| NET_DVR_LANE_TPS_RULE_CONFLICT | 314 | Lane and data rule conflicted. |
| NET_DVR_NOT_SUPPORT_EVENT_TYPE | 315 | This event type is not supported. |
| NET_DVR_LANE_NO_WAY | 316 | The lane has no direction. |
| NET_DVR_SIZE_FILTER_ERROR | 317 | Invalid size of filter frame. |
| NET_DVR_LIB_FFL_NO_FACE | 318 | No face picture exists in the image inputted when positioning feature point. |
| NET_DVR_LIB_FFL_IMG_TOO_SMALL | 319 | The inputted image is too small when positioning feature point. |
| NET_DVR_LIB_FD_IMG_NO_FACE | 320 | No face picture exists in the image inputted when detecting single face picture. |
| NET_DVR_LIB_FACE_TOO_SMALL | 321 | Face picture is too small when building model. |
| NET_DVR_LIB_FACE_QUALITY_TOO_BAD | 322 | The face picture quality is too poor when building model. |
| NET_DVR_KEY_PARAM_ERR | 323 | The configured advanced parameter is incorrect. |
| NET_DVR_CALIBRATE_DATA_ERR | 324 | Calibration sample number error, or data value error, or the sample points are beyond the horizontal line. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_CALIBRATE_DISABLE_FAIL | 325 | Canceling calibration is not allowed for configured rules. |
| NET_DVR_VCA_LIB_FD_SCALE_ OUTRANGE | 326 | The minimum width and height of maximum filter frame are twice or more larger than the maximum width and height of minimum filter frame. |
| NET_DVR_LIB_FD_REGION_TOO_ LARGE | 327 | Too large detection region. The maximum region should be 2/3 of the image. |
| NET_DVR_TRIAL_OVERDUE | 328 | Trial period is ended. |
| NET_DVR_CONFIG_FILE_CONFLICT | 329 | Device type and configuration file conflicted. |
| NET_DVR_FR_FPL_FAIL | 330 | Failed to positioning face feature points. |
| NET_DVR_FR_IQA_FAIL | 331 | Failed to test face picture quality. |
| NET_DVR_FR_FEM_FAIL | 332 | Failed to extract the face feature points. |
| NET_DVR_FPL_DT_CONF_TOO_LOW | 333 | The face detection validity is too low when positioning face feature points. |
| NET_DVR_FPL_CONF_TOO_LOW | 334 | The validity of feature points positionong is too low. |
| NET_DVR_E_DATA_SIZE | 335 | Data size mismatches. |
| NET_DVR_FR_MODEL_VERSION_ERR | 336 | Incorrect model version in face model library. |
| NET_DVR_FR_FD_FAIL | 337 | Failed to detect face in the face recognition library. |
| NET_DVR_FA_NORMALIZE_ERR | 338 | Failed to normalize face attribute. |
| NET_DVR_DOG_PUSTREAM_NOT_ MATCH | 339 | Dongle type and camera type mismatched. |
| NET_DVR_DEV_PUSTREAM_NOT_ MATCH | 340 | Camera version mismatches. |
| NET_DVR_PUSTREAM_ALREADY_ EXISTS | 341 | This camera is already added to other channels of devices. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_SEARCH_CONNECT_FAILED | 342 | Failed to connect to face retrieval server. |
| NET_DVR_INSUFFICIENT_DISK_SPACE | 343 | Insufficient storage space. |
| NET_DVR_DATABASE_CONNECTION_FAILED | 344 | Failed to connect to database. |
| NET_DVR_DATABASE_ADM_PW_ERROR | 345 | Incorrect database user name and password. |
| NET_DVR_DECODE_YUV | 346 | Decoding failed. |
| NET_DVR_IMAGE_RESOLUTION_ERROR | 347 | Invalid picture resolution |
| NET_DVR_CHAN_WORKMODE_ERROR | 348 | Invalid channel working mode. |
| NET_ERROR_TRUNK_LINE | 711 | Sub system is configured as the trunk line. |
| NET_ERROR_MIXED_JOINT | 712 | Mixed joint is not supported. |
| NET_ERROR_DISPLAY_SWITCH | 713 | Switch of display channel is not supported. |
| NET_ERROR_USED_BY_BIG_SCREEN | 714 | Decoded resource is occupied by the big screen. |
| NET_ERROR_USE_OTHER_DEC_RESOURCE | 715 | Using resources of other sub system is not allowed. |
| NET_ERROR_SCENE_USING | 717 | The scene is being used. |
| NET_ERR_NO_ENOUGH_DEC_RESOURCE | 718 | Insufficient resources for decoding. |
| NET_ERR_NO_ENOUGH_FREE_SHOW_RESOURCE | 719 | Insufficient resources for display. |
| NET_ERR_NO_ENOUGH_VIDEO_MEMORY | 720 | Insufficient video storage resources. |
| NET_ERR_MAX _VIDEO_NUM | 721 | Insufficient resources for multiple channels. |
| NET_ERR_WINDOW_COVER_FREE_SHOW_AND_NORMAL | 722 | Windows cover free display output channel and normal output channel. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_ERR_FREE_SHOW_WINDOW_SPLIT | 723 | Window division is not supported for free display windows. |
| NET_ERR_INAPPROPRIATE_WINDOW_FREE_SHOW | 724 | For the windows whose number is not integral multiple of the number of output channels, free display is not supported. |
| NET_DVR_TRANSPARENT_WINDOW_NOT_SUPPORT_SPLIT | 725 | For windows whose transparency configuration is enabled, window division is not supported. |
| NET_DVR_SPLIT_WINDOW_NOT_SUPPORT_TRANSPARENT | 726 | For windows whose window division is enabled, transparency configuration is not supported. |
| NET_ERR_TERMINAL_BUSY | 780 | The terminal busy. |
| NET_DVR_FUNCTION_RESOURCE_USAGE_ERROR | 791 | Failed to enable this function. The resources is occupied by other functions. |
| NET_DVR_DEV_NET_OVERFLOW | 800 | Network traffic is out of the limitation. |
| NET_DVR_STATUS_RECORDFILE_WRITING_NOT_LOCK | 801 | Failed to lock. The video file is recording. |
| NET_DVR_STATUS_CANT_FORMAT_LITTLE_DISK | 802 | Failed to format HDD. The HDD space is too small. |
| NET_SDK_ERR_REMOTE_DISCONNEC | 803 | Failed to connect to the remote terminal. |
| NET_SDK_ERR_RD_ADD_RD | 804 | Spare server cannot be added to spare server. |
| NET_SDK_ERR_BACKUP_DISK_EXCEPT | 805 | Backup disk exception. |
| NET_SDK_ERR_RD_LIMIT | 806 | No more spare server can be added. |
| NET_SDK_ERR_ADDED_RD_IS_WD | 807 | The added spare server is a working server. |
| NET_SDK_ERR_ADD_ORDER_WRONG | 808 | Adding flow error. |
| NET_SDK_ERR_WD_ADD_WD | 809 | Working server cannot be added to working server. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_SDK_ERR_WD_SERVICE_EXCETP | 810 | CVR service exception (For N+1 mode, it refers to CVR working server exception). |
| NET_SDK_ERR_RD_SERVICE_EXCETP | 811 | Spare CVR server exception. |
| NET_SDK_ERR_ADDED_WD_IS_RD | 812 | The added working server is spare server. |
| NET_SDK_ERR_PERFORMANCE_LIMIT | 813 | The performance reaches the upper-limit. |
| NET_SDK_ERR_ADDED_DEVICE_EXIST | 814 | This device already exists. |
| NET_SDK_ERR_INQUEST_RESUMING | 815 | Inquest resuming. |
| NET_SDK_ERR_RECORD_BACKUPING | 816 | Inquest video backing up. |
| NET_SDK_ERR_DISK_PLAYING | 817 | Playing. |
| NET_SDK_ERR_INQUEST_STARTED | 818 | Inquest started. |
| NET_SDK_ERR_LOCAL_OPERATING | 819 | Locally operating. |
| NET_SDK_ERR_INQUEST_NOT_START | 820 | Inquest is not started. |
| NET_SDK_ERR_CHAN_AUDIO_BIND | 821 | The channel is not bound or binding two-way audio failed. |
| NET_DVR_N_PLUS_ONE_MODE | 822 | Ddevice is in N+1 mode. Cloud storage is not supported. |
| NET_DVR_CLOUD_STORAGE_OPENED | 823 | Cloud storage mode is enbaled. |
| NET_DVR_ERR_OPER_NOT_ALLOWED | 824 | Operation failed. The device is in N+0 taken over status. |
| NET_DVR_ERR_NEED_RELOCATE | 825 | The device is in N+0 taken over status. Get re-positioning information and try again. |
| NET_SDK_ERR_IR_PORT_ERROR | 830 | IR output error. |
| NET_SDK_ERR_IR_CMD_ERROR | 831 | IR output port command number error |
| NET_SDK_ERR_NOT_INQUESTING | 832 | Device is not in inquest status. |
| NET_SDK_ERR_INQUEST_NOT_PAUSED | 833 | Device is not in paused status. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_CHECK_PASSWORD_MISTAKE_ERROR | 834 | Incorrect verification code. |
| NET_DVR_CHECK_PASSWORD_NULL_ERROR | 835 | Verification code is required. |
| NET_DVR_UNABLE_CALIB_ERROR | 836 | Failed to calibrate. |
| NET_DVR_PLEASE_CALIB_ERROR | 837 | Calibration first. |
| NET_DVR_ERR_PANORAMIC_CAL_EMPTY | 838 | Panoramic calibration is empty in Flash. |
| NET_DVR_ERR_CALIB_FAIL_PLEASEAGAIN | 839 | Calibration failed, please try again. |
| NET_DVR_ERR_DETECTION_LINE | 840 | Rule line configuration error. Please try again and make sure the line is within the red region. |
| NET_DVR_EXCEED_FACE_IMAGES_ERROR | 843 | No more face pictures can be added. |
| NET_DVR_ANALYSIS_FACE_IMAGES_ERROR | 844 | Picture recognition failed. |
| NET_ERR_ALARM_INPUT_OCCUPIED | 845 | A<-1 alarm number is used for triggering vehicle capture. |
| NET_DVR_FACELIB_DATABASE_ERROR | 846 | Database version in face picture library mismatched. |
| NET_DVR_FACELIB_DATA_ERROR | 847 | Face picture library data error. |
| NET_DVR_FACE_DATA_ID_ERROR | 848 | Invalid face data PID. |
| NET_DVR_FACELIB_ID_ERROR | 849 | Invalid face picture library ID. |
| NET_DVR_EXCEED_FACE_LIBARY_ERROR | 850 | No more face picture libraries can be established.. |
| NET_DVR_PIC_ANALYSIS_NO_TARGET_ERROR | 851 | No target recognized in the picture. |
| NET_DVR_SUBPIC_ANALYSIS_MODELING_ERROR | 852 | Sub picture modeling failed. |
| NET_DVR_PIC_ANALYSIS_NO_RESOURCE_ERROR | 853 | No VCA engine supports picture secondary recognition. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_ANALYSIS_ENGINES_NO_RESOURCE_ERROR | 854 | No VCA engine. |
| NET_DVR_ANALYSIS_ENGINES_USAGE_EXCEED_ERROR | 855 | Overload. The engine CPU reached 100%. |
| NET_DVR_EXCEED_HUMANMISINFO_FILTER_ENABLED_ERROR | 856 | No more false alarm channel can be enabled. |
| NET_DVR_NAME_ERROR | 857 | Name error. |
| NET_DVR_NAME_EXIST_ERROR | 858 | The name already exists. |
| NET_DVR_FACELIB_PIC_IMPORTING_ERROR | 859 | The pictures is importing to face picture library. |
| NET_DVR_PIC_FORMAT_ERROR | 864 | Invalid picture format. |
| NET_DVR_PIC_RESOLUTION_INVALID_ERROR | 865 | Invalid picture resolution. |
| NET_DVR_PIC_SIZE_EXCEED_ERROR | 866 | The picture size is too large. |
| NET_DVR_PIC_ANALYSIS_TARGRT_NUM_EXCEED_ERROR | 867 | Too many targets in the picture. |
| NET_DVR_ANALYSIS_ENGINES_LOADING_ERROR | 868 | Initializing analysis engine. |
| NET_DVR_ANALYSIS_ENGINES_ABNORMA_ERROR | 869 | Analysis engine exception. |
| NET_DVR_ANALYSIS_ENGINES_FACELIB_IMPORTING | 870 | Analysis engine is importing pictures to face picture library. |
| NET_DVR_NO_DATA_FOR_MODELING_ERROR | 871 | No data for modeling. |
| NET_DVR_FACE_DATA_MODELING_ERROR | 872 | Device is modeling picture. Concurrent processing is not supported. |
| NET_ERR_FACELIBDATA_OVERLIMIT | 873 | No more face picture can be added to the device (the data of imported face picture library) |
| NET_DVR_ANALYSIS_ENGINES_ASSOCIATED_CHANNEL | 874 | Channel is linked to the analysis engine. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_ERR_CUSTOMID_LEN | 875 | The minimum length of upper layer custom ID is 32 bytes. |
| NET_DVR_ERR_CUSTOMFACELIBID_REPEAT | 876 | The applied custom face picture library ID is duplicated |
| NET_DVR_ERR_CUSTOMHUMANID_REPEAT | 877 | The applied custom person ID is duplicated. |
| NET_DVR_ERR_URL_DOWNLOAD_FAIL | 878 | URL download failed. |
| NET_DVR_ERR_URL_DOWNLOAD_NOTSTART | 879 | URL download has not started. |
| NET_DVR_CFG_FILE_SECRETKEY_ERROR | 880 | The security verification key of configuration file is error. |
| NET_DVR_THERMOMETRY_REGION_OVERSTEP_ERROR | 883 | Invalid thermometry region |
| NET_DVR_ERR_TOO_SHORT_CALIBRATING_TIME | 894 | Too short time for calibration. |
| NET_DVR_ERR_AUTO_CALIBRATE_FAILED | 895 | Auto calibration failed. |
| NET_DVR_ERR_VERIFICATION_FAILED | 896 | Verification failed. |
| NET_DVR_NO_TEMP_SENSOR_ERROR | 897 | No temperature sensor. |
| NET_DVR_PUPIL_DISTANCE_OVERSIZE_ERROR | 898 | The pupil distance is too large. |
| NET_ERR_WINCHAN_IDX | 901 | Window channel index error. |
| NET_ERR_WIN_LAYER | 902 | Window layer number error(the count of window layers on a single screen exceeds the max number). |
| NET_ERR_WIN_BLK_NUM | 903 | Window block number error(the count of screens that single window overlays exceeds the max number). |
| NET_ERR_OUTPUT_RESOLUTION | 904 | The output resolution error. |
| NET_ERR_LAYOUT | 905 | Layout index error. |
| NET_ERR_INPUT_RESOLUTION | 906 | The input resolution is not supported. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_ERR_SUBDEVICE_OFFLINE | 907 | The sub-device is off-line. |
| NET_ERR_NO_DECODE_CHAN | 908 | There is no free decoding channel. |
| NET_ERR_MAX_WINDOW_ABILITY | 909 | The upper limit of window number. |
| NET_ERR_ORDER_ERROR | 910 | Calling order error. |
| NET_ERR_PLAYING_PLAN | 911 | Be playing plan. |
| NET_ERR_DECODER_USED | 912 | Decoder board is being used. |
| NET_ERR_OUTPUT_BOARD_DATA_ OVERFLOW | 913 | Output board data overflow |
| NET_ERR_SAME_USER_NAME | 914 | Duplicate user name |
| NET_ERR_INVALID_USER_NAME | 915 | Invalid user name |
| NET_ERR_MATRIX_USING | 916 | Input matrix is in use. |
| NET_ERR_DIFFERENT_CHAN_TYPE | 917 | Different channel type (the type of matrix output channel mismatches that of the controller input channel) |
| NET_ERR_INPUT_CHAN_BINDED | 918 | Input channel has been bound by other matrix |
| NET_ERR_BINDED_OUTPUT_CHAN_ OVERFLOW | 919 | The matrix output channels in use exceeded the number bound by matrix and controller |
| NET_ERR_MAX_SIGNAL_NUM | 920 | Number of input signals reached upper limit |
| NET_ERR_INPUT_CHAN_USING | 921 | Input channel is in use |
| NET_ERR_MANAGER_LOGON | 922 | Administrator has logged in, operation failed |
| NET_ERR_USERALREADY_LOGON | 923 | The user has logged in, operation failed |
| NET_ERR_LAYOUT_INIT | 924 | Scene is initializing, operation failed |
| NET_ERR_BASEMAP_SIZE_NOT_ MATCH | 925 | Base image size does not match |
| NET_ERR_WINDOW_OPERATING | 926 | Window is in other operation, operation failed |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_ERR_SIGNAL_UPLIMIT | 927 | Number of signal source window reached upper limit |
| NET_ERR_WINDOW_SIZE_OVERLIMIT | 943 | The window size exceeds the limit. |
| NET_ERR_MAX_WIN_OVERLAP | 951 | The number of windows overlap has reached the maximum limit. |
| NET_ERR_STREAMID_CHAN_BOTH_VALID | 952 | stream ID and channel number are both valid. |
| NET_ERR_NO_ZERO_CHAN | 953 | The device has no zero channel. |
| NEED_RECONNECT | 955 | Need redirection (for transcoding system) |
| NET_ERR_NO_STREAM_ID | 956 | The stream ID does not exist. |
| NET_DVR_TRANS_NOT_START | 957 | The transcoding has not been started. |
| NET_ERR_MAXNUM_STREAM_ID | 958 | The number of stream ID has reached the maximum limit. |
| NET_ERR_WORKMODE_MISMATCH | 959 | The work mode does not match with the requirement. |
| NET_ERR_MODE_IS_USING | 960 | It Has been working in current mode. |
| NET_ERR_DEV_PROGRESSING | 961 | The device is in processing |
| NET_ERR_PASSIVE_TRANSCODING | 962 | It is in transcoding. |
| NET_DVR_ERR_WINDOW_SIZE_PLACE | 975 | Wrong window position. |
| NET_DVR_ERR_RGIONAL_RESTRICTIONS | 976 | Screen distance exceeds the limit. |
| NET_DVR_ERR_CLOSE_WINDOWS | 984 | Operation failed. Close the window first. |
| NET_DVR_ERR_MATRIX_LOOP_ABILITY | 985 | Beyond the cycle decoding capacity. |
| NET_DVR_ERR_MATRIX_LOOP_TIME | 986 | Invalid cycle decoding time. |
| NET_DVR_ERR_LINKED_OUT_ABILITY | 987 | No more linked camera can be added. |
| NET_ERR_RESOLUTION_NOT_SUPPORT_ODD_VOUT | 990 | The resolution is not supported (odd No.). |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_ERR_RESOLUTION_NOT_SUPPORT_EVEN_VOUT | 991 | The resolution is not supported (even No.). |
| NET_ERR_UnitConfig_Failed | 998 | Unit configuration failed. |
| XML_ABILITY_NOTSUPPORT | 1000 | Getting capability node is not supported |
| XML_ANALYZE_NOENOUGH_BUF | 1001 | Not enough output memory |
| XML_ANALYZE_FIND_LOCALXML_ERROR | 1002 | Failed to find related local xml |
| XML_ANALYZE_LOAD_LOCALXML_ERROR | 1003 | Loading local xml error |
| XML_NANLYZE_DVR_DATA_FORMAT_ERROR | 1004 | Device capability data format error |
| XML_ANALYZE_TYPE_ERROR | 1005 | Capability set type error |
| XML_ANALYZE_XML_NODE_ERROR | 1006 | XML capability node format error |
| XML_INPUT_PARAM_ERROR | 1007 | Input capability XML node value error |
| XML_VERSION_MISMATCH | 1008 | XML version does not match |
| NET_ERR_TRANS_CHAN_START | 1101 | Transparent channel has been open, operation failed |
| NET_ERR_DEV_UPGRADING | 1102 | Device is upgrading |
| NET_ERR_MISMATCH_UPGRADE_PACK_TYPE | 1103 | Upgrade pack type does not match |
| NET_ERR_DEV_FORMATTING | 1104 | Device is formatting |
| NET_ERR_MISMATCH_UPGRADE_PACK_VERSION | 1105 | Upgrade pack version does not match |
| NET_ERR_PT_LOCKED | 1106 | PT is locked. |
| NET_DVR_ERR_ILLEGAL_VERIFICATION_CODE | 1111 | Illegal verification code. Change the verification code. |
| NET_DVR_ERR_LACK_VERIFICATION_CODE | 1112 | No verification code. Enter the verification code. |
| NET_DVR_ERR_FORBIDDEN_IP | 1113 | The IP address cannot be configured. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_ERR_HTTP_BKN_EXCEED_ONE | 1125 | Up to one channel's ANR function can be enabled. |
| NET_DVR_ERR_FORMATTING_FAILED | 1131 | Formatting HDD failed. |
| NET_DVR_ERR_ENCRYPTED_FORMATTING_FAILED | 1132 | Formatting encrypted HDD failed. |
| NET_DVR_ERR_WRONG_PASSWORD | 1133 | Verifying password of SD card failed. Incorrect password. |
| NET_ERR_SEARCHING_MODULE | 1201 | Searching peripherals. |
| NET_ERR_REGISTERING_MODULE | 1202 | Registering external module |
| NET_ERR_GETTING_ZONES | 1203 | Getting arming region parameter |
| NET_ERR_GETTING_TRIGGERS | 1204 | Getting trigger |
| NET_ERR_ARMED_STATUS | 1205 | System is in arming status |
| NET_ERR_PROGRAM_MODE_STATUS | 1206 | System is in programming mode |
| NET_ERR_WALK_TEST_MODE_STATUS | 1207 | System is in pacing measuring mode |
| NET_ERR_BYPASS_STATUS | 1208 | Bypass status |
| NET_ERR_DISABLED_MODULE_STATUS | 1209 | Function not enabled |
| NET_ERR_NOT_SUPPORT_OPERATE_ZONE | 1210 | Operation is not supported by arming region |
| NET_ERR_NOT_SUPPORT_MOD_MODULE_ADDR | 1211 | Module address cannot be modified |
| NET_ERR_UNREGISTERED_MODULE | 1212 | Module is not registered |
| NET_ERR_PUBLIC_SUBSYSTEM_ASSOCIATE_SELF | 1213 | Public sub system associate with its self |
| NET_ERR_EXCEEDS_ASSOCIATE_SUBSYSTEM_NUM | 1214 | Number of associated public sub system reached upper limit |
| NET_ERR_BE_ASSOCIATED_BY_PUBLIC_SUBSYSTEM | 1215 | Sub system is associated by other public sub system |
| NET_ERR_ZONE_FAULT_STATUS | 1216 | Arming region is in failure status |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_ERR_SAME_EVENT_TYPE | 1217 | Same event type exists in enable event trigger alarm output and disable event trigger alarm output |
| NET_ERR_ZONE_ALARM_STATUS | 1218 | Arming region is in alarm status |
| NET_ERR_EXPANSION_BUS_SHORT_CIRCUIT | 1219 | Extension bus short-circuit |
| NET_ERR_PWD_CONFLICT | 1220 | Password conflict, e.g., lock password is identical with duress password |
| NET_ERR_DETECTOR_GISTERED_BY_OTHER_ZONE | 1221 | Detector has been registered by other arming regions |
| NET_ERR_DETECTOR_GISTERED_BY_OTHER_PU | 1222 | Detector has been registered by other hosts |
| NET_ERR_DETECTOR_DISCONNECT | 1223 | Detector offline |
| NET_ERR_CALL_BUSY | 1224 | Device in call |
| NET_ERR_FILE_NAME | 1357 | File name error, empty or invalid |
| NET_ERR_BROADCAST_BUSY | 1358 | Device in broadcast |
| NET_DVR_ERR_LANENUM_EXCEED | 1400 | Over the number of lanes. |
| NET_DVR_ERR_PRAREA_EXCEED | 1401 | Recognition area is too large. |
| NET_DVR_ERR_LIGHT_PARAM | 1402 | Signal lamp access parameters error. |
| NET_DVR_ERR_LANE_LINE_INVALID | 1403 | Lane configuration error. |
| NET_DVR_ERR_STOP_LINE_INVALID | 1404 | Stop line configuration error. |
| NET_DVR_ERR_LEFTORRIGHT_LINE_INVALID | 1405 | Turn left / right boundary configuration error. |
| NET_DVR_ERR_LANE_NO_REPEAT | 1406 | Overlay lane number repetition. |
| NET_DVR_ERR_PRAREA_INVALID | 1407 | The polygon does not meet the requirements. |
| NET_DVR_ERR_LIGHT_NUM_EXCEED | 1408 | Video detection of traffic light signal exceeds the maximum number of. |
| NET_DVR_ERR_SUBLIGHT_NUM_INVALID | 1409 | Video detection of traffic signal lamp lights are not legitimate |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_ERR_LIGHT_AREASIZE_ INVALID | 1410 | The size of the video detection of traffic light input signal lamp is not valid. |
| NET_DVR_ERR_LIGHT_COLOR_ INVALID | 1411 | The color of the video detection of traffic light input signal lamp color is not legitimate. |
