Intelligent Security API

Developer Guide

About this Document

- This Document includes instructions for using and managing the Product. Pictures, charts, images and all other information hereinafter are for description and explanation only. Unless otherwise agreed, our company makes no warranties, express or implied.
- Please use this Document with the guidance and assistance of professionals trained in supporting the Product.

Trademarks Acknowledgment

All trademarks and logos mentioned are the properties of their respective owners.

LEGAL DISCLAIMER

- TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, THIS DOCUMENT AND THE PRODUCT DESCRIBED, WITH ITS HARDWARE, SOFTWARE AND FIRMWARE, ARE PROVIDED "AS IS" AND "WITH ALL FAULTS AND ERRORS". OUR COMPANY MAKES NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, MERCHANTABILITY, SATISFACTORY QUALITY, OR FITNESS FOR A PARTICULAR PURPOSE. THE USE OF THE PRODUCT BY YOU IS AT YOUR OWN RISK. IN NO EVENT WILL OUR COMPANY BE LIABLE TO YOU 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), PRODUCT LIABILITY, OR OTHERWISE, IN CONNECTION WITH THE USE OF THE PRODUCT, EVEN IF OUR COMPANY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES OR LOSS.
- YOU ACKNOWLEDGE THAT THE NATURE OF THE INTERNET PROVIDES FOR INHERENT SECURITY RISKS, AND OUR COMPANY SHALL NOT TAKE ANY RESPONSIBILITIES FOR ABNORMAL OPERATION, PRIVACY LEAKAGE OR OTHER DAMAGES RESULTING FROM CYBER-ATTACK, HACKER ATTACK, VIRUS INFECTION, OR OTHER INTERNET SECURITY RISKS; HOWEVER, OUR COMPANY WILL PROVIDE TIMELY TECHNICAL SUPPORT IF REQUIRED.
- YOU AGREE TO USE THIS PRODUCT IN COMPLIANCE WITH ALL APPLICABLE LAWS, AND YOU ARE SOLELY RESPONSIBLE FOR ENSURING THAT YOUR USE CONFORMS TO THE APPLICABLE LAW. ESPECIALLY, YOU ARE RESPONSIBLE, FOR USING THIS PRODUCT IN A MANNER THAT DOES NOT INFRINGE ON THE RIGHTS OF THIRD PARTIES, INCLUDING WITHOUT LIMITATION, RIGHTS OF PUBLICITY, INTELLECTUAL PROPERTY RIGHTS, OR DATA PROTECTION AND OTHER PRIVACY RIGHTS. YOU SHALL NOT USE THIS PRODUCT FOR ANY PROHIBITED END-USES, INCLUDING THE DEVELOPMENT OR PRODUCTION OF WEAPONS OF MASS DESTRUCTION, THE DEVELOPMENT OR PRODUCTION OF CHEMICAL OR BIOLOGICAL WEAPONS, ANY ACTIVITIES IN THE CONTEXT RELATED TO ANY NUCLEAR EXPLOSIVE OR UNSAFE NUCLEAR FUEL-CYCLE, OR IN SUPPORT OF HUMAN RIGHTS ABUSES.
- IN THE EVENT OF ANY CONFLICTS BETWEEN THIS DOCUMENT AND THE APPLICABLE LAW, THE LATTER PREVAILS.

1 Reading Guide

Chapter	Description
Overview	Includes the ISAPI overview, applicable products, terms and definitions, abbreviations, and update history.
ISAPI Framework	Read the chapter to take a quick look at the ISAPI framework and basic functions.
Quick Start Guide	Read the chapter to quickly understand the programming process of basic functions such as authentication, message parsing, real-time live view, playback, and event uploading.
API Reference	Start programming according to API definitions.
How-To Video Guidance	How-to videos demonstrate detailed steps of different integration tasks.

2 Overview

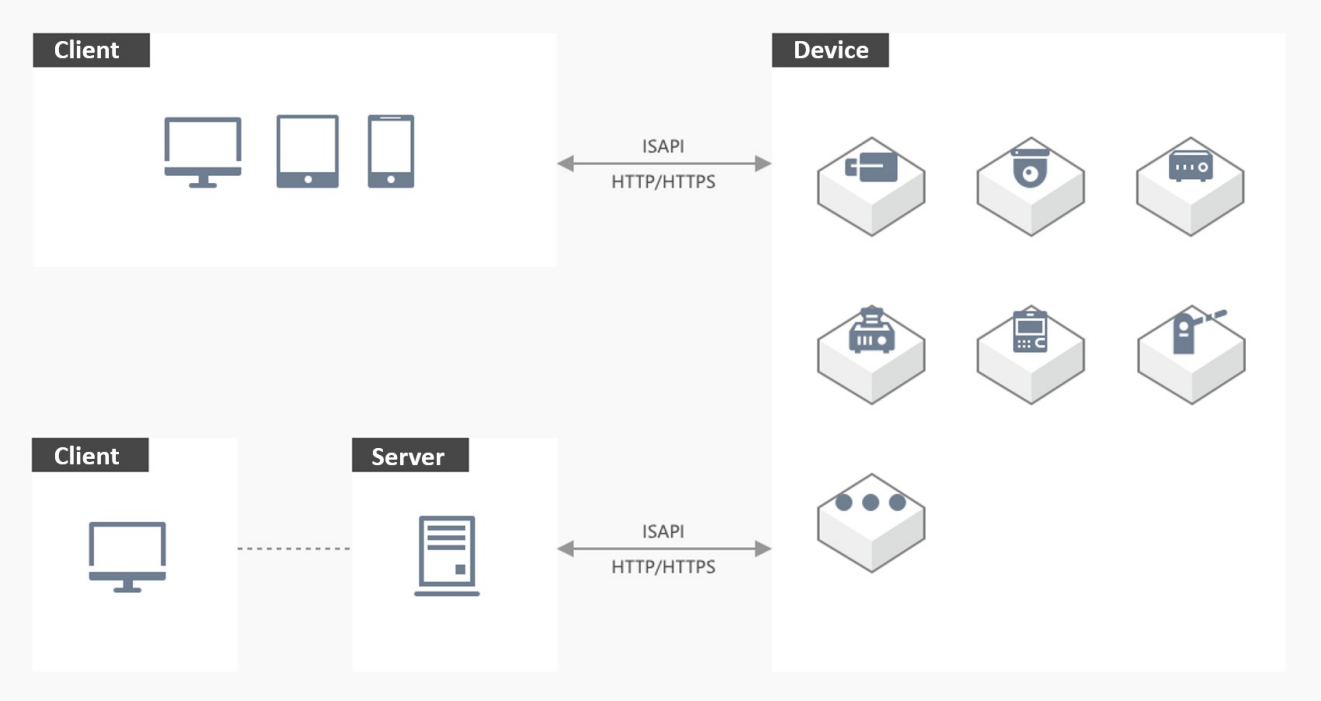
2.1 Introduction

Intelligent Security API (hereinafter referred to as ISAPI) is an application layer protocol based on HTTP (Hypertext Transfer Protocol) and adopts the REST (Representational State Transfer) architecture for communication between security devices (cameras, DVRs, NVRs, etc.) and the platform or client software.

Since established in 2013, ISAPI has included more than 11,000 APIs for different functions, including device management, vehicle recognition, parking lot management, intelligent facial application, access control management, interrogation management, and recording management. It is applicable to industries such as traffic, fire protection, education, and security inspection.

2.1.1 Application Scenario

When you integrate devices via ISAPI, the device acts as the server to listen on the fixed port and the user's application acts as the client to actively log in to the device for communication. To achieve the above goals, the device should be configured with a fixed IP address and the requests from the client can reach the server.



2.1.2 Layers in the Network Model

ISAPI is an application layer protocol based on HTTP, thereby it inherits all specifications and properties from HTTP.

Protocols frequently used along with ISAPI include SADP (Search Active Device Protocol) based on multicast for discovering and activating devices, RTSP (Real-Time Streaming Protocol) based on TCP/UDP for live view and video playback of devices, etc.

<table>
<tr><td rowspan="2">Application Layer</td><td>Device Discovery
SADP</td><td>Signaling Interaction
ISAPI (HTTP)</td><td>Media Stream
RTSP/RTP</td></tr>
<tr><td>Transport Layer</td><td>MCAST</td><td>TCP</td><td>TCP/UDP</td></tr>
<tr><td>Network Layer</td><td colspan="3">IP</td></tr>
<tr><td>Network Interface</td><td colspan="3">Hardware Drive Interface</td></tr>
</table>

2.2 Product Scope

- FingerPrint Terminals
  - Pro Series

    DS-K1A802AEF, DS-K1A802AEF-B, DS-K1A802AF, DS-K1A802AF-B, DS-K1A802AMF, DS-K1A802AMF-B, DS-K1A802EF-B, DS-K1A802MF, DS-K1A802MF-B, DS-K1T501SF, DS-K1T502DBFWX, DS-K1T502DBFWX-C, DS-K1T8005EFWX, DS-K1T8005EFWX-B, DS-K1T8005EFX, DS-K1T8005MFWX, DS-K1T8005MFWX-B, DS-K1T8005MFX, DS-K1T804, DS-K1T804A, DS-K1T804AEF, DS-K1T804AF, DS-K1T804AMF, DS-K1T804BEF, DS-K1T804BF, DS-K1T804BMF, DS-K1T804MF, DS-K1T807EBFWX-E1, DS-K1T807EBFWX-QRE1, DS-K1T807MBFWX-E1, DS-K1T807MBFWX-QRE1, DS-K1T808EFWX, DS-K1T808EFWX-B, DS-K1T808EFX, DS-K1T808MFWX, DS-K1T808MFWX-B, DS-K1T808MFX

2.3 Terms And Definitions

2.3.3 Event

It refers to the information uploaded by the device. The event is uploaded by the device in real time for the immediate response from the client or the platform. If the device is offline, the event will be stored in the cache first and then it will be uploaded again when the connection is restored.

2.3.2 Arming

Arming means that the client establishes connection with the device so that events can be uploaded to the client via the connection. The client can subscribe to some event types, and the device will upload the specified events only, otherwise the device will upload all types of events to the client.

2.3.4 Listen

After the platform starts listening service, when an event occurs, the event information will be sent to the listening port of the platform based on the IP address and port No., then the connection will be closed.

2.3.5 Listening Host

The listening host service can be enabled to receive the event information from devices.

2.3.16 Person-Based Access Control

The person ID is the unique identifier for person management. A person ID is linked to a person's credentials (card,

fingerprint, and face picture) and permissions.

2.3.17 Access Permission

Users can set who can open which doors at what time. The access permission contains information about the person, time, and door.

2.3.8 Credential Type

Credentials are the data which are used for recognizing specific persons. Cards, fingerprints, and face pictures can be the credentials and linked to the specified persons. When the device detects the credential, it can recognize the person whom the credential links to via the comparison algorithm.

2.3.9 Person Type

Persons can be divided into normal persons, visitors, and blocklist persons according to different areas. In the access control system, normal persons have permanent permissions to access the specified areas, visitors have temporary permissions to access the specified areas, and blocklist persons do not have the permissions to access the specified areas.

2.3.10 Group

A group of persons which is used in multi-factor authentication. The same person can belong to different groups at the same time. Up to 4 groups can be added at the same time.

2.3.11 Multi-Factor Authentication

Persons in the group can only open the door by the configured authentication rule, such as swiping card, authenticated by fingerprint, face picture, or iris.

2.3.18 Alarm Receiving Center

The ARC receives alarm information from devices and provides alarm services.

2.3.19 Event Code

An event code is used to identify a specific event and corresponds to a CID code.

2.4 Symbols And Acronyms

HTTP: Hypertext Transfer Protocol

ACS: Access Control System. The access control system controls the entrance and exit channels. The system consists of card readers, access controllers, electric locks, exit buttons, cards, application software, etc.

None.

ARC: Alarm Receiving Center

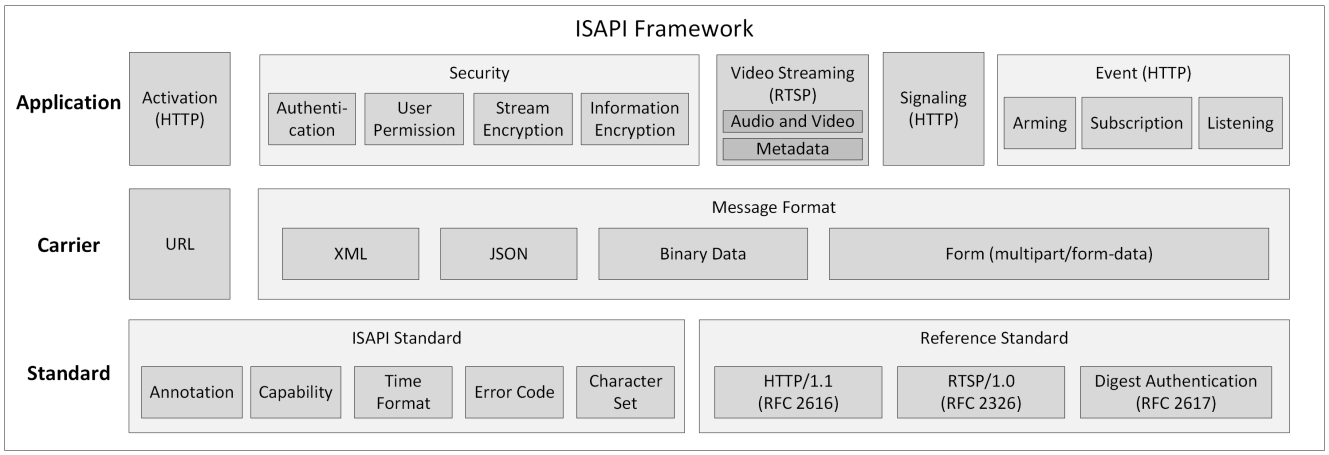
CID: Event Code (Contac ID)

2.5 Update History

No update record

3 ISAPI Framework

3.1 Overview



Notes:

In general, ISAPI refers to the communication protocol based on the HTTP standard. As ISAPI is usually used along with RTSP (Real-Time Streaming Protocol), the RTSP standard is brought into the ISAPI system.

The metadata scheme for transmitting additional information of the stream is extended based on the RTSP standard to transmit the video stream and the structured intelligent information of the stream simultaneously. It is compatible with the RTSP standard.

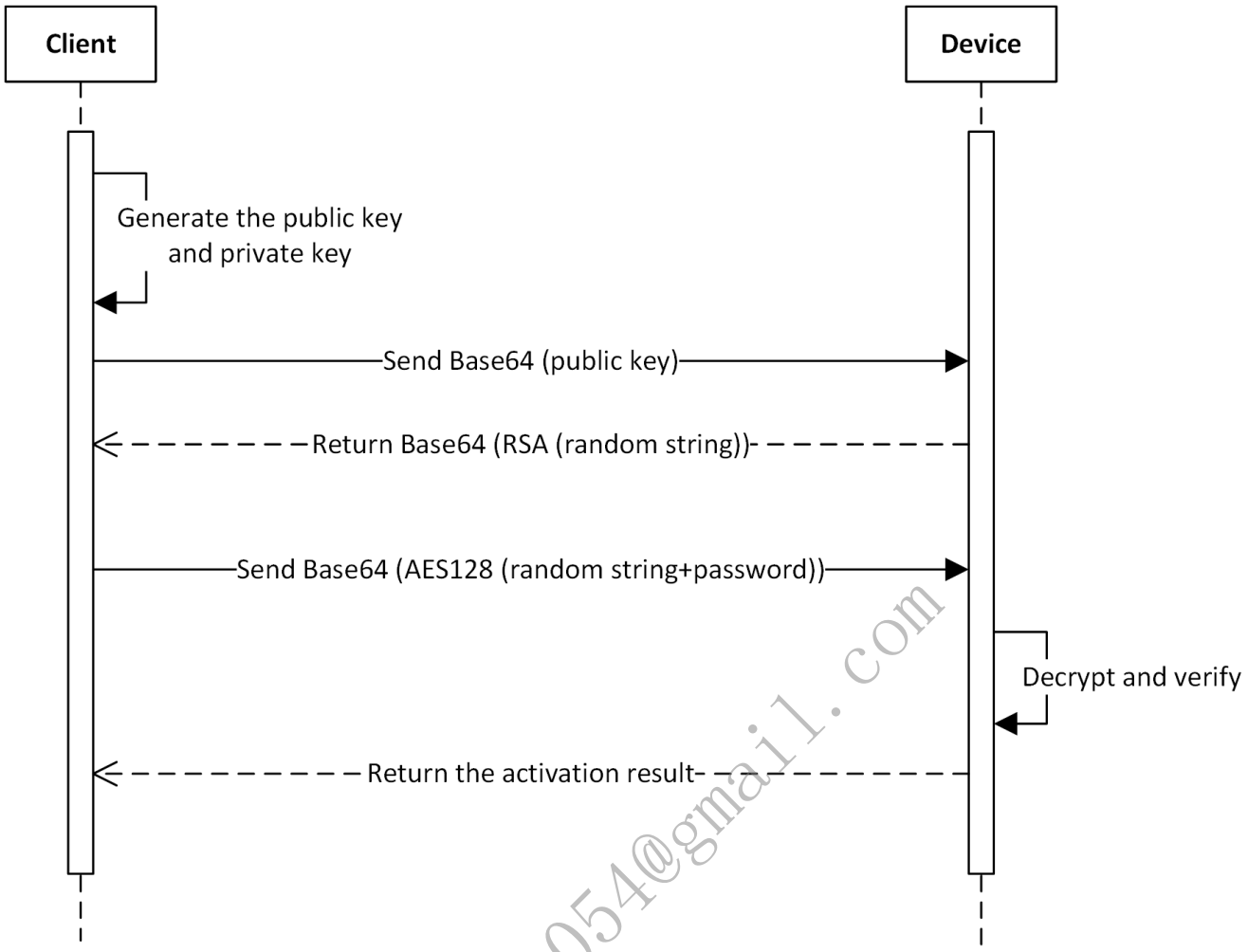
3.2 Activation

The purpose of activation is to ensure that the user can set the password for the device and the password meets the security requirement. After the device is activated, you can use the related functions.

ISAPI is a communication protocol running on the application layer. When activating the device via ISAPI, you should know the device's IP address and make sure that the device is connected to the client.

The web application built in the device supports activating the device via ISAPI. When you enter the device's IP address in the address bar of the web browser on the PC, you can activate the device according to the activation guide.

If you want to activate the device on your own application, you need to integrate the activation function via ISAPI. The API calling flow and related APIs are shown below.



Firstly, two operations are defined:

- bytesToHexstring: it is used to convert a byte array (the length is N) to a hexadecimal string (the length is 2N). For example, `127,10,23` -> `7f0a17`
- hexStringToBytes: it is used to convert a hexadecimal string (the length is 2N) to a byte array (the length is N). For example, `7f0a17` -> `127,10,23`

1. The client generates a public and private key pair (1024 bits), and gets the 128-byte modulus in the public key (hereinafter referred to as public key modulus). If the length is longer than 128, the leading 0 needs to be removed.
2. The client converts the public key modulus to a 256-byte public key string via bytesToHexstring and sends the public key string to the device in XML message (related URI: `POST /ISAPI/Security/challenge`) after being encoded by Base64.
3. The device parses the request to obtain a 256-byte public key string decoded by Base64 and converts it to a 128-byte public key modulus via hexStringToBytes. The complete public key is the combination of obtained public key modulus and public exponent (the default value is `'010001'`).
4. The device generates a 32-byte hexadecimal random string, calls the RSA API to encrypt the random string with the private key, converts the encrypted data to a string via bytesToHexstring, encodes the string by Base64, and then sends it to the client.
5. The client decodes the string from the device by Base64, converts it via hexStringToBytes to get the encrypted data, decrypts the encrypted data with the private key via RSA to obtain a 32-byte hexadecimal random string, converts the obtained string via hexStringToBytes to get a 16-byte AES key. Then the client uses the AES key to encrypt the `"string consisting of the first 16 characters of the random string and the real password"` by AES128 ECB mode (with zero-padding method) to get a ciphertext, and converts the ciphertext via bytesToHexstring, encodes it by Base64, and sends it to the device in XML message (related URI: `PUT /ISAPI/System/activate`). Note: If the first 16 characters of the random string are `aaaabbbbccccdddd` and the real password is `Abc12345`, the data before encryption is `aaaabbbbccccddddAbc12345`. This can ensure that the client uses the random string as the key for encryption.
6. The device decodes the string by Base64, converts it via hexStringToBytes to get the ciphertext, uses the AES key to decrypt the ciphertext by AES128 ECB mode, and gets the real password via removing the first 16 characters.

7. The device verifies the password and returns the activation result.

Notes:

- You can get the device's activation status by calling the URI `GET /SDK/activateStatus` which requires no authentication.
- Devices also support to be activated via SADP (Search Active Device Protocol) which is based on the communication protocol of the data link layer. With SADP, you do not have to know the IP address of the device but need to ensure that the device and the application running SADP are connected to the same router. SADP also supports discovering devices in the LAN, changing the password of the devices, and so on. The HCSadpSDK is provided for SADP integration, including the developer guide, plug-in, and sample demo which can be used as a simple SADP tool.

3.3 Security Mechanism

3.3.1 Authentication

When the client applications send requests to devices, they need to use digest authentication (see details in RFC 7616) for identity authentication.

Currently, all mainstream request class libraries of HTTP have encapsulated digest authentication. See details in Authentication of Quick Start Guide.

3.3.2 User Permission

There are three kinds of users with different permissions for access control and management.

Administrator: Has the permission to access all supported resources and should keep activated all the time. It is also known as "admin".

Operator: Has the permission to access general resources and a part of advanced resources.

Normal User: Only has the permission to access general resources.

3.3.3 Information Encryption

During ISAPI integration, the HTTPS service of devices is enabled by default. When the client applications communicate with devices via HTTPS, the information can be transmitted securely.

3.4 Video Streaming

3.4.1 Audio and Video Stream

ISAPI supports getting and setting stream media parameters of the device, such as video resolution, encoding format, and stream.

Cameras support standard RTSP (Real-Time Streaming Protocol, see details in RFC 7826). Client applications can get the stream from devices via RTSP.

For details about real-time streaming and video playback, refer to **Real-Time Live View** and **Playback** in Quick Start Guide.

3.4.2 Metadata

The metadata is the structured intelligent information generated by intelligent devices. When the client applications get the audio and/or video stream from devices via RTSP, the metadata will be returned by the device at the same time. For example, to display the face target frame, face information, vehicle target frame, license plate number, vehicle information, and other information on the video stream, the client applications can overlay the above information on the video image.

Before using the metadata, you need to enable the metadata function of the device and then get the stream from the device via RTSP. Some devices support subscribing to the metadata by type. For details about the process of integrating the metadata function, refer to **Metadata Management**.

4 Quick Start Guide

4.1 Authentication

When the client applications send requests to the devices, they need to use digest authentication (see details in RFC 7616) for identity authentication.

Client applications only need to call APIs of the class library to implement the digest authentication. The sample code is shown below.