| NET_DVR_ERR_LIGHT_DIRECTION_ INVALID | 1412 | The direction property of the video detection of traffic light input light is not valid. |
| NET_DVR_ERR_LACK_IOABLITY | 1413 | Lack of IO ablity. |
| NET_DVR_ERR_FTP_PORT | 1414 | FTP port error. |
| NET_DVR_ERR_FTP_CATALOGUE | 1415 | FTP catalogue error. |
| NET_DVR_ERR_FTP_UPLOAD_TYPE | 1416 | FTP upload type error. |
| NET_DVR_ERR_FLASH_PARAM_ WRITE | 1417 | Setting param flash write error. |
| NET_DVR_ERR_FLASH_PARAM_READ | 1418 | Getting param flash read error. |
| NET_DVR_ERR_PICNAME_DELIMITER | 1419 | Pic name delimiter error. |
| NET_DVR_ERR_PICNAME_ITEM | 1420 | Pic name item error. |
| NET_DVR_ERR_PLATE_RECOGNIZE_ TYPE | 1421 | Plate recognize type error. |
| NET_DVR_ERR_CAPTURE_TIMES | 1422 | Capture times error. |
| NET_DVR_ERR_LOOP_DISTANCE | 1423 | Loop distance error. |
| NET_DVR_ERR_LOOP_INPUT_STATUS | 1424 | Loop input status error. |
| NET_DVR_ERR_RELATE_IO_CONFLICT | 1425 | Related IO conflict. |
| NET_DVR_ERR_INTERVAL_TIME | 1426 | Interval time error. |
| NET_DVR_ERR_SIGN_SPEED | 1427 | Sign speed error. |
| NET_DVR_ERR_PIC_FLIP | 1428 | Flip is used. |
| NET_DVR_ERR_RELATE_LANE_ NUMBER | 1429 | Related lane number error. |
| NET_DVR_ERR_TRIGGER_MODE | 1430 | Trigger mode error. |
| NET_DVR_ERR_DELAY_TIME | 1431 | Delay time error. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_ERR_EXCEED_RS485_COUNT | 1432 | Exceed RS485 count. |
| NET_DVR_ERR_RADAR_TYPE | 1433 | Radar type error. |
| NET_DVR_ERR_RADAR_ANGLE | 1434 | Radar angle error. |
| NET_DVR_ERR_RADAR_SPEED_VALID_TIME | 1435 | Radar speed valid time error. |
| NET_DVR_ERR_RADAR_LINE_CORRECT | 1436 | Radar line correct error. |
| NET_DVR_ERR_RADAR_CONST_CORRECT | 1437 | Radar const correct error. |
| NET_DVR_ERR_RECORD_PARAM | 1438 | Record param error. |
| NET_DVR_ERR_LIGHT_WITHOUT_COLOR_AND_DIRECTION | 1439 | Light number and other param error. |
| NET_DVR_ERR_LIGHT_WITHOUT_DETECTION_REGION | 1440 | Light number and detection region error. |
| NET_DVR_ERR_RECOGNIZE_PROVINCE_PARAM | 1441 | Plate recognize Province param error. |
| NET_DVR_ERR_SPEED_TIMEOUT | 1442 | IO Speed TimeOut Param error. |
| NET_DVR_ERR_NTP_TIMEZONE | 1443 | NTP TimeZone Param error. |
| NET_DVR_ERR_NTP_INTERVAL_TIME | 1444 | NTP Interval Time error. |
| NET_DVR_ERR_NETWORK_CARD_NUM | 1445 | Network Card Num error. |
| NET_DVR_ERR_DEFAULT_ROUTE | 1446 | Default Route error. |
| NET_DVR_ERR_BONDING_WORK_MODE | 1447 | Banding Work Mode error. |
| NET_DVR_ERR_SLAVE_CARD | 1448 | Sub-Card error. |
| NET_DVR_ERR_PRIMARY_CARD | 1449 | Primary Card error. |
| NET_DVR_ERR_DHCP_PPOE_WORK | 1450 | DHCP and PPOE not Meanwhile start. |
| NET_DVR_ERR_NET_INTERFACE | 1451 | Net Interface invalid. |
| NET_DVR_ERR_MTU | 1452 | Invalid MTU parameters. |
| NET_DVR_ERR_NETMASK | 1453 | Netmask address invalid. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_ERR_IP_INVALID | 1454 | IP address invalid. |
| NET_DVR_ERR_MULTICAST_IP_INVALID | 1455 | Multicast IP address invalid. |
| NET_DVR_ERR_GATEWAY_INVALID | 1456 | Gateway address invalid. |
| NET_DVR_ERR_DNS_INVALID | 1457 | DNS Param invalid. |
| NET_DVR_ERR_ALARMHOST_IP_INVALID | 1458 | AlarmHost IP invalid. |
| NET_DVR_ERR_IP_CONFLICT | 1459 | IP address Conflict. |
| NET_DVR_ERR_NETWORK_SEGMENT | 1460 | IP not support Multi Network segment. |
| NET_DVR_ERR_NETPORT | 1461 | NetPort error. |
| NET_DVR_ERR_PPPOE_NOSUPPORT | 1462 | PPPoE is not supported. |
| NET_DVR_ERR_DOMAINNAME_NOSUPPORT | 1463 | Not Support Domain Name. |
| NET_DVR_ERR_NO_SPEED | 1464 | Speed Not Enabled. |
| NET_DVR_ERR_IOSTATUS_INVALID | 1465 | IO Status invalid. |
| NET_DVR_ERR_BURST_INTERVAL_INVALID | 1466 | Burst Interval invalid. |
| NET_DVR_ERR_RESERVE_MODE | 1467 | Reserve Mode invalid. |
| NET_DVR_ERR_LANE_NO | 1468 | Lane No error. |
| NET_DVR_ERR_COIL_AREA_TYPE | 1469 | Coil Area Type error. |
| NET_DVR_ERR_TRIGGER_AREA_PARAM | 1470 | Trigger Area Param error. |
| NET_DVR_ERR_SPEED_LIMIT_PARAM | 1471 | Speed Limit Param error. |
| NET_DVR_ERR_LANE_PROTOCOL_TYPE | 1472 | Lane Protocol Type error. |
| NET_DVR_ERR_INTERVAL_TYPE | 1473 | Capture Interval Type error. |
| NET_DVR_ERR_INTERVAL_DISTANCE | 1474 | Capture Interval Distance error. |
| NET_DVR_ERR_RS485_ASSOCIATE_DEVTYPE | 1475 | Rs485 Associate DevType error. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_ERR_RS485_ASSOCIATE_LANENO | 1476 | Rs485 Associate LaneNo error. |
| NET_DVR_ERR_LANENO_ASSOCIATE_MULTIRS485 | 1477 | LaneNo Associate MulitRs485 error. |
| NET_DVR_ERR_LIGHT_DETECTION_REGION | 1478 | Light Detection Region error. |
| NET_DVR_ERR_DN2D_NOSUPPORT | 1479 | UnSupport Capture Frame 2D Noise Reduction. |
| NET_DVR_ERR_IRISMODE_NOSUPPORT | 1480 | UnSupport scene Mode. |
| NET_DVR_ERR_WB_NOSUPPORT | 1481 | UnSupport White Balance Mode. |
| NET_DVR_ERR_IO_EFFECTIVENESS | 1482 | IO Effectiveness invalid. |
| NET_DVR_ERR_LIGHTNO_MAX | 1483 | Access Detector Lights Red / Yellow Overrun. |
| NET_DVR_ERR_LIGHTNO_CONFLICT | 1484 | Access Detector Lights Red / Yellow Conflict. |
| NET_DVR_ERR_CANCEL_LINE | 1485 | Trigger straight line error. |
| NET_DVR_ERR_STOP_LINE | 1486 | Subject line area stop line error. |
| NET_DVR_ERR_RUSH_REDLIGHT_LINE | 1487 | Red light trigger lines error. |
| NET_DVR_ERR_IOOUTNO_MAX | 1488 | IO out port error. |
| NET_DVR_ERR_IOOUTNO_AHEADTIME_MAX | 1489 | IO out ahead time error. |
| NET_DVR_ERR_IOOUTNO_IOWORKTIME | 1490 | IO out inwork time error. |
| NET_DVR_ERR_IOOUTNO_FREQMULTI | 1491 | IO out frequency multiplication error. |
| NET_DVR_ERR_IOOUTNO_DUTYRATE | 1492 | IO out duty rate error. |
| NET_DVR_ERR_VIDEO_WITH_EXPOSURE | 1493 | IO out work mode error. |
| NET_DVR_ERR_PLATE_BRIGHTNESS_WITHOUT_FLASHDET | 1494 | Plate enable in plate compensate mode on. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_ERR_RECOGNIZE_TYPE_PARAM | 1495 | Recognize Type error. |
| NET_DVR_ERR_PALTE_RECOGNIZE_AREA_PARAM | 1496 | Plate Recognize Area Param error. |
| NET_DVR_ERR_PORT_CONFLICT | 1497 | Port Conflict. |
| NET_DVR_ERR_LOOP_IP | 1498 | IP cannot be the loopback address. |
| NET_DVR_ERR_DRIVELINE_SENSITIVE | 1499 | Driveline sensitivity error. |
| NET_ERR_VQD_TIME_CONFLICT | 1500 | The time period conflict. |
| NET_ERR_VQD_PLAN_NO_EXIST | 1501 | The diagnostic plan of VQD dese not exist. |
| NET_ERR_VQD_CHAN_NO_EXIST | 1502 | The channel dese not exist. |
| NET_ERR_VQD_CHAN_MAX | 1503 | The total number of VQD plans exceeds the max limit. |
| NET_ERR_VQD_TASK_MAX | 1504 | The total number of VQD tasks exceeds the max limit. |
| NET_DVR_ERR_EXCEED_MAX_CAPTURE_TIMES | 1600 | Capture times exceed 2 in flash mode. |
| NET_DVR_ERR_REDAR_TYPE_CONFLICT | 1601 | Radar type conflict. |
| NET_DVR_ERR_LICENSE_PLATE_NULL | 1602 | The license plate is null. |
| NET_DVR_ERR_WRITE_DATABASE | 1603 | Failed to write data into the database. |
| NET_DVR_ERR_LICENSE_EFFECTIVE_TIME | 1604 | The effective time of license plate error. |
| NET_DVR_ERR_PRERECORDED_STARTTIME_LONG | 1605 | The pre recorded start time is greater than the number of illegal capture. |
| NET_DVR_ERR_TRIGGER_RULE_LINE | 1606 | Trigger rule line error. |
| NET_DVR_ERR_LEFTRIGHT_TRIGGERLINE_NOTVERTICAL | 1607 | Left and right trigger line is not vertical. |
| NET_DVR_ERR_FLASH_LAMP_MODE | 1608 | Flash lamp mode error. |
| NET_DVR_ERR_ILLEGAL_SNAPSHOT_NUM | 1609 | Illegal capture number error. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_ERR_ILLEGAL_DETECTION_TYPE | 1610 | Illegal detection type error. |
| NET_DVR_ERR_POSITIVEBACK_TRIGGERLINE_HIGH | 1611 | Positive back to trigger line height error. |
| NET_DVR_ERR_MIXEDMODE_CAPTYPE_ALLTARGETS | 1612 | Mixed mode only supports capture type all targets. |
| NET_DVR_ERR_CARSIGNSPEED_GREATERTHAN_LIMITSPEED | 1613 | Car sign speed greater than speed limit value. |
| NET_DVR_ERR_BIGCARSIGNSPEED_GREATERTHAN_LIMITSPEED | 1614 | Big car sign speed limit greater than speed limit value. |
| NET_DVR_ERR_BIGCARSIGNSPEED_GREATERTHAN_CARSIGNSPEED | 1615 | Big car sign speed limit is greater than the car sign speed limit value. |
| NET_DVR_ERR_BIGCARLIMITSPEED_GREATERTHAN_CARLIMITSPEED | 1616 | Big car speed limit value is greater than the car speed limit value. |
| NET_DVR_ERR_BIGCARLOWSPEEDLIMIT_GREATERTHAN_CARLOWSPEEDLIMIT | 1617 | Big car low speed limit value is greater than the car low speed limit value. |
| NET_DVR_ERR_CARLIMITSPEED_GREATERTHAN_EXCEPHIGHSPEED | 1618 | Car speed limit greater than exception high speed value. |
| NET_DVR_ERR_BIGCARLIMITSPEED_GREATERTHAN_EXCEPHIGHSPEED | 1619 | Big car speed limit greater than exception high speed value. |
| NET_DVR_ERR_STOPLINE_MORETHAN_TRIGGERLINE | 1620 | Stopping more than straight lines trigger lines. |
| NET_ERR_TIME_OVERLAP | 1900 | Time periods overlap |
| NET_ERR_HOLIDAY_PLAN_OVERLAP | 1901 | Holiday plan overlap |
| NET_ERR_CARDNO_NOT_SORT | 1902 | Card number is not sorted |
| NET_ERR_CARDNO_NOT_EXIST | 1903 | Card number does not exist |
| NET_ERR_ILLEGAL_CARDNO | 1904 | Card number error |
| NET_ERR_ZONE_ALARM | 1905 | Arming region is in arming status (parameter cannot be modified) |
| NET_ERR_ZONE_OPERATION_NOT_SUPPORT | 1906 | Arming region does not support the operation |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_ERR_INTERLOCK_ANTI_ CONFLICT | 1907 | Interlock and anti-passback configuration conflict |
| NET_ERR_DEVICE_CARD_FULL | 1908 | Card full (return after card reached 10,000) |
| NET_ERR_HOLIDAY_GROUP_ DOWNLOAD | 1909 | Failed to download holiday group |
| NET_ERR_LOCAL_CONTROL_OFF | 1910 | Distributed access controller offline |
| NET_ERR_LOCAL_CONTROL_DISADD | 1911 | Distributed access controller is not added |
| NET_ERR_LOCAL_CONTROL_HASADD | 1912 | Distributed access controller is added |
| NET_ERR_LOCAL_CONTROL_ DOORNO_CONFLICT | 1913 | Conflict with added distributed access controller |
| NET_ERR_LOCAL_CONTROL_ COMMUNICATION_FAIL | 1914 | Distributed access controller communication failed |
| NET_ERR_OPERAND_INEXISTENCE | 1915 | Operation object does not exist (operation to door, alarm output, alarm input, return when the object is not added) |
| NET_ERR_LOCAL_CONTROL_OVER_ LIMIT | 1916 | Distributed access controller exceeded device capability upper limit |
| NET_ERR_DOOR_OVER_LIMIT | 1917 | Door exceeded device capability upper limit |
| NET_ERR_ALARM_OVER_LIMIT | 1918 | Alarm input and output exceeded device capability upper limit |
| NET_ERR_LOCAL_CONTROL_ ADDRESS_INCONFORMITY_TYPE | 1919 | Distributed access controller address does not match with type |
| NET_ERR_NOT_SUPPORT_ONE_ MORE_CARD | 1920 | not support one person multi-card |
| NET_ERR_DELETE_NO_EXISTENCE_ FACE | 1921 | The face picture does not exist. |
| NET_ERR_DOOR_SPECIAL_ PASSWORD_REPEAT | 1922 | Repeated door door duress code, the super password, or the dismiss code. |
| NET_ERR_AUTH_CODE_REPEAT | 1923 | Repeated device authentication code |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_ERR_DEPLOY_EXCEED_MAX | 1924 | No more devices can be armed. |
| NET_ERR_NOT_SUPPORT_DEL_FP_BY_ID | 1925 | The fingerprint module does not support deleting fingerprint by finger ID. |
| NET_ERR_TIME_RANGE | 1926 | Invalid range of the effective period. |
| NET_ERR_CAPTURE_TIMEOUT | 1927 | Collection timed out. |
| NET_ERR_LOW_SCORE | 1928 | Low quality of collected data. |
| NET_ERR_OFFLINE_CAPTURING | 1929 | The device is collecting data offline and cannot respond. |
| NET_DVR_ERR_OUTDOOR_COMMUNICATION | 1950 | Communication exception with outdoor terminal |
| NET_DVR_ERR_ROOMNO_UNDEFINED | 1951 | Room number is not set |
| NET_DVR_ERR_NO_CALLING | 1952 | No call |
| NET_DVR_ERR_RINGING | 1953 | Ringing |
| NET_DVR_ERR_IS_CALLING_NOW | 1954 | Call in progress |
| NET_DVR_ERR_LOCK_PASSWORD_WRONG | 1955 | Incorrect smart lock password |
| NET_DVR_ERR_CONTROL_LOCK_FAILURE | 1956 | Lock control failure |
| NET_DVR_ERR_CONTROL_LOCK_OVERTIME | 1957 | Lock control timed out |
| NET_DVR_ERR_LOCK_DEVICE_BUSY | 1958 | Smart lock device busy |
| NET_DVR_ERR_UNOPEN_REMOTE_LOCK_FUNCTION | 1959 | Remote lock control not enabled |
| NET_DVR_ERR_FILE_NOT_COMPLETE | 2100 | Downloaded file is incomplete |
| NET_DVR_ERR_IPC_EXIST | 2101 | The camera already exists |
| NET_DVR_ERR_ADD_IPC | 2102 | Camera has been added to the channel |
| NET_DVR_ERR_OUT_OF_RES | 2103 | Not enough network bandwidth |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_ERR_CONFLICT_TO_ LOCALIP | 2104 | IP address of camera conflicts with that of DVR |
| NET_DVR_ERR_IP_SET | 2105 | Invalid IP address |
| NET_DVR_ERR_PORT_SET | 2106 | Invalid port number |
| NET_ERR_WAN_NOTSUPPORT | 2107 | Not in the same LAN, cannot set security question or export GUID file |
| NET_ERR_MUTEX_FUNCTION | 2108 | Mutually exclusive function |
| NET_ERR_QUESTION_CONFIGNUM | 2109 | Error in number of security question configurations |
| NET_ERR_FACECHAN_NORESOURCE | 2110 | All the face VCA channels are occupied. |
| NET_ERR_DATA_CALLBACK | 2111 | Data is calling back. |
| NET_ERR_ATM_VCA_CHAN_IS_ RELATED | 2112 | The VCA channel is already linked. |
| NET_ERR_ATM_VCA_CHAN_IS_ OVERLAPED | 2113 | The VCA channel is already overlayed. |
| NET_ERR _FACE_CHAN_UNOVERLAP_ EACH_OTHER | 2114 | The face channels cannot be overlayed. |
| NET_DVR_SMD_ENCODING_ NORESOURSE | 2116 | Insufficient SMD encoding resource |
| NET_DVR_SMD_DECODING_ NORESOURSE | 2117 | Insufficient SMD decoding resource |
| NET_DVR_FACELIB_DATA_ PROCESSING | 2118 | Face picture library data is in processing |
| NET_DVR_ERR_LARGE_TIME_ DIFFRENCE | 2119 | There is a great time difference between device and server. |
| NET_DVR_NO_SUPPORT_WITH_ PLAYBACK | 2120 | It is not supported. Playback is enabled. |
| NET_DVR_CHANNEL_NO_SUPPORT_ WITH_SMD | 2121 | It is not supported. SMD of channel is enabled. |
| NET_DVR_CHANNEL_NO_SUPPORT_ WITH_FD | 2122 | It is not supported. Face capture of channel is enabled. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_ILLEGAL_PHONE_NUMBER | 2123 | Invalid telephone number |
| NET_DVR_ILLEGAL_CERITIFICATE_NUMBER | 2124 | Invalid ID No. |
| NET_DVR_ERR_CHANNEL_RESOLUTION_NO_SUPPORT | 2125 | The channel resolution is not supported |
| NET_DVR_ERR_CHANNEL_COMPRESSION_NO_SUPPORT | 2126 | The channel encoding format is not supported |
| NET_DVR_ERR_CLUSTER_DEVICE_TOO_LESS | 2127 | Deleting is not allowed. The number of devices is not enough |
| NET_DVR_ERR_CLUSTER_DEL_DEVICE_CM_PLAYLOAD | 2128 | Deleting is not allowed. The device is cluster host. |
| NET_DVR_ERR_CLUSTER_DEVNUM_OVER_UPPER_LIMIT | 2129 | No more devices can be added. |
| NET_DVR_ERR_CLUSTER_DEVICE_TYPE_INCONFORMITY | 2130 | Device type mismatched. |
| NET_DVR_ERR_CLUSTER_DEVICE_VERSION_INCONFORMITY | 2131 | Device version mismatched. |
| NET_DVR_ERR_CLUSTER_IP_CONFLICT | 2132 | Cluster system IP address conflict: ipv4 address conflict, invalid ipv6. |
| NET_DVR_ERR_CLUSTER_IP_INVALID | 2133 | Invalid cluster system IP address: invalid ipv4, invalid ipv6. |
| NET_DVR_ERR_CLUSTER_PORT_CONFLICT | 2134 | Cluster system port conflict |
| NET_DVR_ERR_CLUSTER_PORT_INVALID | 2135 | Invalid cluster system port |
| NET_DVR_ERR_CLUSTER_USERNAEM_OR_PASSWORD_INVALID | 2136 | Invalid user name or password |
| NET_DVR_ERR_CLUSTER_DEVICE_ALREADY_EXIST | 2137 | The device already exists. |
| NET_DVR_ERR_CLUSTER_DEVICE_NOT_EXIST | 2138 | The device does not exist. |
| NET_DVR_ERR_CLUSTER_NON_CLUSTER_MODE | 2139 | The device working mode is not the cluster mode . |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_ERR_CLUSTER_IP_NOT_SAME_LAN | 2140 | IP addresses are in different LAN. Building cluster or extending capacity for NVRs in different LAN is not allowed. |
| NET_DVR_ERR_IDENTITY_KEY | 2147 | Incorrect interaction password |
| NET_DVR_MISSING_IDENTITY_KEY | 2148 | Interaction password is missing |
| NET_DVR_ERR_CAPTURE_PACKAGE_FAILED | 2141 | Capturing packets failed. |
| NET_DVR_ERR_CAPTURE_PACKAGE_PROCESSING | 2142 | Capturing packet. |
| NET_DVR_ERR_SAFETY_HELMET_NO_RESOURCE | 2143 | No enough hard hat detection resource. |
| NET_DVR_NO_SUPPORT_WITH_ABSTRACT | 2144 | This function is not supported. Video synopsis is already enabled. |
| NET_DVR_INSUFFICIENT_DEEP_LEARNING_RESOURCES | 2146 | No more deep learning resources can be added. |
| NET_DVR_NO_SUPPORT_WITH_PERSON_DENSITY_DETECT | 2149 | People gathering density is enabled, it is not supported |
| NET_DVR_IPC_RESOLUTION_OVERFLOW | 2150 | The network camera resolution is too large |
| NET_DVR_IPC_BITRATE_OVERFLOW | 2151 | The network camera bitrate is too large |
| NET_DVR_ERR_INVALID_TASKID | 2152 | Invalid taskID |
| NET_DVR_PANEL_MODE_NOT_CONFIG | 2153 | The ATM panel mode is not configured. |
| NET_DVR_NO_HUMAN_ENGINES_RESOURCE | 2154 | No enough engine resource |
| NET_DVR_ERR_TASK_NUMBER_OVERFLOW | 2155 | No more task data is allowed |
| NET_DVR_ERR_COLLISION_TIME_OVERFLOW | 2156 | Collision time is over the limit |
| NET_DVR_ERR_EVENT_NOTSUPPORT | 2159 | Subscribing alarm/event is not supported. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_IPC_NUM_REACHES_LIMIT | 2184 | The max. number of network camera channels reached. |
| NET_DVR_IOT_NUM_REACHES_LIMIT | 2185 | The max. number of IoT channels reached |
| NET_DVR_IOT_CHANNEL_DEVICE_EXIST | 2186 | Device of the IoT channel already exists. |
| NET_DVR_IOT_CHANNEL_DEVICE_NOT_EXIST | 2187 | Device of the IoT channel does not exist. |
| NET_DVR_INVALID_IOT_PROTOCOL_TYPE | 2188 | Invalid IoT protocol type |
| NET_DVR_INVALID_EZVIZ_SECRET_KEY | 2189 | Invalid verification code |
| NET_DVR_DUPLICATE_IOT_DEVICE | 2190 | Duplicated IoT device |
| NET_DVR_ERROR_NEED_DOUBLE_VERIFICATION | 2206 | Double verification is required |
| NET_DVR_NO_DOUBLE_VERIFICATION_USER | 2207 | No double verification user |
| NET_DVR_TIMESPAN_NUM_OVER_LIMIT | 2209 | Max. number of time buckets reached |
| NET_DVR_CHANNEL_NUM_OVER_LIMIT | 2210 | Max. number of channels reached |
| NET_DVR_NO_SEARCH_ID_RESOURCE | 2211 | Insufficient searchID resources |
| NET_DVR_SWITCH_TIMEDIFF_LESS_LIMIT | 2249 | Time difference between power on and off should be less than 10 minutes. |
| NET_DVR_NO_SUPPORT_DELETE_STRANGER_LIB | 2262 | Deleting stranger library is not supported |
| NET_DVR_NO_SUPPORT_CREATE_STRANGER_LIB | 2263 | Creating stranger library is not supported |
| NET_DVR_SSD_FILE_SYSTEM_ERROR | 2266 | SSD file system error |
| NET_DVR_INSUFFICIENT_SSD__FOR_FPD | 2267 | Insufficient SSD space for person frequency detection |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_SMRDISK_NOT_SUPPORT_RAID | 2269 | SMR disk does not support RAID. |
| NET_DVR_ERR_NOTSUPPORT_DEICING | 3001 | Device does not support deicing function under current status.(Deicing function is only supported under the power status of POE+, AC24V, and DC12V). |
| NET_DVR_ERR_THERMENABLE_CLOSE | 3002 | Temperature measurement function is not enabled. (The enable function in NET_DVR_THERMOMETRY_BASICPARAM is not turned on) |
| NET_DVR_ERR_PANORAMIC_LIMIT_OPERATED | 3004 | Panoramic map and limit cannot be operated at same time |
| NET_DVR_ERR_SMARTH264_ROI_OPERATED | 3005 | SmartH264 and ROI cannot be enabled at the same time. |
| NET_DVR_ERR_RULENUM_LIMIT | 3006 | No more rules can be added. |
| NET_DVR_ERR_LASER_DEICING_OPERATED | 3007 | Laser and deicing function cannot be enabled at the same time. |
| NET_DVR_ERR_OFFDIGITALZOOM_OR_MINZOOMLIMIT | 3008 | Please disable the digital zoom function or set the zoom limit to the minimum value. Otherwise, when enabling smoke and fire detection, abnormal event detection, ship detection, defective point correction, temperature measurement, smoke and fire shielding function, this error code will be prompted. |
| NET_DVR_SYNCHRONIZEFOV_ERROR | 3010 | Field of view synchronization failed. |
| NET_DVR_RULE_SHIELDMASK_CONFLICT_ERROR | 3013 | The rule region conflicts with the shielded area. |
| NET_DVR_ERR_NO_SAFETY_HELMET_REGION | 3501 | The hard hat detection area is not configured. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_ERR_UNCLOSED_SAFETY_HELMET | 3502 | The hard hat detection is enabled. |
| NET_DVR_UPLOAD_HBDLIBID_ERROR | 3504 | Incorrect ID of human body picture library (incorrect HBDID or customHBDID) |

### RTSP Communication Library Related Errors

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_RTSP_ERROR_NOENOUGHPRI | 401 | Authentication failed: if server returns 401, it will change to this error code |
| NET_DVR_RTSP_ERROR_ALLOC_RESOURCE | 402 | Failed to allocate the resource |
| NET_DVR_RTSP_ERROR_PARAMETER | 403 | Parameter error |
| NET_DVR_RTSP_ERROR_NO_URL | 404 | The assigned URL does not exist: when the server returns 404, SDK turns to this error code. E.g. the channel is not available, or the channel does not support sub stream |
| NET_DVR_RTSP_ERROR_FORCE_STOP | 406 | The user forces to exit midway |
| NET_DVR_RTSP_GETPORTFAILED | 407 | RTSP port getting error. |
| NET_DVR_RTSP_DESCRIBERROR | 410 | RTSP DECRIBE communicate error |
| NET_DVR_RTSP_DESCRIBESENDTIMEOUT | 411 | Sending "RTSP DECRIBE" is timeout. |
| NET_DVR_RTSP_DESCRIBESENDERROR | 412 | Failed to send "RTSP DECRIBE". |
| NET_DVR_RTSP_DESCRIBERECVTIMEOUT | 413 | Receiving "RTSP DECRIBE" is timeout. |
| NET_DVR_RTSP_DESCRIBERECVDATALOST | 414 | Receiving data of "RTSP DECRIBE" error. |
| NET_DVR_RTSP_DESCRIBERECVERROR | 415 | Failed to receive "RTSP DECRIBE". |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_RTSP_DESCRIBESERVERERR | 416 | "RTSP DECRIBE, the device returns the error code: 501 (failed to allocate the resource in the device) |
| NET_DVR_RTSP_SETUPERROR | 420 | (or 419), RTSP SETUP interaction error. Generally, it is that the address(URL) returned by the device is not accessible, or it is rejected by the server |
| NET_DVR_RTSP_SETUPSENDTIMEOUT | 421 | Sending "RTSP SETUP" is timeout. |
| NET_DVR_RTSP_SETUPSENDERROR | 422 | Sending "RTSP SETUP" error. |
| NET_DVR_RTSP_SETUPRECVTIMEOUT | 423 | Receiving "RTSP SETUP" is timeout. |
| NET_DVR_RTSP_SETUPRECVDATALOST | 424 | Receiving data of "RTSP SETUP" error. |
| NET_DVR_RTSP_SETUPRECVERROR | 425 | Failed to receive "RTSP SETUP". |
| NET_DVR_RTSP_OVER_MAX_CHAN | 426 | "RTSP SETUP" device returns the error that values 401 or 501. It exceeds the max connection number. |
| NET_DVR_RTSP_PLAYERROR | 430 | RTSP PLAY interaction error. |
| NET_DVR_RTSP_PLAYSENDTIMEOUT | 431 | Sending "RTSP PLAY" is timeout. |
| NET_DVR_RTSP_PLAYSENDERROR | 432 | Sending "RTSP PLAY" error. |
| NET_DVR_RTSP_PLAYRECVTIMEOUT | 433 | Receiving "RTSP PLAY" is timeout. |
| NET_DVR_RTSP_PLAYRECVDATALOST | 434 | Receiving data of "RTSP PLAY" error. |
| NET_DVR_RTSP_PLAYRECVERROR | 435 | Failed to receive "RTSP PLAY". |
| NET_DVR_RTSP_PLAYSERVERERR | 436 | "RTSP PLAY" device returns the error that values 401 or 501. |
| NET_DVR_RTSP_TEARDOWNERROR | 440 | RTSP TEARDOWN interaction error. |
| NET_DVR_RTSP_TEARDOWNSENDTIMEOUT | 441 | Sending "RTSP TEARDOWN" is timeout. |
| NET_DVR_RTSP_TEARDOWNSENDERROR | 442 | Sending "RTSP TEARDOWN" error. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_DVR_RTSP_ TEARDOWNRECVTIMEOUT | 443 | Receiving "RTSP TEARDOWN" is timeout. |
| NET_DVR_RTSP_ TEARDOWNRECVDATALOST | 444 | Receiving data of "RTSP TEARDOWN" error. |
| NET_DVR_RTSP_ TEARDOWNRECVERROR | 445 | Failed to receive "RTSP TEARDOWN". |
| NET_DVR_RTSP_ TEARDOWNSERVERERR | 446 | "RTSP TEARDOWN" device returns the error that values 401 or 501. |

## Software Decoding Library Related Errors

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_PLAYM4_NOERROR | 500 | No error. |
| NET_PLAYM4_PARA_OVER | 501 | Input parameter is invalid. |
| NET_PLAYM4_ORDER_ERROR | 502 | API calling order error. |
| NET_PLAYM4_TIMER_ERROR | 503 | Failed to create multimedia clock. |
| NET_PLAYM4_DEC_VIDEO_ERROR | 504 | Failed to decode video data. |
| NET_PLAYM4_DEC_AUDIO_ERROR | 505 | Failed to decode audio data. |
| NET_PLAYM4_ALLOC_MEMORY_ ERROR | 506 | Failed to allocate memory. |
| NET_PLAYM4_OPEN_FILE_ERROR | 507 | Failed to open the file. |
| NET_PLAYM4_CREATE_OBJ_ERROR | 508 | Failed to create thread event. |
| NET_PLAYM4_CREATE_DDRAW_ ERROR | 509 | Failed to create DirectDraw object. |
| NET_PLAYM4_CREATE_OFFSCREEN_ ERROR | 510 | Failed to create backstage cache for OFFSCREEN mode. |
| NET_PLAYM4_BUF_OVER | 511 | Buffer overflow, failed to input stream. |
| NET_PLAYM4_CREATE_SOUND_ ERROR | 512 | Failed to create audio equipment. |
| NET_PLAYM4_SET_VOLUME_ ERROR | 513 | Failed to set the volume. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_PLAYM4_SUPPORT_FILE_ONLY | 514 | This API can be called only for file playback mode. |
| NET_PLAYM4_SUPPORT_STREAM_ONLY | 515 | This API can be called only when playing stream. |
| NET_PLAYM4_SYS_NOT_SUPPORT | 516 | Not support by the system. Decoder can only work on the system above Pentium 3. |
| NET_PLAYM4_FILEHEADER_UNKNOWN | 517 | There is no file header. |
| NET_PLAYM4_VERSION_INCORRECT | 518 | The version mismatch between decoder and encoder. |
| NET_PLAYM4_INIT_DECODER_ERROR | 519 | Failed to initialize the decoder. |
| NET_PLAYM4_CHECK_FILE_ERROR | 520 | The file is too short, or the stream data is unknown. |
| NET_PLAYM4_INIT_TIMER_ERROR | 521 | Failed to initialize multimedia clock. |
| NET_PLAYM4_BLT_ERROR | 522 | BLT failure. |
| NET_PLAYM4_UPDATE_ERROR | 523 | Failed to update overlay surface |
| NET_PLAYM4_OPEN_FILE_ERROR_MULTI | 524 | Failed to open video & audio stream file. |
| NET_PLAYM4_OPEN_FILE_ERROR_VIDEO | 525 | Failed to open video stream file. |
| NET_PLAYM4_JPEG_COMPRESS_ERROR | 526 | JPEG compression error. |
| NET_PLAYM4_EXTRACT_NOT_SUPPORT | 527 | Don't support the version of this file. |
| NET_PLAYM4_EXTRACT_DATA_ERROR | 528 | Extract video data failed. |

### Container Format Conversion Library Related Errors

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_CONVERT_ERROR_NOT_ SUPPORT | 581 | This container format is not supported. |

### Two Way Audio Library Related Errors

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_AUDIOINTERCOM_OK | 600 | No error. |
| NET_AUDIOINTECOM_ERR_NOTSUPORT | 601 | Not support. |
| NET_AUDIOINTECOM_ERR_ALLOC_MEMERY | 602 | Memory allocation error. |
| NET_AUDIOINTECOM_ERR_PARAMETER | 603 | Parameter error. |
| NET_AUDIOINTECOM_ERR_CALL_ORDER | 604 | API calling order error. |
| NET_AUDIOINTECOM_ERR_FIND_DEVICE | 605 | No audio device |
| NET_AUDIOINTECOM_ERR_OPEN_DEVICE | 606 | Failed to open the audio device |
| NET_AUDIOINTECOM_ERR_NO_CONTEXT | 607 | Context error. |
| NET_AUDIOINTECOM_ERR_NO_WAVFILE | 608 | WAV file error. |
| NET_AUDIOINTECOM_ERR_INVALID_TYPE | 609 | The type of WAV parameter is invalid |
| NET_AUDIOINTECOM_ERR_ENCODE_FAIL | 610 | Failed to encode data |
| NET_AUDIOINTECOM_ERR_DECODE_FAIL | 611 | Failed to decode data |
| NET_AUDIOINTECOM_ERR_NO_PLAYBACK | 612 | Failed to play audio |
| NET_AUDIOINTECOM_ERR_DENOISE_FAIL | 613 | Failed to denoise |
| NET_AUDIOINTECOM_ERR_UNKOWN | 619 | Unknown |

### QoS Stream Control Library Related Errors

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_QOS_ERR_SCHEDPARAMS_BAD_ MINIMUM_INTERVAL | 678 | Incorrect predefined minimum interval. |
| NET_QOS_ERR_SCHEDPARAMS_BAD_ FRACTION | 679 | Incorrect predefined score. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_QOS_ERR_SCHEDPARAMS_INVALID_ BANDWIDTH | 680 | Invalid predefined bandwidth. |
| NET_QOS_ERR_PACKET_TOO_BIG | 687 | The packet size is too large. |
| NET_QOS_ERR_PACKET_LENGTH | 688 | Invalid packet size. |
| NET_QOS_ERR_PACKET_VERSION | 689 | Incorrect packet versio information. |
| NET_QOS_ERR_PACKET_UNKNOW | 690 | Unknown packet. |
| NET_QOS_ERR_OUTOFMEM | 695 | Out of memory. |
| NET_QOS_ERR_LIB_NOT_INITIALIZED | 696 | The library is not initialized. |
| NET_QOS_ERR_SESSION_NOT_FOUND | 697 | No session found. |
| NET_QOS_ERR_INVALID_ARGUMENTS | 698 | Invalid parameters. |
| NET_QOS_ERROR | 699 | QoS Stream Control Library error. |
| NET_QOS_OK | 700 | No error. |

## NPQ (Network Protocol Quality) Related Error

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_ERR_NPQ_PARAM | 8001 | NPQ library: Incorrect parameter. |
| NET_ERR_NPQ_SYSTEM | 8002 | NPQ library: Operating system error. |
| NET_ERR_NPQ_GENRAL | 8003 | NPQ library: Internal error. |
| NET_ERR_NPQ_PRECONDITION | 8004 | NPQ library: Calling sequence error. |
| NET_ERR_NPQ_NOTSUPPORT | 8005 | NPQ library: This function is not supported. |
| NET_ERR_NPQ_NOTCALLBACK | 8100 | No data is called back. |
| NET_ERR_NPQ_LOADLIB | 8101 | Loading NPQ library failed. |
| NET_ERR_NPQ_STEAM_CLOSE | 8104 | The NPQ function of this stream is not enabled. |

| Error Name | Error Code | Error Description |
|---|---|---|
| NET_ERR_NPQ_MAX_LINK | 8110 | No more streaming channel's NPQ function can be enabled. |
| NET_ERR_NPQ_STREAM_CFG_ CONFLICT | 8111 | The configured encoding parameters conflicted. |

## C.4 HCNetSDK Log Types

The logs generated by the devices during the HCNetSDK integration are classified as five major types, i.e., alarm log (MAJOR_ALARM-01), exception log (MAJOR_EXCEPTION-0x2), operation log (MAJOR_OPERATION-0x3), additional information log (MAJOR_INFORMATION-0x4), and event log (MAJOR_EVENT-0x5). Each major log type corresponds to multiple minor types, see details below.

### MAJOR_ALARM

**Table C-1 Minor Types of Alarm Log**

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_ALARM_IN | 0x1 | Alarm Input |
| MINOR_ALARM_OUT | 0x2 | Alarm output |
| MINOR_MOTDET_START | 0x3 | Motion detection alarm started |
| MINOR_MOTDET_STOP | 0x4 | Motion detection alarm ended |
| MINOR_HIDE_ALARM_START | 0x5 | Tampering alarm started |
| MINOR_HIDE_ALARM_STOP | 0x6 | Tampering alarm ended |
| MINOR_VCA_ALARM_START | 0x7 | VCA alarm started |
| MINOR_VCA_ALARM_STOP | 0x8 | VCA alarm ended |
| MINOR_ITS_ALARM_START | 0x09 | Traffic event alarm started |
| MINOR_ITS_ALARM_STOP | 0x0a | Traffic event alarm ended |
| MINOR_NETALARM_START | 0x0b | Network alarm started |
| MINOR_NETALARM_STOP | 0x0c | Network alarm ended |
| MINOR_NETALARM_RESUME | 0x0d | Network alarm recovery |
| MINOR_WIRELESS_ALARM_START | 0x0e | Wireless alarm started |
| MINOR_WIRELESS_ALARM_STOP | 0x0f | Wireless alarm ended |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_PIR_ALARM_START | 0x10 | Human induction alarm started |
| MINOR_PIR_ALARM_STOP | 0x11 | Human induction alarm ended |
| MINOR_CALLHELP_ALARM_START | 0x12 | Emergency alarm started |
| MINOR_CALLHELP_ALARM_STOP | 0x13 | Emergency alarm ended |
| MINOR_DETECTFACE_ALARM_START | 0x16 | Face detection alarm started |
| MINOR_DETECTFACE_ALARM_STOP | 0x17 | Face detection alarm ended |
| MINOR_VCA_SECNECHANGE_ DETECTION | 0x1a | Scene change detection alarm |
| MINOR_SMART_REGION_EXITING_ BEGIN | 0x1b | Region exiting detection started |
| MINOR_SMART_REGION_EXITING_ END | 0x1c | Region exiting detection ended |
| MINOR_SMART_LOITERING_BEGIN | 0x1d | Loitering detection started |
| MINOR_SMART_LOITERING_END | 0x1e | Loitering detection ended |
| MINOR_DREDGERDETECTION _ ALARM | 0x11a | Dredger detection alarm |
| MINOR_VCA_ALARM_LINE_ DETECTION_BEGIN | 0x20 | Line crossing detection started |
| MINOR_VCA_ALARM_LINE_ DETECTION_END | 0x21 | Line crossing detection ended |
| MINOR_VCA_ALARM_INTRUDE_ BEGIN | 0x22 | Intrusion detection started |
| MINOR_VCA_ALARM_INTRUDE_END | 0x23 | Intrusion detection ended |
| MINOR_VCA_ALARM_AUDIOINPUT | 0x24 | Audio loss detection |
| MINOR_VCA_ALARM_ AUDIOABNORMAL | 0x25 | Audio exception detection |
| MINOR_VCA_DEFOCUS_DETECTION_ BEGIN | 0x26 | Defocus detection started |
| MINOR_VCA_DEFOCUS_DETECTION_ END | 0x27 | Defocus detection ended |
| MINOR_VCA_FACE_ALARM_BEGIN | 0x29 | Face detection started |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_SMART_REGION_ENTRANCE_BEGIN | 0x2a | Region entrance detection started |
| MINOR_SMART_REGION_ENTRANCE_END | 0x2b | Region entrance detection ended |
| MINOR_SMART_PEOPLE_GATHERING_BEGIN | 0x2c | People gathering detection started |
| MINOR_SMART_PEOPLE_GATHERING_END | 0x2d | People gathering detection ended |
| MINOR_SMART_FAST_MOVING_BEGIN | 0x2e | Fast moving detection started |
| MINOR_SMART_FAST_MOVING_END | 0x2f | Fast moving detection ended |
| MINOR_VCA_FACE_ALARM_END | 0x30 | Face detection ended |
| MINOR_VCA_SCENE_CHANGE_ALARM_BEGIN | 0x31 | Scene change detection started |
| MINOR_VCA_SCENE_CHANGE_ALARM_END | 0x32 | Scene change detection ended |
| MINOR_VCA_ALARM_AUDIOINPUT_BEGIN | 0x33 | Audio loss detection started |
| MINOR_VCA_ALARM_AUDIOINPUT_END | 0x34 | Audio loss detection ended |
| MINOR_VCA_ALARM_AUDIOABNORMAL_BEGIN | 0x35 | Sudden change of sound intensity detection started |
| MINOR_VCA_ALARM_AUDIOABNORMAL_END | 0x36 | Sudden change of sound intensity detection ended |
| MINOR_VCA_ALARM_AUDIOSTEEPDROP | 0x39 | Sudden decrease of sound intensity detection |
| MINOR_SMART_PARKING_BEGIN | 0x3c | Parking detection started |
| MINOR_SMART_PARKING_END | 0x3d | Parking detection ended |
| MINOR_SMART_UNATTENDED_BAGGAGE_BEGIN | 0x3e | Unattended baggage detection started |
| MINOR_SMART_UNATTENDED_BAGGAGE_END | 0x3f | Unattended baggage detection ended |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_SMART_OBJECT_REMOVAL_BEGIN | 0x40 | Object removal detection started |
| MINOR_SMART_OBJECT_REMOVAL_END | 0x41 | Object removal detection ended |
| MINOR_VCA_LEAVE_POSITION_START | 0x42e | Absence detection started |
| MINOR_VCA_LEAVE_POSITION_STOP | 0x42f | Absence detection ended |
| MINOR_VCA_PEOPLENUM_CHANGE_START | 0x434 | The people number change started |
| MINOR_VCA_PEOPLENUM_CHANGE_STOP | 0x435 | The people number change ended |
| MINOR_VCA_RUNNING_START | 0x438 | People running started |
| MINOR_VCA_RUNNING_STOP | 0x439 | People running ended |
| MINOR_VCA_VIOLENT_MOTION_START | 0x43a | Violent motion started |
| MINOR_VCA_VIOLENT_MOTION_STOP | 0x43b | Violent motion ended |
| MINOR_VCA_FAIL_DOWN_START | 0x43c | People falling started |
| MINOR_VCA_FAIL_DOWN_STOP | 0x43d | People falling ended |
| MINOR_VCA_RETENTION_START | 0x43e | Overstay detection started |
| MINOR_VCA_RETENTION_STOP | 0x43f | Overstay detection ended |
| MINOR_SMART_VEHICLE_ALARM_START | 0x46 | License plate detection started |
| MINOR_SMART_VEHICLE_ALARM_STOP | 0x47 | License plate detection ended |
| MINOR_THERMAL_FIREDETECTION | 0x48 | Thermal imaging fire point detection started |
| MINOR_THERMAL_FIREDETECTION_END | 0x49 | Thermal imaging fire point detection ended |
| MINOR_SMART_VANDALPROOF_BEGIN | 0x50 | Vandal-proof detection started |
| MINOR_SMART_VANDALPROOF_END | 0x51 | Vandal-proof detection ended |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_FACESNAP_MATCH_ALARM_START | 0x55 | Face picture comparison alarm started |
| MINOR_FACESNAP_MATCH_ALARM_STOP | 0x56 | Face picture comparison alarm ended |
| MINOR_ALLOWLIST_FACESNAP_MATCH_ALARM_START | 0x57 | Face picture in allowlist comparison alarm started |
| MINOR_ALLOWLIST_FACESNAP_MATCH_ALARM_STOP | 0x58 | Face picture in allowlist comparison alarm ended |
| MINOR_THERMAL_SHIPSDETECTION | 0x5a | Thermal imaging ship detection |
| MINOR_THERMAL_THERMOMETRY_EARLYWARNING_BEGIN | 0x5b | Thermal imaging temperature measurement pre-alarm started |
| MINOR_THERMAL_THERMOMETRY_EARLYWARNING_END | 0x5c | Thermal imaging temperature measurement pre-alarm ended |
| MINOR_THERMAL_THERMOMETRY_ALARM_BEGIN | 0x5d | Thermal imaging temperature measurement alarm started |
| MINOR_THERMAL_THERMOMETRY_ALARM_END | 0x5e | Thermal imaging temperature measurement alarm ended |
| MINOR_THERMAL_THERMOMETRY_DIFF_ALARM_BEGIN | 0x5f | Thermal imaging temperature difference alarm started |
| MINOR_THERMAL_THERMOMETRY_DIFF_ALARM_END | 0x60 | Thermal imaging temperature difference alarm ended |
| MINOR_ FACE_THERMOMETRY_ALARM | 0x63 | Body thermometry alarm |
| MINOR_SAFETY_HELMET_ALARM_START | 0x72 | Hard hat detection alarm stated |
| MINOR_SAFETY_HELMET_ALARM_STOP | 0x73 | Hard hat detection alarm ended |
| MINOR_HFPD_ALARM_START | 0x74 | Frequently appeared person detection alarm started |
| MINOR_HFPD_ALARM_STOP | 0x75 | Frequently appeared person detection alarm ended |
| MINOR_MIXED_TARGET_ALARM_START | 0x76 | Milti-target-type detection alarm started |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_MIXED_TARGET_ALARM_STOP | 0x77 | Milti-target-type detection alarm ended |
| MINOR_VCA_GET_UP_ALARM_BEGIN | 0x80 | Getting up alarm started |
| MINOR_VCA_GET_UP_ALARM_END | 0x81 | Getting up alarm ended |
| MINOR_VCA_ADV_REACH_HEIGHT_ALARM_BEGIN | 0x82 | Climbing alarm started |
| MINOR_VCA_ADV_REACH_HEIGHT_ALARM_END | 0x83 | Climbing alarm ended |
| MINOR_VCA_TOILET_TARRY_ALARM_BEGIN | 0x84 | Toilet overtime alarm started |
| MINOR_VCA_TOILET_TARRY_ALARM_END | 0x85 | Toilet overtime alarm ended |
| MINOR_HUMAN_RECOGNITION_ALARM_BEGIN | 0x86 | Target alarm started |
| MINOR_HUMAN_RECOGNITION_ALARM_END | 0x87 | Target alarm ended |
| MINOR_ACCESS_CONTROLLER_EVENT | 0x100 | Access controller event |
| MINOR_VIDEO_INTERCOM_EVENT | 0x101 | Video intercom event |
| MINOR_GJD_EVENT | 0x102 | GJD security control panel event |
| MINOR_LUMINITE_EVENT | 0x103 | LUMINITE security control panel event |
| MINOR_OPTEX_EVENT | 0x104 | OPTEX security control panel event |
| MINOR_CAMERA_DETECTOR_EVENT | 0x105 | Detector event |
| MINOR_SECURITY_CONTROL_PANEL_EVENT | 0x106 | Security control panel event |
| MINOR_LFPD_ALARM_START | 0x124 | Low frequency person alarm started |
| MINOR_LFPD_ALARM_STOP | 0x125 | Low frequency person alarm stopped |
| MINOR_DATA_PREALARM_ALARM | 0x127 | Network traffic pre-alarm |
| MINOR_VIBRATION_DETECTION_ALARM_BEGIN | 0x132 | Vibration detection alarm started |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_VIBRATION_DETECTION_ALARM_END | 0x133 | Vibration detection alarm stopped |
| MINOR_ALARMIN_SHORT_CIRCUIT | 0x400 | Zone short circuited alarm |
| MINOR_ALARMIN_BROKEN_CIRCUIT | 0x401 | Zone disconnected alarm |
| MINOR_ALARMIN_EXCEPTION | 0x402 | Zone exception alarm |
| MINOR_ALARMIN_RESUME | 0x403 | Zone alarm recovery |
| MINOR_HOST_DESMANTLE_ALARM | 0x404 | Device anti-tamper alarm |
| MINOR_HOST_DESMANTLE_RESUME | 0x405 | Device anti-tamper recovery |
| MINOR_CARD_READER_DESMANTLE_ALARM | 0x406 | Card reader anti-tamper alarm |
| MINOR_CARD_READER_DESMANTLE_RESUME | 0x407 | Card reader anti-tamper recovery |
| MINOR_CASE_SENSOR_ALARM | 0x408 | Event input alarm |
| MINOR_CASE_SENSOR_RESUME | 0x409 | Event input recovery |
| MINOR_STRESS_ALARM | 0x40a | Duress alarm |
| MINOR_OFFLINE_ECENT_NEARLY_FULL | 0x40b | No memory alarm |
| MINOR_CARD_MAX_AUTHENTICATE_FAIL | 0x40c | Card reading failure alarm |
| MINOR_POS_START_ALARM | 0x411 | POS enabled |
| MINOR_POS_END_ALARM | 0x412 | POS disabled |