4.1.1 C/C++ (libcurl)

```
// #include <curl/curl.h>
// Callback Function
static size_t OnWriteData(void* buffer, size_t size, size_t nmemb, void* lpVoid)
{
    std::string* str = dynamic_cast<std::string*>((std::string *)lpVoid);
    if( NULL == str || NULL == buffer )
    {
        return -1;
    }

    char* pData = (char*)buffer;
    str->append(pData, size * nmemb);
    return nmemb;
}

std::string strUrl = "http://192.168.18.84:80/ISAPI/System/deviceInfo";
std::string strResponseData;
CURL *pCurlHandle = curl_easy_init();
curl_easy_setopt(pCurlHandle, CURLOPT_CUSTOMREQUEST, "GET");
curl_easy_setopt(pCurlHandle, CURLOPT_URL, strUrl.c_str());
// Set the user name and password
curl_easy_setopt(pCurlHandle, CURLOPT_USERPWD, "admin:admin12345");
// Set the authentication method to the digest authentication
curl_easy_setopt(pCurlHandle, CURLOPT_HTTPAUTH, CURLAUTH_DIGEST);
// Set the callback function
curl_easy_setopt(pCurlHandle, CURLOPT_WRITEFUNCTION, OnWriteData);
// Set the parameters of the callback function to get the returned information
curl_easy_setopt(pCurlHandle, CURLOPT_WRITEDATA, &strResponseData);
// Timeout settings for receiving the data. If receiving data is not completed within 5 seconds, the application will exit directly
curl_easy_setopt(pCurlHandle, CURLOPT_TIMEOUT, 5);
// Set the redirection times to avoid too many redirections
curl_easy_setopt(pCurlHandle, CURLOPT_MAXREDIRS, 1);
// Connection timeout duration. If the duration is too short, the client application will be disconnected before the data request sent by the application
reaches the device
curl_easy_setopt(pCurlHandle, CURLOPT_CONNECTTIMEOUT, 5);
CURLcode nRet = curl_easy_perform(pCurlHandle);
if (0 == nRet)
{
    // Output the received message
    std::cout << strResponseData << std::endl;
}
curl_easy_cleanup(pCurlHandle);
```

4.1.2 C# (WebClient)

```
// using System.Net;
// using System.Net.Security;
try
{
    string strUrl = "http://192.168.18.84:80/ISAPI/System/deviceInfo";
    WebClient client = new WebClient();
    // Set the user name and password
    client.Credentials = new NetworkCredential("admin", "admin12345");
    byte[] responseData = client.DownloadData(strUrl);
    string strResponseData = Encoding.UTF8.GetString(responseData);
    // Output received information
    Console.WriteLine(strResponseData);
}
catch (Exception ex)
{
    Console.WriteLine(ex.Message);
}
```

4.1.3 Java (HttpClient)

```java
// import org.apache.commons.httpclient.HttpClient;
String url = "http://192.168.18.84:80/ISAPI/System/deviceInfo";
HttpClient client = new HttpClient();
// Set the user name and password
UsernamePasswordCredentials creds = new UsernamePasswordCredentials("admin", "admin12345");
client.getState().setCredentials(AuthScope.ANY, creds);
GetMethod method = new GetMethod(url);
method.setDoAuthentication(true);
int statusCode = client.executeMethod(method);
byte[] responseData = method.getResponseBodyAsString().getBytes(method.getResponseCharSet());
String strResponseData = new String(responseData, "utf-8");
method.releaseConnection();
// Output received information
System.out.println(strResponseData);
```

4.1.4 Python (requests)

```python
# - *- coding: utf-8 -*-

import requests
request_url = 'http://192.168.18.84:80/ISAPI/System/deviceInfo'
# Set the authentication information
auth = requests.auth.HTTPDigestAuth('admin', 'admin12345')
# Send the request and receive response
response = requests.get(request_url, auth=auth)
# Output response content
print(response.text)
```

4.2 Message Parsing

4.2.1 Message Format

During the process of communication and interaction via ISAPI, the request and response messages are often text data in XML or JSON format. Besides that, the data of firmware packages and configuration files is in binary format. A request can also be in form format with multiple formats of data (multipart/form-data).

4.2.1.1 XML

Generally, the `Content-Type` in the headers of the HTTP request is `application/xml; charset="UTF-8"`.

Request and response messages in XML format are all encoded with UTF-8 standards in ISAPI.

The namespace `http://www.isapi.org/ver20/XMLSchema` and ISAPI version number `2.0` of XML messages are configured by default, see the example below.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<NodeList xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <Node>
    <id>1</id>
    <enabled>true</enabled>
    <nodeName>nodeName</nodeName>
    <level>level1</level>
  </Node>
</NodeList>
```

4.2.1.2 JSON

The `Content-Type` in the headers of the HTTP request is often `application/json`.

To distinguish between APIs with XML messages and those with JSON messages, ISAPI adds the query parameter format=json to all request URLs with JSON messages, e.g., `http://192.168.1.1:80/ISAPI/System/Sensor/thermometrySensor?format=json`. Messages of request URLs without the query parameter format=json are usually in XML format. However, there may be some exceptions, and the message format is subject to the API definition.

Request and response messages in JSON format are all encoded by UTF-8 in ISAPI.

4.2.1.3 Binary Data

For the firmware and configuration files, the `Content-Type` in the header of an HTTP request is often `application/octet-stream`.

4.2.1.4 Form (multipart/form-data)

When multiple pieces of data are submitted at the same time in an ISAPI request (e.g., the person information and face picture need to be submitted at the same time when a face record is added to the face picture library), the `Content-Type` in the header of the corresponding HTTP request is usually `multipart/form-data, boundary=AaB03x`, where the boundary is a variable used to separate the entire HTTP body into multiple units and each unit is a piece of data with its own headers and body. In `Content-Disposition` of form unit headers, the `name` property refers to the form unit name, which is required for all form units; the `filename` property refers to the file name of form unit body, which is required only when the form unit body is a file. In headers of form units, `Content-Length` refers to the body length, which starts after CRLF(`\r\n`) and ends before two hyphens (`--`) of next form. There should be a CRLF used as the delimiter of two form units before two hyphens (`--`), and the `Content-Length` of previous form unit does not include the CRLF length. For the detailed format description, refer to RFC 1867 (Form-Based File Upload in HTML). Pay attention to two hyphens (`--`) before and after the boundary.

Notes

- In RFC specifications, it is strongly recommended to contain the field `Content-Length` in the entity header, and there is no requirement that the field `Content-Length` should be contained in the header of each form element. The absence of field `Content-Length` in the header should be considered when the client and device programs parse the form data.
- To avoid the conflict between message content and boundary value, it is recommended to use a longer and more complex string as the boundary value.

The example of ISAPI form data submitted by a client to a device is as follows.

```
POST /ISAPI/Intelligent/FDLib/pictureUpload
Content-Type: multipart/form-data; boundary=e5c2f8c5461142aea117791dade6414d
Content-Length: 56789

--e5c2f8c5461142aea117791dade6414d
Content-Disposition: form-data; name="PictureUploadData";
Content-Type: application/xml
Content-Length: 1234

<PictureUploadData/>
--e5c2f8c5461142aea117791dade6414d
Content-Disposition: form-data; name="face_picture"; filename="face_picture.jpg";
Content-Type: image/jpeg
Content-Length: 34567

Picture Data
--e5c2f8c5461142aea117791dade6414d--
```


The example of ISAPI form data responded from a device to a client is as follows.

In ISAPI messages, when there are multiple form units, three nodes (`pid`, `contentid`, and `filename`) are used for linking form units. The corresponding relations are as follows:

Node Name	Form Field	Description
pid	name	pid in XML/JSON messages corresponds to the name property of Content-Disposition in form headers.
contentid	Content-ID	contentid in XML/JSON messages corresponds to Content-ID in form headers.
filename	filename	filename in XML/JSON messages corresponds to filename property of Content-Disposition in form headers.

```
HTTP/1.1 200 OK
Content-Type: multipart/form-data; boundary=136a73438ecc4618834b999409d05bb9
Content-Length: 56789

--136a73438ecc4618834b999409d05bb9
Content-Disposition: form-data; name="mixedTargetDetection";
Content-Type: application/json
Content-Length: 811

{
    "ipAddress": "172.6.64.7",
    "macAddress": "01:17:24:45:D9:F4",
    "channelID": 1,
    "dateTime": "2009-11-14T15:27+08:00",
    "eventType": "mixedTargetDetection",
    "eventDescription": "Mixed target detection",
    "deviceID": "123456789",
    "CaptureResult": [{
        "targetID": 1,
        "Human": {
            "Rect": {
                "height": 1.0,
                "width": 1.0,
                "x": 0.0,
                "y": 0.0
            },
            "contentID1": "humanImage", /*human body thumbnail*/
            "contentID2": "humanBackgroundImage", /*human body background picture*/
            "pId1": "9d48a26f7b8b4f2390c16808f93f3534", /*human body thumbnail ID */
            "pId2": "5EE7078E07BB47CF860DE8E4E9A85F28" /*ID of human body background picture*/
        }
    }]
}
--136a73438ecc4618834b999409d05bb9
Content-Disposition: form-data; name="9d48a26f7b8b4f2390c16808f93f3534"; filename="humanImage.jpg";
Content-Type: image/jpeg
Content-Length: 34567
Content-ID: humanImage

Picture Data
--136a73438ecc4618834b999409d05bb9
Content-Disposition: form-data; name="5EE7078E07BB47CF860DE8E4E9A85F28"; filename="humanBackgroundImage.jpg";
Content-Type: image/jpeg
Content-Length: 345678
Content-ID: humanBackgroundImage

Picture Data
--136a73438ecc4618834b999409d05bb9--
```


4.2.2 Annotation

The field descriptions of ISAPI request and response messages are marked as annotations in the example messages as shown below.

```
<?xml version="1.0" encoding="UTF-8"?>

<NodeList xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, node list, attr:version{req, string, version No., range:[,]}-->
  <Node>
    <!--ro, opt, object, node information-->
    <id>
      <!--ro, req, int, node No., range:[,], step:, unit:, unitType:-->1
    </id>
    <enabled>
      <!--ro, opt, bool, whether to enable-->true
    </enabled>
    <nodeName>
      <!--ro, req, string, node name, range:[1,32]-->test
    </nodeName>
    <level>
      <!--ro, opt, enum, level, subType:string,
      [level1#Level 1,level2#Level 2,level3#Level 3]-->level1
    </level>
  </Node>
</NodeList>
```

```
{
    "name":  "test",
    /*ro, req, string, name, range:[1,32]*/
    "type":  "type1",
    /*ro, req, enum, type, subType:string, [type1#type 1,type2#type 2]*/
    "enabled":  true,
    /*ro, opt, bool, enable or not, desc:xxxxxxx*/
    "NodeList": {
    /*opt, object, node list, dep:and,{$.enabled,eq,true}*/
        "scene":  1,
        /*req, enum, scene, subType:int, [1#scene 1; 2#scene 2; 3#scene 3]*/
        "ID":  1
        /*req, int, No., range:[1,8], step:, unit:, unitType:*/
    }
}
```

Key annotations are shown in the table below.

Annotation	Description	Remark
ro	Attribute: Read-Only	This field can only be obtained and cannot be edited.
wo	Attribute: Write-Only	This field can only be edited and cannot be obtained.
req	Attribute: Required	This field is required for request messages sent to the device and response messages returned from the device.
opt	Attribute: Optional	This field is optional for request messages sent to the device and response messages returned from the device.
dep	Attribute: Dependent	This field is valid and required when specific conditions are satisfied.
object	Field Type: Object	The field of type object contains multiple sub-fields.
list	Field Type: List	The subType following it refers to the data type of sub-items in the list.
subType	Field Type: String	The range following it refers to the maximum and the minimum string size of the field.
int	Field Type: Int	The range following it refers to the maximum and the minimum value of the field.
float	Field Type: Float	The range following it refers to the maximum and the minimum value of the field.
bool	Field Type: Boolean	The value can be true or false.
enum	Field Type: Enumeration	The subType following it indicates that the enumerators are of type string or int. The [] following the subType contains all enumerators.
subType	Sub-Type of Field	When the type of field is list or enum, the value of subType is the data type of each sub-object.
desc	Field Description	The detailed description of the field.

4.2.3 Capability Set

ISAPI has designed capability sets for almost all functions, APIs, and fields. URLs for getting the capability set end with `/capabilities`. Some URLs may contain query parameters in the format: `/capabilities?format=json&type=xxx`.

There are two types of fields in the capability message of ISAPI: whether the device supports a function and the value range of a field in an API.

Whether the device supports a function: it is often in the format `isSupportXxxxxxxx`, which indicates that whether the device supports a function and a set of APIs for implementing this function.

The capability message example in JSON format is shown below:

```
{
    "isSupportMap":  true,
    /*ro, opt, bool, whether it supports the e-map function, desc:/ISAPI/SDT/Management/map/capabilities?format=json*/
    "isSupportAlgTrainResourceInfo":  true,
    /*ro, opt, bool, whether it supports only getting the resource information of the algorithm training platform,
desc:/ISAPI/SDT/algorithmTraining/ResourceInfo?format=json*/
    "isSupportAlgTrainAuthInfo":  true,
    /*ro, opt, bool, whether it supports ony getting the authorization information of the algorithm training platform,
desc:/ISAPI/SDT/algorithmTraining/SoftLock/AuthInfo?format=json*/
    "isSupportAlgTrainNodeList":  true,
    /*ro, opt, bool, whether it supports only getting the node information of the algorithm training platform, desc:/ISAPI/SDT/algorithmTraining/NodeList?
format=json*/
    "isSupportNAS":  true
    /*ro, opt, bool, whether it supports mounting and unmounting NAS, desc:/ISAPI/SDT/Management/NAS/capabilities?format=json*/
}
```

The capability message example in XML format is shown below:

```
<isSupportNetworkStatus>
    <!--ro, opt, bool, whether it supports searching the network status, desc: related API (/ISAPI/System/Network/status?format=json)-->true
</isSupportNetworkStatus>
```

The value range of the field: the maximum value, minimum value, the maximum size, the minimum size, options, and so on of each field of the API.

The example of JSON format is shown below:

```
{
    "boolType": {
        /*req, object, example of the capability of type bool*/
        "@opt": [true, false]
        /*req, array, options, subType: bool*/
    },
    "integerType": {
        /*req, object, example of the capability of type integer*/
        "@min": 0,
        /*ro, req, int, the minimum value*/
        "@max": 100
        /*ro, req, int, the maximum value*/
    },
    "stringType": {
        /*req, object, example of the capability of type string*/
        "@min": 0,
        /*ro, req, int, the minimum string size*/
        "@max": 32
        /*ro, req, int, the maximum string size*/
    },
    "enumType": {
        /*req, object, capability example of type enum*/
        "@opt": ["enum1", "enum2", "enum3"]
        /*req, array, options, subType: string*/
    }
}
```

The example of XML format is shown below:

```
<boolType opt="true,false" def="true">
    <!--ro, opt, bool, example of the capability of type bool-->true
</boolType>
<integerType min="0" max="100">
    <!--ro, opt, int, example of the capability of type int-->50
</integerType>
<stringType min="0" max="64">
    <!--ro, opt, string, example of the capability of type string-->test
</stringType>
<enumType opt="red,white,black" def="red">
    <!--ro, opt, string, example of the capability of type enum-->white
</enumType>
```

Note: For the same capability set, devices of different models and versions may return different results. The values shown in this document are only examples for reference. The capability set actually returned by the device takes precedence.

4.2.4 Time Format

ISAPI adopts ISO 8601 Standard Time Format, which is the same as W3C Standard Date and Time Formats.

Format: `YYYY-MM-DDThh:mm:ss.sTZD`

```
YYYY = the year consisting of four decimal digits
MM = the month consisting of two decimal digits (01-January, 02-February, and so forth)
DD = the day consisting of two decimal digits (01 to 31)
hh = the hour consisting of two decimal digits (00 to 23, a.m. and p.m. are not allowed)
mm = the minute consisting of two decimal digits (00 to 59)
ss = the second consisting of two decimal digits (00 to 59)
s = one or more digits representing the fractional part of a second
TZD = time zone identifier (Z or +hh:mm or -hh:mm)
```

Example: 2017-08-16T20:17:06.123+08:00 refers to 20:17:06.123 on August 16, 2017 (local time which is 8 hours ahead of UTC). The plus sign (+) indicates that the local time is ahead of UTC, and the minus sign (-) means that the local time is behind UTC.

After the DST is enabled, the local time and time difference will change compared with UTC, and the values of related fields also need to be changed. Disabling the DST will bring into the opposite effect.

Example: In 1986, the DST was in effect from May 4 at 2:00 a.m. (GMT+8). During the DST period, the clocks were moved one hour ahead, which means that there was one less hour on that day. When the DST ends at 2:00 a.m. on September 14, 1986, the clocks were moved one hour back and there was an extra hour on that day. The changes of the time are as follows:

- DST Starts: 1986-05-04T02:00:00+08:00 --> 1986-05-04T03:00:00+09:00
- DST Ends: 1986-09-14T02:00:00+09:00 --> 1986-09-14T01:00:00+08:00

Notes:

- The time difference cannot be simply used to determine the time zone. Because when the DST starts, the time difference will change and it cannot represent the actual time zone.
- Both TZ (UTC time, e.g., 1986-05-03T18:00:00Z) and TD (local time and time difference, e.g., 1986-05-04T02:00:00+08:00) meet the time format standards of ISO 8601. In ISAPI, the TD format is recommended to be used in messages sent from the user applications and the devices.
- A few old-version devices will return the time in TZ format. For representing the time difference information and forward compatibility, an extra field `timeDiff` is added as shown in the example below. User applications need to support both TD format and TZ format when parsing the time in the messages returned by devices.

```
{
    "dateTime": "1986-05-03T18:00:00Z", /*device time. The value in TZ format is the UTC time and the value in TD format is the time difference between the device's local time and UTC*/
    "timeDiff": "+08:00" /*optional, time difference between the local time and UTC time. If this field does not exist, the user application will convert the dateTime into the local time for use*/
}
```


4.2.5 Character Set

To prevent characters not commonly used from resulting in exceptions in device programs and user applications, ISAPI limits the valid field values of type string to a specific range of characters. Character sets allowed to be used in the fields of type string in ISAPI are listed below.

- Single-byte character set: lowercase letters (`a-z`), uppercase letters (`A-Z`), digits (`0-9`), and special characters (see details in the table below).
- Multi-byte character set: language characters based on Unicode and encoded by UTF-8 (UTF-8 encoding is a transformation format of Unicode character set. For details, refer to RFC 2044).

No.	Name	Special Character	No.	Name	Special Character
1	Open Parenthesis	(	18	Dollar Sign	$
2	Close Parenthesis	)	19	Percent Sign	%
3	Plus Sign	+	20	Ampersand	&
4	Comma	,	21	Close Single Quotation Mark	'
5	Minus Sign	-	22	Asterisk	*
6	Period	.	23	Slash	/
7	Semicolon	;	24	Smaller Than	<
8	Equal Sign	=	25	Greater Than	>
9	At Sign	@	26	Question Mark	?
10	Open Square Bracket	[	27	Caret	^
11	Close Square Bracket	]	28	Open Single Quotation Mark	`
12	Underscore	_	29	Vertical Bar	|
13	Open Brace	{	30	Tilde	~
14	Close Brace	}	31	Double Quotation Marks	"
15	Space		32	Colon	:
16	Exclamation Mark	!	33	Backslash	|
17	Octothorpe	#			

The valid characters that can be used in some special fields are listed below.

- User name: lowercase letters (`a-z`), uppercase letters (`A-Z`), digits (`0-9`), and characters from No. 1 to No. 30 in the special character table.
- Password: User Name: lowercase letters (`a-z`), uppercase letters (`A-Z`), digits (`0-9`), and characters from No. 1 to No. 33 in the special character table.
- Names displayed on the UI (device name, person name, face picture library name, etc.): lowercase letters (`a-z`), uppercase letters (`A-Z`), digits (`0-9`), characters from No. 1 to No. 15 in the special character table, and multi-byte characters.
- Normal fields of type string support lowercase letters (`a-z`), uppercase letters (`A-Z`), digits (`0-9`), characters from No. 1 to No. 15 in the special character table, and multi-byte characters by default.

4.2.6 Error Processing

When requesting via ISAPI failed (the HTTP status code is not 200), the device will return the HTTP status code and ISAPI error code. For HTTP status codes, refer to 10 Status Code Definitions in RFC 2616. For ISAPI error codes, refer to Error Code Dictionary.

Message Example:

```
HTTP/1.1 403 Forbidden
Content-Type: application/json; charset="UTF-8"
Date: Thu, 15 Jul 2021 20:43:30 GMT
Content-Length: 229
Connection: Keep-Alive

{
    "requestURL": "/ISAPI/Event/triggers/notifications/channels/whiteLightAlarm",
    "statusCode": 4,
    "statusString": "Invalid Operation",
    "subStatusCode": "notSupport",
    "errorCode": 1073741825,
    "errorMsg": "notSupport"
}
```

4.3 Event Uploading

When the rules configured on the device are triggered, the device will generate event messages (e.g., motion detection, etc.) and actively upload them to the client. ISAPI supports three methods to receive event messages uploaded by the device, that is, in arming mode, in listening mode, and via subscription.

4.3.1 Arming

The client establishes a HTTP persistent connection with the device to receive event messages from the device.

There are two methods (arming with subscription and arming without subscription) to receive events from the device. The arming without subscription is to get all event messages from the device via HTTP GET method, while the arming with subscription is to get messages of subscribed events via HTTP POST method.

Notes

- ISAPI arming (with or without subscription) uses the HTTP/HTTPS persistent connection. Due to the simplex channel communication mode of HTTP, after establishing the arming connection, the device will send event messages continuously, while it's not supported for clients to send any message to the device via the connection.
- When the heartbeat timed out and no message is received from the device, you should terminate the arming connection and try establishing a new one.

4.3.1.1 Arming without Subscription

1. Establish the connection of arming without subscription: `GET /ISAPI/Event/notification/alertStream` and keep the connection alive via configuring `Connection: keep-alive` in HTTP headers on the client.
2. Receive events sent by the device. The event message will be separated and parsed by boundary. For parsing details, see **Event Message Parsing** below.
3. Terminate the arming connection when no event message needs to be received.

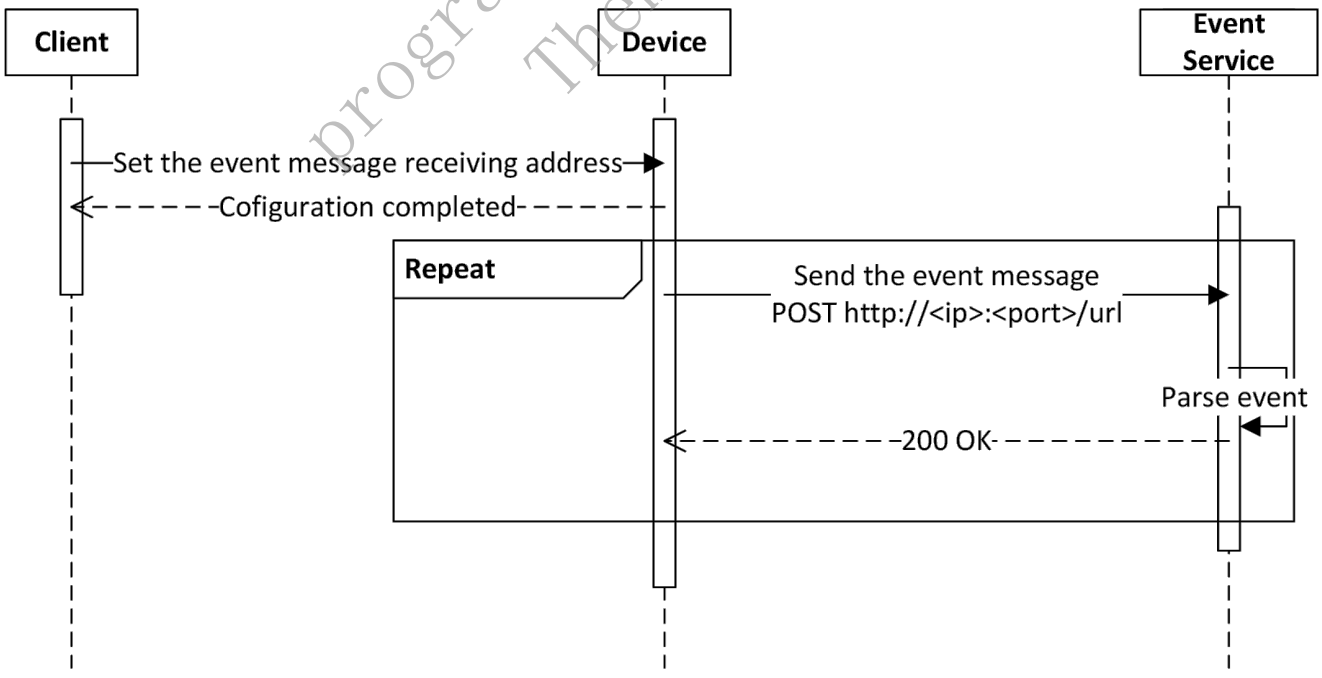
Event Message Parsing:

```
GET /ISAPI/Event/notification/alertStream HTTP/1.1
Host: <data_gateway_ip>
Connection: Keep-Alive

HTTP/1.1 401 Unauthorized
Date: Sun, 01 Apr 2018 18:58:53 GMT
Server:
Content-Length: 178
Content-Type: text/html
Connection: keep-alive
Keep-Alive: timeout=10, max=99
WWW-Authenticate: Digest qop="auth", realm="IP Camera(C2183)", nonce="4e5468694e7a42694e7a4d364f4449354d7a6b354d54513d", stale="FALSE"

GET /ISAPI/Event/notification/alertStream HTTP/1.1
Authorization: Digest username="admin",realm="IP
Camera(C2183)",nonce="4e5468694e7a42694e7a4d364f4449354d7a6b354d54513d",uri="/ISAPI/Event/notification/alertStream",cnonce="3d183a245b8729121ae4ca3d41b90f18
",nc=00000001,qop="auth",response="f2e0728991bb031f83df557a8f185178"
Host: 10.6.165.192

HTTP/1.1 200 OK
MIME-Version: 1.0
Connection: close
Content-Type: multipart/mixed; boundary=<frontier>

--<frontier>
Content-Type: application/xml; charset="UTF-8"  <!--some event messages are uploaded in JSON format, and the upper layer needs to distinguish the message
format accroding to Content-Type when parsing event messages-->
Content-Length: text_length

<EventNotificationAlert/>
--<frontier>
Content-Disposition: form-data; name="Picture_Name"
Content-Type: image/pjpeg
Content-Length: image_length

[Picture Data]
--<frontier>
```

Note: `<data_gateway_ip>` and `<frontier>` are variables, `[Picture Data]` indicates the raw data of a picture.

4.3.2 Listening

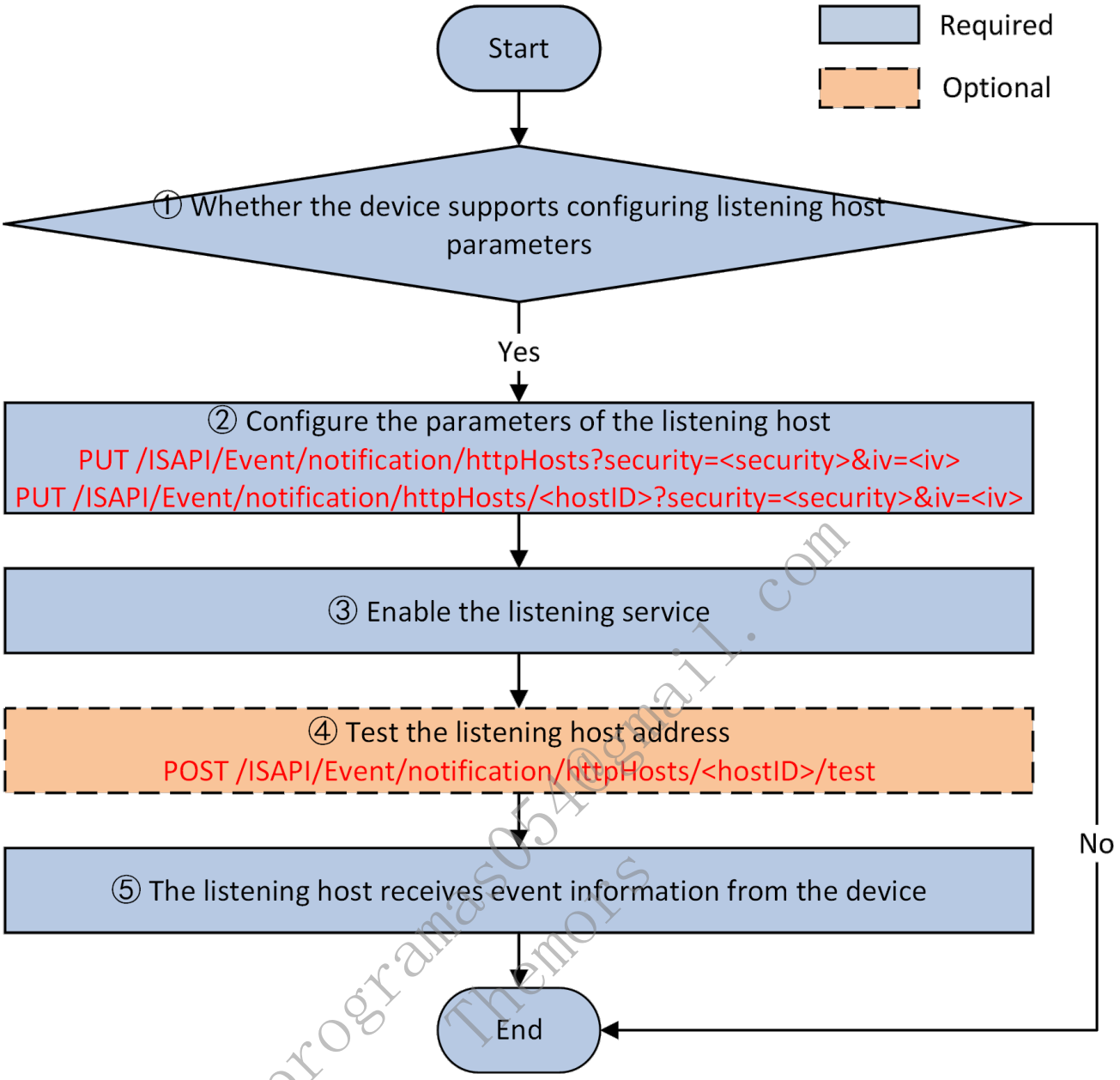
After a client enables the listening service, when an event occurs, the device will send the event information actively to the configured event receiving address. The event receiving address should be valid and configured on the device.



Notes:

- The client and event service can be the same program.
- In listening mode, no heartbeat information is generated on devices.

4.3.2.1 API Calling Flow



1. Check whether the device supports configuring listening host parameters.

   Get the configuration capability of the listening host: `GET /ISAPI/Event/notification/httpHosts/capabilities`. If the node `<HttpHostNotificationCap>` is returned and its value is true, it indicates that the device supports configuring listening host parameters.

2. Configure parameters of the listening host.

   Configure parameters of all listening hosts: `PUT /ISAPI/Event/notification/httpHosts?security=<security>&iv=<iv>`;

   Get parameters of all listening hosts: `GET /ISAPI/Event/notification/httpHosts?security=<security>&iv=<iv>`;

   Configure parameters of a listening host: `PUT /ISAPI/Event/notification/httpHosts/<hostID>?security=<security>&iv=<iv>`;

   Get parameters of a listening host: `GET /ISAPI/Event/notification/httpHosts/<hostID>?security=<security>&iv=<iv>`.

3. Enable the listening service.

   You need to enable the listening service of the listening host.

4. (Optional) Test the listening service.

   The platform applies the command to the device to test whether the listening host is available for the device: `POST /ISAPI/Event/notification/httpHosts/<hostID>/test`.

5. The listening host receives event information from the device.

   When an event occurs, the device creates an connection with the client and uploads alarm information actively. Meanwhile, the listening host receives data from the device. See details in **Event Messages**.

Note: You can also configure the listening parameters such as the time out via URL `/ISAPI/Event/notification/httpHosts/<hostID>/uploadCtrl`.

4.3.2.2 Event Messages

When an event occurs or an alarm is triggered in listening mode, the event/alarm information can be uploaded with binary data (such as pictures) and without binary data.

1. **Without Binary Data:**

The `Content-Type` in headers of the HTTP request sent by the device is usually `application/xml` or `application/json` as follows:

Alarm Message Sent by the Device

```
POST Request_URI HTTP/1.1 <!--Request_URI, related URI: POST /ISAPI/Event/notification/httpHosts-->
Host: data_gateway_ip:port <!--HTTP server's domain name / IP address and port No., related URI: POST /ISAPI/Event/notification/httpHosts-->
Accept-Language: en-us
Date: YourDate
Content-Type: application/xml; <!--Content Type, which is used for the upper layer to distinguish different formats when parsing the message-->
Content-Length: text_length
Connection: keep-alive  <!--maintain the connection between the device and the server for better transmission performance-->

<EventNotificationAlert/>
```

Response by the Listening Host

```
HTTP/1.1 200 OK
Date: YourDate
Connection: close
```

2. **With Binary Data:**

The format of the data sent by the device is HTTP form (multipart/form-data). The `Content-Type` in headers of the HTTP request is usually `multipart/form-data, boundary=<frontier>`, of which boundary is a variable used to divide the HTTP body into multiple units, and each unit has its headers and body. See details in RFC 1867 (Form-based File Upload in HTML). An example is shown below. Please note two hyphens `--` before and after the boundary.

Alarm Message Sent by the Device

```
POST Request_URI HTTP/1.1 <!--Request_URI, , related URI: POST /ISAPI/Event/notification/httpHosts-->
Host: device_ip:port <!--HTTP server's domain name / IP address and port No., related URI: POST /ISAPI/Event/notification/httpHosts-->
Accept-Language: en-us
Date: YourDate
Content-Type: multipart/form-data;boundary=<frontier>
Content-Length: text_length
Connection: keep-alive <!--maintain the connection between the device and the server for better transmission performance-->

--<frontier>
Content-Disposition: form-data; name="Event_Type"
Content-Type: text/xml <!--some event messages are uploaded in JSON format, and the upper layer needs to distinguish the message format accroding to
Content-Type when parsing event messages-->

<EventNotificationAlert/>
--<frontier>
Content-Disposition: form-data; name="Picture_Name"
Content-Length: image_length
Content-Type: image/jpeg

[Picture Data]
--<frontier>--
```

Response by the Listening Host

```
HTTP/1.1 200 OK
Date: YourDate
Connection: close
```

The description of some keywords are as follows:

Keyword	Example	Description
Content-Type	multipart/form-data; boundary=frontier	Content type, multipart/form-data refers to data in form format.
boundary	frontier	Delimiter of the form message. A form message which starts with `--boundary` and ends with `--boundary--`.
Content-Disposition	form-data; name="Picture_Name";	Content description. `form-data` is a piece of form data.
filename	"Picture_Name"	File name. The file refers to the form message.
Content-Length	10	Content length, starting from the next `\r\n` to the next `--boundary`.

4.3.2.3 Exception Handling

Error Codes

statusCode	statusString	subStatusCode	errorCode	Description
6	Invalid Content	eventNotSupport	0x60001024	

5 Device (General)

5.1 Arming and Subscription

5.1.1 Introduction to the Function

With arming and subscription, the client can establish HTTP persistent connection with the device, and continuously receive the event messages from the device.

For ISAPI event arming, the client can receive all types of events by GET method, or receive the subscribed events only by POST method.

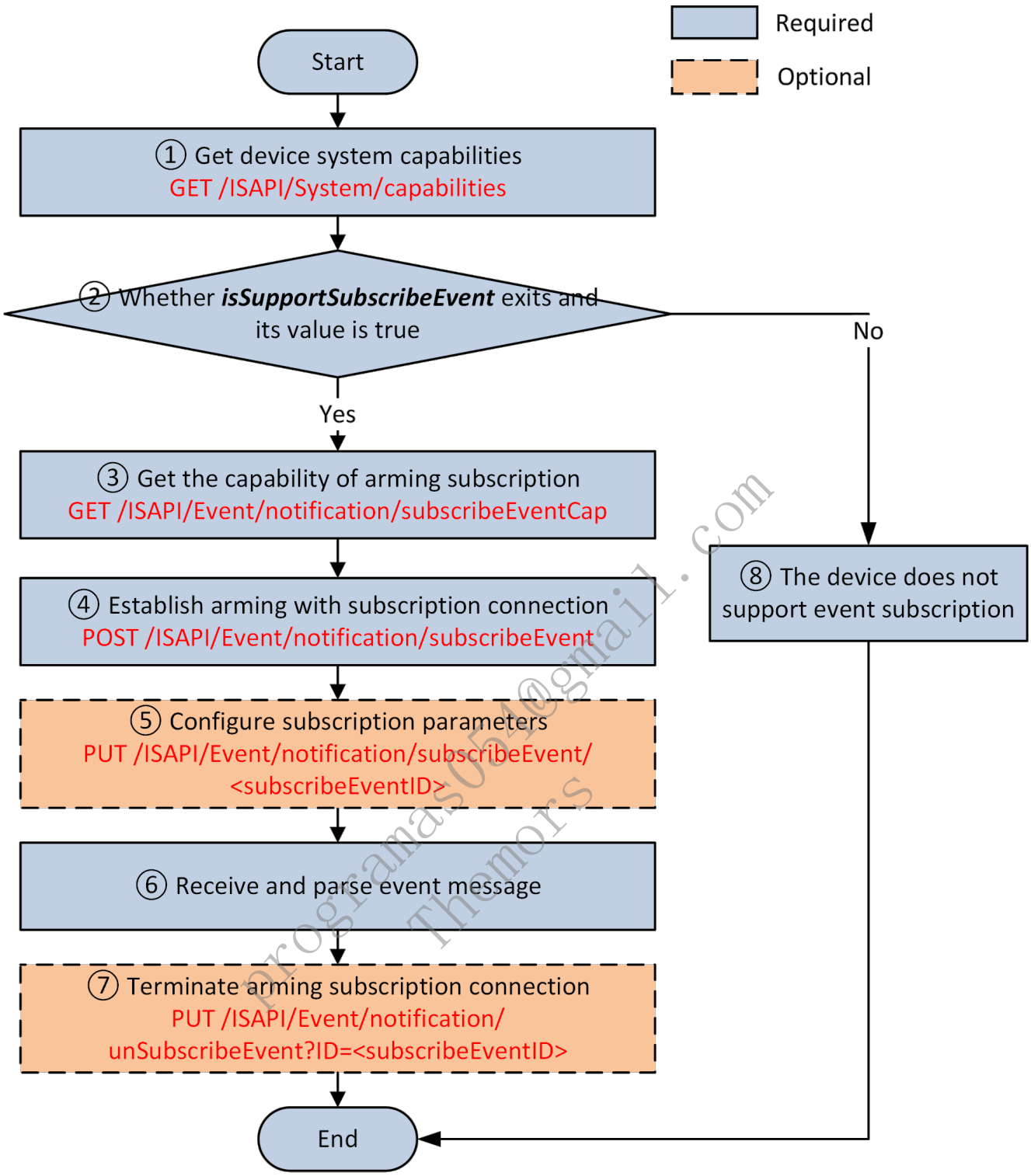
5.1.2 API Calling Flow

5.1.2.1 Without Subscription

1. Establish a connection for arming: `GET /ISAPI/Event/notification/alertStream`. You need to set `Connection: keep-alive` in HTTP Headers.
2. When receiving events sent by the device, the event messages can be separated and parsed by boundary. See "Parsing Event Messages" below for details.
3. Disable the arming connection when you do not need to receive event messages.

5.1.2.2 Subscription

5.1.2.2.1 API Calling Flow



1. Get device system capabilities: `GET /ISAPI/System/capabilities`.
2. Check if event subscription is supported: `isSupportSubscribeEvent` exists and its value is `true`. When `isSupportSubscribeEvent` does not exist or its value is false, the device does not support event subscription.
3. Get the capability of arming with subscription: `GET /ISAPI/Event/notification/subscribeEventCap`.
4. Establish a connection of arming with subscription: `POST /ISAPI/Event/notification/subscribeEvent`. You need to set `Connection: keep-alive` in HTTP Headers.
5. (Optional) Edit parameters of the existing subscription. You need to get the subscription parameters first: `GET /ISAPI/Event/notification/subscribeEvent/<subscribeEventID>`. Then, edit the parameters based on the existing subscription configurations: `PUT /ISAPI/Event/notification/subscribeEvent/<subscribeEventID>`.
6. Receive events sent by the device. The event messages will be separated and parsed by boundary. For parsing description, see **Event Messages Parsing** below.

7. (Optional) Terminate the connection of arming with subscription: `PUT /ISAPI/Event/notification/unSubscribeEvent?ID=<subscribeEventID>`. When communicating with the device via HTTP directly, there is no need to call this API. You can just terminate the connection.

Note:

Three types of data will be transmitted on the arming link: `<SubscribeEventResponse/>`, `<EventNotificationAlert/>`, and picture data. `<SubscribeEventResponse/>` is the data of first form sent by the device after arming established, see the response parameters of URL (`POST /ISAPI/Event/notification/subscribeEvent`) for details; and `<EventNotificationAlert/>` is the event content or heartbeat, you can identify the event type via field `eventType`, e.g., for heartbeat, the value of `eventType` is `heartBeat`.

5.1.2.2.2 Example

```
POST /ISAPI/Event/notification/subscribeEvent HTTP/1.1
Host: device_ip
Accept-Language: zh-cn
Date: YourDate
Content-Type: application/xml;
Content-Length: text_length
Connection: Keep-Alive

<SubscribeEvent/>
HTTP/1.1 401 Unauthorized
Date: Sun, 01 Apr 2018 18:58:53 GMT
Server:
Content-Length: 178
Content-Type: text/html
Connection: keep-alive
Keep-Alive: timeout=10, max=99
WWW-Authenticate: Digest qop="auth", realm="IP Camera(C2183)", nonce="4e5468694e7a42694e7a4d364f4449354d7a6b354d54513d", stale="FALSE"

POST /ISAPI/Event/notification/subscribeEvent HTTP/1.1
Authorization: Digest username="admin",realm="IP
Camera(C2183)",nonce="4e5468694e7a42694e7a4d364f4449354d7a6b354d54513d",uri="/ISAPI/Event/notification/alertStream",cnonce="3d183a245b8729121ae4ca3d41b90f18
",nc=00000001,qop="auth",response="f2e0728991bb031f83df557a8f185178"
Host: device_ip

<SubscribeEvent/>
HTTP/1.1 200 OK
MIME-Version: 1.0
Connection: close
Content-Type: multipart/mixed; boundary=<frontier>

--<frontier>
Content-Type: application/xml; charset="UTF-8" <!--some event messages are uploaded in JSON format, and the upper layer needs to distinguish the message
format accroding to Content-Type when parsing event messages-->
Content-Length: text_length

<SubscribeEventResponse/>
--<frontier>
Content-Type: application/xml; charset="UTF-8"  <!--some event messages are uploaded in JSON format, and the upper layer needs to distinguish the message
format accroding to Content-Type when parsing event messages-->
Content-Length: text_length

<EventNotificationAlert/>
--<frontier>
Content-Disposition: form-data; name="Picture_Name"
Content-Type: image/pjpeg
Content-Length: image_length

[Picture Data]
--<frontier>
```


5.1.2.3 Event Messages Parsing

After the arming connection with the device is established, the data sent by the device is in HTTP form format (multipart/form-data). In an HTTP request, `Content-Type` in Headers is usually `multipart/form-data; boundary=AaB03x`, and boundary is a variable used to divide HTTP Body into multiple units, each being a set of data and has its own Headers and Body. For detailed format description, see RFC 1867 (Form-based File Upload in HTML). An example is shown below. Note the dash `--` before and after boundary.