## MAJOR_EXCEPTION

**Table C-2 Minor Types of Exception Log**

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_RAID_ERROR | 0x20 | RAID exception |
| MINOR_VI_LOST | 0x21 | Video loss |
| MINOR_ILLEGAL_ACCESS | 0x22 | Illegal login |
| MINOR_HD_FULL | 0x23 | HDD full |
| MINOR_HD_ERROR | 0x24 | HDD error |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_DCD_LOST | 0x25 | MODEM offline (reserved) |
| MINOR_IP_CONFLICT | 0x26 | IP address conflicted |
| MINOR_NET_BROKEN | 0x27 | Network disconnected |
| MINOR_REC_ERROR | 0x28 | Recording error |
| MINOR_IPC_NO_LINK | 0x29 | IPC connection exception |
| MINOR_VI_EXCEPTION | 0x2a | Video input exception (only for analog channel) |
| MINOR_IPC_IP_CONFLICT | 0x2b | IP address conflicted of IPC |
| MINOR_SENCE_EXCEPTION | 0x2c | Sence exception |
| MINOR_PIC_REC_ERROR | 0x2d | Capture error. Failed to get pictures. |
| MINOR_VI_MISMATCH | 0x2e | Video format mismatches |
| MINOR_RESOLUTION_MISMATCH | 0x2f | Encoding resolution does not match with the front-end resolution |
| MINOR_RS485_DEVICE_ABNORMAL | 0x3a | RS485 connection status exception |
| MINOR_RS485_DEVICE_REVERT | 0x3b | RS485 connection status exception recovery |
| MINOR_SCREEN_SUBSYSTEM_ABNORMALREBOOT | 0x3c | Sub-board abnormal startup |
| MINOR_SCREEN_SUBSYSTEM_ABNORMALINSERT | 0x3d | Sub-board inserted |
| MINOR_SCREEN_SUBSYSTEM_ABNORMALPULLOUT | 0x3e | Sub-board pulled out |
| MINOR_SCREEN_ABNARMALTEMPERATURE | 0x3f | Temperature exception |
| MINOR_RECORD_OVERFLOW | 0x41 | Buffer overflow |
| MINOR_DSP_ABNORMAL | 0x42 | DSP exception |
| MINOR_ANR_RECORD_FAIED | 0x43 | ANR recording failed |
| MINOR_SPARE_WORK_DEVICE_EXCEPT | 0x44 | Hot spare device working exception |
| MINOR_START_IPC_MAS_FAILED | 0x45 | Failed to enable IPC MAS |
| MINOR_IPCM_CRASH | 0x46 | IPCM abnormal rebooting |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_POE_POWER_EXCEPTION | 0x47 | POE power supply exception |
| MINOR_UPLOAD_DATA_CS_EXCEPTION | 0x48 | Failed to upload data to cloud storage |
| MINOR_DIAL_EXCEPTION | 0x49 | Dial-up exception |
| MINOR_DEV_EXCEPTION_OFFLINE | 0x50 | Device abnormal offline |
| MINOR_UPGRADEFAIL | 0x51 | Remote upgrading failed. |
| MINOR_AI_LOST | 0x52 | Audio loss |
| MINOR_SYNC_IPC_PASSWD | 0x53 | IPC password synchronization exception |
| MINOR_EZVIZ_OFFLINE | 0x54 | Ezviz offline exception |
| MINOR_ACCESSORIES_PLATE | 0x57 | Accessory board exception |
| MINOR_CAMERA_ANGLE_ANOMALY | 0x60 | Camera view angle exception |
| MINOR_FACESNAP_RESOLUTION_OVERFLOW | 0x63 | Overlimit face capture stream resolution |
| MINOR_SMD_RESOLUTION_OVERFLOW | 0x64 | Overlimit SMD stream resolution |
| MINOR_AUDIO_LOSS_EXCEPTION | 0x65 | Audio loss |
| MINOR_SAFETY_HELMET_EXCEPTION | 0x66 | Hard hat detection exception |
| MINOR_VCA_PIC_LENGTH_OVERFLOW | 0x67 | The VCA picture size is too large |
| MINOR_FACE_MODEL_EXCEPTION | 0x68 | Face picture library model synchronization error |
| MINOR_CLUSTER_DEVICE_OFFLINE | 0x70 | The device in cluster is offline |
| MINOR_CLUSTER_CONFIG_FAILED | 0x71 | Configuring the devices in cluster failed. |
| MINOR_CLUSTER_DISASTER_TOLERANCE_EXCEPT | 0x72 | Cluster disaster recovery exception: cluster CM election failed, no enough cluster storage period, no enough cluster bandwidth, no enough channel resource, no enough device. |
| MINOR_CLUSTER_STORFULL_EXCEPTION | 0x73 | The cluster HDD is full. |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_CLUSTER_VERSION_ EXCEPTION | 0x74 | Cluster version exception |
| MINOR_CLUSTER_OFFLINENODE_ EXCEPTION | 0x75 | The offline devices in cluster exceed the limit. |
| MINOR_CLUSTER_RECORDCYCLE_ EXCEPTION | 0x76 | Cluster storage period is not enough. |
| MINOR_CLUSTER_IPCTRANSFER_ EXCEPTION | 0x77 | Cluster network camera migration failed. |
| MINOR_CLUSTER_IPCONFLICT_ EXCEPTION | 0x78 | Cluster IP conflict. |
| MINOR_EVENT_UPLOAD_EXCEPTION | 0x7c | Uploading event failed/Uploaded event lost |
| MINOR_DEV_POWER_ON | 0x400 | Device power on |
| MINOR_DEV_POWER_OFF | 0x401 | Device power off |
| MINOR_WATCH_DOG_RESET | 0x402 | Watch dog resumed |
| MINOR_LOW_BATTERY | 0x403 | Low battery |
| MINOR_BATTERY_RESUME | 0x404 | Battery voltage recovery |
| MINOR_AC_OFF | 0x405 | AC power interrupt |
| MINOR_AC_RESUME | 0x406 | AC power recovery |
| MINOR_NET_RESUME | 0x407 | Network recovery |
| MINOR_FLASH_ABNORMAL | 0x408 | FLASH reading/writing exception |
| MINOR_CARD_READER_OFFLINE | 0x409 | Card reader offline |
| MINOR_CARD_READER_RESUME | 0x40a | Card reader offline recovery |
| MINOR_DSP_START_FAILED | 0x43a | Starting up DSP failed. |
| MINOR_SMART_REGULATION_NOT_ ALLOWED | 0x43b | Intelligent rule is not supported. |
| MINOR_AUXILIARY_BOARD_OFFLINE | 0x43c | Auxiliary board disconnected |
| MINOR_AUXILIARY_BOARD_RESUME | 0x43d | Auxiliary board connected |
| MINOR_IDCARD_SECURITY_ MOUDLE_EXCEPTION | 0x43e | ID card module exception |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_IDCARD_SECURITY_ MOUDLE_RESUME | 0x43f | ID card module restored |
| MINOR_FP_PERIPHERAL_EXCEPTION | 0x440 | Fingerprint recorder exception |
| MINOR_FP_PERIPHERAL_RESUME | 0x441 | Fingerprint recorder restored |
| MINOR_SUBSYSTEM_IP_CONFLICT | 0x4000 | IP conflicted of sub-board |
| MINOR_SUBSYSTEM_NET_BROKEN | 0x4001 | Sub-board offline |
| MINOR_FAN_ABNORMAL | 0x4002 | Fan exception |
| MINOR_BACKPANEL_TEMPERATURE_ ABNORMAL | 0x4003 | Back board temperature exception |
| MINOR_SDCARD_ABNORMAL | 0x4004 | SD card defective |
| MINOR_SDCARD_DAMAGE | 0x4005 | SD card damaged |
| MINOR_OVERVOLTAGE | 0x4019 | High supply voltage |
| MINOR_UNDERVOLTAGE | 0x401a | Low supply voltage |
| MINOR_EZVIZ_UPGRADE_EXCEPTION | 0x401e | Guarding Vision upgrade exception |
| MINOR_HIGH_HD_TEMPERATURE | 0x80 | HDD high temperature |
| MINOR_LOW_HD_TEMPERATURE | 0x81 | HDD low temperature |
| MINOR_HD_IMPACT | 0x82 | HDD impact |
| MINOR_HD_BAD_BLOCK | 0x83 | HDD bad sector |
| MINOR_SEVERE_HD_FAILURE | 0x84 | HDD severe fault |

## MAJOR_OPERATION

**Table C-3 Minor Types of Operation Log**

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_START_DVR | 0x41 | Power on |
| MINOR_STOP_DVR | 0x42 | Shutdown |
| MINOR_STOP_ABNORMAL | 0x43 | Abnormal shutdown |
| MINOR_REBOOT_DVR | 0x44 | Reboot device (local) |
| MINOR_LOCAL_LOGIN | 0x50 | Logged in (local) |
| MINOR_LOCAL_LOGOUT | 0x51 | Logged out (Local) |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_LOCAL_CFG_PARM | 0x52 | Local configuration |
| MINOR_LOCAL_PLAYBYFILE | 0x53 | Playback or download by file (local) |
| MINOR_LOCAL_PLAYBYTIME | 0x54 | Playback or download by time (local) |
| MINOR_LOCAL_START_REC | 0x55 | Start recording (local) |
| MINOR_LOCAL_STOP_REC | 0x56 | Stop recording (local) |
| MINOR_LOCAL_PTZCTRL | 0x57 | PTZ control (local) |
| MINOR_LOCAL_PREVIEW | 0x58 | Live view (local,reserved) |
| MINOR_LOCAL_MODIFY_TIME | 0x59 | Edit time (local,reserved) |
| MINOR_LOCAL_UPGRADE | 0x5a | Local upgrade |
| MINOR_LOCAL_RECFILE_OUTPUT | 0x5b | Backup video files (local) |
| MINOR_LOCAL_FORMAT_HDD | 0x5c | Initialize HDD (local) |
| MINOR_LOCAL_CFGFILE_OUTPUT | 0x5d | Export local configuration files |
| MINOR_LOCAL_CFGFILE_INPUT | 0x5e | Import local configuration files |
| MINOR_LOCAL_COPYFILE | 0x5f | Backup files (local) |
| MINOR_LOCAL_LOCKFILE | 0x60 | Lock video files (local) |
| MINOR_LOCAL_UNLOCKFILE | 0x61 | Unlock video files (local) |
| MINOR_LOCAL_DVR_ALARM | 0x62 | Clear manually and trigger alarm (local) |
| MINOR_IPC_ADD | 0x63 | Add IPC (local) |
| MINOR_IPC_DEL | 0x64 | Delete IPC (local) |
| MINOR_IPC_SET | 0x65 | Set IPC (local) |
| MINOR_LOCAL_START_BACKUP | 0x66 | Start backup (local) |
| MINOR_LOCAL_STOP_BACKUP | 0x67 | Stop backup (local) |
| MINOR_LOCAL_COPYFILE_START_TIME | 0x68 | Start time of local backup |
| MINOR_LOCAL_COPYFILE_END_TIME | 0x69 | End time of local backup |
| MINOR_LOCAL_ADD_NAS | 0x6a | Add NetHDD (local) |
| MINOR_LOCAL_DEL_NAS | 0x6b | Delete NAS (local) |
| MINOR_LOCAL_SET_NAS | 0x6c | Set NAS (local) |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_REMOTE_LOGIN | 0x70 | Login (remote) |
| MINOR_REMOTE_LOGOUT | 0x71 | Logout (local) |
| MINOR_REMOTE_START_REC | 0x72 | Start recording (remote) |
| MINOR_REMOTE_STOP_REC | 0x73 | Stop recording (remote) |
| MINOR_START_TRANS_CHAN | 0x74 | Start transparent transmission |
| MINOR_STOP_TRANS_CHAN | 0x75 | Stop transparent transmission |
| MINOR_REMOTE_GET_PARM | 0x76 | Get parameters (remote) |
| MINOR_REMOTE_CFG_PARM | 0x77 | Remote configuration |
| MINOR_REMOTE_GET_STATUS | 0x78 | Get status (remote) |
| MINOR_REMOTE_ARM | 0x79 | Arm (remote) |
| MINOR_REMOTE_DISARM | 0x7a | Disarm (remote) |
| MINOR_REMOTE_REBOOT | 0x7b | Reboot (remote) |
| MINOR_START_VT | 0x7c | Start two-way audio |
| MINOR_STOP_VT | 0x7d | Stop two-way audio |
| MINOR_REMOTE_UPGRADE | 0x7e | Remote upgrade |
| MINOR_REMOTE_PLAYBYFILE | 0x7f | Playback by file (remote) |
| MINOR_REMOTE_PLAYBYTIME | 0x80 | Playback by time (remote) |
| MINOR_REMOTE_PTZCTRL | 0x81 | PTZ control (remote) |
| MINOR_REMOTE_FORMAT_HDD | 0x82 | Format HDD (remote) |
| MINOR_REMOTE_STOP | 0x83 | Shutdown (remote) |
| MINOR_REMOTE_LOCKFILE | 0x84 | Lock files (remote) |
| MINOR_REMOTE_UNLOCKFILE | 0x85 | Unlock files (remote) |
| MINOR_REMOTE_CFGFILE_OUTPUT | 0x86 | Export configuration files (remote) |
| MINOR_REMOTE_CFGFILE_INTPUT | 0x87 | Import configuration files (remote) |
| MINOR_REMOTE_RECFILE_OUTPUT | 0x88 | Export video files (remote) |
| MINOR_REMOTE_DVR_ALARM | 0x89 | Clear manually and trigger alarm (remote) |
| MINOR_REMOTE_IPC_ADD | 0x8a | Add IPC (remote) |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_REMOTE_IPC_DEL | 0x8b | Delete IPC (remote) |
| MINOR_REMOTE_IPC_SET | 0x8c | Set IPC (remote) |
| MINOR_REBOOT_VCA_LIB | 0x8d | Reboot intelligent library |
| MINOR_REMOTE_ADD_NAS | 0x8e | Add NAS (remote) |
| MINOR_REMOTE_DEL_NAS | 0x8f | Delete NAS (remote) |
| MINOR_REMOTE_SET_NAS | 0x90 | Set NAS (remote) |
| MINOR_LOCAL_START_REC_CDRW | 0x91 | Start burning (local) |
| MINOR_LOCAL_STOP_REC_CDRW | 0x92 | Stop burning (local) |
| MINOR_REMOTE_START_REC_CDRW | 0x93 | Start burning (remote) |
| MINOR_REMOTE_STOP_REC_CDRW | 0x94 | Stop burning (remote) |
| MINOR_LOCAL_PIC_OUTPUT | 0x95 | Back up pictures (local) |
| MINOR_REMOTE_PIC_OUTPUT | 0x96 | Back up pictures (remote) |
| MINOR_LOCAL_INQUEST_RESUME | 0x97 | Resume inquest event (local) |
| MINOR_REMOTE_INQUEST_RESUME | 0x98 | Resume inquest event (remote) |
| MINOR_LOCAL_ADD_FILE | 0x99 | Import files (local) |
| MINOR_REMOTE_DELETE_HDISK | 0x9a | Delete exception or nonexistent HDD |
| MINOR_REMOTE_LOAD_HDISK | 0x9b | Load HDD (remote) |
| MINOR_REMOTE_UNLOAD_HDISK | 0x9c | Unload HDD (remote) |
| MINOR_LOCAL_OPERATE_LOCK | 0x9d | Lock (local) |
| MINOR_LOCAL_OPERATE_UNLOCK | 0x9e | Unlock (local) |
| MINOR_LOCAL_DEL_FILE | 0x9f | Delete inquest files (local) |
| MINOR_REMOTE_BYPASS | 0xd0 | Bypass (remote) |
| MINOR_REMOTE_UNBYPASS | 0xd1 | Bypass recovery (remote) |
| MINOR_REMOTE_SET_ALARMIN_CFG | 0xd2 | Set alarm input parameters (remote) |
| MINOR_REMOTE_GET_ALARMIN_CFG | 0xd3 | Get alarm input parameters (remote) |
| MINOR_REMOTE_SET_ALARMOUT_CFG | 0xd4 | Set alarm output parameters (remote) |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_REMOTE_GET_ALARMOUT_CFG | 0xd5 | Get alarm output parameters (remote) |
| MINOR_REMOTE_ALARMOUT_OPEN_MAN | 0xd6 | Enable alarm output manually (remote) |
| MINOR_REMOTE_ALARMOUT_CLOSE_MAN | 0xd7 | Disable alarm output manually (remote) |
| MINOR_REMOTE_ALARM_ENABLE_CFG | 0xd8 | Enable/Disable RS-485 serial port of security control panel (remote) |
| MINOR_DBDATA_OUTPUT | 0xd9 | Export database records |
| MINOR_DBDATA_INPUT | 0xda | Import database records |
| MINOR_MU_SWITCH | 0xdb | Cascading switch |
| MINOR_MU_PTZ | 0xdc | Cascading PTZ control |
| MINOR_REMOTE_INQUEST_DEL_FILE | 0xde | Delete file (remote) |
| MINOR_LOCAL_CONF_REB_RAID | 0x101 | Configure auto-rebuild (local) |
| MINOR_LOCAL_CONF_SPARE | 0x102 | Configure hot spare (local) |
| MINOR_LOCAL_ADD_RAID | 0x103 | Create array (local) |
| MINOR_LOCAL_DEL_RAID | 0x104 | Delete array (local) |
| MINOR_LOCAL_MIG_RAID | 0x105 | Migrate array (local) |
| MINOR_LOCAL_REB_RAID | 0x106 | Rebuild array manually (local) |
| MINOR_LOCAL_QUICK_CONF_RAID | 0x107 | One-touch configuration (local) |
| MINOR_LOCAL_ADD_VD | 0x108 | Create virtual disk (local) |
| MINOR_LOCAL_DEL_VD | 0x109 | Delete virtual disk (local) |
| MINOR_LOCAL_RP_VD | 0x10a | Repair virtual disk (local) |
| MINOR_LOCAL_FORMAT_EXPANDVD | 0x10b | Expand virtual disk (local) |
| MINOR_LOCAL_RAID_UPGRADE | 0x10c | Upgrade RAID (local) |
| MINOR_LOCAL_STOP_RAID | 0x10d | Pause RAID (local, unplug safely) |
| MINOR_REMOTE_CONF_REB_RAID | 0x111 | Configure auto-rebuild (remote) |
| MINOR_REMOTE_CONF_SPARE | 0x112 | Configure hot spare (remote) |
| MINOR_REMOTE_ADD_RAID | 0x113 | Create array (remote) |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_REMOTE_DEL_RAID | 0x114 | Delete array (remote) |
| MINOR_REMOTE_MIG_RAID | 0x115 | Migrate array (remote) |
| MINOR_REMOTE_REB_RAID | 0x116 | Rebuild array manually (remote) |
| MINOR_REMOTE_QUICK_CONF_RAID | 0x117 | One-touch configuration (remote) |
| MINOR_REMOTE_ADD_VD | 0x118 | Create virtual disk (remote) |
| MINOR_REMOTE_DEL_VD | 0x119 | Delete virtual disk (remote) |
| MINOR_REMOTE_RP_VD | 0x11a | Repair virtual disk (remote) |
| MINOR_REMOTE_FORMAT_ EXPANDVD | 0x11b | Expand virtual disk (remote) |
| MINOR_REMOTE_RAID_UPGRADE | 0x11c | Upgrade RAID (remote) |
| MINOR_REMOTE_STOP_RAID | 0x11d | Pause RAID (remote, unplug safely) |
| MINOR_LOCAL_START_PIC_REC | 0x121 | Start capture (local) |
| MINOR_LOCAL_STOP_PIC_REC | 0x122 | Stop capture (local) |
| MINOR_LOCAL_SET_SNMP | 0x125 | Set SNMP (local) |
| MINOR_LOCAL_TAG_OPT | 0x126 | Tag operation (local) |
| MINOR_REMOTE_START_PIC_REC | 0x131 | Start capture (remote) |
| MINOR_REMOTE_STOP_PIC_REC | 0x132 | Stop capture (remote) |
| MINOR_REMOTE_SET_SNMP | 0x135 | Set SNMP (remote) |
| MINOR_REMOTE_TAG_OPT | 0x136 | Tag operation (remote) |
| MINOR_SCHEDULE_ ANGLECALIBRATION | 0x139 | Scheduled angle calibration |
| MINOR_LOCAL_VOUT_SWITCH | 0x140 | Switch output (local) |
| MINOR_STREAM_CABAC | 0x141 | Encoding performance configuration |
| MINOR_LOCAL_SPARE_OPT | 0x142 | N+1 hot spare operation (local) |
| MINOR_REMOTE_SPARE_OPT | 0x143 | N+1 hot spare operation (remote) |
| MINOR_LOCAL_IPCCFGFILE_OUTPUT | 0x144 | Export IPC configuration file (local) |
| MINOR_LOCAL_IPCCFGFILE_INPUT | 0x145 | Import IPC configuration file (local) |
| MINOR_LOCAL_IPC_UPGRADE | 0x146 | Upgrade IPC (local) |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_REMOTE_IPCCFGFILE_ OUTPUT | 0x147 | Export IPC configuration file (remote) |
| MINOR_REMOTE_IPCCFGFILE_INPUT | 0x148 | Import IPC configuration file (remote) |
| MINOR_REMOTE_IPC_UPGRADE | 0x149 | Upgrade IPC (remote) |
| MINOR_LOCAL_UNLOAD_HDISK | 0x150 | Uninstall HDD (local) |
| MINOR_LOCAL_AUDIO_MIX | 0x151 | Set audio mix parameters (local) |
| MINOR_REMOTE_AUDIO_MIX | 0x152 | Set audio mix parameters (remote) |
| MINOR_LOCAL_TRIAL_PAUSE | 0x153 | Pause inquest (local) |
| MINOR_LOCAL_TRIAL_RESUME | 0x154 | Resume inquest (local) |
| MINOR_REMOTE_TRIAL_PAUSE | 0x155 | Pause inquest (remote) |
| MINOR_REMOTE_TRIAL_RESUME | 0x156 | Resume inquest (remote) |
| MINOR_REMOTE_MODIFY_ VERIFICATION_CODE | 0x157 | Change the verification code of the system |
| MINOR_SET_MULTI_MASTER | 0x201 | Set main screen of multi-screen controller |
| MINOR_SET_MULTI_SLAVE | 0x202 | Set sub-screen of multi-screen controller |
| MINOR_CANCEL_MULTI_MASTER | 0x203 | Cancel main screen of multi-screen controller |
| MINOR_CANCEL_MULTI_SLAVE | 0x204 | Cancel sub-screen of multi-screen controller |
| MINOR_SCREEN_SET_INPUT | 0x251 | Edit input source |
| MINOR_SCREEN_SET_OUTPUT | 0x252 | Edit output channel |
| MINOR_SCREEN_SET_OSD | 0x253 | Edit virtual LED |
| MINOR_SCREEN_SET_LOGO | 0x254 | Edit LOGO |
| MINOR_SCREEN_SET_LAYOUT | 0x255 | Set scene |
| MINOR_SCREEN_PICTUREPREVIEW | 0x256 | Display operation |
| MINOR_SCREEN_GET_OSD | 0x257 | Get virtual LED |
| MINOR_SCREEN_GET_LAYOUT | 0x258 | Get scene |
| MINOR_SCREEN_LAYOUT_CTRL | 0x259 | Scene control |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_GET_ALL_VALID_WND | 0x260 | Get all the valid windows |
| MINOR_GET_SIGNAL_WND | 0x261 | Get single window information |
| MINOR_REMOTE_CLUSTER_MODE_CONFIG | 0x261c | Remote operation: cluster mode configuration |
| MINOR_LOCAL_CLUSTER_MODE_CONFIG | 0x261d | Local operation: cluster mode configuration |
| MINOR_REMOTE_CLUSTER_NETWORK_CONFIG | 0x261e | Remote operation: NVR in cluster configuration |
| MINOR_LOCAL_CLUSTER_NETWORK_CONFIG | 0x261f | Local operation: NVR in cluster configuration |
| MINOR_REMOTE_CLUSTER_ADD_DEVICE | 0x2620 | Remote operation: Add device to cluster |
| MINOR_WINDOW_CTRL | 0x262 | Window control |
| MINOR_LOCAL_CLUSTER_ADD_DEVICE | 0x2621 | Local operation: Add device to cluster |
| MINOR_REMOTE_CLUSTER_DEL_DEVICE | 0x2622 | Remote operation: Delete device from cluster |
| MINOR_LOCAL_CLUSTER_DEL_DEVICE | 0x2623 | Local operation: Delete device from cluster |
| MINOR_REMOTE_HFPD_CFG | 0x2624 | Remote operation: frequently appeared person detection configuration |
| MINOR_REMOTE_FACE_CONTRAST_TASK | 0x2625 | Remote operation: face picture comparison task configuration |
| MINOR_REMOTE_LFPD_CFG | 0x2626 | Remote configuration of low frequency person detection |
| MINOR_REMOTE_IOTCFGFILE_INPUT | 0x2627 | Remote operation: import IoT configuration file |
| MINOR_REMOTE_IOTCFGFILE_OUTPUT | 0x2628 | Remote operation: export IoT configuration file |
| MINOR_LOCAL_IOT_ADD | 0x2629 | Local operation: add IoT channel |
| MINOR_REMOTE_IOT_ADD | 0x262a | Remote operation: add IoT channel |
| MINOR_LOCAL_IOT_DEL | 0x262b | Local operation: delete IoT channel |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_REMOTE_IOT_DEL | 0x262c | Remote operation: delete IoT channel |
| MINOR_LOCAL_IOT_SET | 0x262d | Local operation: configure IoT channel |
| MINOR_REMOTE_IOT_SET | 0x262e | Remote operation: configure IoT channel |
| MINOR_LOCAL_IOTCFGFILE_INPUT | 0x262f | Local operation: import IoT configuration file |
| MINOR_LOCAL_IOTCFGFILE_OUTPUT | 0x2630 | Local operation: export IoT configuration file |
| MINOR_GET_LAYOUT_LIST | 0x263 | Get scene list |
| MINOR_LAYOUT_CTRL | 0x264 | Scene control |
| MINOR_SET_LAYOUT | 0x265 | Set single scene |
| MINOR_GET_SIGNAL_LIST | 0x266 | Get input signal source list |
| MINOR_GET_PLAN_LIST | 0x267 | Get plan list |
| MINOR_SET_PLAN | 0x268 | Edit plan |
| MINOR_CTRL_PLAN | 0x269 | Control plan |
| MINOR_CTRL_SCREEN | 0x270 | Screen control |
| MINOR_ADD_NETSIG | 0x271 | Add signal source |
| MINOR_SET_NETSIG | 0x272 | Edit signal source |
| MINOR_SET_DECBDCFG | 0x273 | Set decoding board parameters |
| MINOR_GET_DECBDCFG | 0x274 | Get decoding board parameters |
| MINOR_GET_DEVICE_STATUS | 0x275 | Get device information |
| MINOR_UPLOAD_PICTURE | 0x276 | Upload background |
| MINOR_SET_USERPWD | 0x277 | Set password |
| MINOR_ADD_LAYOUT | 0x278 | Add scene |
| MINOR_DEL_LAYOUT | 0x279 | Delete scene |
| MINOR_DEL_NETSIG | 0x280 | Delete signal source |
| MINOR_ADD_PLAN | 0x281 | Add plan |
| MINOR_DEL_PLAN | 0x282 | Delete plan |
| MINOR_GET_EXTERNAL_MATRIX_CFG | 0x283 | Get external matrix settings |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_SET_EXTERNAL_MATRIX_CFG | 0x284 | Set external matrix |
| MINOR_GET_USER_CFG | 0x285 | Get user settings |
| MINOR_SET_USER_CFG | 0x286 | Set user |
| MINOR_GET_DISPLAY_PANEL_LINK_CFG | 0x287 | Get video wall connection settings |
| MINOR_SET_DISPLAY_PANEL_LINK_CFG | 0x288 | Set video wall connection |
| MINOR_GET_WALLSCENE_PARAM | 0x289 | Get video wall scene |
| MINOR_SET_WALLSCENE_PARAM | 0x28a | Set video wall scene |
| MINOR_GET_CURRENT_WALLSCENE | 0x28b | Get current scene |
| MINOR_SWITCH_WALLSCENE | 0x28c | Scene switch |
| MINOR_LOCAL_LOAD_HDISK | 0x300 | Load HDD (local) |
| MINOR_LOCAL_DELETE_HDISK | 0x301 | Delete exception or nonexistence HDD (local) |
| MINOR_REMOTE_CFG_POE_WORK_MODE | 0x361 | Remotely set PoE working mode |
| MINOR_LOCAL_CFG_POE_WORK_MODE | 0x362 | Locally set PoE working mode |
| MINOR_REMOTE_CFG_FACE_CONTRAST | 0x363 | Remotely set face comparison |
| MINOR_LOCAL_CFG_FACE_CONTRAST | 0x364 | Locally set face comparison |
| MINOR_REMOTE_CFG_ALLOWLIST_FACE_CONTRAST | 0x365 | Remotely set face comparison in allowlist |
| MINOR_LOCAL_CHECK_TIME | 0x367 | Manual time synchronization (local) |
| MINOR_LOCAL_CFG_ALLOWLIST_FACE_CONTRAST | 0x366 | Locally set face comparison in allowlist |
| MINOR_REMOTE_CFG_WIRELESS_DIALPARAM | 0x36c | Configure wireless dial-up parameters remotely |
| MINOR_LOCAL_CFG_WIRELESS_DIALPARAM | 0x36d | Configure wireless dial-up parameters locally |
| MINOR_REMOTE_CFG_WIRELESS_SMSPARAM | 0x36e | Configure wireless message parameters remotely |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_LOCAL_CFG_WIRELESS_ SMSPARAM | 0x36f | Configure wireless message parameters locally |
| MINOR_REMOTE_CFG_WIRELESS_ SMSSElFHELP | 0x370 | Configure SMS self-service parameters remotely |
| MINOR_LOCAL_CFG_WIRELESS_ SMSSElFHELP | 0x371 | Configure SMS self-service parameters locally |
| MINOR_REMOTE_CFG_WIRELESS_ NETFLOWPARAM | 0x372 | Configure wireless traffic parameters remotely |
| MINOR_LOCAL_CFG_WIRELESS_ NETFLOWPARAM | 0x373 | Configure wireless traffic parameters locally |
| MINOR_REMOTE_OPEN_DOOR | 0x400 | Open door (remote) |
| MINOR_REMOTE_CLOSE_DOOR | 0x401 | Close door (remote) |
| MINOR_REMOTE_ALWAYS_OPEN | 0x402 | Remain open (remote) |
| MINOR_REMOTE_ALWAYS_CLOSE | 0x403 | Remain closed (remote) |
| MINOR_REMOTE_CHECK_TIME | 0x404 | Manual time synchronization (remote) |
| MINOR_NTP_CHECK_TIME | 0x405 | NTP auto time synchronization |
| MINOR_REMOTE_CLEAR_CARD | 0x406 | Clear card No. (remote) |
| MINOR_REMOTE_RESTORE_CFG | 0x407 | Resume default parameters (remote) |
| MINOR_ALARMIN_ARM | 0x408 | Zone arming |
| MINOR_ALARMIN_DISARM | 0x409 | Zone disarming |
| MINOR_LOCAL_RESTORE_CFG | 0x40a | Resume default parameters (local) |
| MINOR_OFFLINE_DATA_OUTPUT | 0x423 | Exported offline collection data |
| MINOR_CREATE_SSH_LINK | 0x42d | Connected with SSH |
| MINOR_CLOSE_SSH_LINK | 0x42e | Disconnected with SSH |
| MINOR_SET_TRIGGERMODE_CFG | 0x1001 | Set trigger mode parameters |
| MINOR_GET_TRIGGERMODE_CFG | 0x1002 | Get trigger mode parameters |
| MINOR_SET_IOOUT_CFG | 0x1003 | Set IO output parameters |
| MINOR_GET_IOOUT_CFG | 0x1004 | Get IO output parameters |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_GET_TRIGGERMODE_DEFAULT | 0x1005 | Get recommended parameters of trigger mode |
| MINOR_GET_ITCSTATUS | 0x1006 | Get status detection parameters |
| MINOR_SET_STATUS_DETECT_CFG | 0x1007 | Set status detection parameters |
| MINOR_GET_STATUS_DETECT_CFG | 0x1008 | Get status detection parameters |
| MINOR_GET_VIDEO_TRIGGERMODE_CFG | 0x1009 | Gt parameters of video e-police mode |
| MINOR_SET_VIDEO_TRIGGERMODE_CFG | 0x100a | Set parameters of video e-police mode |
| MINOR_WEB_AUTHENTICATION | 0x111d | Web authentication method configuration |
| MINOR_HTTPS_ENABLED | 0x111f | HTTPS switch configuration |
| MINOR_SET_NETWORK_CFG | 0x112b | Set network parameters |
| MINOR_GET_NETWORK_CFG | 0x112c | Get network parameters |
| MINOR_LOCAL_ADD_CAR_INFO | 0x2001 | Add vehicle information (local) |
| MINOR_LOCAL_MOD_CAR_INFO | 0x2002 | Edit vehicle information (local) |
| MINOR_LOCAL_DEL_CAR_INFO | 0x2003 | Delete vehicle information (local) |
| MINOR_LOCAL_FIND_CAR_INFO | 0x2004 | Search vehicle information (local) |
| MINOR_LOCAL_ADD_MONITOR_INFO | 0x2005 | Add arming information (local) |
| MINOR_LOCAL_MOD_MONITOR_INFO | 0x2006 | Edit arming information (local) |
| MINOR_LOCAL_DEL_MONITOR_INFO | 0x2007 | Delete arming information (local) |
| MINOR_LOCAL_FIND_MONITOR_INFO | 0x2008 | Search arming information (local) |
| MINOR_LOCAL_FIND_NORMAL_PASS_INFO | 0x2009 | Search normal passing information (local) |
| MINOR_LOCAL_FIND_ABNORMAL_PASS_INFO | 0x200a | Search abnormal passing information (local) |
| MINOR_LOCAL_FIND_PEDESTRIAN_PASS_INFO | 0x200b | Search normal passing information (local) |
| MINOR_LOCAL_PIC_PREVIEW | 0x200c | Preview local picture |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_LOCAL_SET_GATE_PARM_CFG | 0x200d | Set local exit&entrance parameters |
| MINOR_LOCAL_GET_GATE_PARM_CFG | 0x200e | Get local exit&entrance parameters |
| MINOR_LOCAL_SET_DATAUPLOAD_PARM_CFG | 0x200f | Set local data uploading parameters |
| MINOR_LOCAL_GET_DATAUPLOAD_PARM_CFG | 0x2010 | Get local data uploading parameters |
| MINOR_LOCAL_DEVICE_CONTROL | 0x2011 | Control local device (Open/close barrier) |
| MINOR_LOCAL_ADD_EXTERNAL_DEVICE_INFO | 0x2012 | Add peripheral information (local) |
| MINOR_LOCAL_MOD_EXTERNAL_DEVICE_INFO | 0x2013 | Edit peripheral information (local) |
| MINOR_LOCAL_DEL_EXTERNAL_DEVICE_INFO | 0x2014 | Delete peripheral information (local) |
| MINOR_LOCAL_FIND_EXTERNAL_DEVICE_INFO | 0x2015 | Search peripheral information (local) |
| MINOR_LOCAL_ADD_CHARGE_RULE | 0x2016 | Add parking rule (local) |
| MINOR_LOCAL_MOD_CHARGE_RULE | 0x2017 | Edit parking rule (local) |
| MINOR_LOCAL_DEL_CHARGE_RULE | 0x2018 | Delete parking rule (local) |
| MINOR_LOCAL_FIND_CHARGE_RULE | 0x2019 | Search parking rule (local) |
| MINOR_LOCAL_COUNT_NORMAL_CURRENTINFO | 0x2020 | Normal passing information statistics (local) |
| MINOR_LOCAL_EXPORT_NORMAL_CURRENTINFO_REPORT | 0x2021 | Export normal passing information report (local) |
| MINOR_LOCAL_COUNT_ABNORMAL_CURRENTINFO | 0x2022 | Abnormal passing information statistics (local) |
| MINOR_LOCAL_EXPORT_ABNORMAL_CURRENTINFO_REPORT | 0x2023 | Export abnormal passing information report (local) |
| MINOR_LOCAL_COUNT_PEDESTRIAN_CURRENTINFO | 0x2024 | Pedestrian passing information statistics (local) |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_LOCAL_EXPORT_ PEDESTRIAN_CURRENTINFO_REPORT | 0x2025 | Export pedestrian passing information report (local) |
| MINOR_LOCAL_FIND_CAR_ CHARGEINFO | 0x2026 | Search vehicle passing fee information (local) |
| MINOR_LOCAL_COUNT_CAR_ CHARGEINFO | 0x2027 | Vehicle passing fee information statistics (local) |
| MINOR_LOCAL_EXPORT_CAR_ CHARGEINFO_REPORT | 0x2028 | Export vehicle passing fee information report (local) |
| MINOR_LOCAL_FIND_SHIFTINFO | 0x2029 | Search shift information (local) |
| MINOR_LOCAL_FIND_CARDINFO | 0x2030 | Search card information (local) |
| MINOR_LOCAL_ADD_RELIEF_RULE | 0x2031 | Add discount rule (local) |
| MINOR_LOCAL_MOD_RELIEF_RULE | 0x2032 | Edit discount rule (local) |
| MINOR_LOCAL_DEL_RELIEF_RULE | 0x2033 | Delete discount rule (local) |
| MINOR_LOCAL_FIND_RELIEF_RULE | 0x2034 | Search discount rule (local) |
| MINOR_LOCAL_GET_ENDETCFG | 0x2035 | Get configuration parameters for entrance&exit station offline detection (local) |
| MINOR_LOCAL_SET_ENDETCFG | 0x2036 | Set configuration parameters for entrance&exit station offline detection (local) |
| MINOR_LOCAL_SET_ENDEV_ ISSUEDDATA | 0x2037 | Set card applying information for entrance&exit station (local) |
| MINOR_LOCAL_DEL_ENDEV_ ISSUEDDATA | 0x2038 | Clear card applying information for entrance&exit station (local) |
| MINOR_REMOTE_DEVICE_CONTROL | 0x2101 | Remote device control |
| MINOR_REMOTE_SET_GATE_PARM_ CFG | 0x2102 | Set entrance&exit parameters for remote configuration |
| MINOR_REMOTE_GET_GATE_PARM_ CFG | 0x2103 | Get entrance&exit parameters for remote configuration |
| MINOR_REMOTE_SET_DATAUPLOAD_ PARM_CFG | 0x2104 | Set data uploading parameters for remote configuration |
| MINOR_REMOTE_GET_DATAUPLOAD_ PARM_CFG | 0x2105 | Get data uploading parameters for remote configuration |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_REMOTE_GET_BASE_INFO | 0x2106 | Get terminal basic information (remote) |
| MINOR_REMOTE_GET_OVERLAP_CFG | 0x2107 | Get text overlay parameters (remote) |
| MINOR_REMOTE_SET_OVERLAP_CFG | 0x2108 | Set text overlay parameters (remote) |
| MINOR_REMOTE_GET_ROAD_INFO | 0x2109 | Get crossing information (remote) |
| MINOR_REMOTE_START_TRANSCHAN | 0x210a | Build data synchronizing server (remote) |
| MINOR_REMOTE_GET_ECTWORKSTATE | 0x210b | Get entrance&exit terminal working status (remote) |
| MINOR_REMOTE_GET_ECTCHANINFO | 0x210c | Get entrance&exit terminal channel status (remote) |
| MINOR_REMOTE_ADD_EXTERNAL_DEVICE_INFO | 0x210d | Add peripheral information (remote) |
| MINOR_REMOTE_MOD_EXTERNAL_DEVICE_INFO | 0x210e | Edit peripheral information (remote) |
| MINOR_REMOTE_GET_ENDETCFG | 0x210f | Get configuration parameters for entrance&exit station offline detection (remote) |
| MINOR_REMOTE_SET_ENDETCFG | 0x2110 | Set configuration parameters for entrance&exit station offline detection (remote) |
| MINOR_REMOTE_ENDEV_ISSUEDDATA | 0x2111 | Set card applying information for entrance&exit station (remote) |
| MINOR_REMOTE_DEL_ENDEV_ISSUEDDATA | 0x2112 | Clear card applying information for entrance&exit station (remote) |
| MINOR_REMOTE_ON_CTRL_LAMP | 0x2115 | Enable remote control parking indicator |
| MINOR_REMOTE_OFF_CTRL_LAMP | 0x2116 | Disable remote control parking indicator |
| MINOR_SET_VOICE_LEVEL_PARAM | 0x2117 | Set volume |
| MINOR_SET_VOICE_INTERCOM_PARAM | 0x2118 | Set recording volume |
| MINOR_SET_INTELLIGENT_PARAM | 0x2119 | VCA configuration |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_LOCAL_SET_RAID_SPEED | 0x211a | Set raid speed (local) |
| MINOR_REMOTE_SET_RAID_SPEED | 0x211b | Set raid speed (remote) |
| MINOR_REMOTE_CREATE_STORAGE_POOL | 0x211c | Add storage pool (remote) |
| MINOR_REMOTE_DEL_STORAGE_POOL | 0x211d | Delete storage pool (remote) |
| MINOR_REMOTE_DEL_PIC | 0x2120 | Delete picture data (remote) |
| MINOR_REMOTE_DEL_RECORD | 0x2121 | Delete recording data (remote) |
| MINOR_REMOTE_CLOUD_ENABLE | 0x2123 | Enable cloud storage (remote) |
| MINOR_REMOTE_CLOUD_DISABLE | 0x2124 | Disable cloud storage (remote) |
| MINOR_REMOTE_CLOUD_MODIFY_PARAM | 0x2125 | Edit cloud storage pool parameters (remote) |
| MINOR_REMOTE_CLOUD_MODIFY_VOLUME | 0x2126 | Edit cloud storage pool capacity (remote) |
| MINOR_REMOTE_CREATE_MOD_VIEWLIB_SPACE | 0x2200 | Create/edit image library space (remote) |
| MINOR_REMOTE_DELETE_VIEWLIB_FILE | 0x2201 | Delete image library file (remote) |
| MINOR_REMOTE_DOWNLOAD_VIEWLIB_FILE | 0x2202 | Download image library file(s) (remote) |
| MINOR_REMOTE_UPLOAD_VIEWLIB_FILE | 0x2203 | Upload image library file(s) (remote) |
| MINOR_LOCAL_CREATE_MOD_VIEWLIB_SPACE | 0x2204 | Create/edit image library space (local) |
| MINOR_REMOTE_CONFERENCE_CONFIG | 0x2501 | MCU meeting configuration |
| MINOR_REMOTE_TERMINAL_CONFIG | 0x2502 | MCU terminal configuration |
| MINOR_REMOTE_GROUP_CONFIG | 0x2503 | MCU group configuration |
| MINOR_REMOTE_CONFERENCE_CTRL | 0x2504 | MCU meeting control |
| MINOR_REMOTE_TERMINAL_CTRL | 0x2505 | MCU terminal control |
| MINOR_LOCAL_RESET_LOGIN_PASSWORD | 0x2600 | Reset password for admin user (local) |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_REMOTE_RESET_LOGIN_ PASSWORD | 0x2601 | Reset password for admin user (remote) |
| MINOR_LOCAL_FACE_BASE_CREATE | 0x2602 | Create local face picture library |
| MINOR_REMOTE_FACE_BASE_CREATE | 0x2603 | Create remote face picture library |
| MINOR_LOCAL_FACE_BASE_MODIFY | 0x2604 | Edit local face picture library |
| MINOR_REMOTE_FACE_BASE_ MODIFY | 0x2605 | Edit remote face picture library |
| MINOR_LOCAL_FACE_BASE_DELETE | 0x2606 | Delete local face picture library |
| MINOR_REMOTE_FACE_BASE_DELETE | 0x2607 | Delete remote face picture library |
| MINOR_LOCAL_FACE_DATA_APPEND | 0x2608 | Add face data locally |
| MINOR_REMOTE_FACE_DATA_ APPEND | 0x2609 | Add face data remotely |
| MINOR_LOCAL_FACE_DATA_SEARCH | 0x2610 | Search local face comparison data |
| MINOR_REMOTE_FACE_DATA_ SEARCH | 0x2611 | Search remote face comparison data |
| MINOR_LOCAL_FACE_DATA_ANALYSIS | 0x2612 | Analysis picture locally |
| MINOR_REMOTE_FACE_DATA_ ANALYSIS | 0x2613 | Analysis picture remotely |
| MINOR_LOCAL_FACE_DATA_EDIT | 0x2614 | Edit local face data |
| MINOR_REMOTE_FACE_DATA_EDIT | 0x2615 | Edit remote face data |
| MINOR_LOCAL_FACE_DATA_DELETE | 0x2616 | Delete local face data |
| MINOR_REMOTE_FACE_DATA_DELET | 0x2617 | Delete remote face data |
| MINOR_LOCAL_VCA_ANALYSIS_CFG | 0x2618 | Set local intelligent analysis |
| MINOR_REMOTE_VCA_ANALYSIS_CFG | 0x2619 | Set remote intelligent analysis |
| MINOR_LOCAL_FACE_BASE_IMPORT | 0x261a | Import face picture library locally |
| MINOR_LOCAL_FACE_BASE_EXPORT | 0x261b | Export face picture library locally |
| MINOR_LOCAL_ADDRESS_FILTER_ CONFIG | 0x2633 | Local address filter configuration |
| MINOR_REMOTE_ADDRESS_FILTER_ CONFIG | 0x2634 | Remote address filter configuration |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_LOCAL_SSD_UPGRADE_START | 0x2639 | Upgrade of local SSD file system started |
| MINOR_LOCAL_SSD_UPGRADE_STOP | 0x2640 | Upgrade of local SSD file system ended |
| MINOR_REMOTE_SSD_UPGRADE_START | 0x2641 | Upgrade of remote SSD file system started |
| MINOR_REMOTE_SSD_UPGRADE_STOP | 0x2642 | Upgrade of remote SSD file system ended |
| MINOR_LOCAL_AUTO_SWITCH_CONFIG | 0x2647 | Configure auto power on or off locally |
| MINOR_REMOTE_AUTO_SWITCH_CONFIG | 0x2648 | Configure auto power on or off remotely |
| MINOR_REMOTE_AI_MODEL_ADD | 0x2650 | Add model package |
| MINOR_REMOTE_AI_MODEL_QUERY | 0x2651 | Search for model package |
| MINOR_REMOTE_AI_MODEL_DELETE | 0x2652 | Delete model package |
| MINOR_REMOTE_AI_MODEL_UPDATE | 0x2653 | Update model package |
| MINOR_REMOTE_AI_PICTURE_POLLING_TASK_ADD | 0x2654 | Add picture polling task |
| MINOR_REMOTE_AI_PICTURE_POLLING_TASK_QUERY | 0x2655 | Search for picture polling task |
| MINOR_REMOTE_AI_PICTURE_POLLING_TASK_DELETE | 0x2656 | Delete picture polling task |
| MINOR_REMOTE_AI_PICTURE_POLLING_TASK_MODIFY | 0x2657 | Edit picture polling task |
| MINOR_REMOTE_AI_VIDEO_POLLING_TASK_ADD | 0x2658 | Add video polling task |
| MINOR_REMOTE_AI_VIDEO_POLLING_TASK_QUERY | 0x2659 | Search for video polling task |
| MINOR_REMOTE_AI_VIDEO_POLLING_TASK_DELETE | 0x265A | Delete video polling task |
| MINOR_REMOTE_AI_VIDEO_POLLING_TASK_MODIFY | 0x265B | Edit video polling task |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_REMOTE_AI_PICTURE_TASK_ADD | 0x265C | Add picture task |
| MINOR_REMOTE_AI_PICTURE_TASK_QUERY | 0x265D | Search for picture task |
| MINOR_REMOTE_AI_PICTURE_TASK_DELETE | 0x265E | Delete picture task |
| MINOR_REMOTE_AI_PICTURE_TASK_MODIFY | 0x265F | Edit picture task |
| MINOR_REMOTE_AI_VIDEO_TASK_ADD | 0x2660 | Add video task |
| MINOR_REMOTE_AI_VIDEO_TASK_QUERY | 0x2661 | Search for video task |
| MINOR_REMOTE_AI_VIDEO_TASK_DELETE | 0x2662 | Delete video task |
| MINOR_REMOTE_AI_VIDEO_TASK_MODIFY | 0x2663 | Edit video task |
| MINOR_LOCAL_SSD_OPERATE_START | 0x2705 | Local SSD operation started |
| MINOR_LOCAL_SSD_OPERATE_STOP | 0x2706 | Local SSD operation ended |
| MINOR_REMOTE_SSD_OPERATE_START | 0x2707 | Remote SSD operation started |
| MINOR_REMOTE_SSD_OPERATE_STOP | 0x2708 | Remote SSD operation ended |
| MINOR_LOCAL_EZVIZ_OPERATION | 0x2671 | Local EZVIZ operations |
| MINOR_REMOTE_EZVIZ_OPERATION | 0x2672 | Remote EZVIZ operations |
| MINOR_LOCAL_PARA_FACTORY_DEFAULT | 0x3002 | Restore to default settings locally |
| MINOR_REMOTE_PARA_FACTORY_DEFAULT | 0x3003 | Restore to default settings remotely |
| MIMOR_REMOTE_DELETE_ALL_VERIFYORCAP_PICS | 0x3004 | Delete all authenticated or captured face pictures remotely |
| MIMOR_LOCAL_DELETE_ALL_VERIFYORCAP_PICS | 0x3005 | Delete all authenticated or captured face pictures locally |

| Log Minor Type | Value | Description |
|---|---|---|
| MIMOR_REMOTE_DELETE_EVENTS_AT_SPECTIME | 0x3006 | Delete events by specified time remotely |
| MIMOR_LOCAL_DELETE_EVENTS_AT_SPECTIME | 0x3007 | Delete events by specified time locally |
| MIMOR_REMOTE_OPEN_SUMMER_TIME | 0x3008 | Enable DST (Daylight Saving Time) remotely |
| MIMOR_LOCAL_OPEN_SUMMER_TIME | 0x3009 | Enable DST (Daylight Saving Time) locally |
| MIMOR_REMOTE_CLOSE_SUMMER_TIME | 0x3010 | Disable DST (Daylight Saving Time) remotely |
| MIMOR_LOCAL_CLOSE_SUMMER_TIME | 0x3011 | Disable DST (Daylight Saving Time) locally |
| MIMOR_REMOTE_EZVIZ_UNBIND | 0x3012 | Unbind from EZVIZ cloud remotely |
| MIMOR_LOCAL_EZVIZ_UNBIND | 0x3013 | Unbind from EZVIZ cloud locally |
| MIMOR_ENTER_LOCALUI_BACKGROUND | 0x3014 | Enter UI background |
| MIMOR_REMOTE_DELETE_FACEBASEMAP | 0x3015 | Delete registered face pictures remotely |
| MIMOR_LOCAL_DELETE_FACEBASEMAP | 0x3016 | Delete registered face pictures locally |
| MINOR_SSH_ENABLE | 0xc55 | SSH switch configuration |