```
HTTP/1.1 200 OK
Content-Type: multipart/form-data; boundary=AaB03x
Connection: keep-alive
--AaB03x
Content-Disposition: form-data; name="ANPR.xml"; filename="ANPR.xml";
Content-Type: application/xml
Content-Length: 9

<ANPR/>
--AaB03x
Content-Disposition: form-data; name="licensePlatePicture.jpg"; filename="licensePlatePicture.jpg";
Content-Type: image/jpeg
Content-Length: 14

Image Data
--AaB03x--
```

The description of some keywords are as follows:

Keyword	Example	Description
Content-Type	multipart/form-data; boundary=AaB03x	Content type. multipart/form-data means the message is in form format.
boundary	AaB03x	Delimiter of the form message. --boundary is the start of a form. --boundary-- is the end of the whole HTTP form message.
Content-Disposition	form-data; name="ANPR.xml"; filename="ANPR.xml";	Content description.
name	"ANPR.xml"	Form name.
filename	"ANPR.xml"	File name of the form.
Content-Length	9	Content length, starting from the next \r\n to the next --boundary.

5.1.3 Restriction Description

Note that ISAPI arming (with or without subscription) uses HTTP/HTTPS persistent connection. Due to the simplex channel communication mode of HTTP, after establishing the arming connection, the device will send out event messages continuously, while you cannot send any message to the device via the connection.

After the heartbeat time, if you do not receive any message from the device, you should disable the arming connection and try establishing a new one.

5.1.4 Sample Messages

5.1.4.1 Establish Arming Subscription

```
POST /ISAPI/Event/notification/subscribeEvent HTTP/1.1
Authorization: Digest username="admin",realm="IP
Camera(C2183)",nonce="4e5468694e7a42694e7a4d364f4449354d7a6b354d54513d",uri="/ISAPI/Event/notification/alertStream",cnonce="3d183a245b8729121ae4ca3d41b90f18
",nc=00000001,qop="auth",response="f2e0728991bb031f83df557a8f185178"
Host: device_ip

<SubscribeEvent/>
```

5.1.4.2 The Device Responses and Uploads an Event Message

```
HTTP/1.1 200 OK
MIME-Version: 1.0
Connection: close
Content-Type: multipart/mixed; boundary=<frontier>

--<frontier>
Content-Type: application/xml; charset="UTF-8"  <!--Some alarm messages are in JSON format, so when parsing messages, the upper-layer should distinguish them according to the Content-Type field.-->
Content-Length: text_length

<SubscribeEventResponse/>
--<frontier>
Content-Type: application/xml; charset="UTF-8"  <!--Some alarm messages are in JSON format, so when parsing messages, the upper-layer should distinguish them according to the Content-Type field.-->
Content-Length: text_length

<EventNotificationAlert version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
    <ipAddress>10.17.133.46</ipAddress>
    <portNo>80</portNo>
    <protocol>HTTP</protocol>
    <macAddress>44:19:b6:6d:24:85</macAddress>
    <channelID>1</channelID>
    <dateTime>2017-05-04T11:20:02+08:00</dateTime>
    <activePostCount>0</activePostCount>
    <eventType>heartBeat</eventType>
    <eventState>active</eventState>
    <eventDescription>heartBeat</eventDescription>
</EventNotificationAlert>
--<frontier>
Content-Disposition: form-data; name="Picture_Name"
Content-Type: image/pjpeg
Content-Length: image_length
Content-ID: image_ID

[Picture Data]
--<frontier>
```

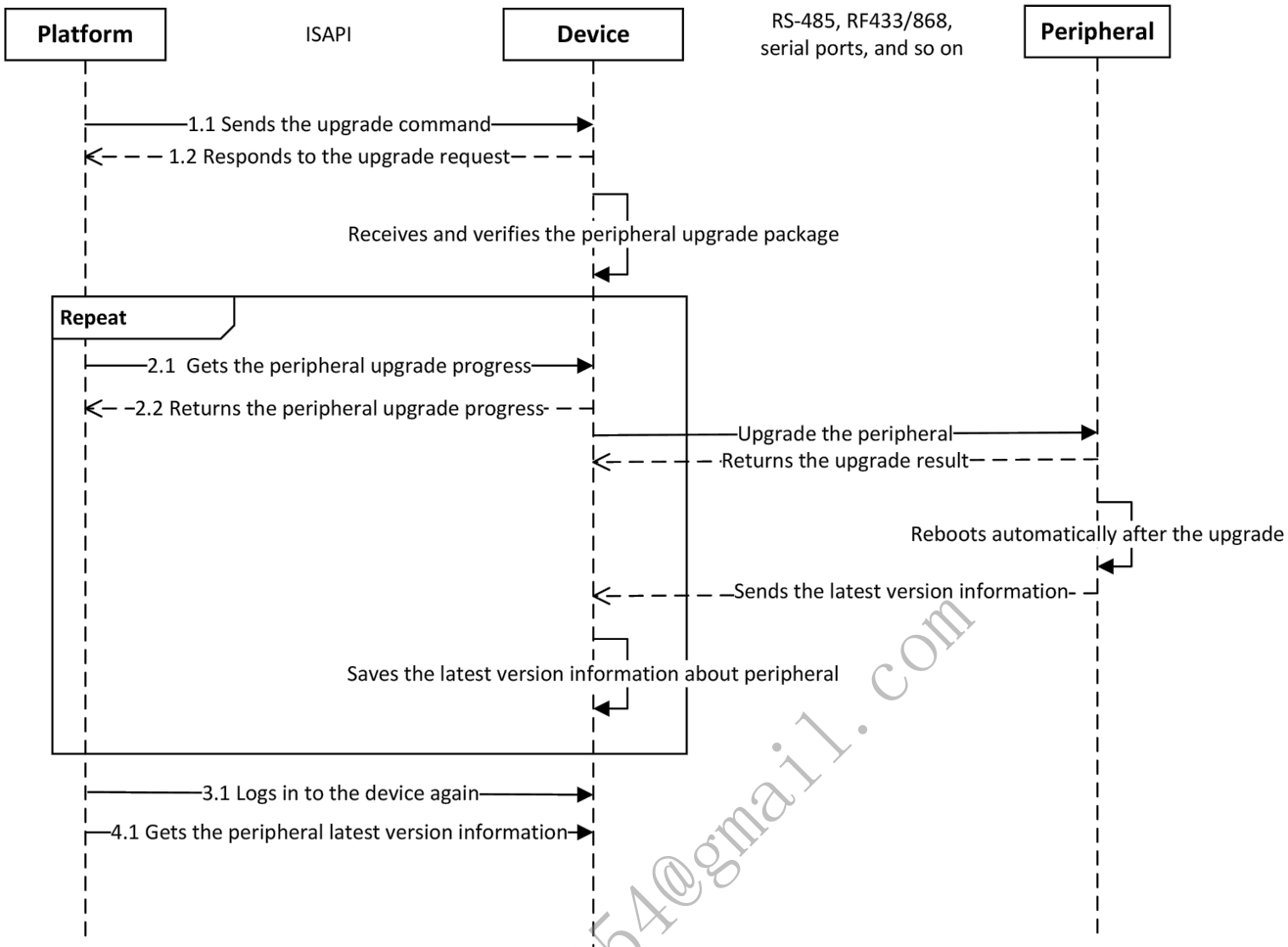
5.2 Device Peripherals Upgrade

5.2.1 Introduction to the Function

The platform or client software or web client under the LAN upgrades device peripherals via ISAPI.

5.2.2 API Calling Flow

The sequence diagram of upgrading device peripherals by the platform is shown below.



1. Get the device system capability `GET /ISAPI/System/capabilities` and check whether the device supports upgrading peripherals. If the field `isSupportAcsUpdate` is returned and its value is true, it indicates that the device supports this function, otherwise, the device does not support this function.
2. Get the capability of upgrading the peripherals module `GET /ISAPI/System/AcsUpdate/capabilities`, and get the types and IDs of peripherals that support upgrading.
3. The platform sends the upgrade command `POST /ISAPI/System/updateFirmware?type=<type>&moduleAddress=<moduleAddress>&id=<indexID>`. In the URL `type` refers to the peripheral type, `moduleAddress` refers to the peripheral module address, and `indexID` refers to the ID of peripheral to be upgraded. The platform will apply the upgrade peripheral package to the device.
4. Get the peripheral upgrade progress `GET /ISAPI/System/upgradeStatus?type=<Type>`.
5. Log in to the device again.
6. Get the peripheral latest version information.

5.3 Device Time Sync

5.3.1 Introduction to the Function

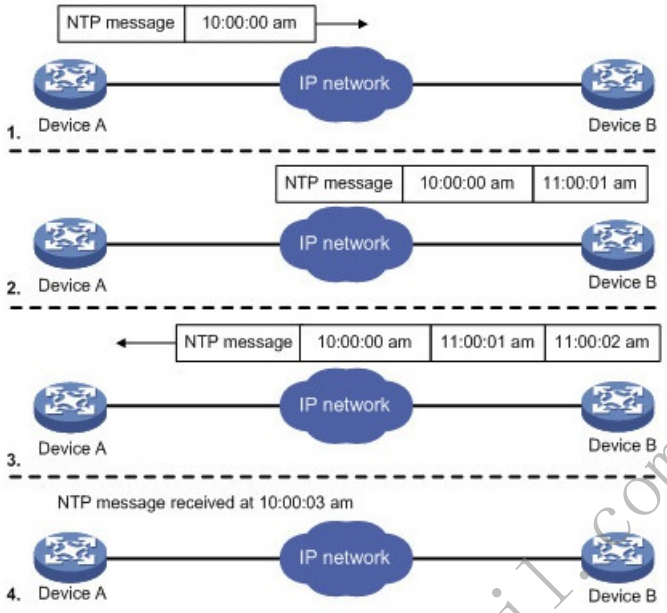
```
Time sync is a method to synchronize the time of all devices connecting to the NTP server, so that all devices can share the same clock time for
providing related functions based on time. Supported time sync types: NTP time sync, manual sync, satellite time sync, platform time synchronization,
GB28181 time sync, etc. The following describes the method of NTP time sync.
```

5.3.1.1 NTP Time Sync

The local system of running NTP can receive sync from other clock sources (self as client), other clocks can sync from the local system (self as server), and sync with other devices.

The basic working principle of NTP is shown in the picture. Device A and Device B are connected via the network, and their systems follow their own independent system time. To auto sync their time, you can set device time auto sync via NTP. For example:

```
Before time sync between Device A and Device B, the time of Device A is 10:00:00 am, and that of Device B is 11:00:00 am. Device B is set as the server of
NTP server, so that the time of Device A should be synchronized with that of Device B. The time of NTP message transmitted between Device A and Device B is
1 second.
```



The working process of system clock synchronization is as follows:

Device A sends an NTP message to Device B with a timestamp of 10:00:00 am (T1) that is when it leaves Device A.

When the NTP message reaches Device B. Device B will add its own timestamp, which is 11:00:01 am (T2).

Then the NTP message leaves Device B with Device B's timestamp, which is 11:00:02 am (T3).

Device A receives the response message, and the local time of Device A is 10:00:03 am (T4).

Above all, Device A can calculate two important parameters:

Round-trip delay of NTP message: Delay = (T4-T1) - (T3-T2) = 2 seconds.

Time difference between Device A and Device B: offset = ((T2-T1)+(T3-T4))/2=1 h.

Device A can sync its own time with that of Device B according to calculation results.

5.3.2 API Calling Flow

5.3.2.1 Time Sync Configuration

1. Get the Capability of Device Time synchronization Management

You can call this API to get the time sync types currently supported by the device, such as NTP time sync, manual time sync, satellite time sync, Connect platform time sync, and GB28181 time sync.

Get the capability: `GET /ISAPI/System/time/capabilities`.

2. Set device time synchronization management parameters

You can configure the time synchronization mode as follows.

NTP time synchronization: See 4.2.2 NTP Time Sync (Client).

Manual time synchronization: Set the value of `timeMode` to `manual`, and set the device local time in nodes `localTime`、`timeZone`.

Satellite time synchronization: Set the value of `timeMode` to `satellite`, and set the device local time in nodes `satelliteInterval`.

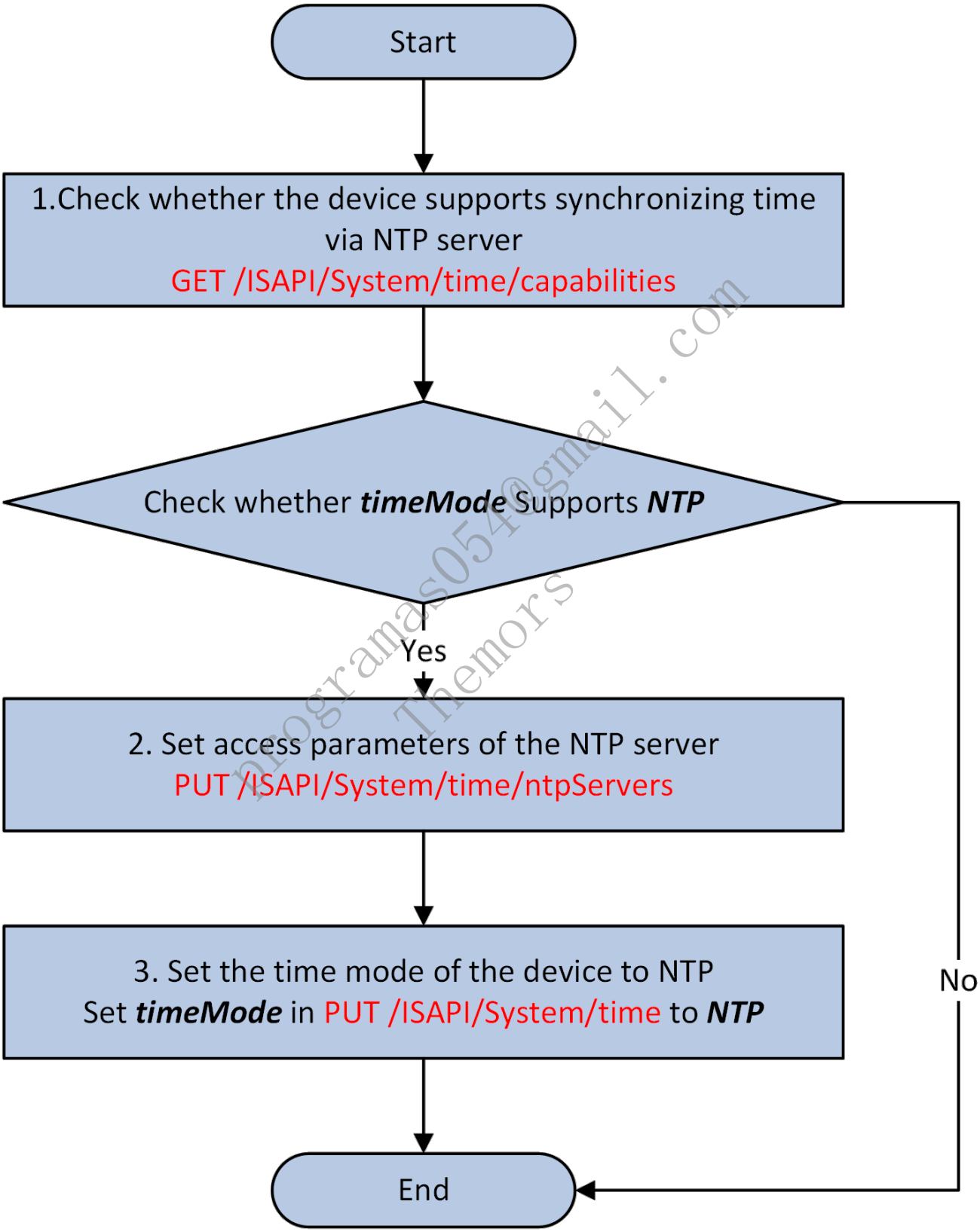
Platform time synchronization: Set the value of `timeMode` to `platform`.

GB28181 time synchronization: Set the value of `timeMode` to `GB28181`. If there are two GB28181 platforms, you can select platform No. via `platformNo`. it is the only ID, which is configured via `platformNo` in `GB28181List`, related URI: `/ISAPI/System/Network/SIP/<SIPServerID>`

Get device time synchronization management parameters: `GET /ISAPI/System/time`; Set device time synchronization management parameters: `PUT /ISAPI/System/time`.

5.3.2.2 NTP Time Sync (Client)

The local system running the NTP server can receive sync information from other clock sources (self as client), sync other clocks (self as server) as clock sources, and sync with other devices. Calling flow (self as client):



1. Check whether the device supports synchronizing time via NTP server Get the capability of the device: `GET /ISAPI/System/time/capabilities`; and check whether `timeMode` supports `NTP`.

2. Set access parameters of the NTP server

Supports accessing the NTP server by IP address to synchronize the device time.

Get the access parameter capability of the NTP server: `GET /ISAPI/System/time/ntpServers/capabilities`

Set access parameters of the NTP server: `PUT /ISAPI/System/time/ntpServers`

Get access parameters of the NTP server: `GET /ISAPI/System/time/ntpServers`

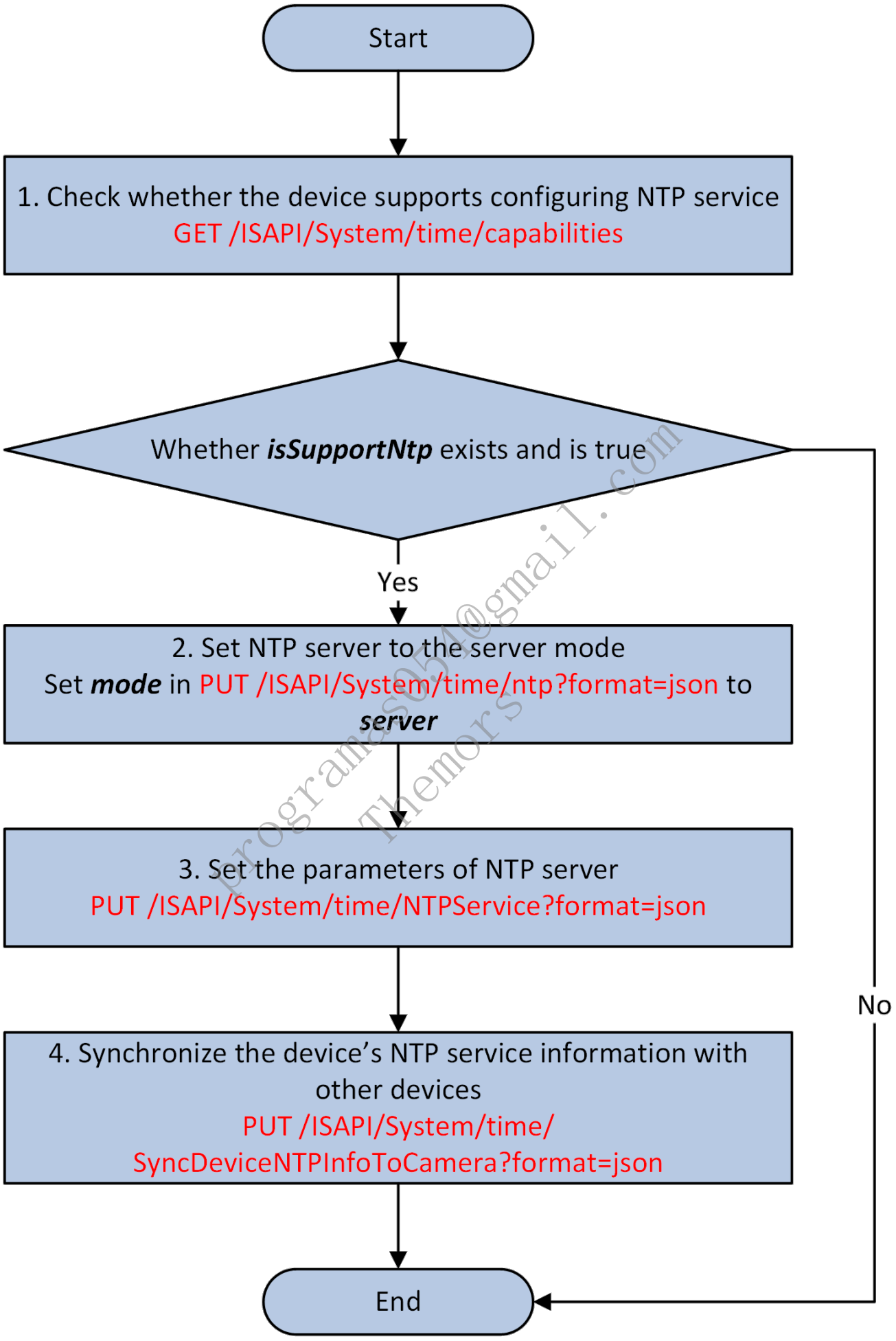
3. Set the time mode of the device to NTP

Supports setting the value of `timeMode` to `NTP`.

Get device time synchronization management parameters: `GET /ISAPI/System/time` Set device time synchronization management parameters: `PUT /ISAPI/System/time`

5.3.2.2 NTP Time Sync (Server Mode)

The local system running the NTP server can receive sync information from other clock sources (self as client), sync other clocks (self as server) as clock sources, and sync with other devices. Calling flow (self as server):



1. Check whether the device supports configuring NTP service Get the capability of device time synchronization management: `GET /ISAPI/System/time/capabilities`; If `isSupportNtp` is returned, it indicates that the device supports time synchronization management.

2. Set NTP server to the server mode

Supports setting the value of `mode` to `server`.

Get the capability of server mode: `GET /ISAPI/System/time/ntp/capabilities?format=json`

Set NTP to server mode: `PUT /ISAPI/System/time/ntp?format=json`

Get parameters of NTP server mode: `GET /ISAPI/System/time/ntp?format=json`

3. Set the parameters of NTP server

Supports setting the IP address of the NTP server.

Get the capability of NTP server: `GET /ISAPI/System/time/NTPService/capabilitis?format=json`

Set the NTP server parameters: `PUT /ISAPI/System/time/NTPService?format=json`

Get the parameters of the NTP server: `GET /ISAPI/System/time/NTPService?format=json`

4. Synchronize the device's NTP service information with other devices

Supports synchronizing the time information to the camera.

Get the capability set of synchronizing device's NTP service information with the camera: `GET /ISAPI/System/time/SyncDeviceNTPInfoToCamera/capabilities?format=json`

Synchronize device's NTP service information with the camera: `PUT /ISAPI/System/time/SyncDeviceNTPInfoToCamera?format=json`

Get the progress of synchronizing device's NTP service information with the camera: `GET /ISAPI/System/time/SyncDeviceNTPInfoToCamera/Progress?format=json`

Search for the results of synchronizing device's NTP service information with the camera: `POST /ISAPI/System/time/SyncDeviceNTPInfoToCamera/SearchResult?format=json`

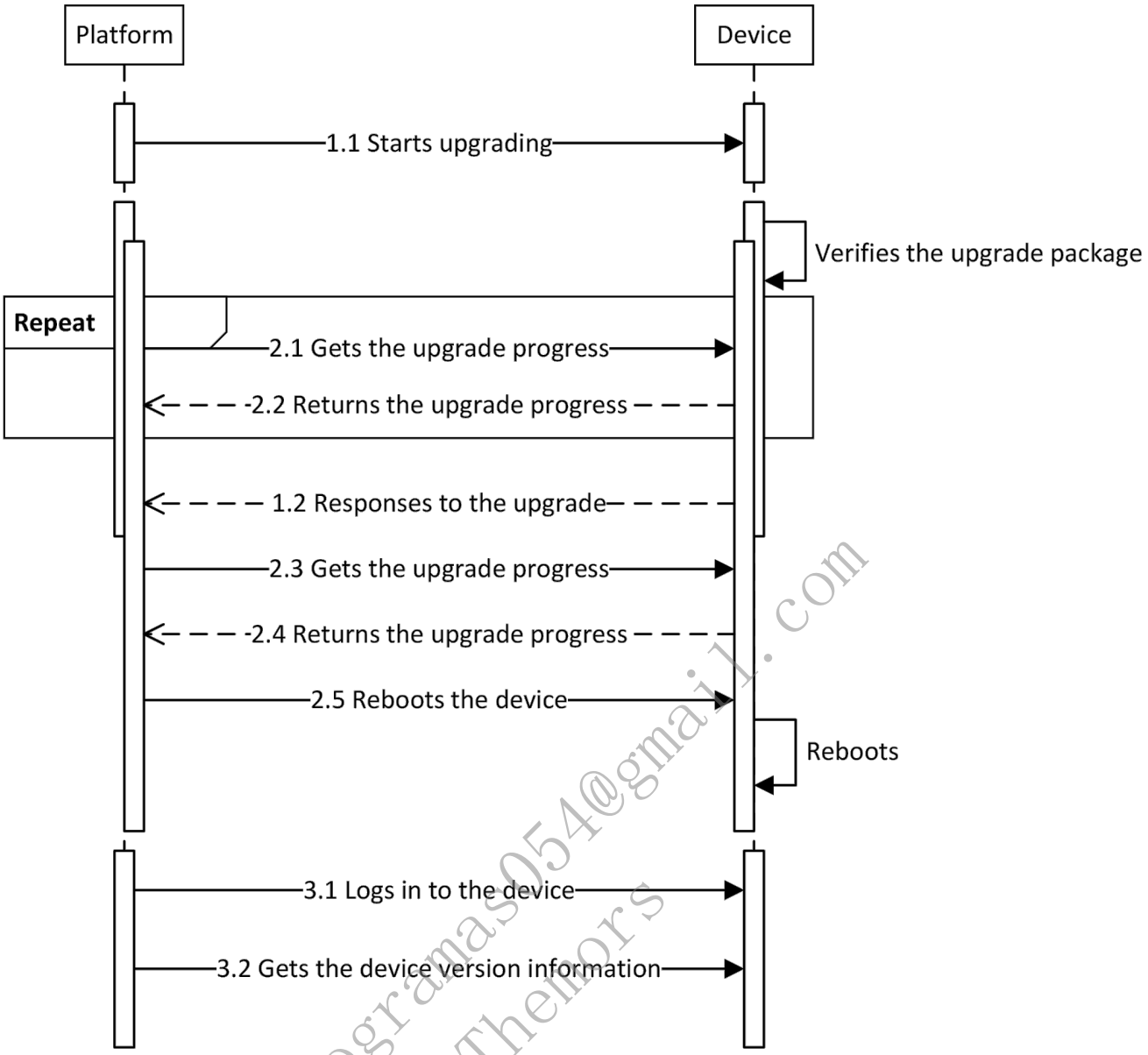
5.4 Device Upgrade

5.4.1 Introduction to the Function

The platform or client software or web client under the LAN upgrades devices via ISAPI.

5.4.2 API Calling Flow

The sequence diagram of upgrading devices by the platform is shown below.



1. **Upgrade devices.**

   Upgrade the device firmware: `POST /ISAPI/System/updateFirmware`.

2. **Get the device upgrade progress.**

   Get the device upgrade progress: `GET /ISAPI/System/upgradeStatus`.

3. **Reboot devices.**

   Reboot devices: `PUT /ISAPI/System/reboot`.

5.5 Management of Devices Access Serial Port

5.5.1 Introduction to the Function

Information management of device accessed the serial are as follows: 1. Configure manufacturer, type, and model information of the specific serial port access device. 2. Search for the device type or model supported by the specific serial port.

5.5.2 API Calling Flow

1. **Check whether the device supports information management of devices access the serial port**: `GET /ISAPI/System/Serial/capabilities`; If `<isSupportDeviceInfo>` is returned, it indicates that the device supports information configuration of devices access the serial port.

3. Set the information of devices access the serial port:

Get the capability of device information parameters of a single serial port: `GET /ISAPI/System/Serial/ports/<portID>/deviceInfo?format=json`;

Get device information parameters access single serial port: `GET /ISAPI/System/Serial/ports/<portID>/deviceInfo?format=json`;

Set device information parameter of single serial port: `PUT /ISAPI/System/Serial/ports/<portID>/deviceInfo?format=json`;

4. **Check whether the device supports linking information of devices access the serial port**: `GET /ISAPI/System/Serial/capabilities`; If `<isSupportSearchDeviceInfoRelations>` is returned, it indicates that the device supports searching for linked information od devices access the serial port.

5. **Search for linked information of devices access the serial port**:

Get the capability of searching for linked parameters of information of devices access a single serial port: `GET /ISAPI/System/Serial/ports/<portID>/searchDeviceInfoRelations/capabilities?format=json`;

Search for linked parameters of information of devices access a single serial port:`POST /ISAPI/System/Serial/ports/<portID>/searchDeviceInfoRelations?format=json`;

5.6 Mutually Exclusive Functions

5.6.1 Introduction to the Function

Some functions are mutually exclusive due to the device performance (for example, function A and function B cannot run at the same time, i.e, only one of them is allowed at one time).

5.6.2 API Calling Flow

The following three APIs are available for the integration of mutually exclusive functions:

1. Get the information of mutually exclusive functions: `GET /ISAPI/System/mutexFunction/capabilities?format=json`. Call this URL to get the list of existing mutually exclusive functions supported by the device. Note: NVR devices only support setting exlusive function "perimeter" (perimeter protection), and do not support "linedetection" (line crossing detection), "fielddetection" (intrusion detection), "regionEntrance" (region entrance), or "regionExiting" (region exiting).

2. Search for the functions that are mutually exclusive with a specified function: `POST /ISAPI/System/mutexFunction?format=json`. Based on the list of mutually exclusive functions returned by `GET /ISAPI/System/mutexFunction/capabilities?format=json`, you can search for the mutual exclusion status of a specified function and see whether to change the settings and disbale the mutually exclusive function.

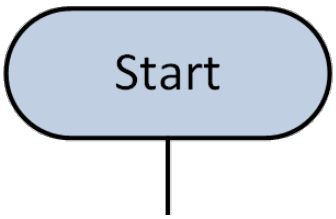
3. Get the mutual exclusion information when device function exception occurs: `GET /ISAPI/System/mutexFunctionErrorMsg`. After getting the error code, you can call this API to get the current mutually exclusive functions.

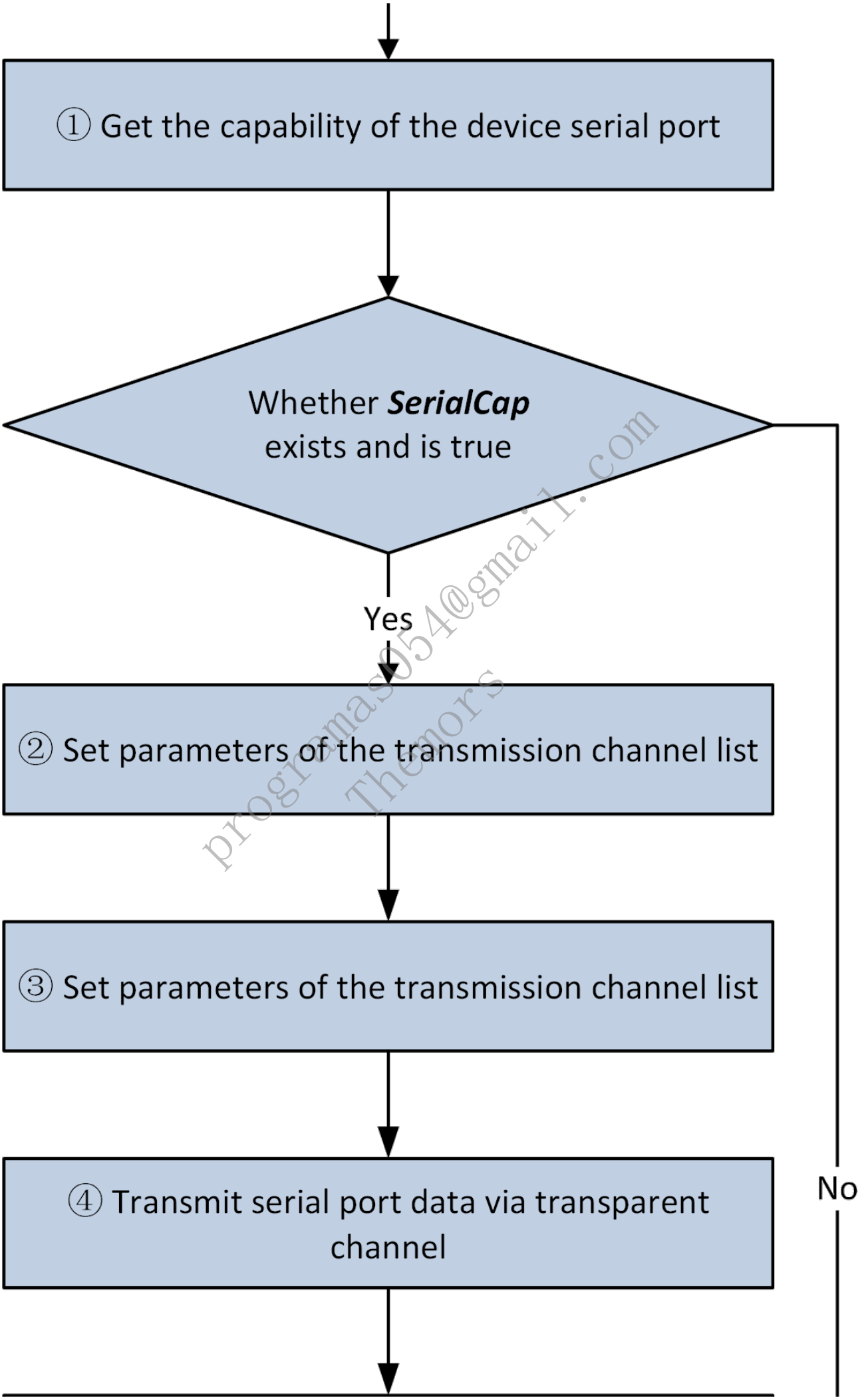
5.7 Serial Port Data Transparent Transmission

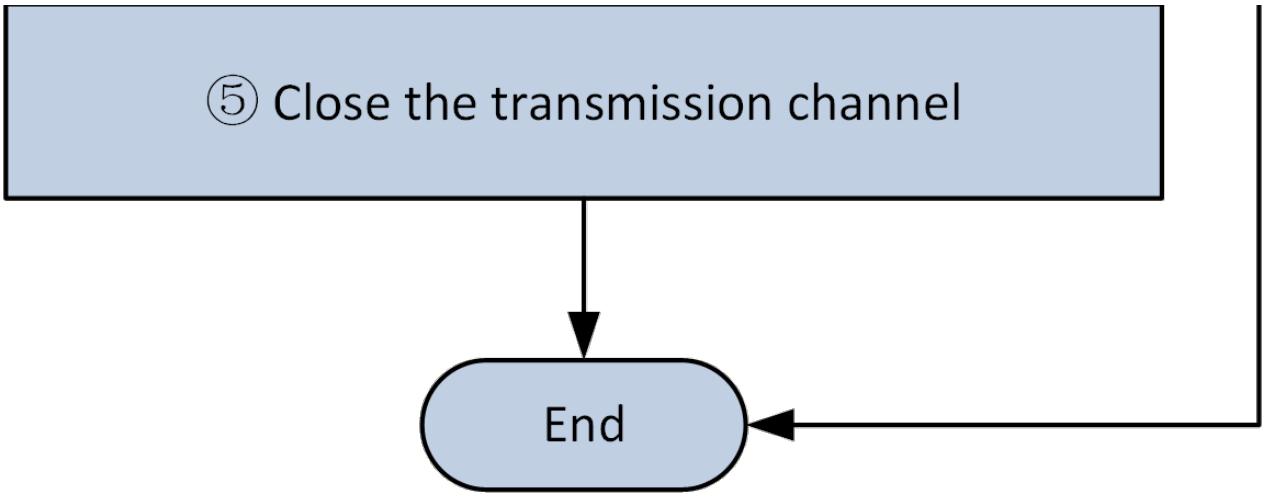
5.7.1 Introduction to the Function

RS485, RS422 and RS232 serial ports external to the device are used as transparent channels to transmit serial port data.

5.7.2 API Calling Flow







1. **Check whether the device supports serial port data transmission:**

Get the capability of the device serial port: `GET /ISAPI/System/capabilities`. If `SerialCap` is returned and the value is true, it indicates that the device supports the functions of the serial port.

2. **Set parameters of the transmission channel list**:

Get parameters of the specific transmission channel: `GET /ISAPI/System/Serial/ports/<portID>/Transparent/channels/<channelID>`;

Configure parameters of the specific transmission channel: `GET /ISAPI/System/Serial/ports/<portID>/Transparent/channels/<channelID>`.

3. **Open the transmission channel**: `PUT /ISAPI/System/Serial/ports/<portID>/Transparent/channels/<channelID>/open`;

4. **Transmit serial port data via transparent channel**:

Receive data uploaded by device serial port through transmission channel: `GET /ISAPI/System/Serial/ports/<portID>/Transparent/channels/<channelID>/transData`;

Send data to device serial port through transmission channel: `PUT /ISAPI/System/Serial/ports/<portID>/Transparent/channels/<channelID>/transData`;

5. **Close the transmission channel**: `PUT /ISAPI/System/Serial/ports/<portID>/Transparent/channels/<channelID>/close`;

5.8 Serial Port Parameter Configuration

5.8.1 Introduction to the Function

Serial port parameter configuration.

5.8.2 API Calling Flow

1. **Check whether the device supports configuring parameters of the serial port:**

Get the capability of device serial port: `GET /ISAPI/System/capabilities;如果报文返回节点`. If is returned and its value is true, it indicates that the device supports functions of serial port.

3. **Get parameters of all serial ports**:

Get the capability of all serial ports: `GET /ISAPI/System/Serial/capabilities`;

Get control parameters of all serial ports: `GET /ISAPI/System/Serial/ports?permissionController=<indexID>`;

4. **Set control parameters of single serial port**:

Get control parameters of single serial port: `GET /ISAPI/System/Serial/ports/<portID>?permissionController=<indexID>`;

Configure control parameters of single serial port: `PUT /ISAPI/System/Serial/ports/<portID>?permissionController=<indexID>`;

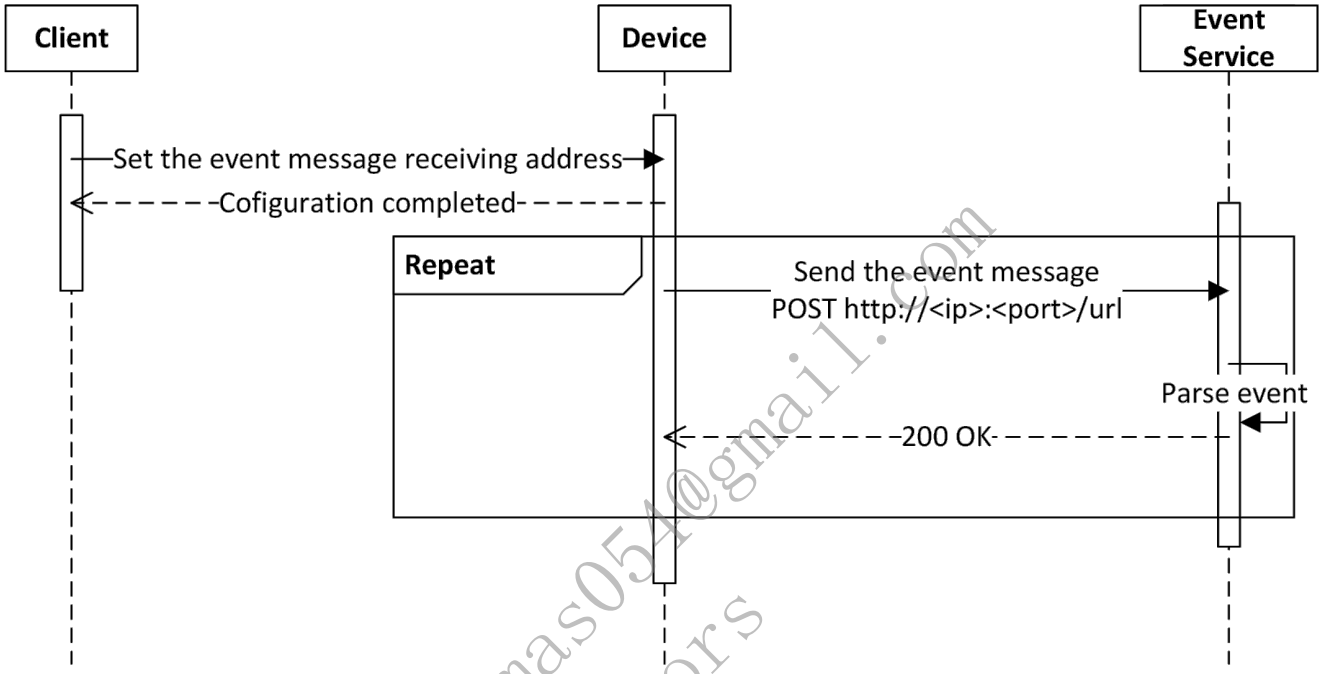
5. **Get the status of single serial port**:`GET /ISAPI/System/Serial/ports/<portID>/status`

6 Network Settings

6.1 Listening Service

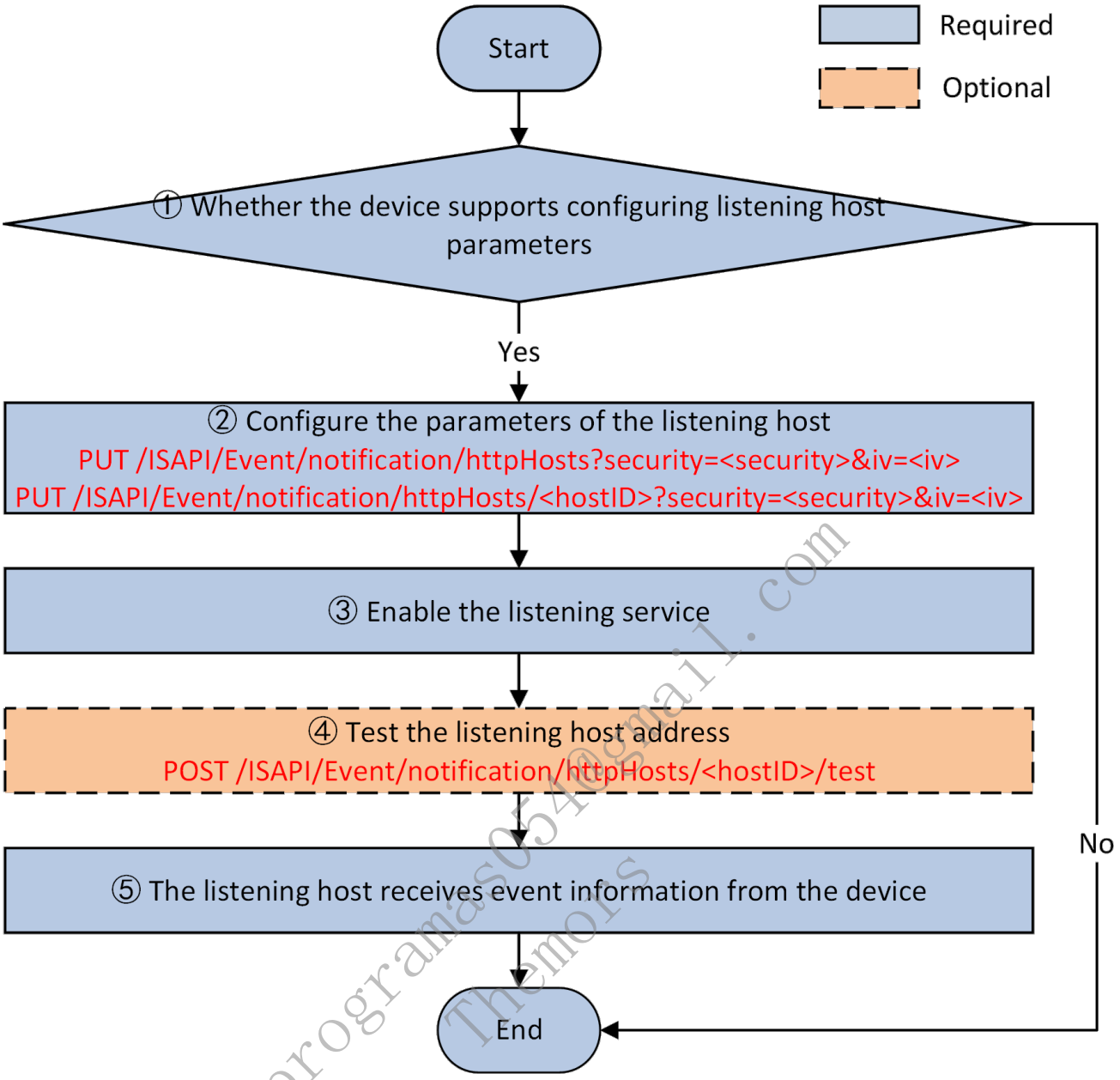
6.1.1 Introduction to the Function

When an event occurs, the device creates connection with the client and uploads alarm information. Meanwhile, the listening host receives data from the device. The IP address and the port No. of the listening host should be configured for the device. The HTTP listening service supports subscribing to specific events when adding or editing the listening host. Only the specified events will be uploaded by the device.(currently not available for the device)



6.1.2 API Calling Flow

6.1.2.1 Listening Service



1. **Check whether the device supports configuring listening host parameters:**

   Get the configuration capability of the listening host: `GET /ISAPI/Event/notification/httpHosts/capabilities`. If the node `<HttpHostNotificationCap>` exists in the returned message and its value is true, it indicates that the device supports configuring listening host parameters.

2. **Configure the parameters of the listening host:**

   Configure the parameters of all listening hosts: `PUT /ISAPI/Event/notification/httpHosts?security=<security>&iv=<iv>`;

   Get the parameters of all listening hosts: `GET /ISAPI/Event/notification/httpHosts?security=<security>&iv=<iv>`;

   Configure the parameters of a listening host: `PUT /ISAPI/Event/notification/httpHosts/<hostID>?security=<security>&iv=<iv>`;

   Get the parameters of a listening host: `GET /ISAPI/Event/notification/httpHosts/<hostID>?security=<security>&iv=<iv>`;

3. **Enable the listening service:**

   The user needs to enable the listening service of the listening host.

4. **(Optional) Test the listening service:**

   The platform applies the command to the device to test whether the listening host is available for the device: `POST /ISAPI/Event/notification/httpHosts/<hostID>/test`.

5. **The listening host receives event information from the device:**

   When an event occurs, the device creates connection with the client and uploads alarm information. Meanwhile, the listening host receives data from the device. See details in **Event Message Grammar**.

Remark: You can also configure the listening parameters such as the timeout.

6.1.2.2 Event Message Grammar

When an event occurs or an alarm is triggered, the event/alarm information can be with binary data (such as pictures) and without binary data.

1. **Without Binary Data:**

The `Content-Type` in the Headers of the HTTP request sent by the device is usually `application/xml` or `application/json` as follows:

Alarm Message Sent by the Device

```
POST Request_URI HTTP/1.1 <!--/Request_URI, related URI: POST /ISAPI/Event/notification/httpHosts-->
Host: data_gateway_ip:port <!--Host: HTTP server's domain name / IP address and port No., related URI: POST /ISAPI/Event/notification/httpHosts-->
Accept-Language: zh-cn
Date: YourDate
Content-Type: application/xml; <!--content type, which is used for the upper layer to distinguish different formats when parsing the message-->
Content-Length: text_length
Connection: keep-alive  <!--maintain the connection between the device and the server for better transmission performance-->

<EventNotificationAlert/>
```

Response by the Listening Host

```
HTTP/1.1 200 OK
Date: YourDate
Connection: close
```

2. **With Binary Data**

The format of the data sent by the device is HTTP form (multipart/form-data). The `Content-Type` in the Headers of the HTTP request sent by the device is usually `multipart/form-data, boundary=<frontier>`: boundary is a variable, which is used to divide the HTTP Body into multiple units and each unit has its Headers and Body. See details in RFC 1867 (Form-based File Upload in HTML). Please note the `--` before and after the boundary.

Alarm Message Sent by the Device

```
POST Request_URI HTTP/1.1 <!--Request_URI, related URI: POST /ISAPI/Event/notification/httpHosts-->
Host: device_ip:port <!--Host: HTTP server's domain name / IP address and port No., related URI: POST /ISAPI/Event/notification/httpHosts-->
Accept-Language: zh-cn
Date: YourDate
Content-Type: multipart/form-data;boundary=<frontier>
Content-Length: text_length
Connection: keep-alive <!--maintain the connection between the device and the server for better transmission performance-->

--<frontier>
Content-Disposition: form-data; name="Event_Type"
Content-Type: text/xml <!--maintain the connection between the device and the server for better transmission performance-->

<EventNotificationAlert/>
--<frontier>
Content-Disposition: form-data; name="Picture_Name"
Content-Length: image_length
Content-Type: image/jpeg