## MAJOR_INFORMATION

**Table C-4 Minor Types of Additional Information Log**

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_HDD_INFO | 0xa1 | HDD Information |
| MINOR_SMART_INFO | 0xa2 | S.M.A.R.T Information |
| MINOR_REC_START | 0xa3 | Start recording |
| MINOR_REC_STOP | 0xa4 | Stop recording |
| MINOR_REC_OVERDUE | 0xa5 | Delete expired video files |
| MINOR_LINK_START | 0xa6 | Connect front-end device |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_LINK_STOP | 0xa7 | Disconnect front-end device |
| MINOR_NET_DISK_INFO | 0xa8 | Network HDD information |
| MINOR_RAID_INFO | 0xa9 | raid related information |
| MINOR_RUN_STATUS_INFO | 0xaa | System running status information |
| MINOR_SPARE_START_BACKUP | 0xab | Hot spare system starts backing up working device |
| MINOR_SPARE_STOP_BACKUP | 0xac | Hot spare system stops backing up working device |
| MINOR_SPARE_CLIENT_INFO | 0xad | Hot spare customer device information |
| MINOR_ANR_RECORD_START | 0xae | Start ANR recording |
| MINOR_ANR_RECORD_END | 0xaf | Stop ANR recording |
| MINOR_ANR_ADD_TIME_QUANTUM | 0xb0 | Add ANR time period |
| MINOR_ANR_DEL_TIME_QUANTUM | 0xb1 | Delete ANR time period |
| MINOR_PIC_REC_START | 0xb3 | Start capturing |
| MINOR_PIC_REC_STOP | 0xb4 | Stop Capturing |
| MINOR_PIC_REC_OVERDUE | 0xb5 | Delete expired picture |
| MINOR_CLIENT_LOGIN | 0xb6 | Logging in to server completed |
| MINOR_CLIENT_RELOGIN | 0xb7 | Log in to server again |
| MINOR_CLIENT_LOGOUT | 0xb8 | Exiting server completed |
| MINOR_CLIENT_SYNC_START | 0xb9 | Start Synchronous Recording |
| MINOR_CLIENT_SYNC_STOP | 0xba | Stop Synchronous Recording |
| MINOR_CLIENT_SYNC_SUCC | 0xbb | Synchronous Recording Completed |
| MINOR_CLIENT_SYNC_EXCP | 0xbc | Synchronous recording exception |
| MINOR_GLOBAL_RECORD_ERR_INFO | 0xbd | Global Error Information |
| MINOR_BUFFER_STATE | 0xbe | Buffer Status Log File |
| MINOR_DISK_ERRORINFO_V2 | 0xbf | HDD Error Details V2 |
| MINOR_UNLOCK_RECORD | 0xc3 | Lock Record |
| MINOR_VIS_ALARM | 0xc4 | Zone Alarm |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_TALK_RECORD | 0xc5 | Calling Record |
| MINOR_ACCESSORIES_MESSAGE | 0xc6 | Accessories Information |
| MINOR_IPC_CONNECT | 0xc8 | Network connection information |
| MINOR_INTELLIGENT_BOARD_STATUS | 0xc9 | Intelligent board status |
| MINOR_IPC_CONNECT_STATUS | 0xca | Network camera connection status |
| MINOR_EZVIZ_OPERATION | 0xcc | EZVIZ Running Status |
| MINOR_CLUSTER_DEVICE_ONLINE | 0xcd | Cluster device is online |
| MINOR_CLUSTER_MGR_SERVICE_STARTUP | 0xce | Cluster management service is enabled |
| MINOR_CLUSTER_BUSINESS_TRANSFER | 0xcf | Cluster migration |
| MINOR_CLUSTER_STATUS | 0xd0 | Cluster status information |
| MINOR_CLUSTER_CS_STATUS | 0xd1 | Sending device status to CM failed. Record the IP address of CS and CM. |
| MINOR_CLUSTER_CM_STATUS | 0xd2 | CM status switching. |
| MINOR_DOUBLE_VERIFICATION_PASS | 0xd5 | Double verification completed |
| MINOR_HD_FORMAT_START | 0xd8 | Formatting HDD started. |
| MINOR_HD_FORMAT_STOP | 0xd9 | Formatting HDD stopped. |
| MINOR_802_1X_AUTH_SUCC | 0x320 | 802.1x authentication succeeded. |
| MINOR_802_1X_AUTH_FAIL | 0x321 | 802.1x authentication failed. |
| MINOR_LIVE_DETECT_OPEN | 0x400 | Enabled face anti-spoofing detection |
| MINOR_LIVE_DETECT_CLOSE | 0x401 | Disabled face anti-spoofing detection |
| MINOR_CLEAR_DATA_COLLECTION | 0x402 | Cleared collected data |
| MINOR_DELETE_DATA_COLLECTION | 0x403 | Deleted collected data |
| MINOR_EXPORT_DATA_COLLECTION | 0x404 | Exported collected data |
| MINOR_CARD_LEN_CONFIG | 0x405 | Configured card number size |
| MINOR_DATA_BASE_INIT_FAILED | 0x406 | Initializing database failed |
| MINOR_DATA_BASE_PATCH_UPDATE | 0x407 | Upgraded database patch |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_PSAM_CARD_INSERT | 0x408 | Inserted PSAM card |
| MINOR_PSAM_CARD_REMOVE | 0x409 | Pulled out PSAM card |
| MINOR_HARD_FAULT_REBOOT | 0x40a | Reboot as hardware exception |
| MINOR_PSAM_CARD_OCP | 0x40b | Overflow protection of PSAM card |
| MINOR_STACK_OVERFLOW | 0x40c | Stack overflow |
| MINOR_PARM_CFG | 0x40d | Parameter configuration |
| MINOR_CLR_USER | 0x40e | Clear all users |
| MINOR_CLR_CARD | 0x40f | Clear all cards |
| MINOR_CLR_FINGER_BY_READER | 0x410 | Clear all fingerprints by fingerprint and card reader |
| MINOR_CLR_FINGER_BY_CARD | 0x411 | Clear all fingerprints by card No. |
| MINOR_CLR_FINGER_BY_EMPLOYEE_ON | 0x412 | Clear all fingerprints by employee ID |
| MINOR_DEL_FINGER | 0x413 | Delete a fingerprint |
| MINOR_CLR_WEEK_PLAN | 0x414 | Clear week schedules of access permission control |
| MINOR_SET_WEEK_PLAN | 0x415 | Set the week schedule of access permission control |
| MINOR_SET_HOLIDAY_PLAN | 0x416 | Set the holiday schedule of access permission control |
| MINOR_CLR_HOLIDAY_PLAN | 0x417 | Clear holiday schedules of access permission control |
| MINOR_SET_HOLIDAY_GROUP | 0x418 | Set the holiday group of access permission control schedule |
| MINOR_CLR_HOLIDAY_GROUP | 0x419 | Clear holiday groups of access permission control schedule |
| MINOR_CLR_TEMPLATE_PLAN | 0x41a | Clear access permission control schedules |
| MINOR_SET_TEMPLATE_PLAN | 0x41b | Set the access permission control schedule |
| MINOR_ADD_CARD | 0x41c | Add a card |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_MOD_CARD | 0x41d | Edit a card |
| MINOR_ADD_FINGER_BY_CARD | 0x41e | Add a fingerprint by card No. |
| MINOR_ADD_FINGER_BY_EMPLOYEE_NO | 0x41f | Add a fingerprint by employee ID |
| MINOR_MOD_FINGER_BY_CARD | 0x420 | Edit a fingerprint by card No. |
| MINOR_MOD_FINGER_BY_EMPLOYEE_NO | 0x421 | Edit a fingerprint by employee ID |
| MINOR_IMPORT_USER_LIST | 0x422 | Imported user list (offline data collection) |
| MINOR_USB_LOGIN | 0x423 | Log in via USB |
| MINOR_USB_LOGOUT | 0x424 | Log out via USB |
| MINOR_ISAPI_HTTP_LOGIN | 0x425 | Log in via text protocol (HTTP) |
| MINOR_ISAPI_HTTP_LOGOUT | 0x426 | Log out via text protocol (HTTP) |
| MINOR_ISAPI_HTTPS_LOGIN | 0x427 | Log in via text protocol (HTTPS) |
| MINOR_ISAPI_HTTPS_LOGOUT | 0x428 | Log out via text protocol (HTTPS) |
| MINOR_ISUP_ONLINE | 0x429 | ISUP online |
| MINOR_ISUP_OFFLINE | 0x42a | ISUP offline |
| MINOR_FP_ISSUE_REC | 0x42b | Issuing record of card containing fingerprint information |
| MINOR_FACE_ISSUE_REC | 0x42c | Issuing record of card containing face picture information |
| MINOR_ADD_USER_INFO | 0x432 | Added person information (access control permission) |
| MINOR_MODIFY_USER_INFO | 0x433 | Edit person information (access control permission) |
| MINOR_CLR_USER_INFO | 0x434 | Delete person information by employee No. (access control permission) |
| MINOR_CLR_CARD_BY_CARD_OR_EMPLOYEE | 0x435 | Delete cards by card No. or employee No. |
| MINOR_WIRELESS_RUNNING_STATUS | 0xd6 | Wireless network running status |

## MAJOR_EVENT

**Table C-5 Minor Types of Event Log**

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_LEGAL_CARD_PASS | 0x01 | Legal Card Authenticated |
| MINOR_CARD_AND_PSW_PASS | 0x02 | Card and Password Authenticated |
| MINOR_CARD_AND_PSW_FAIL | 0x03 | Card and password authentication failed. |
| MINOR_CARD_AND_PSW_TIMEOUT | 0x04 | Card and password authentication timed out. |
| MINOR_CARD_AND_PSW_OVER_TIME | 0x05 | Card and password timed out. |
| MINOR_CARD_NO_RIGHT | 0x06 | No Permission |
| MINOR_CARD_INVALID_PERIOD | 0x07 | Invalid Duration |
| MINOR_CARD_OUT_OF_DATE | 0x08 | Expired Card |
| MINOR_INVALID_CARD | 0x09 | No card No. |
| MINOR_ANTI_SNEAK_FAIL | 0x0a | Anti-passing back authentication failed. |
| MINOR_INTERLOCK_DOOR_NOT_CLOSE | 0x0b | Interlocking Door Not Closed |
| MINOR_NOT_BELONG_MULTI_GROUP | 0x0c | The card does not belong to multiple authentication group. |
| MINOR_INVALID_MULTI_VERIFY_PERIOD | 0x0d | The card is not in the multiple authentication duration. |
| MINOR_MULTI_VERIFY_SUPER_RIGHT_FAIL | 0x0e | Multiple Authentication: Super Permission Authentication Failed |
| MINOR_MULTI_VERIFY_REMOTE_RIGHT_FAIL | 0x0f | Multiple Authentication: Remote Authentication Failed |
| MINOR_MULTI_VERIFY_SUCCESS | 0x10 | Pass Multiple Authentication |
| MINOR_LEADER_CARD_OPEN_BEGIN | 0x11 | Open Door with First Card Started |
| MINOR_LEADER_CARD_OPEN_END | 0x12 | Open Door with First Card Stopped |
| MINOR_ALWAYS_OPEN_BEGIN | 0x13 | Remain Open Started |
| MINOR_ALWAYS_OPEN_END | 0x14 | Remain Open Stopped |

| Log Minor Type | Value | Description |
|---|---|---|
| MINOR_LOCK_OPEN | 0x15 | Unlock Door |
| MINOR_LOCK_CLOSE | 0x16 | Lock Door |
| MINOR_DOOR_BUTTON_PRESS | 0x17 | Press Door Button |
| MINOR_DOOR_BUTTON_RELEASE | 0x18 | Release Door Button |
| MINOR_DOOR_OPEN_NORMAL | 0x19 | Normal Open (Door Magnetic) |
| MINOR_DOOR_CLOSE_NORMAL | 0x1a | Normal Closed (Door Magnetic) |
| MINOR_DOOR_OPEN_ABNORMAL | 0x1b | Abnormal Open (Door Magnetic) |
| MINOR_DOOR_OPEN_TIMEOUT | 0x1c | Open Door Timeout (Door Magnetic) |
| MINOR_ALARMOUT_ON | 0x1d | Alarm Output On |
| MINOR_ALARMOUT_OFF | 0x1e | Alarm Output Off |
| MINOR_ALWAYS_CLOSE_BEGIN | 0x1f | Remain Open Started |
| MINOR_ALWAYS_CLOSE_END | 0x20 | Remain Open Stopped |
| MINOR_MULTI_VERIFY_NEED_ REMOTE_OPEN | 0x21 | Multiple Authentication: Remote Open Door |
| MINOR_MULTI_VERIFY_ SUPERPASSWD_VERIFY_SUCCESS | 0x22 | Multiple Authentication: Super Password Authentication Passed |
| MINOR_MULTI_VERIFY_REPEAT_ VERIFY | 0x23 | Multiple Authentication: Repeat Authentication |
| MINOR_MULTI_VERIFY_TIMEOUT | 0x24 | Multiple Authentication: Repeat Authentication Event |

## C.5 Response Codes of Text Protocol

The response codes returned during the text protocol integration is based on the status codes of HTTP. 7 kinds of status codes are predefined, including 1 (OK), 2 (Device Busy), 3 (Device Error), 4 (Invalid Operation), 5 (Invalid Message Format), 6 (Invalid Message Content), and 7 (Reboot Required). Each kind of status code contains multiple sub status codes, and the response codes are in a one-to-one correspondence with the sub status codes.

### StatusCode=1

| SubStatusCode | Error Code | Description |
|---|---|---|
| ok | 0x1 | Operation completed. |
| riskPassword | 0x10000002 | Risky password. |
| armProcess | 0x10000005 | Arming process. |

### StatusCode=2

| Sub Status Code | Error Code | Description |
|---|---|---|
| noMemory | 0x20000001 | Insufficient memory. |
| serviceUnavailable | 0x20000002 | The service is not available. |
| upgrading | 0x20000003 | Upgrading. |
| deviceBusy | 0x20000004 | The device is busy or no response. |
| reConnectIpc | 0x20000005 | The video server is reconnected. |
| transferUpgradePackageFailed | 0x20000006 | Transmitting device upgrade data failed. |
| startUpgradeFailed | 0x20000007 | Starting upgrading device failed. |
| getUpgradeProcessfailed. | 0x20000008 | Getting upgrade status failed. |
| certificateExist | 0x2000000B | The Authentication certificate already exists. |

### StatusCode=3

| Sub Status Code | Error Code | Description |
|---|---|---|
| deviceError | 0x30000001 | Hardware error. |
| badFlash | 0x30000002 | Flash operation error. |
| 28181Uninitialized | 0x30000003 | The 28181 configuration is not initialized. |
| socketConnectError | 0x30000005 | Connecting to socket failed. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| receiveError | 0x30000007 | Receive response message failed. |
| deletePictureError | 0x3000000A | Deleting picture failed. |
| pictureSizeExceedLimit | 0x3000000C | Too large picture size. |
| clearCacheError | 0x3000000D | Clearing cache failed. |
| updateDatabasError | 0x3000000F | Updating database failed. |
| searchDatabaseError | 0x30000010 | Searching in the database failed. |
| writeDatabaseError | 0x30000011 | Writing to database failed. |
| deleteDatabaseError | 0x30000012 | Deleting database element failed. |
| searchDatabaseElementError | 0x30000013 | Getting number of database elements failed. |
| cloudAutoUpgradeException | 0x30000016 | Downloading upgrade packet from cloud and upgrading failed. |
| HBPException | 0x30001000 | HBP exception. |
| UDEPException | 0x30001001 | UDEP exception |
| elasticSearchException | 0x30001002 | Elastic exception. |
| kafkaException | 0x30001003 | Kafka exception. |
| HBaseException | 0x30001004 | Hbase exception. |
| sparkException | 0x30001005 | Spark exception. |
| yarnException | 0x30001006 | Yarn exception. |
| cacheException | 0x30001007 | Cache exception. |
| trafficException | 0x30001008 | Monitoring point big data server exception. |
| faceException | 0x30001009 | Human face big data server exception. |
| SSDFileSystemIsError | 0x30001013 | SSD file system error (Error occurs when it is non-Ext4 file system) |

| Sub Status Code | Error Code | Description |
|---|---|---|
| insufficientSSDCapacityForFPD | 0x30001014 | Insufficient SSD space for person frequency detection. |
| wifiException | 0x3000100A | Wi-Fi big data server exception |
| structException | 0x3000100D | Video parameters structure server exception. |
| noLinkageResource | 0x30001015 | Insufficient linkage resources. |
| engineAbnormal | 0x30002015 | Engine exception. |
| engineInitialization | 0x30002016 | Initializing the engine. |
| algorithmLoadingFailed | 0x30002017 | Loading the model failed. |
| algorithmDownloadFailed | 0x30002018 | Downloading the model failed. |
| algorithmDecryptionFailed | 0x30002019 | Decrypting the model failed. |
| unboundChannel | 0x30002020 | Delete the linked channel to load the new model. |
| unsupportedResolution | 0x30002021 | Invalid resolution. |
| unsupportedSteamType | 0x30002022 | Invalid stream type. |
| insufficientDecRes | 0x30002023 | Insufficient decoding resources. |
| insufficientEnginePerformance | 0x30002024 | Insufficient engine performance (The number of channels to be analyzed exceeds the engine's capability). |
| improperResolution | 0x30002025 | Improper resolution (The maximum resolution allowed is 4096×4096). |
| improperPicSize | 0x30002026 | Improper picture size (The maximum size allowed is 5MB). |
| URLDownloadFailed | 0x30002027 | Downloading the picture via the URI failed. |
| unsupportedImageFormat | 0x30002028 | Invalid picture format (Only JPG is supported currently). |

| Sub Status Code | Error Code | Description |
|---|---|---|
| unsupportedPollingIntervalTime | 0x30002029 | Invalid polling interval (The interval should be more than 10s). |
| exceedImagesNumber | 0x30002030 | The number of pictures exceeds the limit (The platform can apply 1 to 100 picture URIs per time, the maximum number allowed is 100). |
| unsupportedMPID | 0x30002031 | The applied MPID does not exist in the device, so updating this MPID is not supported. |
| modelPackageNotMatchLabel | 0x30002032 | The model and the description file mismatch. |
| modelPackageNotMatchTask | 0x30002033 | The task and the model type mismatch. |
| insufficientSpace | 0x30002034 | Insufficient space (When the number of model packages does not reach the maximum number allowed but their size together exceeds the free space, the model packages cannot be added). |
| engineUnLoadingModelPackage | 0x30002035 | Applying the task failed. This engine is not linked to a model package (Canceling the linkage failed, this engine is not linked to a model package). |
| engineWithModelPackage | 0x30002036 | Linking the engine to this model package failed. The engine has been linked to another model package. Please cancel their linkage first. |
| modelPackageDelete | 0x30002037 | Linking the model package failed. The model package has been deleted. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| deleteTaskFailed | 0x30002038 | Deleting the task failed (It is returned when the user fails to end a task). |
| modelPackageNumberslimited | 0x30002039 | Adding the model package failed. The number of model package has reached the maximum number allowed. |
| modelPackageDeleteFailed | 0x30002040 | Deleting the model package failed. |
| noArmingResource | 0x30001016 | Insufficient arming resources. |
| calibrationTimeout | 0x30002051 | Calibration timed out. |
| captureTimeout | 0x30006000 | Data collection timed out. |
| lowScore | 0x30006001 | Low quality of collected data. |
| uploadingFailed | 0x30007004 | Uploading failed. |

### StatusCode=4

| Sub Status Code | Error Code | Description |
|---|---|---|
| notSupport | 0x40000001 | Not supported. |
| lowPrivilege | 0x40000002 | No permission. |
| badAuthorization | 0x40000003 | Authentication failed. |
| methodNotAllowed | 0x40000004 | Invalid HTTP method. |
| notSetHdiskRedund | 0x40000005 | Setting spare HDD failed. |
| invalidOperation | 0x40000006 | Invalid operation. |
| notActivated | 0x40000007 | Inactivated. |
| hasActivated | 0x40000008 | Activated. |
| certificateAlreadyExist | 0x40000009 | The certificate already exists. |
| operateFailed | 0x4000000F | Operation failed. |
| USBNotExist | 0x40000010 | USB device is not connected. |
| upgradePackageMorethan2GB | 0x40001000 | Up to 2GB upgrade package is allowed to be uploaded. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| IDNotexist | 0x40001001 | The ID does not exist. |
| interfaceOperationError | 0x40001002 | API operation failed. |
| synchronizationError | 0x40001003 | Synchronization failed. |
| synchronizing | 0x40001004 | Synchronizing. |
| importError | 0x40001005 | Importing failed. |
| importing | 0x40001006 | Importing. |
| fileAlreadyExists | 0x40001007 | The file already exists. |
| invalidID | 0x40001008 | Invalid ID. |
| backupnodeNotAlloweLog | 0x40001009 | Accessing to backup node is not allowed. |
| exportingError | 0x4000100A | Exporting failed. |
| exporting | 0x4000100B | Exporting. |
| exportEnded | 0x4000100C | Exporting stopped. |
| exported | 0x4000100D | Exported. |
| IPOccupied | 0x4000100E | The IP address is already occupied. |
| IDAlreadyExists | 0x4000100F | The ID already exists. |
| exportItemsExceedLimit | 0x40001010 | No more items can be exported. |
| noFiles | 0x40001011 | The file does not exist. |
| beingExportedByAnotherUser | 0x40001012 | Being exported by others. |
| needReAuthentication | 0x40001013 | Authentication is needed after upgrade. |
| unitAddNotOnline | 0x40001015 | The added data analysis server is offline. |
| unitControl | 0x40001016 | The data analysis server is already added. |
| analysis unitFull | 0x40001017 | No more data analysis server can be added. |
| unitIDError | 0x40001018 | The data analysis server ID does not exist. |
| unitExit | 0x40001019 | The data analysis server already exists in the list. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| unitSearch | 0x4000101A | Searching data analysis server in the list failed. |
| unitNotOnline | 0x4000101B | The data analysis server is offline. |
| unitInfoEror | 0x4000101C | Getting data analysis server information failed. |
| unitGetNodeInfoError | 0x4000101D | Getting node information failed. |
| unitGetNetworkInfoError | 0x4000101E | Getting the network information of data analysis server failed |
| unitSetNetworkInfoError | 0x4000101F | Setting the network information of data analysis server failed |
| setSmartNodeInfoError | 0x40001020 | Setting node information failed. |
| setUnitNetworkInfoError | 0x40001021 | Setting data analysis server network information failed. |
| unitRestartCloseError | 0x40001022 | Rebooting or shutting down data analysis server failed. |
| virtualIPnotAllowed | 0x40001023 | Adding virtual IP address is not allowed. |
| unitInstalled | 0x40001024 | The data analysis server is already installed. |
| badSubnetMask | 0x40001025 | Invalid subnet mask. |
| uintVersionMismatched | 0x40001026 | Data analysis server version mismatches. |
| deviceMOdelMismatched | 0x40001027 | Adding failed. Device model mismatches. |
| unitAddNotSelf | 0x40001028 | Adding peripherals is not allowed. |
| noValidUnit | 0x40001029 | No valid data analysis server. |
| unitNameDuplicate | 0x4000102A | Duplicated data analysis server name. |
| deleteUnitFirst | 0x4000102B | Delete the added data analysis server of the node first. |
| getLocalInfoFailed | 0x4000102C | Getting the server information failed. |
| getClientAddedNodeFailed | 0x4000102D | Getting the added node information of data analysis server failed. |
| taskExit | 0x4000102E | The task already exists. |
| taskInitError | 0x4000102F | Initializing task failed. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| taskSubmitError | 0x40001030 | Submiting task failed. |
| taskDelError | 0x40001031 | Deleting task failed. |
| taskPauseError | 0x40001032 | Pausing task failed. |
| taskContinueError | 0x40001033 | Starting task failed. |
| taskSeverNoCfg | 0x40001035 | Full-text search server is not configured. |
| taskPicSeverNoCfg | 0x40001036 | The picture server is not configured. |
| taskStreamError | 0x40001037 | Streaming information exception. |
| taskRecSDK | 0x40001038 | History recording is not supported. |
| taskCasaError | 0x4000103A | Cascading is not supported. |
| taskVCARuleError | 0x4000103B | Invalid VCA rule. |
| taskNoRun | 0x4000103C | The task is not executed. |
| unitLinksNoStorageNode | 0x4000103D | No node is linked with the data analysis server. Configure the node first. |
| searchFailed | 0x4000103E | Searching video files failed. |
| searchNull | 0x4000103F | No video clip. |
| userScheOffline | 0x40001040 | The task scheduler service is offline. |
| updateTypeUnmatched | 0x40001041 | The upgrade package type mismatches. |
| userExist | 0x40001043 | The user already exists. |
| userCannotDelAdmin | 0x40001044 | The administrator cannot be deleted. |
| userInexistence | 0x40001045 | The user name does not exist. |
| userCannotCreatAdmin | 0x40001046 | The administrator cannot be created. |
| monitorCamExceed | 0x40001048 | Up to 3000 cameras can be added. |
| monitorCunitOverLimit | 0x40001049 | Adding failed. Up to 5 lower-levels are supported by the control center. |
| monitorReginOverLimit | 0x4000104A | Adding failed. Up to 5 lower-levels are supported by the area. |
| monitorArming | 0x4000104B | The camera is already armed. Disarm the camera and try again. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| monitorSyncCfgNotSet | 0x4000104C | The system parameters are not configured. |
| monitorFdSyncing | 0x4000104E | Synchronizing. Try again after completing the synchronization. |
| monitorParseFailed | 0x4000104F | Parsing camera information failed. |
| monitorCreatRootFailed | 0x40001050 | Creating resource node failed. |
| deleteArmingInfo | 0x40001051 | The camera is already . Disarm the camera and try again. |
| cannotModify | 0x40001052 | Editing is not allowed. Select again. |
| cannotDel | 0x40001053 | Deletion is not allowed. Select again. |
| deviceExist | 0x40001054 | The device already exists. |
| IPErrorConnectFailed | 0x40001056 | Connection failed. Check the network port. |
| cannotAdd | 0x40001057 | Only the capture cameras can be added. |
| serverExist | 0x40001058 | The server already exists. |
| fullTextParamError | 0x40001059 | Incorrect full-text search parameters. |
| storParamError | 0x4000105A | Incorrect storage server parameters. |
| picServerFull | 0x4000105B | The storage space of picture storage server is full. |
| NTPUnconnect | 0x4000105C | Connecting to NTP server failed. Check the parameters. |
| storSerConnectFailed | 0x4000105D | Connecting to storage server failed. Check the network port. |
| storSerLoginFailed | 0x4000105E | Logging in to storage server failed. Check the user name and password. |
| searchSerConnectFailed | 0x4000105F | Connecting to full-text search server failed. Check the network port. |
| searchSerLoginFailed | 0x40001060 | Logging in to full-text search server failed. Check the user name and password. |
| kafkaConnectFailed | 0x40001061 | Connecting to Kafka failed. Check the network port. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| mgmtConnectFailed | 0x40001062 | Connecting to system failed. Check the network port. |
| mgmtLoginFailed | 0x40001063 | Logging in to system failed. Check the user name and password. |
| TDAConnectFailed | 0x40001064 | Connecting to traffic data access server failed. Checking the server status. |
| 86sdkConnectFailed | 0x40001065 | Connecting to listening port of iVMS-8600 System failed. Check the parameters. |
| nameExist | 0x40001066 | Duplicated server name. |
| batchProcessFailed | 0x40001067 | Processing in batch failed. |
| IDNotExist | 0x40001068 | The server ID does not exist. |
| serviceNumberReachesLimit | 0x40001069 | No more service can be added. |
| invalidServiceType. | 0x4000106A | Invalid service type. |
| clusterGetInfo | 0x4000106B | Getting cluster group information failed. |
| clusterDelNode | 0x4000106C | Deletion node failed. |
| clusterAddNode | 0x4000106D | Adding node failed. |
| clusterInstalling | 0x4000106E | Creating cluster…Do not operate. |
| clusterUninstall | 0x4000106F | Reseting cluster…Do not operate. |
| clusterInstall | 0x40001070 | Creating cluster failed. |
| clusterIpError | 0x40001071 | Invalid IP address of task scheduler server. |
| clusterNotSameSeg | 0x40001072 | The main node and sub node must be in the same network segment. |
| clusterVirIpError | 0x40001073 | Automatically getting virtual IP address failed. Enter manually. |
| clusterNodeUnadd | 0x40001074 | The specified main (sub) node is not added. |
| clusterNodeOffline | 0x40001075 | The task scheduler server is offline. |
| nodeNotCurrentIP | 0x40001076 | The analysis node of the current IP address is required when adding main and sub nodes. |
| addNodeNetFailed | 0x40001077 | Adding node failed. The network disconnected. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| needTwoMgmtNode | 0x40001078 | Two management nodes are required when adding main and sub nodes. |
| ipConflict | 0x40001079 | The virtual IP address and data analysis server's IP address conflicted. |
| ipUsed | 0x4000107A | The virtual IP address has been occupied. |
| cloudAlalyseOnline | 0x4000107B | The cloud analytic server is online. |
| virIP&mainIPnotSameNetSegment | 0x4000107C | The virtual IP address is not in the same network segment with the IP address of main/ sub node. |
| getNodeDispatchInfoFailed | 0x4000107D | Getting node scheduler information failed. |
| unableModifyManagementNetworkIP | 0x4000107E | Editing management network interface failed. The analysis board is in the cluster. |
| notSpecifyVirtualIP | 0x4000107F | Virtual IP address should be specified for main and sub cluster. |
| armingFull | 0x40001080 | No more device can be armed. |
| armingNoFind | 0x40001081 | The arming information does not exist. |
| disArming | 0x40001082 | Disarming failed. |
| getArmingError | 0x40001084 | Getting arming information failed. |
| refreshArmingError | 0x40001085 | Refreshing arming information failed. |
| ArmingPlateSame | 0x40001086 | The license plate number is repeatedly armed. |
| ArmingParseXLSError | 0x40001087 | Parsing arming information file failed. |
| ArmingTimeError | 0x40001088 | Invalid arming time period. |
| ArmingSearchTimeError | 0x40001089 | Invalid search time period. |
| armingRelationshipReachesLimit | 0x4000108A | No more relation can be created. |
| duplicateAarmingName | 0x4000108B | The relation name already exists. |
| noMoreArmingListAdded | 0x4000108C | No more blocklist library can be armed. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| noMoreCamerasAdded | 0x4000108D | No more camera can be armed. |
| noMoreArmingListAddedWithCamera | 0x4000108E | No more library can be linked to the camera. |
| noMoreArmingPeriodAdded | 0x4000108F | No more time period can be added to the arming schedule. |
| armingPeriodsOverlapped | 0x40001090 | The time periods in the arming schedule are overlapped. |
| noArmingAlarmInfo | 0x40001091 | The alarm information does not exist. |
| armingAlarmUnRead | 0x40001092 | Getting number of unread alarms failed. |
| getArmingAlarmError | 0x40001093 | Getting alarm information failed. |
| searchByPictureTimedOut | 0x40001094 | Searching picture by picture timeout. Search again. |
| comparisonTimeRangeError | 0x40001095 | Comparison time period error. |
| selectMonitorNumberUpperLimit | 0x40001096 | No more monitoring point ID can be filtered. |
| noMoreComparisonTasksAdded | 0x40001097 | No more comparison task can be executed at the same time. |
| GetComparisonResultFailed | 0x40001098 | Getting comparison result failed. |
| comparisonTypeError | 0x40001099 | Comparison type error. |
| comparisonUnfinished | 0x4000109A | The comparison is not completed. |
| facePictureModelInvalid | 0x4000109B | Invalid face model. |
| duplicateLibraryName. | 0x4000109C | The library name already exists. |
| noRecord | 0x4000109D | No record found. |
| countingRecordsFailed. | 0x4000109E | Calculate the number of records failed. |
| getHumanFaceFrameFailed | 0x4000109F | Getting face thumbnail from the picture failed. |
| modelingFailed. | 0x400010A0 | Modeling face according to picture URL failed. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| 1V1FacePictureComparisonFailed | 0x400010A1 | Comparison 1 VS 1 face picture failed. |
| libraryArmed | 0x400010A2 | The blocklist library is armed. |
| licenseExeedLimit | 0x400010A3 | Dongle limited. |
| licenseExpired | 0x400010A4 | Dongle expired. |
| licenseDisabled | 0x400010A5 | Unavailable dongle. |
| licenseNotExist | 0x400010A6 | The dongle does not exist. |
| SessionExpired | 0x400010A7 | Session expired . |
| beyondConcurrentLimit | 0x400010A8 | Out of concurrent limit. |
| stopSync | 0x400010A9 | Synchronization stopped. |
| getProgressFaild | 0x400010AA | Getting progress failed. |
| uploadExtraCaps | 0x400010AB | No more files can be uploaded. |
| timeRangeError | 0x400010AC | Time period error. |
| dataPortNotConnected | 0x400010AD | The data port is not connected. |
| addClusterNodeFailed | 0x400010AE | Adding to the cluster failed. The device is already added to other cluster. |
| taskNotExist | 0x400010AF | The task does not exist. |
| taskQueryFailed | 0x400010B0 | Searching task failed. |
| modifyTimeRuleFailed | 0x400010B2 | The task already exists. Editing time rule is not allowed. |
| modifySmartRuleFailed | 0x400010B3 | The task already exists. Editing VAC rule is not allowed. |
| queryHistoryVideoFailed | 0x400010B4 | Searching history video failed. |
| addDeviceFailed | 0x400010B5 | Adding device failed. |
| addVideoFailed | 0x400010B6 | Adding video files failed. |
| deleteAllVideoFailed | 0x400010B7 | Deleting all video files failed. |
| createVideoIndexFailed | 0x400010B8 | Indexing video files failed. |
| videoCheckTypeFailed | 0x400010B9 | Verifying video files types failed. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| configStructuredAddressFailed | 0x400010BA | Configuring IP address of structured server failed. |
| configPictureServerAddressFailed | 0x400010BB | Configuring IP address of picture storaged server failed. |
| storageServiceIPNotExist | 0x400010BD | The storage server IP address does not exist. |
| syncBackupDatabaseFailed | 0x400010BE | Synchronizing sub database failed. Try again. |
| syncBackupNTPTimeFailed | 0x400010BF | Synchronizing NTP time of sub server failed. |
| clusterNotSelectLoopbackAddress | 0x400010C0 | Loopbacl address is not supported by the main or sub cluster. |
| addFaceRecordFailed | 0x400010C1 | Adding face record failed. |
| deleteFaceRecordFailed | 0x400010C2 | Deleting face record failed. |
| modifyFaceRecordFailed | 0x400010C3 | Editing face record failed. |
| queryFaceRecordFailed | 0x400010C4 | Searching face record failed. |
| faceDetectFailed | 0x400010C5 | Detecting face failed. |
| libraryNotExist | 0x400010C6 | The library does not exist. |
| blackListQueryExporting | 0x400010C7 | Exporting matched blocklists. |
| blackListQueryExported | 0x400010C8 | The matched blocklists are exported. |
| blackListQueryStopExporting | 0x400010C9 | Exporting matched blocklists is stopped. |
| blackListAlarmQueryExporting | 0x400010CA | Exporting matched blocklist alarms. |
| blackListAlarmQueryExported | 0x400010CB | The matched blocklists alarms are exported. |
| blackListAlarmQueryStopExporting | 0x400010CC | Exporting matched blocklist alarms is stopped. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| getBigDataCloudAnalysisFailed | 0x400010CD | Getting big data cloud analytic information failed. |
| setBigDataCloudAnalysisFailed | 0x400010CE | Configuring big data cloud analytic failed. |
| submitMapSearchFailed | 0x400010CF | Submitting search by picture task failed. |
| controlRelationshipNotExist | 0x400010D0 | The relation does not exist. |
| getHistoryAlarmInfoFailed | 0x400010D1 | Getting history alarm information failed. |
| getFlowReportFailed | 0x400010D2 | Getting people counting report failed. |
| addGuardFailed | 0x400010D3 | Adding arming configuration failed. |
| deleteGuardFailed | 0x400010D4 | Deleting arming configuration failed. |
| modifyGuardFailed | 0x400010D5 | Editing arming configuration failed. |
| queryGuardFailed | 0x400010D6 | Searching arming configurations failed. |
| uploadUserSuperCaps | 0x400010D7 | No more user information can be uploaded. |
| bigDataServerConnectFailed | 0x400010D8 | Connecting to big data server failed. |
| microVideoCloudRequestInfoBuildFailed | 0x400010D9 | Adding response information of micro video cloud failed. |
| microVideoCloudResponseInfoBuildFailed | 0x400010DA | Parsing response information of micro video cloud failed. |
| transcodingServerRequestInfoBuildFailed | 0x400010DB | Adding response information of transcoding server failed. |
| transcodingServerResponseInfoParseFailed | 0x400010DC | Parsing response information of transcoding server failed. |
| transcodingServerOffline | 0x400010DD | Transcoding server is offline. |
| microVideoCloudOffline | 0x400010DE | Micro video cloud is offline. |
| UPSServerOffline | 0x400010DF | UPS monitor server is offline. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| statisticReportRequestInfoBuildFailed | 0x400010E0 | Adding response information of statistics report failed. |
| statisticReportResponseInfoParseFailed | 0x400010E1 | Parsing response information of statistics report failed. |
| DisplayConfigInfoBuildFailed | 0x400010E2 | Adding display configuration information failed. |
| DisplayConfigInfoParseFailed | 0x400010E3 | Parsing display configuration information failed. |
| DisplayConfigInfoSaveFailed | 0x400010E4 | Saving display configuration information failed. |
| notSupportDisplayConfigType | 0x400010E5 | The display configuration type is not supported. |
| passError | 0x400010E7 | Incorrect password. |
| upgradePackageLarge | 0x400010EB | Too large upgrade package. |
| sesssionUserReachesLimit | 0x400010EC | No more user can log in via session. |
| ISO 8601TimeFormatError | 0x400010ED | Invalid ISO8601 time format. |
| clusterDissolutionFailed | 0x400010EE | Deleting cluster failed. |
| getServiceNodeInfoFailed | 0x400010EF | Getting service node information failed. |
| getUPSInfoFailed | 0x400010F0 | Getting UPS configuration information failed. |
| getDataStatisticsReportFailed | 0x400010F1 | Getting data statistic report failed. |
| getDisplayConfigInfoFailed | 0x400010F2 | Getting display configuration failed. |
| namingAnalysisBoardNotAllowed | 0x400010F3 | Renaming analysis board is not allowed. |
| onlyDrawRegionsOfConvexPolygon | 0x400010F4 | Only drawing convex polygon area is supported. |
| bigDataServerResponseInfoParseFailed | 0x400010F5 | Parsing response message of big data service failed. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| bigDataServerReturnFailed | 0x400010F6 | No response is returned by big data service. |
| microVideoReturnFailed | 0x400010F7 | No response is returned by micro video cloud service. |
| transcodingServerReturnFailed | 0x400010F8 | No response is returned by transcoding service. |
| UPSServerReturnFailed | 0x400010F9 | No response is returned by UPS monitoring service. |
| forwardingServer ReturnFailed | 0x400010FA | No response is returned by forwarding service. |
| storageServer ReturnFailed | 0x400010FB | No response is returned by storage service. |
| cloudAnalysisServerReturnFailed | 0x400010FC | No response is returned by cloud analytic service. |
| modelEmpty | 0x400010FD | No model is obtained. |
| mainAndBackupNodeCannotModifyManagementNetworkInterfaceIP | 0x400010FE | Editing the management interface IP address of main node and backup node is not allowed. |
| IDTooLong | 0x400010FF | The ID is too long. |
| pictureCheckFailed | 0x40001100 | Detecting picture failed. |
| pictureModelingFailed | 0x40001101 | Modeling picture failed. |
| setCloudAnalsisDefaultProvinceFailed | 0x40001102 | Setting default province of cloud analytic service failed. |
| InspectionAreasNumberExceedLimit | 0x40001103 | No more detection regions can be added. |
| picturePixelsTooLarge | 0x40001105 | The picture resolution is too high. |
| picturePixelsTooSmall | 0x40001106 | The picture resolution is too low. |
| storageServiceIPEmpty | 0x40001107 | The storage server IP address is required. |
| bigDataServerRequestInfoBuildFail | 0x40001108 | Creating request message of big data service failed. |
| analysiTimedOut | 0x40001109 | Analysis time out. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| high-performanceModeDisabled. | 0x4000110A | Please enable high-performance mode. |
| configuringUPSMonitoringServerTimedOut | 0x4000110B | Configurating the UPS monitoring server time out. Check IP address. |
| cloudAnalysisRequestInformationBuildFailed | 0x4000110C | Creating request message of cloud analytic service failed. |
| cloudAnalysisResponseInformationParseFailed | 0x4000110D | Parsing response message of cloud analytic service failed. |
| allCloudAnalysisInterfaceFailed | 0x4000110E | Calling API for cloud analytic service failed. |
| cloudAnalysisModelCompareFailed | 0x4000110F | Model comparison of cloud analytic service failed. |
| cloudAnalysisFacePictureQualityRatingFailed | 0x40001110 | Getting face quality grading of cloud analytic service failed. |
| cloudAnalysisExtractFeaturePointsFailed | 0x40001111 | Extracting feature of cloud analytic service failed. |
| cloudAnalysisExtractPropertyFailed | 0x40001112 | Extracting property of cloud analytic service failed. |
| getAddedNodeInformationFailed | 0x40001113 | Getting the added nodes information of data analysis server failed. |
| noMoreAnalysisUnitsAdded | 0x40001114 | No more data analysis servers can be added. |
| detectionAreaInvalid | 0x40001115 | Invalid detection region. |
| shieldAreaInvalid | 0x40001116 | Invalid shield region. |
| noMoreShieldAreasAdded | 0x40001117 | No more shield region can be drawn. |
| onlyAreaOfRectangleShapeAllowed | 0x40001118 | Only drawing rectangle is allowed in detection area. |
| numberReachedLlimit | 0x40001119 | Number reached the limit. |
| wait1~3MinutesGetIPAfterSetupDHCP | 0x4000111A | Wait 1 to 3 minutes to get IP address after configuring DHCP. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| plannedTimeMustbeHalfAnHour | 0x4000111B | Schedule must be half an hour. |
| oneDeviceCannotBuildCluster | 0x4000111C | Creating main and backup cluster requires at least two devices. |
| updatePackageFileNotUploaded | 0x4000111E | Upgrade package is not uploaded. |
| highPerformanceTasksNotSupportDrawingDetectionRegions | 0x4000111F | Drawing detection area is not allowed under high-performance mode. |
| controlCenterIDDoesNotExist | 0x40001120 | The control center ID does not exist. |
| regionIDDoesNotExist | 0x40001121 | The area ID does not exist. |
| licensePlateFormatError | 0x40001122 | Invalid license plate format. |
| managementNodeDoesNotSupportThisOperation | 0x40001123 | The operation is not supported. |
| searchByPictureResourceNotConfiged | 0x40001124 | The conditions for searching picture by picture are not configured. |
| videoFileEncapsulationFormatNotSupported | 0x40001125 | The video container format is not supported. |
| videoPackageFailure | 0x40001126 | Converting video container format failed. |
| videoCodingFormatNotSupported | 0x40001127 | Video coding format is not supported. |
| monitorOfDeviceArmingdeleteArmingInfo | 0x40001129 | The camera is armed. Disarm it and try again. |
| getVideoSourceTypeFailed | 0x4000112A | Getting video source type failed. |
| smartRulesBuildFailed | 0x4000112B | Creating VAC rule failed. |
| smartRulesParseFailed | 0x4000112C | Parsing VAC rule failed. |
| timeRulesBuildFailed | 0x4000112D | Creating time rule failed. |
| timeRulesParseFailed | 0x4000112E | Parsing time rule failed. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| monitoInfoInvalid | 0x4000112F | Invalid camera information. |
| addingFailedVersionMismatches | 0x40001130 | Adding failed. The device version mismatches. |
| theInformationReturnedAfterCloudAnalysisIsEmpty | 0x40001131 | No response is returned by the cloud analytic service. |
| selectingIpAddressOfHostAndSpareNodeFailedCheckTheStatus | 0x40001132 | Setting IP address for main node and backup node failed. Check the node status. |
| theSearchIdDoesNotExist | 0x40001133 | The search ID does not exist. |
| theSynchronizationIdDoesNotExist | 0x40001134 | The synchronization ID does not exist. |
| theUserIdDoesNotExist | 0x40001136 | The user ID does not exist. |
| theIndexCodeDoesNotExist | 0x40001138 | The index code does not exist. |
| theControlCenterIdDoesNotExist | 0x40001139 | The control center ID does not exist. |
| theAreaIdDoesNotExist | 0x4000113A | The area ID does not exist. |
| theArmingLinkageIdDoesNotExist | 0x4000113C | The arming relationship ID does not exist. |
| theListLibraryIdDoesNotExist | 0x4000113D | The list library ID does not exist. |
| invalidCityCode | 0x4000113E | Invalid city code. |
| synchronizingThePasswordOfSpareServerFailed | 0x4000113F | Synchronizing backup system password failed. |
| editingStreamingTypeIsNotSupported | 0x40001140 | Editing streaming type is not supported. |
| switchingScheduledTaskToTemporaryTaskIsNotSupported | 0x40001141 | Switching scheduled task to temporary task is not supported. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| switchingTemporaryTaskToScheduledTaskIsNotSupported | 0x40001142 | Switching temporary task to scheduled task is not supported. |
| theTaskIsNotDispatchedOrItIsUpdating | 0x40001143 | The task is not dispatched or is updating. |
| thisTaskDoesNotExist | 0x40001144 | This task does not exist in the cloud analytic serice. |
| duplicatedSchedule | 0x40001145 | Schedule period cannot be overlapped. |
| continuousScheduleWithSameAlgorithmTypeShouldBeMerged | 0x40001146 | The continuous schedule periods with same algorithm type should be merged. |
| invalidStreamingTimeRange | 0x40001147 | Invalid streaming time period. |
| invalidListLibraryType | 0x40001148 | Invalid list library type. |
| theNumberOfMatchedResultsShouldBeLargerThan0 | 0x40001149 | The number of search results should be larger than 0. |
| invalidValueRangeOfSimilarity | 0x4000114A | Invalid similarity range. |
| invalidSortingType | 0x4000114B | Invalid sorting type. |
| noMoreListLibraryCanBeLinkedToTheDevice | 0x4000114C | No more lists can be added to one device. |
| InvalidRecipientAddressFormat | 0x4000114D | Invalid address format of result receiver. |
| creatingClusterFailedTheDongleIsNotPluggedIn | 0x4000114E | Insert the dongle before creating cluster. |
| theURLIsTooLong | 0x4000114F | No schedule configured for the task. |
| noScheduleIsConfiguredForTheTask | 0x40001150 | No schedule configured for the task. |
| theDongleIsExpiried | 0x40001151 | Dongle has expired. |
| dongleException | 0x40001152 | Dongle exception. |
| invalidKey | 0x40001153 | Invalid authorization service key. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| decryptionFailed | 0x40001154 | Decrypting authorization service failed. |
| encryptionFailed | 0x40001155 | Encrypting authorization service failed. |
| AuthorizeServiceResponseError | 0x40001156 | Authorization service response exception. |
| incorrectParameter | 0x40001157 | Authorization service parameters error. |
| operationFailed | 0x40001158 | Operating authorization service error. |
| noAnalysisResourceOrNoDataInTheListLibrary | 0x40001159 | No cloud analytic resources or no data in the list library. |
| calculationException | 0x4000115A | Calculation exception. |
| allocatingList | 0x4000115B | Allocating list. |
| thisOperationIsNotSupportedByTheCloudAnalytics | 0x4000115C | This operation is not supported by the cloud analytic serice. |
| theCloudAnalyticsIsInterrupted | 0x4000115D | The operation of cloud analytic serice is interrupted. |
| theServiceIsNotReady | 0x4000115E | The service is not ready. |
| searchingForExternalApiFailed | 0x4000115F | Searching external interfaces failed. |
| noOnlineNode | 0x40001160 | No node is online. |
| noNodeAllocated | 0x40001161 | No allocated node. |
| noMatchedList | 0x40001162 | No matched list. |
| allocatingFailedTooManyFacePictureLists | 0x40001163 | Allocation failed. Too many lists of big data service. |
| searchIsNotCompletedSearchAgain | 0x40001164 | Current searching is not completed. Search again. |
| allocatingListIsNotCompleted | 0x40001165 | Allocating list is not completed. |
| searchingForCloudAnalyticsResultsFailed | 0x40001166 | Searching cloud analytic serice overtime. |
| noDataOfTheCurrentLibraryFound | 0x40001167 | No data in the current library. Make sure there is data in the Hbase. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| noFacePictureLibraryIsArmed | 0x40001168 | No face picture library is armed for big data service. |
| noAvailableDataSlicingVersionInformationArmFirstAndSliceTheData | 0x40001169 | Invalid standard version information. |
| duplicatedOperationDataSlicingIsExecuting | 0x4000116A | Slicing failed. Duplicated operation. |
| slicinDataFailedNoArmedFacePictureLibrary | 0x4000116B | Slicing failed. No arming information in the face big data. |
| GenerateBenchmarkFileFailedSlicingAgain | 0x4000116C | Generating sliced file failed. Slice again. |
| NonprimaryNodeIsProhibitedFromSlcingData | 0x4000116D | Slicing is not allowed by the backup node. |
| NoReadyNodeToClusterServers | 0x4000116E | Creating the cluster failed. No ready node. |
| NodeManagementServiceIsOffline | 0x4000116F | The node management server is offline. |
| theCamera(s)OfTheControlCenterAreAlreadyArmed.DisarmThemFirst | 0x40001170 | Some cameras in control center are already armed. Disarm them and try again. |
| theCamera(s)OfTheAreaAreAlreadyArmed.DisarmThemFirst | 0x40001171 | Some cameras in this area are already armed. Disarm them and try again. |
| configuringHigh-frequencyPeopleDetectionFailed | 0x40001172 | Configuring high frequency people detection failed. |
| searchingForHigh-frequencyPeopleDetectionLogsFailed. | 0x40001173 | Searching detection event logs of high-frequency people detection failed. |
| gettingDetailsOfSearchedHigh-frequencyPeopleDetectionLogsFailed. | 0x40001174 | Getting the search result details of frequently appeared person alarms failed. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| theArmedCamerasAlreadyExistInTheControlCenter | 0x40001175 | Some cameras in control center are already armed. |
| disarmingFailedTheCameraIsNotArmed | 0x40001177 | Disarming failed. The camera is not armed. |
| noDataReturned | 0x40001178 | No response is returned by the big data service. |
| preallocFailure | 0x40001179 | Pre-allocating algorithm resource failed. |
| overDogLimit | 0x4000117A | Configuration failed. No more resources can be pre-allocated. |
| analysisServicesDoNotSupport | 0x4000117B | Not supported. |
| commandAndDispatchServiceError | 0x4000117C | Scheduling service of cloud analytic serice error. |
| engineModuleError | 0x4000117D | Engine module of cloud analytic serice error. |
| streamingServiceError | 0x4000117E | Streaming component of cloud analytic serice error. |
| faceAnalysisModuleError | 0x4000117F | Face analysis module of cloud analytic serice error. |
| vehicleAnalysisModuleError | 0x40001180 | Vehicle pictures analytic module of cloud analytic serice error. |
| videoStructuralAnalysisModuleError | 0x40001181 | Video structuring module of cloud analytic serice error. |
| postprocessingModuleError | 0x40001182 | Post-processing module of cloud analytic serice error. |
| frequentlyAppearedPersonAlarmIsAlreadyConfiguredForListLibrary | 0x40001183 | Frequently appeared person alarm is already armed for blocklist library. |
| creatingListLibraryFailed | 0x40001184 | Creating list library failed. |
| invalidIdentiryKeyOfListLibrary | 0x40001185 | Invalid identity key of list library. |
| noMoreDevicesCanBeArmed | 0x40001186 | No more camera can be added. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| settingAlgorithmTypeForDeviceFailed | 0x40001187 | Allocating task resource failed. |
| gettingHighFrequencyPersonDetectionAlarmInformationFailed | 0x40001188 | Setting frequently appeared person alarm failed. |
| invalidSearchConfition | 0x40001189 | Invalid result. |
| theTaskIsNotCompleted | 0x4000118B | The task is not completed. |
| resourceOverRemainLimit | 0x4000118C | No more resource can be pre-allocated. |
| frequentlyAppearedPersonAlarmIs AlreadyConfiguredForTheCameraDisarmFirstAndTryAgain | 0x4000118D | The frequently appeared person alarm of this camera is configured. Delete the arming information and try again. |
| switchtimedifflesslimit | 0x4000123b | Time difference between power on and off should be less than 10 minutes. |
| associatedFaceLibNumOverLimit | 0x40001279 | Maximum number of linked face picture libraries reached. |
| noMorePeopleNumChangeRulesAdded | 0x4000128A | Maximum number of people number changing rules reached. |
| noMoreViolentMotionRulesAdded | 0x4000128D | Maximum number of violent motion rules reached. |
| noMoreLeavePositionRulesAdded | 0x4000128E | Maximum number of leaving position rules reached. |
| SMRDiskNotSupportRaid | 0x40001291 | SMR disk does not support RAID. |
| OnlySupportHikAndCustomProtocol | 0x400012A3 | IPv6 camera can only be added via Device Network SDK or custom protocols. |
| vehicleEnginesNoResource | 0x400012A6 | Insufficient vehicle engine resources. |
| noMoreRunningRulesAdded | 0x400012A9 | Maximum number of running rules reached. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| noMoreGroupRulesAdded | 0x400012AA | Maximum number of people gathering rules reached. |
| noMoreFailDownRulesAdded | 0x400012AB | Maximum number of people falling down rules reached. |
| noMorePlayCellphoneRulesAdded | 0x400012AC | Maximum number of playing cellphone rules reached. |
| ruleEventTypeDuplicate | 0x400012C8 | Event type duplicated. |
| noMoreRetentionRulesAdded | 0x400015AD | Maximum number of people retention rules reached. |
| noMoreSleepOnDutyRulesAdded | 0x400015AE | Maximum number of sleeping on duty rules reached. |
| polygonNotAllowCrossing | 0x400015C2 | Polygons are not allowed to cross. |
| configureRuleBeforeAdvanceParam | 0x400015F8 | Advanced parameters fail to be configured as no rule is configured, please configure rule information first. |
| behaviorCanNotPackToPic | 0x40001603 | The behavior model cannot be packaged as a picture algorithm. |
| noCluster | 0x40001608 | No cluster created. |
| NotAssociatedWithOwnChannel | 0x400019C1 | Current channel is not linked. |
| AITargetBPCaptureFail | 0x400019C5 | Capturing reference picture for AI target comparison failed. |
| AITargetBPToDSPFail | 0x400019C6 | Sending reference picture to DSP for AI target comparison failed. |
| AITargetBPDuplicateName | 0x400019C7 | Duplicated name of reference picture for AI target comparison. |
| audioFileNameWrong | 0x400019D0 | Incorrect audio file name. |
| audioFileImportFail | 0x400019D1 | Importing audio file failed. |
| NonOperationalStandbyMachine | 0x400019F0 | Non-operational hot spare. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| MaximumNumberOfDevices | 0x400019F1 | The maximum number of devices reached. |
| StandbyMmachineCannotBeDeleted | 0x400019F2 | The hot spare cannot be deleted. |
| alreadyRunning | 0x40002026 | The application program is running. |
| notRunning | 0x40002027 | The application program is stopped. |
| packNotFound | 0x40002028 | The software packet does not exist. |
| alreadyExist | 0x40002029 | The application program already exists. |
| noMemory | 0x4000202A | Insufficient memory. |
| invalLicense | 0x4000202B | Invalid License. |
| noClientCertificate | 0x40002036 | The client certificate is not installed. |
| noCACertificate | 0x40002037 | The CA certificate is not installed. |
| authenticationFailed | 0x40002038 | Authenticating certificate failed. Check the certificate. |
| clientCertificateExpired | 0x40002039 | The client certificate is expired. |
| clientCertificateRevocation | 0x4000203A | The client certificate is revoked. |
| CACertificateExpired | 0x4000203B | The CA certificate is expired. |
| CACertificateRevocation | 0x4000203C | The CA certificate is revoked. |
| connectFail | 0x4000203D | Connection failed. |
| loginNumExceedLimit | 0x4000203F | No more user can log in. |
| HDMIResolutionIllegal | 0x40002040 | The HDMI video resolution cannot be larger than that of main and sub stream. |
| hdFormatFail | 0x40002049 | Formatting HDD failed. |
| formattingFailed | 0x40002056 | Formatting HDD failed. |
| encryptedFormattingFailed | 0x40002057 | Formatting encrypted HDD failed. |
| wrongPassword | 0x40002058 | Verifying password of SD card failed. Incorrect password. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| audioIsPlayingPleaseWait | 0x40002067 | Audio is playing. Please wait. |
| twoWayAudioInProgressPleaseWait | 0x40002068 | Two-way audio in progress. Please wait. |
| calibrationPointNumFull | 0x40002069 | The maximum number of calibration points reached. |
| completeTheLevelCalibrationFirst | 0x4000206A | The level calibration is not set. |
| completeTheRadarCameraCalibrationFirst | 0x4000206B | The radar-camera calibration is not set. |
| pointsOnStraightLine | 0x4000209C | Calibrating failed. The calibration points cannot be one the same line. |
| TValueLessThanOrEqualZero | 0x4000209D | Calibration failed. The T value of the calibration points should be larger than 0. |
| HBDLibNumOverLimit | 0x40002092 | The number of human body picture libraries reaches the upper limit |
| theShieldRegionError | 0x40002093 | Saving failed. The shielded area should be the ground area where the shielded object is located. |
| theDetectionAreaError | 0x40002094 | Saving failed. The detection area should only cover the ground area. |
| invalidLaneLine | 0x40002096 | Saving failed. Invalid lane line. |
| enableITSFunctionOfThisChannelFirst | 0x400020A2 | Enable ITS function of this channel first. |
| noCloudStorageServer | 0x400020C5 | No cloud storage server |
| NotSupportWithVideoTask | 0x400020F3 | This function is not supported. |
| noDetectionArea | 0x400050df | No detection area |
| armingFailed | 0x40008000 | Arming failed. |
| disarmingFailed | 0x40008001 | Disarming failed. |
| clearAlarmFailed | 0x40008002 | Clearing alarm failed. |
| bypassFailed | 0x40008003 | Bypass failed. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| bypassRecoverFailed | 0x40008004 | Bypass recovery failed. |
| outputsOpenFailed | 0x40008005 | Opening relay failed. |
| outputsCloseFailed | 0x40008006 | Closing relay failed. |
| registerTimeOut | 0x40008007 | Registering timed out. |
| registerFailed | 0x40008008 | Registering failed. |
| addedByOtherHost | 0x40008009 | The peripheral is already added by other security control panel. |
| alreadyAdded | 0x4000800A | The peripheral is already added. |
| armedStatus | 0x4000800B | The partition is armed. |
| bypassStatus | 0x4000800C | Bypassed. |
| zoneNotSupport | 0x4000800D | This operation is not supported by the zone. |
| zoneFault | 0x4000800E | The zone is in fault status. |
| pwdConflict | 0x4000800F | Password conflicted. |
| audioTestEntryFailed | 0x40008010 | Enabling audio test mode failed. |
| audioTestRecoveryFailed | 0x40008011 | Disabling audio test mode failed. |
| addCardMode | 0x40008012 | Adding card mode. |
| searchMode | 0x40008013 | Search mode. |
| addRemoterMode | 0x40008014 | Adding keyfob mode. |
| registerMode | 0x40008015 | Registration mode. |
| exDevNotExist | 0x40008016 | The peripheral does not exist. |
| theNumberOfExDevLimited | 0x40008017 | No peripheral can be added. |
| sirenConfigFailed | 0x40008018 | Setting siren failed. |
| chanCannotRepeatedBinded | 0x40008019 | This channel is already linked by the zone. |
| inProgramMode | 0x4000801B | The keypad is in programming mode. |
| inPaceTest | 0x4000801C | In pacing mode. |
| arming | 0x4000801D | Arming. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| masterSlaveIsEnable | 0x4000802c | The main-sub relationship has taken effect, the sub radar does not support this operation. |
| forceTrackNotEnabled | 0x4000802d | Mandatory tracking is disabled. |
| isNotSupportZoneConfigByLocalArea | 0x4000802e | This area does not support the zone type. |
| alarmLineCross | 0x4000802f | Trigger lines are overlapped. |
| zoneDrawingOutOfRange | 0x40008030 | The drawn zone is out of detection range. |
| alarmLineDrawingOutOfRange | 0x40008031 | The drawn alarm trigger line is out of detection range. |
| hasTargetInWarningArea | 0x40008032 | The warning zone already contains targets. Whether to enable mandatory arming? |
| radarMoudleConnectFail | 0x40008033 | Radar module communication failed. |
| importCfgFilePasswordErr | 0x40008034 | Incorrect password for importing configuration files. |
| overAudioFileNumLimit | 0x40008038 | The number of audio files exceeds the limit. |
| audioFileNameIsLong | 0x40008039 | The audio file name is too long. |
| audioFormatIsWrong | 0x4000803a | The audio file format is invalid. |
| audioFileIsLarge | 0x4000803b | The size of the audio file exceeds the limit. |
| pircamCapTimeOut | 0x4000803c | Capturing of pircam timed out. |
| pircamCapFail | 0x4000803d | Capturing of pircam failed. |
| pircamIsCaping | 0x4000803e | The pircam is capturing. |
| audioFileHasExisted | 0x4000803f | The audio file already exists. |
| subscribeTypeErr | 0x4000a016 | This metadata type is not supported to be subscribed. |
| EISError | 0x4000A01C | Electronic image stabilization failed. The smart event function is enabled. |
| jpegPicWithAppendDataError | 0x4000A01D | Capturing the thermal graphic failed. Check if the temperature measurement parameters |