[picture data]
--<frontier>--
```

Response by the Listening Host

```
HTTP/1.1 200 OK
Date: YourDate
Connection: close
```

Here are the descriptions of the main keywords.

keyword	example	description
Content-Type	multipart/form-data; boundary=frontier	Content type, multipart/form-data refers to data in form format.
boundary	frontier	Separator of the form message. A form message starts with --boundary and ends with --boundary--.
Content-Disposition	form-data; name="Picture_Name";	Content description. form-data refers to data in the form format.
filename	"Picture_Name"	File name. The file refers to the form message.
Content-Length	10	Content length. The length of the content which starts from \r\n to the next --boundary.

6.1.3 Exception Handling

6.1.3.1 Error Codes

statusCode	statusString	subStatusCode	errorCode	errorMsg	Description	Remarks
6	Invalid Content	eventNotSupport	0x60001024		Event subscription is not supported.	

7 Access Control (General)

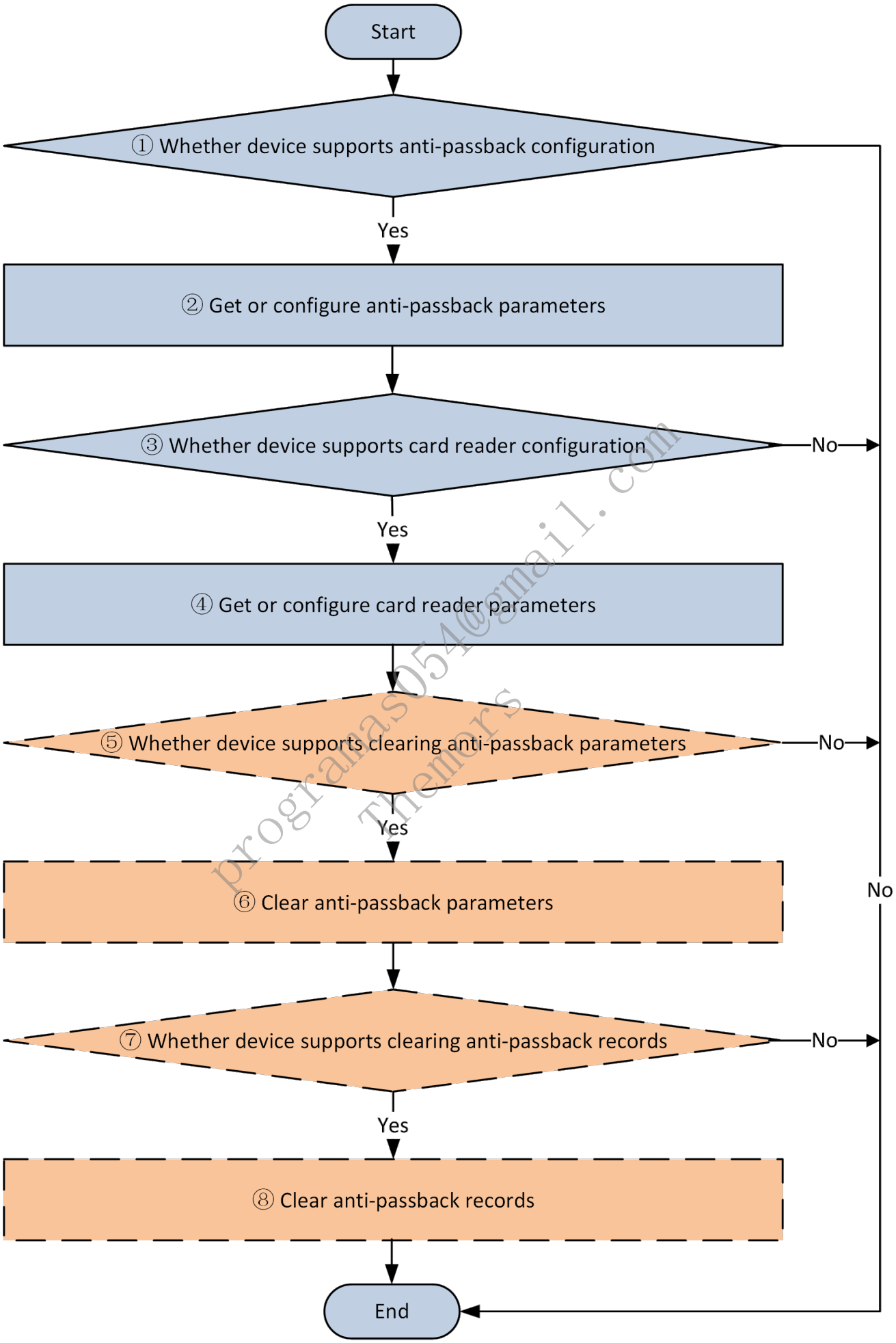
7.1 . Configure Parameters of Stand-Alone Anti-Passback

7.1.1 Introduction to the Function

Anti-passback function allows person to pass entrance and exit by specific route. For areas with multiple entrance/exit, person need to authenticate and pass specific doors. Only one time of authentication and passing is allowed for each door, that is the person need to follow the specific order to pass the doors. Stand-alone anti-passback is designed to minimize the misuse or fraudulent use of access credentials such as passing back card to an unauthorized person, or tailed access. It is applicable to exhibitions, scenic spots, or metro entrances where one card one person is required.

7.1.2 API Calling Flow

Calling Flow:



ISAPI Protocol Calling Flow:

1. 1. Get the capability of access control: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportAntiSneakCfg is returned and its value is "true", it indicates that the device supports configuring

parameters of stand-alone anti-passback.

2. Get the parameters of anti-passback configuration: `GET /ISAPI/AccessControl/AntiSneakCfg?format=json`; Set the anti-passing back parameters: `PUT /ISAPI/AccessControl/AntiSneakCfg?format=json`; enable the anti-passback function and the first card reader (first entrance), see details in `GET /ISAPI/AccessControl/AntiSneakCfg/capabilities?format=json`.
3. Get the capability of access control: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportCardReaderAntiSneakCfg is returned and its value is "true", it indicates that the device supports configuring parameters of card readers.
4. Get the anti-passing back configuration parameters of a specified card reader: `GET /ISAPI/AccessControl/CardReaderAntiSneakCfg/<cardReaderID>?format=json`; Set anti-passing back parameters of a card reader: `PUT /ISAPI/AccessControl/CardReaderAntiSneakCfg/<cardReaderID>?format=json`; the node cardReaderID refers to the card reader No. The person's passing route will follow the card reader No. in the API. Note: the anti-passback route should be closed-loop. Improper configuration will affect normal door opening. For example: card reader 1 -> card reader 2 -> card reader 3. The card reader 1 should be set after card reader 3, or authentication in card reader 1 after one loop will fail. Get the configuration capability of anti-passing back parameters of card readers: `GET /ISAPI/AccessControl/CardReaderAntiSneakCfg/capabilities?format=json`.
5. Get the capability of access control: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportClearAntiSneakCfg is returned and its value is "true", it indicates that the device supports clearing parameters of anti-passback.
6. Clear anti-passing back parameters: `PUT /ISAPI/AccessControl/ClearAntiSneakCfg?format=json`; Get the capability of clearing anti-passback parameters: `GET /ISAPI/AccessControl/ClearAntiSneakCfg/capabilities?format=json`, set the value of antiSneak to false to disable the anti-passback function.
7. Get the capability of access control: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportClearAntiSneak is returned and its value is "true", it indicates that the device supports clearing records of anti-passback.
8. Clear anti-passback records in the device: `PUT /ISAPI/AccessControl/ClearAntiSneak?format=json`; It supports clearing anti-passback records by person ID. Get the capability of clearing anti-passback records: `GET /ISAPI/AccessControl/ClearAntiSneak/capabilities?format=json`.

Note: clear the historic anti-passback parameters before configuring new parameters.

7.2 Arming Information

7.2.1 Introduction to the Function

The device supports getting the arming information, such as the armed device IP and port, arming type, and protocol type.

7.2.2 API Calling Flow

1. Get the functional capability of access control: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportDeployInfo is returned and its value is "true", it indicates that the device supports getting arming information.
2. Get arming information capability: `GET /ISAPI/AccessControl/DeployInfo/capabilities`.
3. Get arming information: `GET /ISAPI/AccessControl/DeployInfo`.

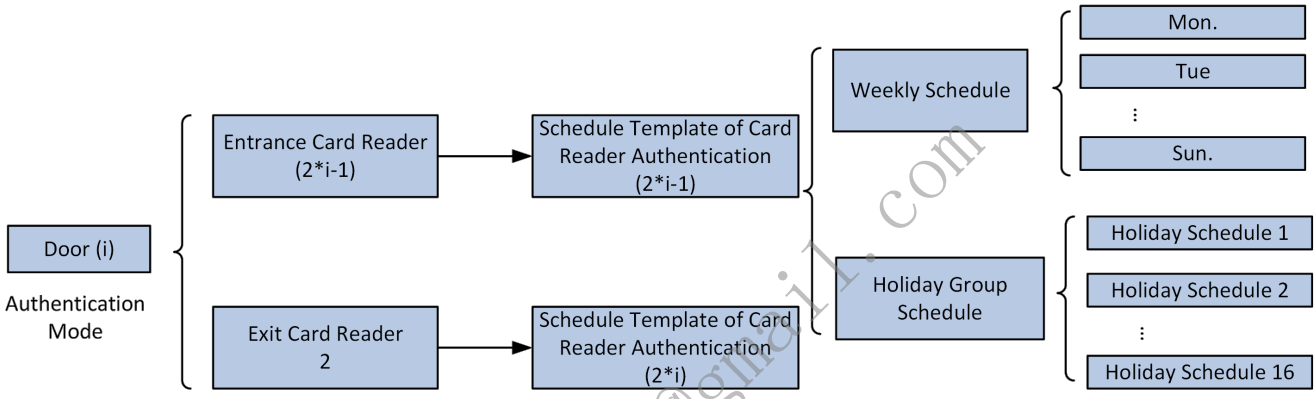
7.3 Authentication Schedule Management

7.3.1 Introduction to the Function

By calling the following APIs, the entrance&exit time schedule and authentication mode can be applied to the card reader. An access control terminal is controlled by two card readers. For example: access control terminal 1 is controlled by entrance card reader 1 and exit card reader 2; access control terminal 2 is controlled by entrance card reader 3 and exit card reader 4, etc. Card reader n is corresponding to schedule template n.

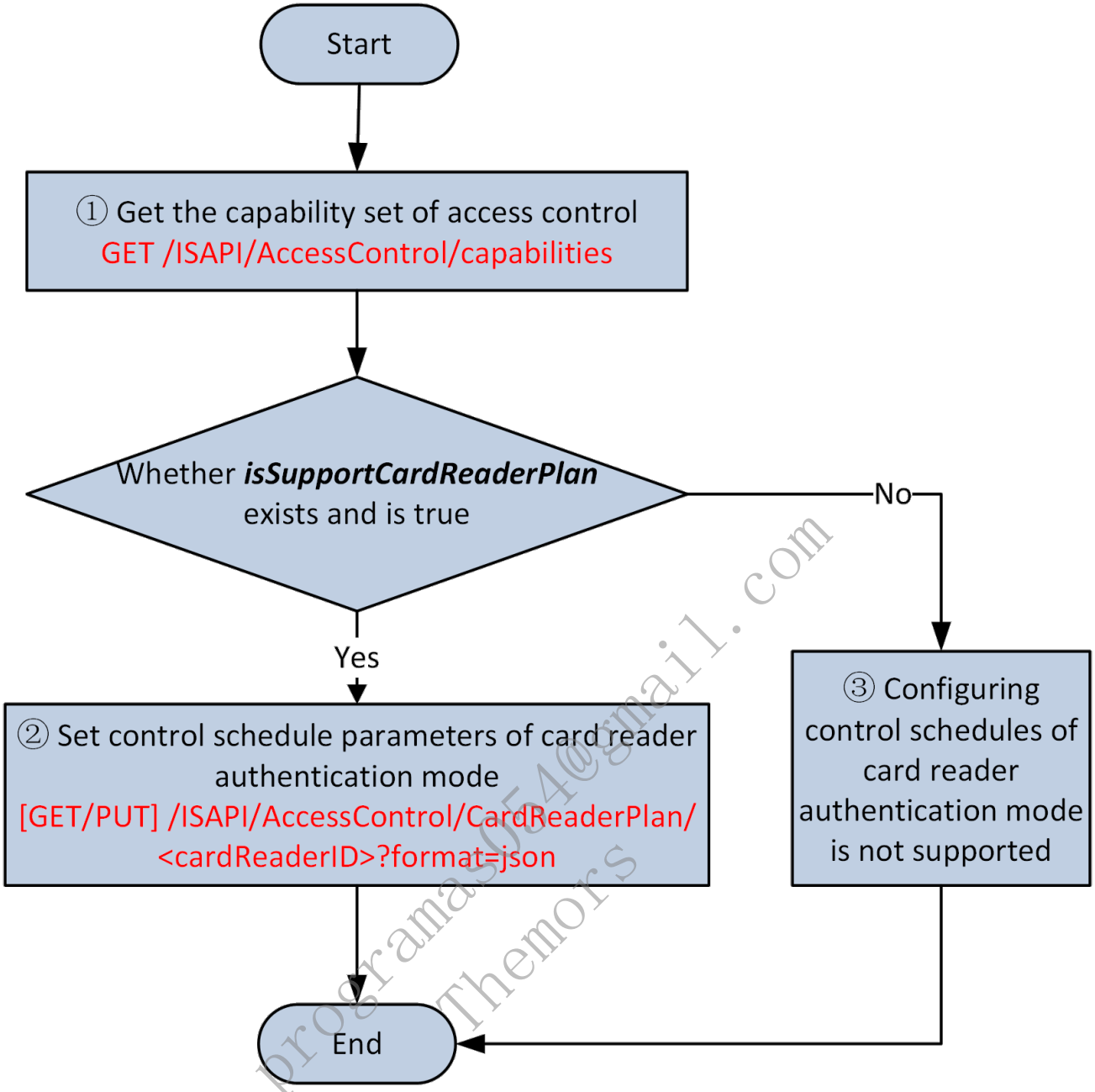
1 weekly schedule and 4 holiday groups can be added in each schedule template. The priority of holiday schedule is higher than that of weekly schedule. A weekly schedule can be configured by date of a week and 8 different time periods of a day. 16 holiday schedules can be added to a holiday group schedule. Each holiday schedule has its start and end date, and the time period is same in the range (8 time periods can be added). The access control can follow the schedule template to manage the time of person's permissions.

For Person Management of Person and Credential Management, the priority of the authentication method for person is higher than that of the authentication schedule. If the authentication method (the node is userVerifyMode) is applied to a person, the person can access according to the authentication method for person.