| Sub Status Code | Error Code | Description |
|---|---|---|
| | | (emissivity, distance, reflective temperature) are configured correctly. |
| startAppFail | / | Starting running application program failed. |
| yuvconflict | / | The raw video stream conflicted. |
| overMaxAppNum | / | No more application program can be uploaded. |
| noFlash | / | Insufficient flash. |
| platMismatch | / | The platform mismatches. |
| emptyEventName | 0x400015E0 | Event name is empty. |
| sameEventName | 0x400015E1 | A same event name already exists. |
| emptyEventType | 0x400015E2 | Event type is required. |
| sameEventType | 0x400015E3 | A same event type already exists. |
| maxEventNameReached | 0x400015E4 | Maximum of events reached. |
| hotSpareNotAllowedExternalStorage | 0x400015FC | External storage is not allowed when hot spare is enabled. |
| sameCustomProtocolName | 0x400015FD | A same protocol name already exists. |
| maxPTZTriggerChannelReached | 0x400015FE | Maximum of channels linked with PTZ reached. |
| POSCanotAddHolidayPlan | 0x400015FF | No POS events during holidays. |
| eventTypeIsTooLong | 0x40001600 | Event type is too long. |
| eventNameIsTooLong | 0x40001601 | Event name is too long. |
| PerimeterEnginesNoResource | 0x40001602 | No more perimeter engines. |
| invalidProvinceCode | 0x40001607 | Invalid province code. |

### StatusCode=5

| Sub Status Code | Error Code | Description |
|---|---|---|
| badXmlFormat | 0x50000001 | Invalid XML format. |

### StatusCode=6

| Sub Status Code | Error Code | Description |
|---|---|---|
| badParameters | 0x60000001 | Invalid parameter. |
| badHostAddress | 0x60000002 | Invalid host IP address. |
| badXmlContent | 0x60000003 | Invalid XML content. |
| badIPv4Address | 0x60000004 | Invalid IPv4 address. |
| badIPv6Address | 0x60000005 | Invalid IPv6 address. |
| conflictIPv4Address | 0x60000006 | IPv4 address conflicted. |
| conflictIPv6Address | 0x60000007 | IPv6 address conflicted. |
| badDomainName | 0x60000008 | Invalid domain name. |
| connectSreverFail | 0x60000009 | Connecting to server failed. |
| conflictDomainName | 0x6000000A | Domain name conflicted. |
| badPort | 0x6000000B | Port number conflicted. |
| portError | 0x6000000C | Port error. |
| exportErrorData | 0x6000000D | Importing data failed. |
| badNetMask | 0x6000000E | Invalid sub-net mask. |
| badVersion | 0x6000000F | Version mismatches. |
| badDevType | 0x60000010 | Device type mismatches. |
| badLanguage | 0x60000011 | Language mismatches. |
| incorrentUserNameOrPassword | 0x600000012 | Incorrect user name or password. |
| invalidStoragePoolOfCloudServer | 0x600000013 | Invalid storage pool. The storage pool is not configured or incorrect ID. |
| noFreeSpaceOfStoragePool | 0x600000014 | Storage pool is full. |
| riskPassword | 0x600000015 | Risky password. |
| UnSupportCapture | 0x600000016 | Capturing in 4096*2160 or 3072*2048 resolution is not supported when H.264+ is enabled. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| userPwdLenUnder8 | 0x60000023 | At least two kinds of characters, including digits, letters, and symbols, should be contained in the password. |
| userPwdNameSame | 0x60000025 | Duplicated password. |
| userPwdNameMirror | 0x60000026 | The password cannot be the reverse order of user name. |
| beyondARGSRangeLimit | 0x60000027 | The parameter value is out of limit. |
| DetectionLineOutofDetectionRegion | 0x60000085 | The rule line is out of region. |
| DetectionRegionError | 0x60000086 | Rule region error. Make sure the rule region is convex polygon. |
| DetectionRegionOutOfCountingRegion | 0x60000087 | The rule region must be marked as red frame. |
| PedalAreaError | 0x60000088 | The pedal area must be in the rule region. |
| DetectionAreaABError | 0x60000089 | The detection region A and B must be in the a rule frame. |
| ABRegionCannotIntersect | 0x6000008a | Region A and B cannot be overlapped. |
| customHBPIDError | 0x6000008b | Incorrect ID of custom human body picture library |
| customHBPIDRepeat | 0x6000008c | Duplicated ID of custom human body picture library |
| dataVersionsInHBDLibMismatches | 0x6000008d | Database versions mismatches of human body picture library |
| invalidHBPID | 0x6000008e | Invalid human body picture PID |
| invalidHBDID | 0x6000008f | Invalid ID of human body picture library |
| humanLibraryError | 0x60000090 | Error of human body picture library |

| Sub Status Code | Error Code | Description |
|---|---|---|
| humanLibraryNumError | 0x60000091 | No more human body picture library can be added |
| humanImagesNumError | 0x60000092 | No more human body picture can be added |
| noHumanInThePicture | 0x60000093 | Modeling failed, no human body in the picture |
| analysisEnginesNoResourceError | 0x60001000 | No analysis engine. |
| analysisEnginesUsageExcced | 0x60001001 | The engine usage is overloaded. |
| PicAnalysisNoResourceError | 0x60001002 | No analysis engine provided for picture secondary recognition. |
| analysisEnginesLoadingError | 0x60001003 | Initializing analysis engine. |
| analysisEnginesAbnormaError | 0x60001004 | Analysis engine exception. |
| analysisEnginesFacelibImporting | 0x60001005 | Importing pictures to face picture library. Failed to edit analysis engine parameters. |
| analysisEnginesAssociatedChannel | 0x60001006 | The analysis engine is linked to channel. |
| smdEncodingNoResource | 0x60001007 | Insufficient motion detection encoding resources. |
| smdDecodingNoResource | 0x60001008 | Insufficient motion detection decoding resources. |
| diskError | 0x60001009 | HDD error. |
| diskFull | 0x6000100a | HDD full. |
| facelibDataProcessing | 0x6000100b | Handling face picture library data. |
| capturePackageFailed | 0x6000100c | Capturing packet failed. |
| capturePackageProcessing | 0x6000100d | Capturing packet. |
| noSupportWithPlaybackAbstract | 0x6000100e | This function is not supported. Playback by video synopsis is enabled. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| insufficientNetworkBandwidth | 0x6000100f | Insufficient network bandwidth. |
| tapeLibNeedStopArchive | 0x60001010 | Stop the filing operation of tape library first. |
| identityKeyError | 0x60001011 | Incorrect interaction command. |
| identityKeyMissing | 0x60001012 | The interaction command is lost. |
| noSupportWithPersonDensityDetect | 0x60001013 | This function is not supported. The people density detection is enabled. |
| ipcResolutionOverflow | 0x60001014 | The configured resolution of network camera is invalid. |
| ipcBitrateOverflow | 0x60001015 | The configured bit rate of network camera is invalid. |
| tooGreatTimeDifference | 0x60001016 | Too large time difference between device and server. |
| noSupportWithPlayback | 0x60001017 | This function is not supported. Playback is enabled. |
| channelNoSupportWithSMD | 0x60001018 | This function is not supported. Motion detection is enabled. |
| channelNoSupportWithFD | 0x60001019 | This function is not supported. Face capture is enabled. |
| illegalPhoneNumber | 0x6000101a | Invalid phone number. |
| illegalCertificateNumber | 0x6000101b | Invalid certificate No. |
| linkedCameraOutLimit | 0x6000101c | Connecting camera timed out. |
| achieveMaxChannelLimit | 0x6000101e | No more channels are allowed. |
| humanMisInfoFilterEnabledChanNumError | 0x6000101f | No more channels are allowed to enable preventing false alarm. |
| humanEnginesNoResource | 0x60001020 | Insufficient human body analysis engine resources. |
| taskNumberOverflow | 0x60001021 | No more tasks can be added. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| collisionTimeOverflow | 0x60001022 | No more comparison duration can be configured. |
| invalidTaskID | 0x60001023 | Invalid task ID. |
| eventNotSupport | 0x60001024 | Event subscription is not supported. |
| invalidEZVIZSecretKey | 0x60001034 | Invalid verification code for Hik-Connect. |
| needDoubleVerification | 0x60001042 | Double verification required |
| noDoubleVerificationUser | 0x60001043 | No double verification user |
| timeSpanNumOverLimit | 0x60001044 | Max. number of time buckets reached |
| channelNumOverLimit | 0x60001045 | Max. number of channels reached |
| noSearchIDResource | 0x60001046 | Insufficient searchID resources |
| noSupportDeleteStrangerLib | 0x60001051 | Deleting stranger library is not supported |
| noSupportCreateStrangerLib | 0x60001052 | Creating stranger library is not supported |
| behaviorAnalysisRuleInfoError | 0x60001053 | Abnormal event detection rule parameters error. |
| safetyHelmetParamError | 0x60001054 | Hard hat parameters error. |
| OneChannelOnlyCanBindOneEngine | 0x60001077 | No more engines can be bound. |
| engineTypeMismatch | 0x60001079 | Engine type mismatched. |
| badUpgradePackage | 0x6000107A | Invalid upgrade package. |
| AudioFileNameDuplicate | 0x60001135 | Duplicated audio file name. |
| CurrentAudioFileAIRuleInUseAlreadyDelete | 0x60001136 | The AI rule linkage related to current audio file has been deleted. |
| TransitionUseEmmc | 0x60002000 | Starting device failed. The EMMC is overused. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| AdaptiveStreamNotEnabled | 0x60002001 | The stream self-adaptive function is not enabled. |
| AdaptiveStreamAndVariableBitrateEnabled | 0x60002002 | Stream self-adptive and variable bitrate function cannot be enabled at the same time. |
| noSafetyHelmetRegion | 0x60002023 | The hard hat detection area is not configured (if users save their settings without configuring the arming area, they should be prompted to configure one). |
| unclosedSafetyHelmet | 0x60002024 | The hard hat detection is enabled (If users save their settings after deleting the arming area, they should be prompted to disable hard hat detection first and then delete the arming area). |
| width/ heightRatioOfPictureError | 0x6000202C | The width/height ratio of the uploaded picture should be in the range from 1:2 to 2:1. |
| PTZNotInitialized | 0x6000202E | PTZ is not initialized. |
| PTZSelfChecking | 0x6000202F | PTZ is self-checking. |
| PTZLocked | 0x60002030 | PTZ is locked. |
| advancedParametersError | 0x60002031 | Auto-switch interval in advanced parameters cannot be shorter than parking tolerance for illegal parking detection in speed dome rule settings. |
| resolutionError | 0x60005003 | Invalid resolution |
| deployExceedMax | 0x60006018 | The arming connections exceed the maximum number. |
| detectorTypeMismatch | 0x60008000 | The detector type mismatched. |
| nameExist | 0x60008001 | The name already exists. |

| Sub Status Code | Error Code | Description |
|---|---|---|
| uploadImageSizeError | 0x60008016 | The size of the uploaded picture is larger than 5 MB. |
| laneAndRegionOverlap | / | The lanes are overlapped. |
| unitConfigurationNotInEffect | / | Invalid unit parameter. |
| ruleAndShieldingMaskConflict | / | The line-rule region overlaps with the shielded area. |
| wholeRuleInShieldingMask | / | There are complete temperature measurement rules in the shielded area. |
| LogDiskNotSetReadOnlyInGroupMode | 0x60001100 | The log HDD in the HDD group cannot be set to read-only. |
| LogDiskNotSetReDundancyInGroupMode | 0x60001101 | The log HDD in the HDD group cannot be set to redundancy. |
| holidayNameContainChineseOrSpecialChar | 0x60001080 | No Chinese and special characters allowed in holiday name. |
| genderValueError | 0x60001081 | Invalid gender. |
| certificateTypeValueError | 0x60001082 | Invalid identification type. |
| personInfoExtendValueIsTooLong | 0x60001083 | The length of customized tags exceeds limit. |
| personInfoExtendValueContainsInvalidChar | 0x60001084 | Invalid characters are not allowed in customized tags of the face picture library. |
| excelHeaderError | 0x60001085 | Excel header error. |
| intelligentTrafficMutexWithHighFrames | 0x60008014 | Please disable all functions of traffic incident detection, violation enforcement, and traffic data collection, or adjust the video frame rate to that lower than 50 fps. |
| intelligentTrafficMutexWithHighFramesEx | 0x60008018 | Please disable all functions of traffic incident detection, violation enforcement, traffic data collection, and vehicle |

| Sub Status Code | Error Code | Description |
|---|---|---|
| | | detection, or adjust the video frame rate to that lower than 50 fps. |

### StatusCode=7

| SubStatusCode | Error Code | Description |
|---|---|---|
| rebootRequired | 0x70000001 | Reboot to take effect. |