7.3.2 API Calling Flow

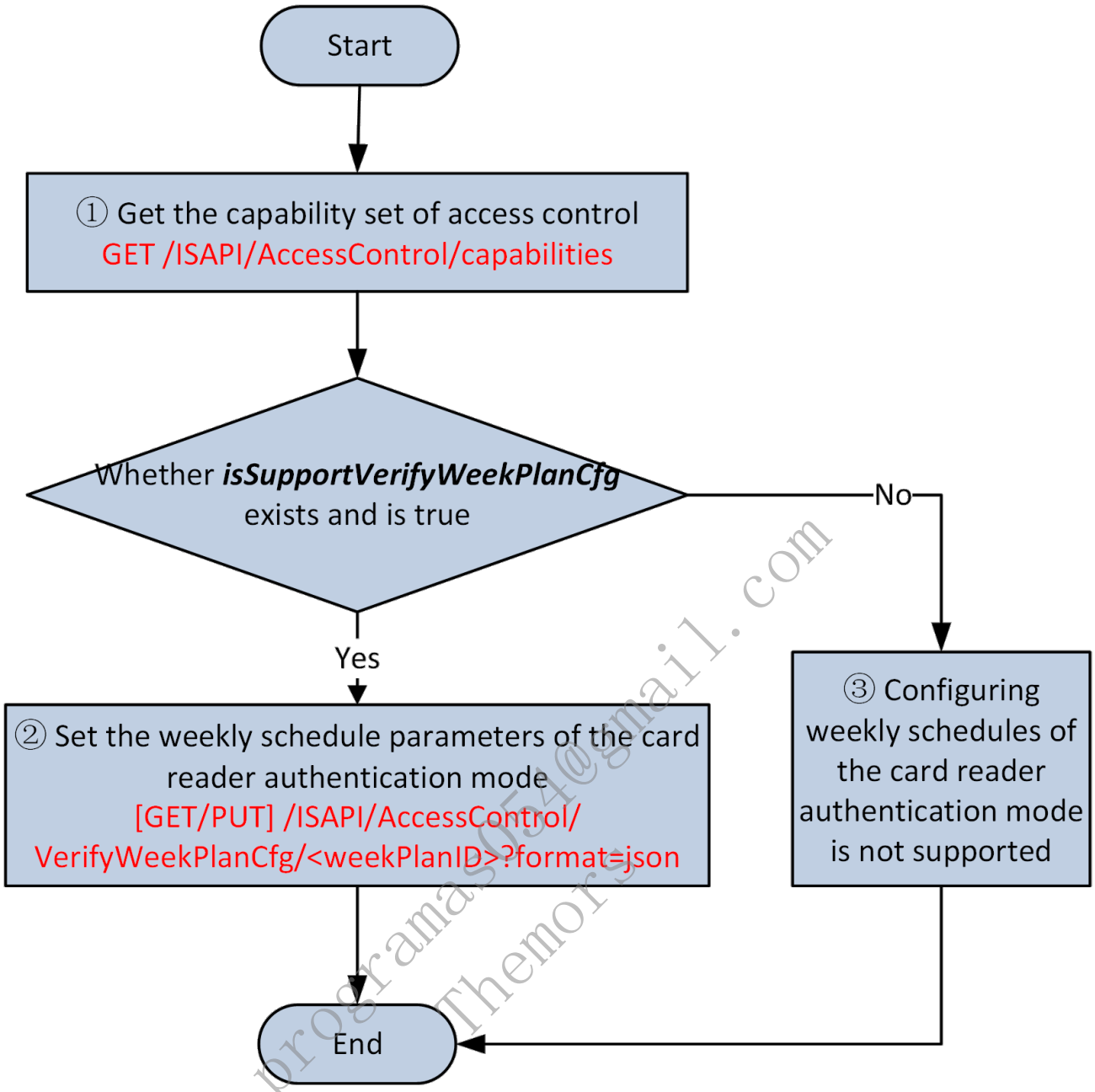
7.3.2.1 Card Reader's Authentication Schedule Configuration



The API calling flow is as follows:

1. Check whether the device supports configuring control schedules of card reader authentication mode: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportCardReaderPlan is returned and its value is "true", it indicates that the device supports configuring acontrol schedules of card reader authentication mode (and the device also supports configuring schedule templates of the card reader authentication mode).
2. Set control schedule parameters of card reader authentication mode: `[GET/PUT] /ISAPI/AccessControl/CardReaderPlan/<cardReaderID>?format=json`; the value of cardReaderID should be the same as the value of templateNo.
3. If the node isSupportCardReaderPlan is returned and its value is "false", it indicates that the device does not support configuring control schedules of card reader authentication mode.

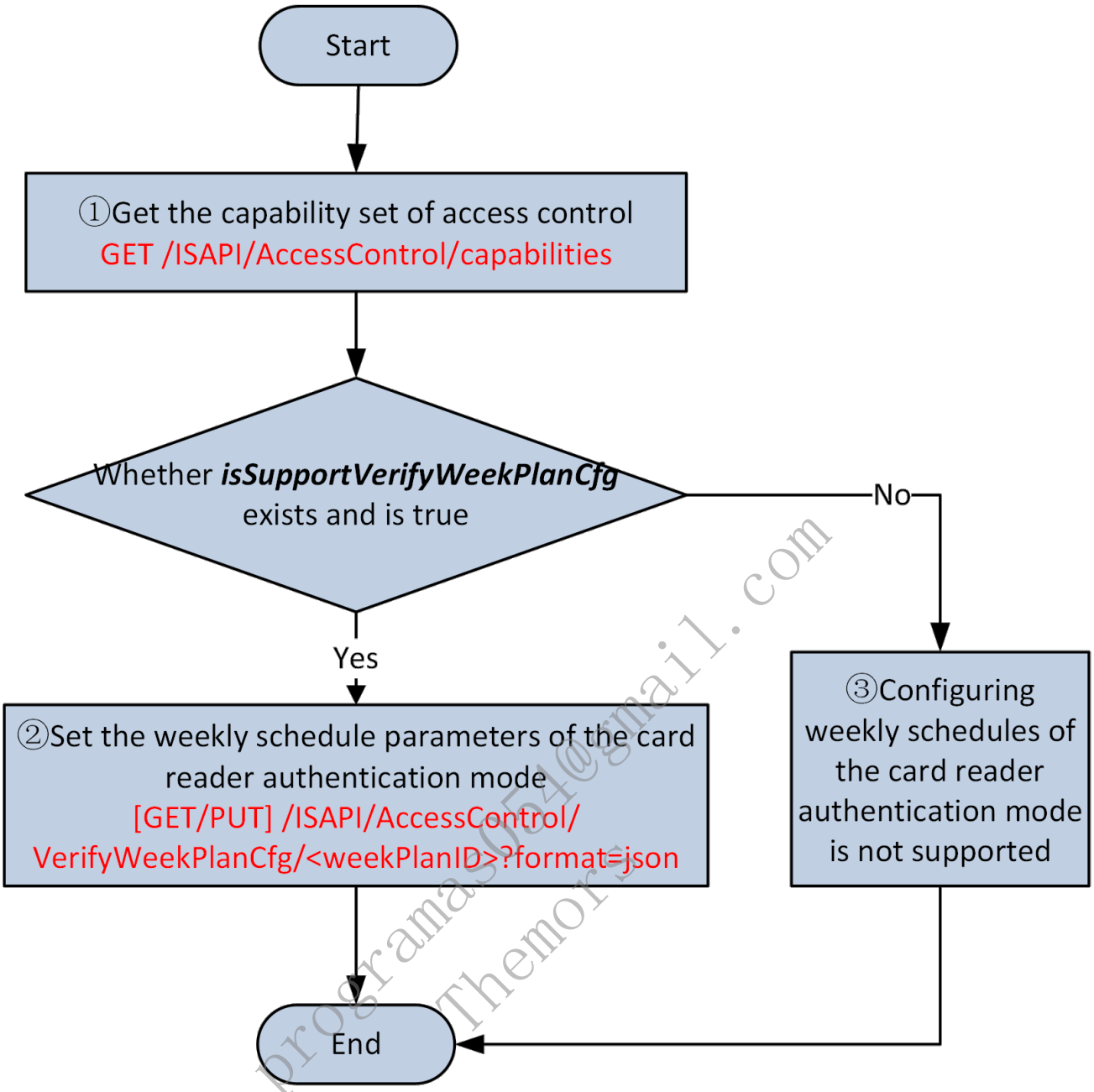
7.3.2.2 Authentication Schedule Template Configuration



The API calling flow is as follows:

1. Check whether the device supports configuring schedule templates of the card reader authentication mode: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportVerifyPlanTemplate is returned and its value is "true", it indicates that the device supports configuring schedule templates of the card reader authentication mode (and the device also supports configuring weekly schedules of the card reader authentication mode).
2. Set the schedule template parameters of the card reader authentication mode: `[GET/PUT] /ISAPI/AccessControl/VerifyPlanTemplate/<planTemplateID>?format=json`.
3. If the node isSupportVerifyPlanTemplate is returned and its value is "false", it indicates that the device does not support configuring schedule templates of the card reader authentication mode.

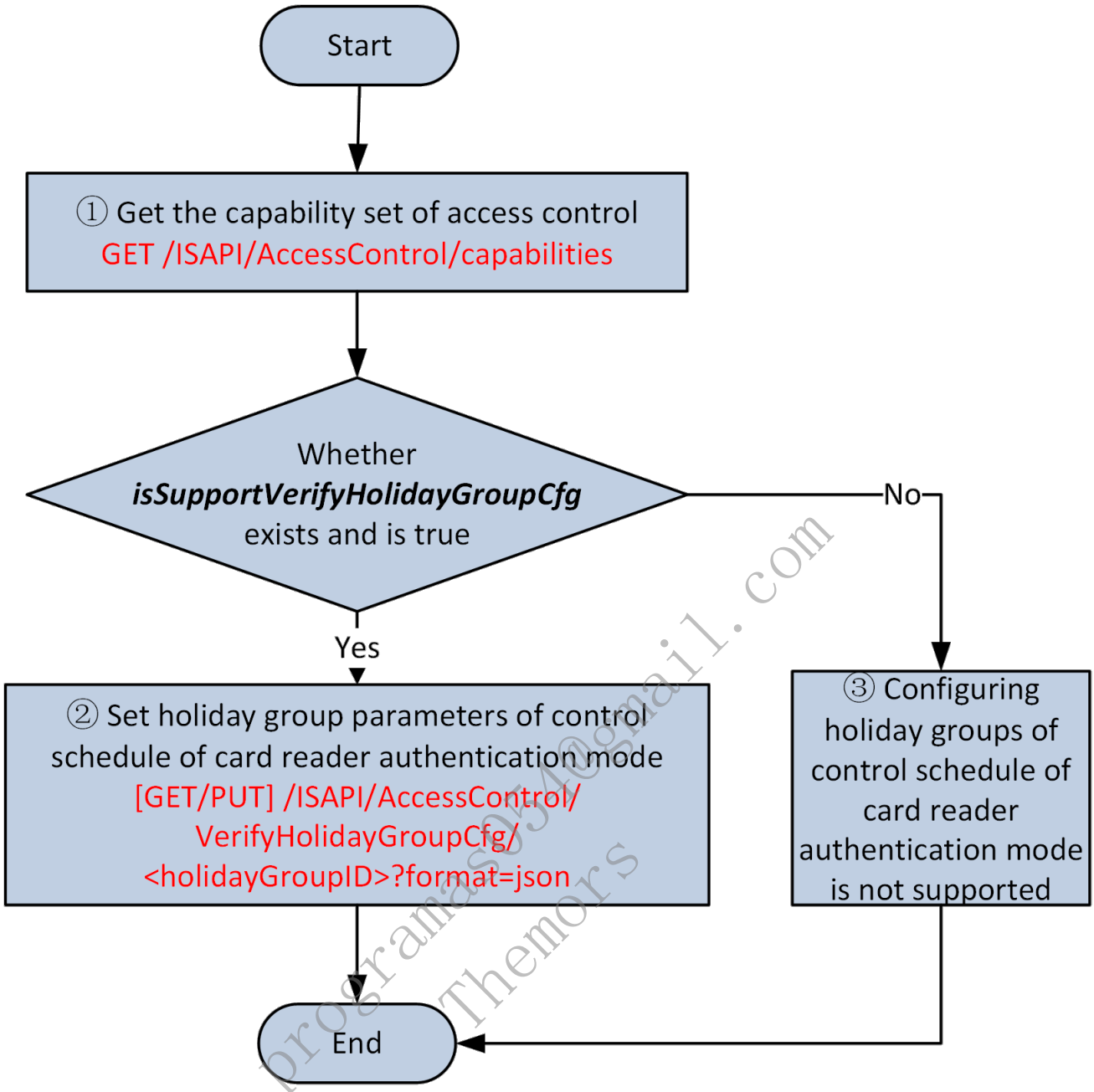
7.3.2.3 Weekly Authentication Schedule Template Configuration



The API calling flow is as follows:

1. Check whether the device supports configuring weekly schedules of the card reader authentication mode: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportCardRightWeekPlanCfg is returned and its value is "true", it indicates that the device supports configuring weekly schedules of the card reader authentication mode.
2. Set the weekly schedule parameters of the card reader authentication mode: `[GET/PUT] /ISAPI/AccessControl/VerifyWeekPlanCfg/<weekPlanID>?format=json`.
3. If the node isSupportVerifyWeekPlanCfg is returned and its value is "false", it indicates that the device does not support configuring weekly schedules of the card reader authentication mode.

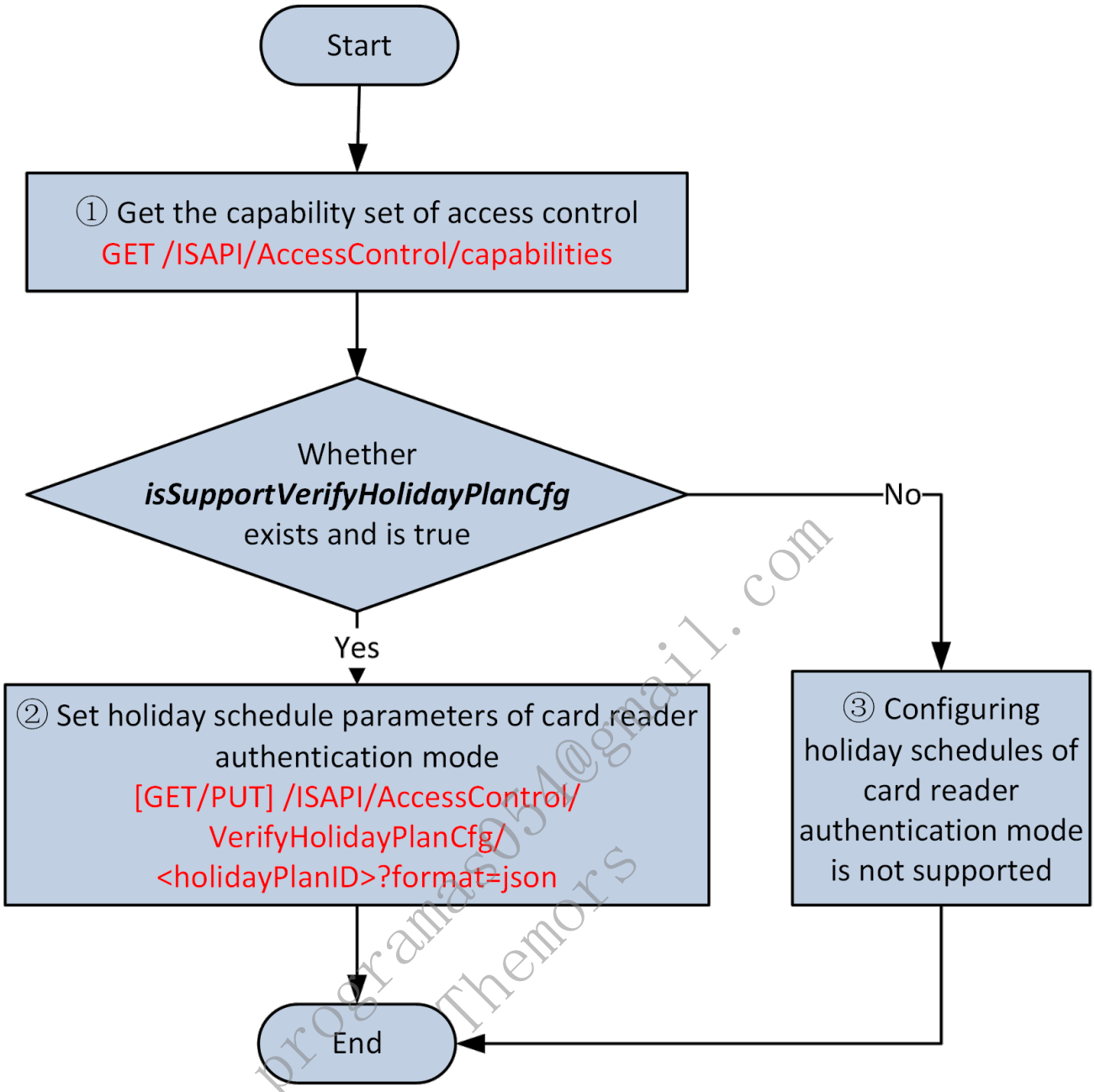
7.3.2.4 Holiday Authentication Group Configuration



The API calling flow is as follows:

1. Check whether the device supports configuring holiday groups of control schedule of card reader authentication mode: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportVerifyHolidayGroupCfg is returned and its value is "true", it indicates that the device supports configuring holiday groups of control schedule of card reader authentication mode (and the device also supports configuring holiday schedules of card reader authentication mode).
2. Set holiday group parameters of control schedule of card reader authentication mode: `[GET/PUT] /ISAPI/AccessControl/VerifyHolidayGroupCfg/<holidayGroupID>?format=json`.
3. If the node isSupportVerifyHolidayGroupCfg is returned and its value is "false", it indicates that the device does not support configuring holiday groups of control schedule of card reader authentication mode.

7.3.2.5 Holiday Authentication Schedule Configuration



The API calling flow is as follows:

1. Check whether the device supports configuring holiday schedules of card reader authentication mode: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportVerifyHolidayPlanCfg is returned and its value is "true", it indicates that the device supports configuring holiday schedules of card reader authentication mode.
2. Set holiday schedule parameters of card reader authentication mode: `[GET/PUT] /ISAPI/AccessControl/VerifyHolidayPlanCfg/<holidayPlanID>?format=json`.
3. If the node isSupportVerifyHolidayPlanCfg is returned and its value is "false", it indicates that the device does not support configuring holiday schedules of card reader authentication mode.

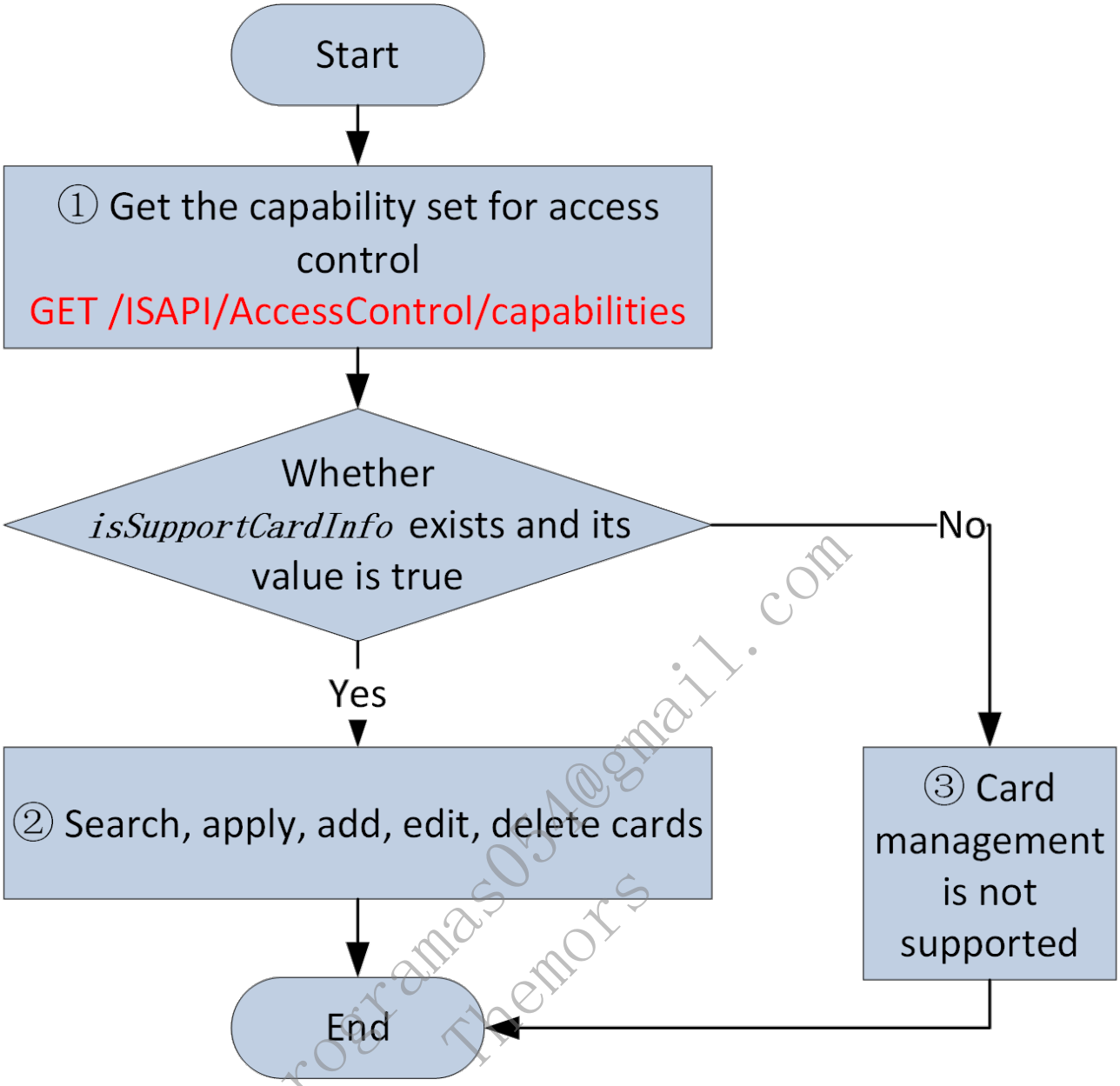
7.4 Card Management

7.4.1 Introduction to the Function

Card management includes searching, applying, adding, editing, deleting, and collecting cards.

7.4.2 API Calling Flow

7.4.2.1 Check Whether the Device Supports Card Management



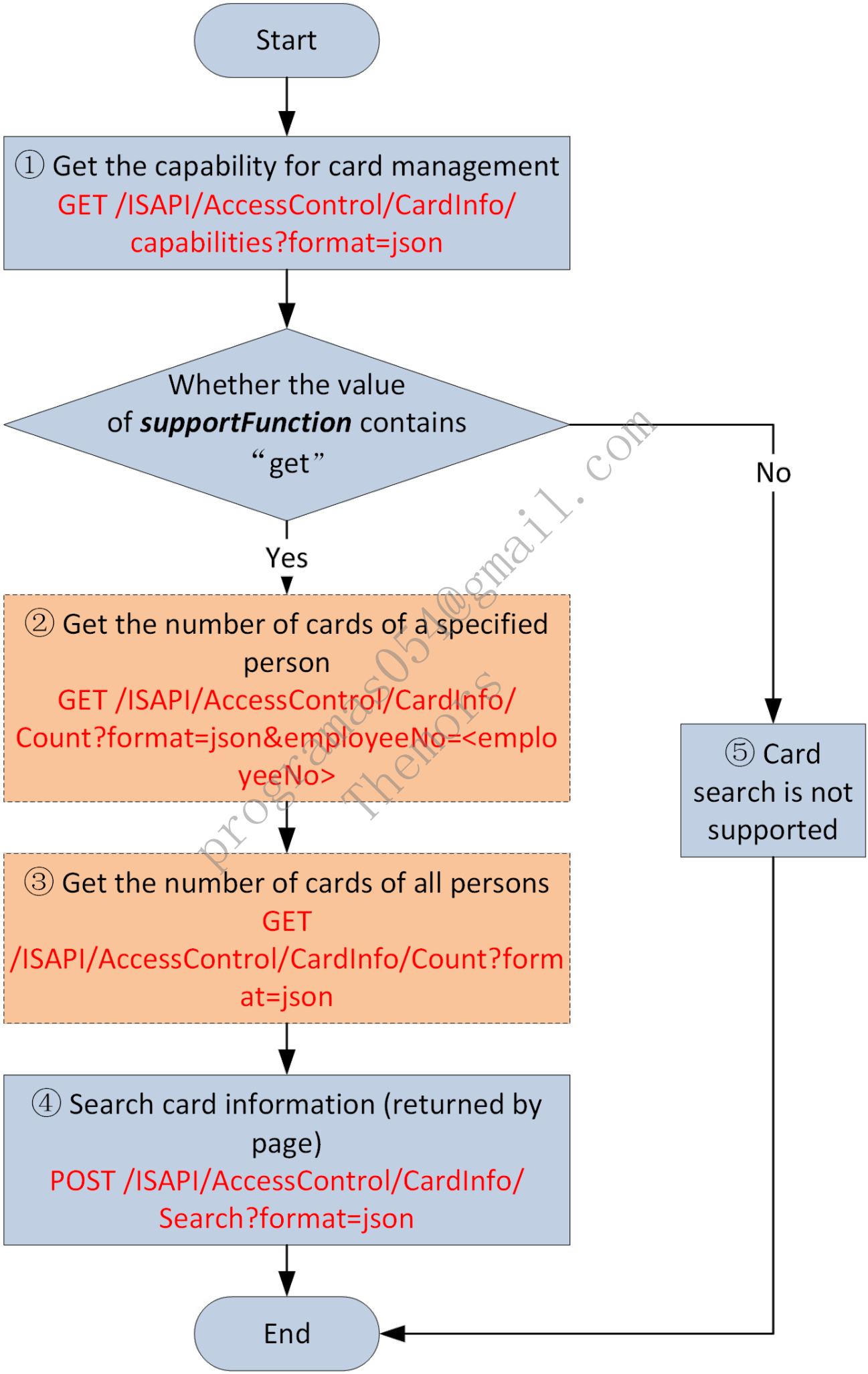
Before calling the API for card management, make sure that the device supports card management.

1. Check whether the device supports card management: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportCardInfo is returned and its value is "true", it indicates that the device supports card management.
2. Search, apply, add, edit, and delete cards.
3. If the node isSupportCardInfo is returned and its value is "false", it indicates that the device does not support card management.

Note:

- Before applying, adding, or editing cards on the device, make sure that the related person information linked to the person ID has been applied to the device.
- The value of the node numberPerPerson returned by calling `GET /ISAPI/AccessControl/CardInfo/capabilities?format=json` is the maximum number of cards supported per person. If the value returned is 255, it indicates that the number of cards per person is unlimited. If the node is not returned, it indicates that the maximum number of cards can be applied is 5.
- Manage cards of different card number lengths by calling `[GET/PUT] /ISAPI/AccessControl/CardVerificationRule?format=json`.

7.4.2.2 Card Search



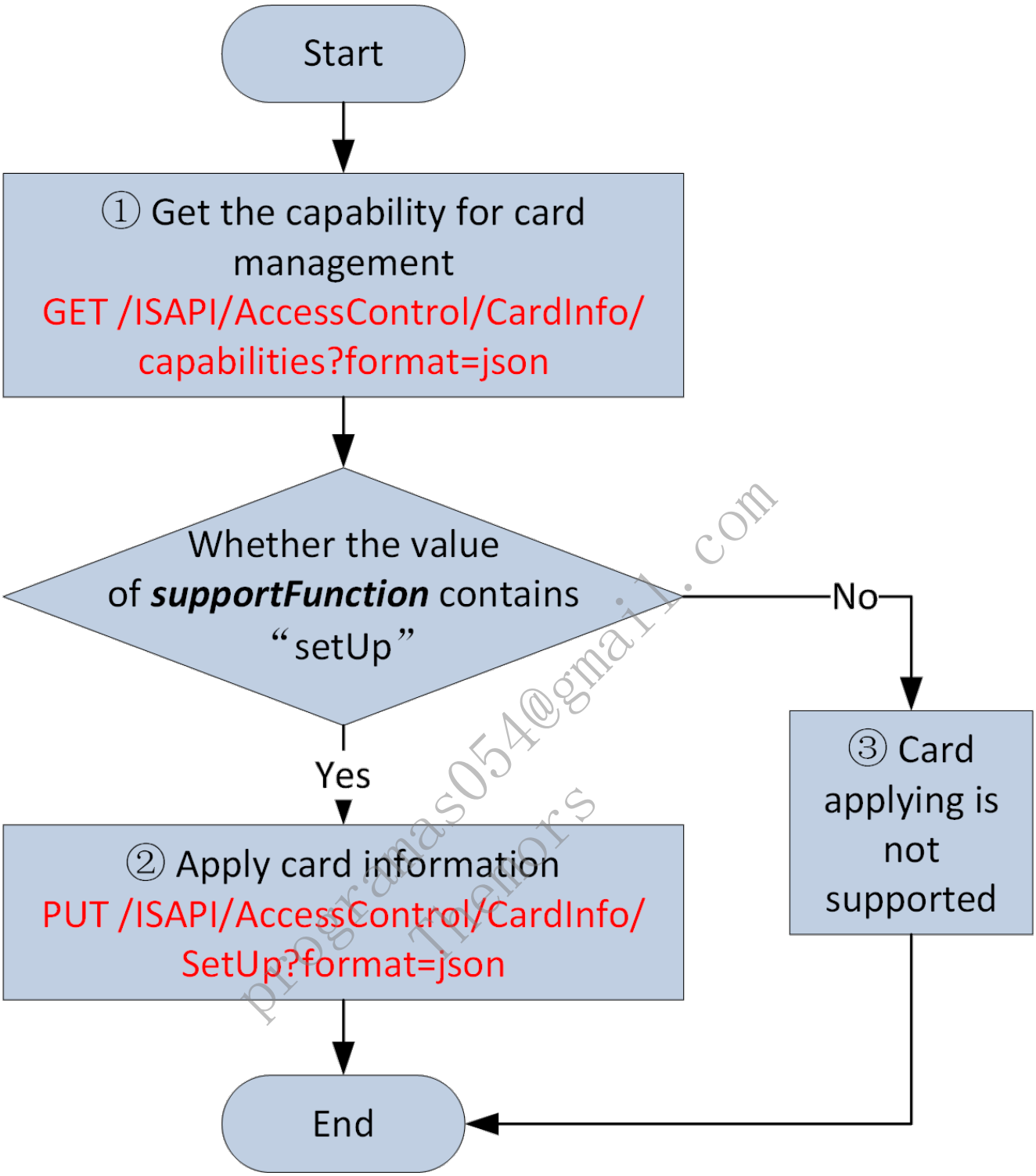
The card search function is for searching the number of cards and card information applied to the device.

1. Check whether the device supports card search: `GET /ISAPI/AccessControl/CardInfo/capabilities?format=json`; if the value of the node supportFunction contains "get", it indicates that the device supports card search.
2. Search the number of specified persons' cards: `GET /ISAPI/AccessControl/CardInfo/Count?format=json&employeeNo=<employeeNo>`; the returned value of the node cardNumber is the number of the cards added to the specified persons.
3. Search the number of all persons' cards: `GET /ISAPI/AccessControl/CardInfo/Count?format=json`; the returned value of the node cardNumber is the number of the cards added to all persons.
4. Search card information: `POST /ISAPI/AccessControl/CardInfo/Search?format=json`; the card information is returned by page.
5. If the value of the node supportFunction does not contain "get", it indicates that the device does not support card search.

Note:

The value of the node maxRecordNum returned by calling `GET /ISAPI/AccessControl/CardInfo/capabilities?format=json` is the maximum number of cards supported by the device.

7.4.2.3 Card Applying



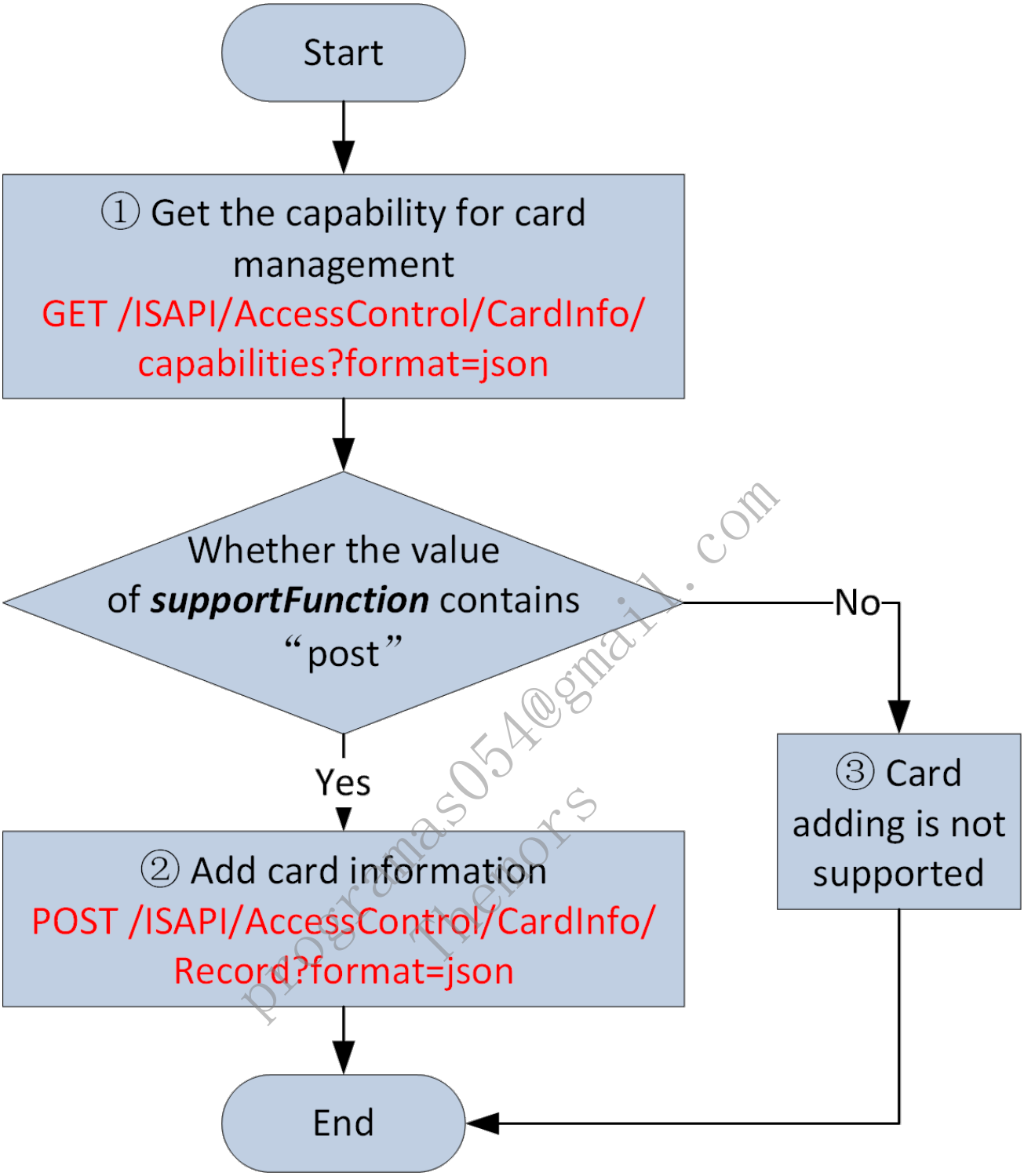
Card information can be applied to the device via the card applying function. If the card has been added to the device, the card information will be edited; if the card has not been added to the device, the card information will be added to the device.

1. Check whether the device supports card applying: `GET /ISAPI/AccessControl/CardInfo/capabilities?format=json`; if the value of the node supportFunction contains "setUp", it indicates that the device supports card applying.
2. Apply card information: `PUT /ISAPI/AccessControl/CardInfo/SetUp?format=json`.
3. If the value of the node supportFunction does not contain "setUp", it indicates that the device does not support card applying.

Note:

Check whether the card has been added to the device via the node cardNo returned after calling the API for card applying.

7.4.2.4 Card Adding



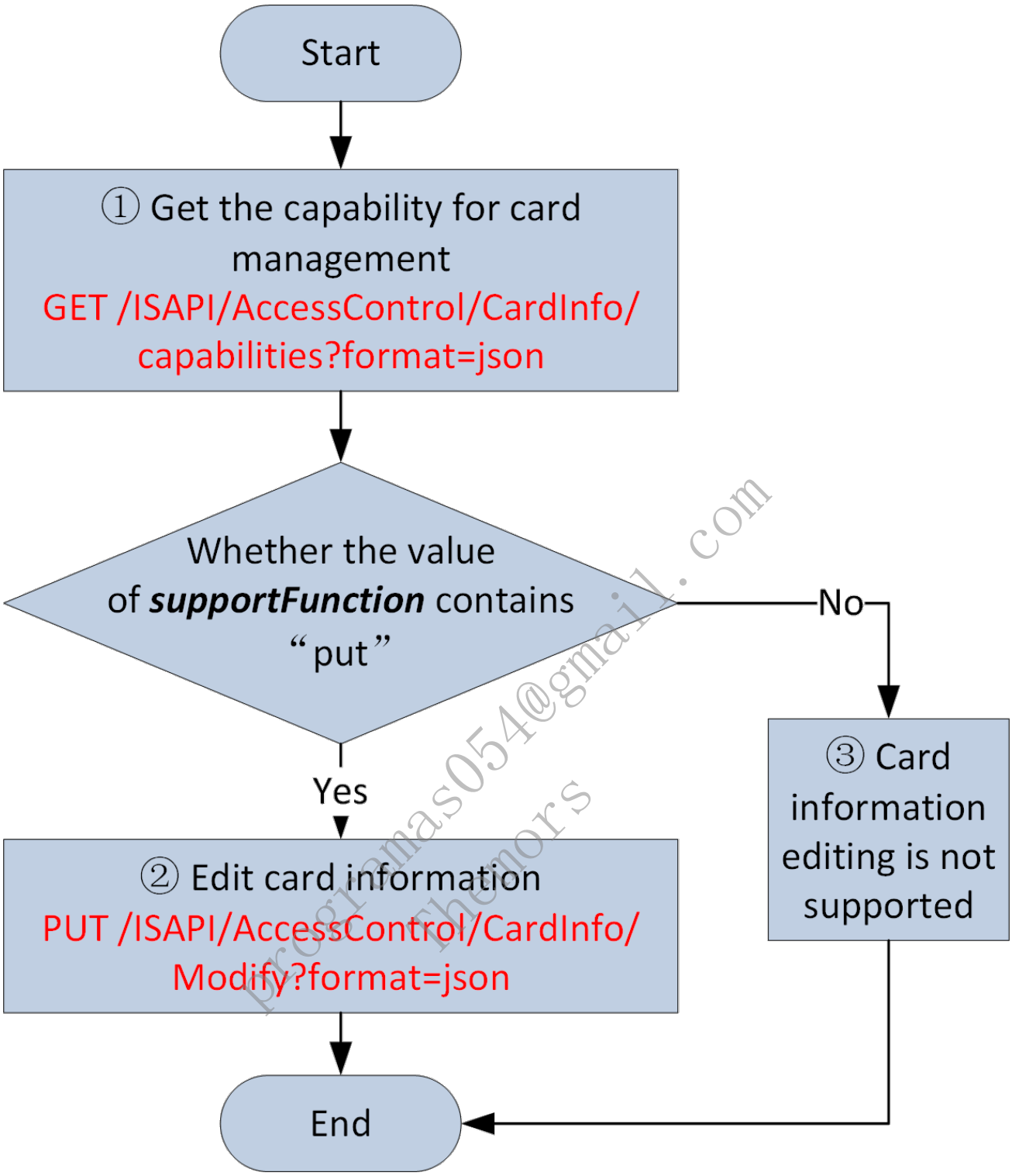
Card information can be added to the device via the card adding function. If the card has been added to the device, the device will report an error; if the card has not been added to the device, the card information will be added to the device.

1. Check whether the device supports card adding: `GET /ISAPI/AccessControl/CardInfo/capabilities?format=json`; if the value of the node supportFunction contains "post", it indicates that the device supports card adding.
2. Add card information: `POST /ISAPI/AccessControl/CardInfo/Record?format=json`.
3. If the value of the node supportFunction does not contain "post", it indicates that the device does not support card adding.

Note:

Check whether the card has been added to the device via the node cardNo returned after calling the API for card adding.

7.4.2.5 Card Information Editing



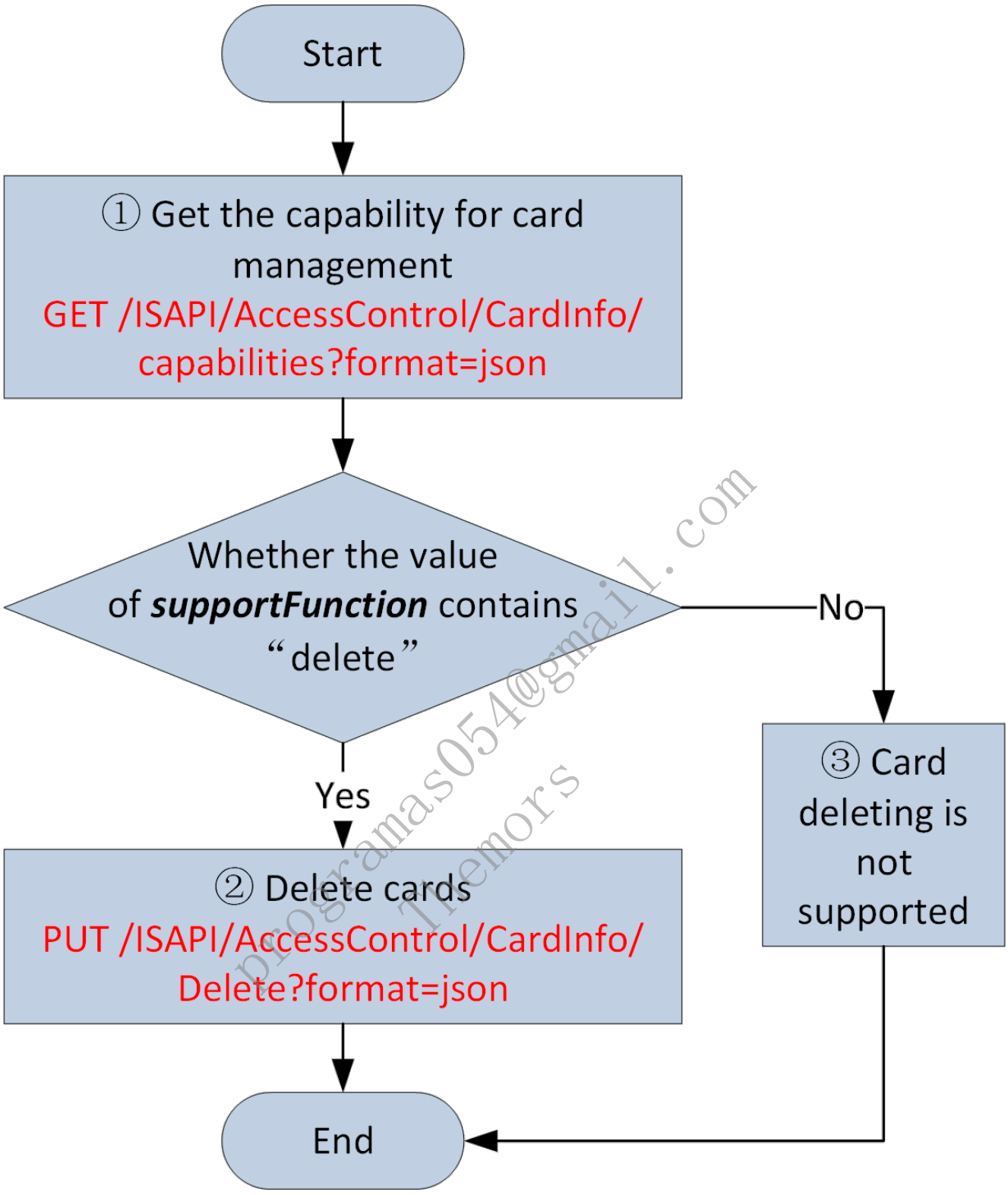
Card information on the device can be edited via the card information editing function. If the card has been added to the device, the card information will be edited; if the card has not been added to the device, the device will report an error.

1. Check whether the device supports card information editing: `GET /ISAPI/AccessControl/CardInfo/capabilities?format=json`; if the value of the node supportFunction contains "put", it indicates that the device supports card information editing.
2. Edit card information: `PUT /ISAPI/AccessControl/CardInfo/Modify?format=json`.
3. If the value of the node supportFunction does not contain "put", it indicates that the device does not support card information editing.

Note:

Check whether the card has been added to the device via the node cardNo returned after calling the API for card information editing.

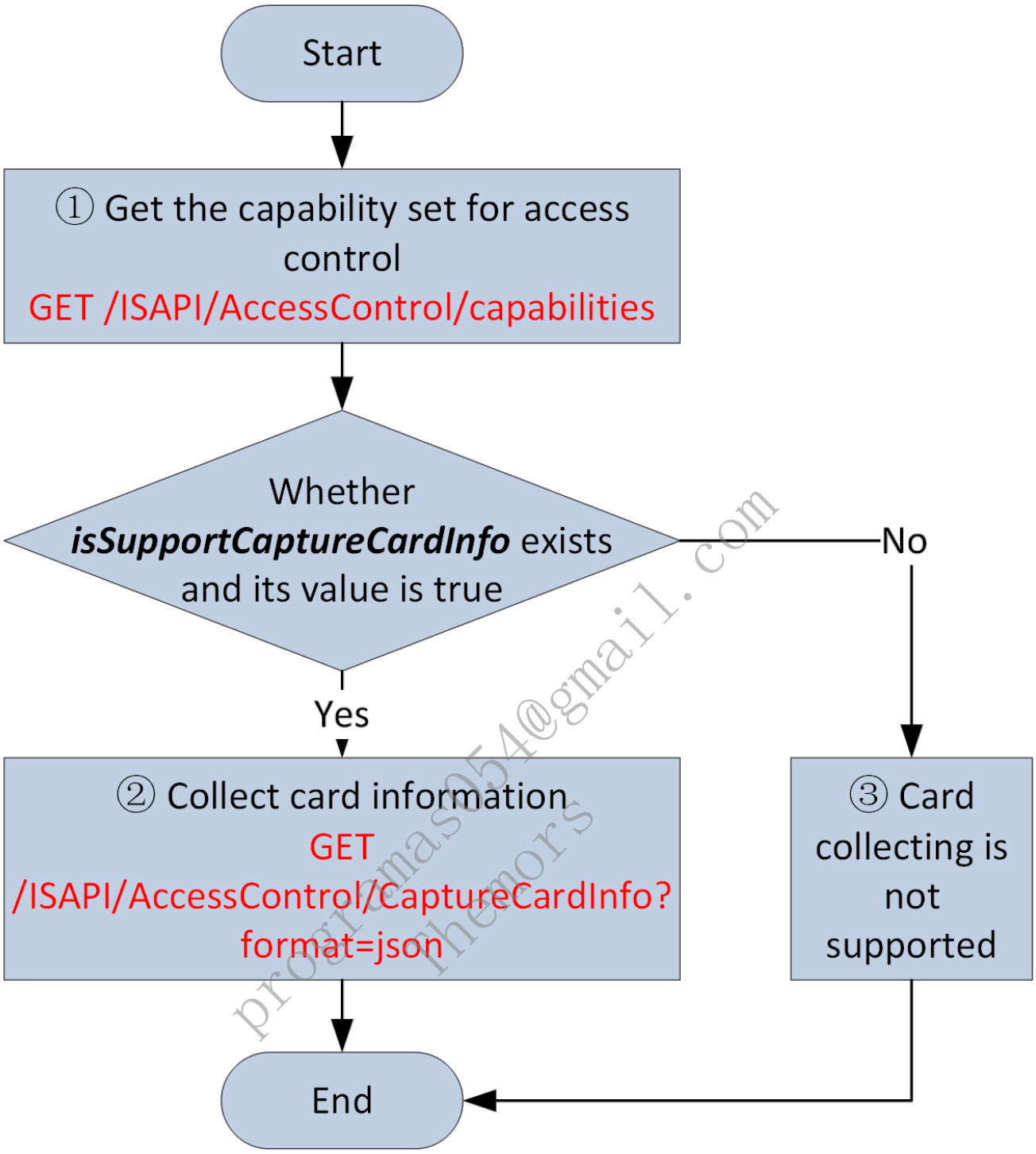
7.4.2.6 Card Deleting



The card information on the device can be deleted via the card deleting function. The device will not report an error if the card information to be deleted is not added to the device.

1. Check whether the device supports card deleting: `GET /ISAPI/AccessControl/CardInfo/capabilities?format=json`; if the value of the node supportFunction contains "delete", it indicates that the device supports card deleting.
2. Delete cards: `PUT /ISAPI/AccessControl/CardInfo/Delete?format=json`; if calling succeeded, it indicates that the device has deleted the cards.
3. If the value of the node supportFunction does not contain "delete", it indicates that the device does not support card deleting.

7.4.2.7 Card Collecting



The card collecting function is for collecting the card No., card type, etc.

1. Check whether the device supports card collecting: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportCaptureCardInfo is returned and its value is "true", it indicates that the device supports card collecting.
2. Collect card information: `GET /ISAPI/AccessControl/CaptureCardInfo?format=json`.
3. If the node isSupportCaptureCardInfo is returned and its value is "false", it indicates that the device does not support card collecting.

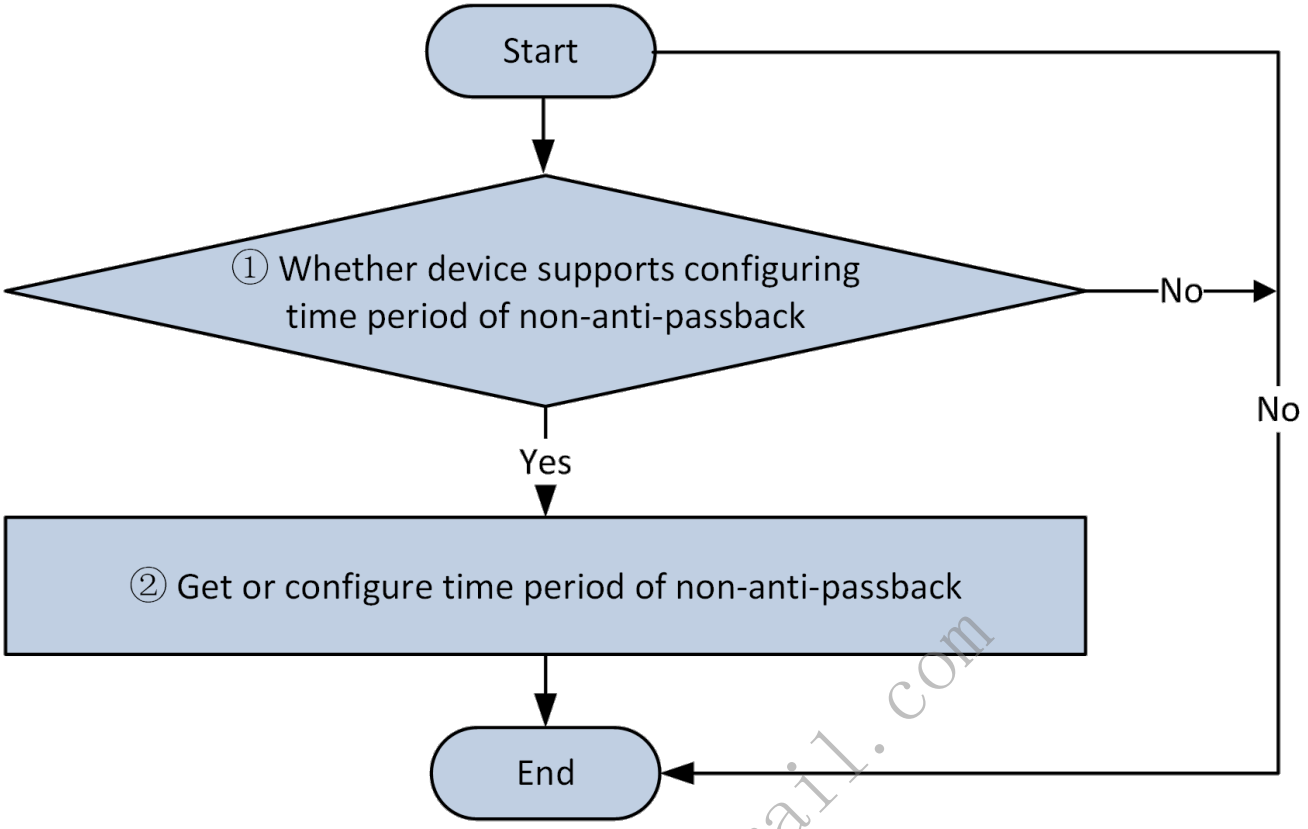
7.5 Configure Non-anti-passback Time Period

Non-anti-passback time period: anti-passback is not triggered in the set time period. Application Scenarios: In rush hour, anti-passback can always happens since the person might follow others in the people flow. Non-anti-passback can help normal entry&exit in rush hour.

7.5.1 Introduction to the Function

7.5.2 API Calling Flow

Calling Flow:



ISAPI Protocol Calling Flow:

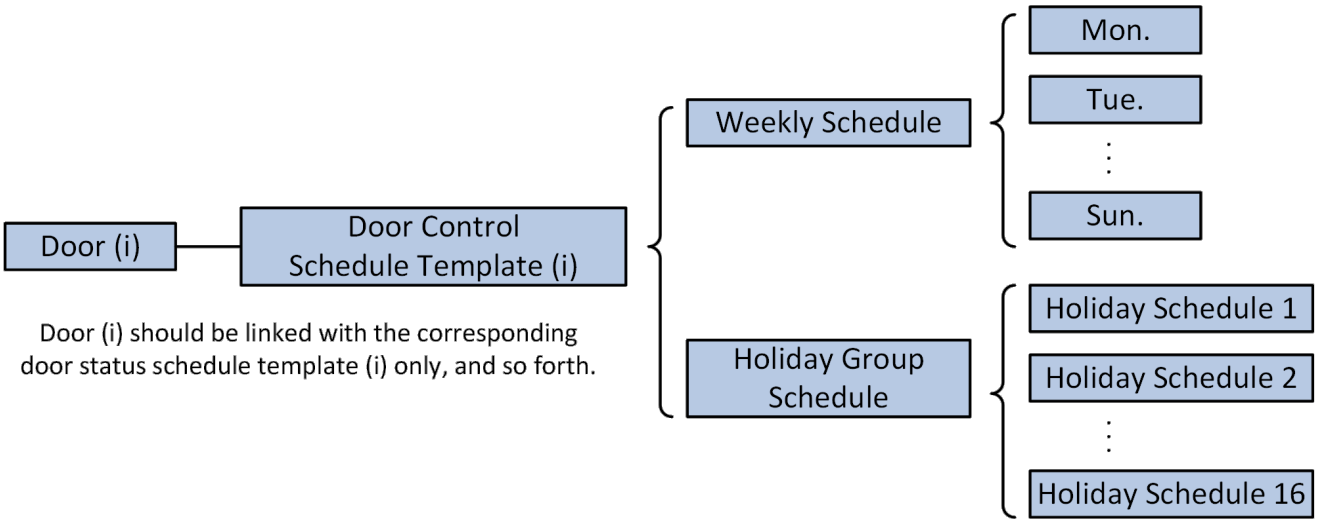
1. Get the capability of access control: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportAntiPassbackTimeRange is returned and its value is "true", it indicates that the device supports configuring time period of anti-passback.
2. Get time period parameters of non anti-passback: `GET /ISAPI/AccessControl/AntiPassback/timeRange?format=json`; configure time period parameters of non anti-passback: `PUT /ISAPI/AccessControl/AntiPassback/timeRange?format=json`; it can specify the time period of anti-passback; Get the capacity of configuring time period of anti-passback: `GET /ISAPI/AccessControl/AntiPassback/timeRange/capabilities?format=json`.

7.6 Door Control Schedule Management

7.6.1 Introduction to the Function

For doors of access control or floors of elevator control, you can set schedule templates. The schedule template information include time period and status (remain open, remain closed, sleep, and normal). Configuring door control schedule is not required. If it is not configured, No device configuration has permission for all the doors by default. The priority of remote door control is higher than that of door control schedule. The operation of remote door control can take effect when the door is in the status of remain open/closed, sleep, and normal.

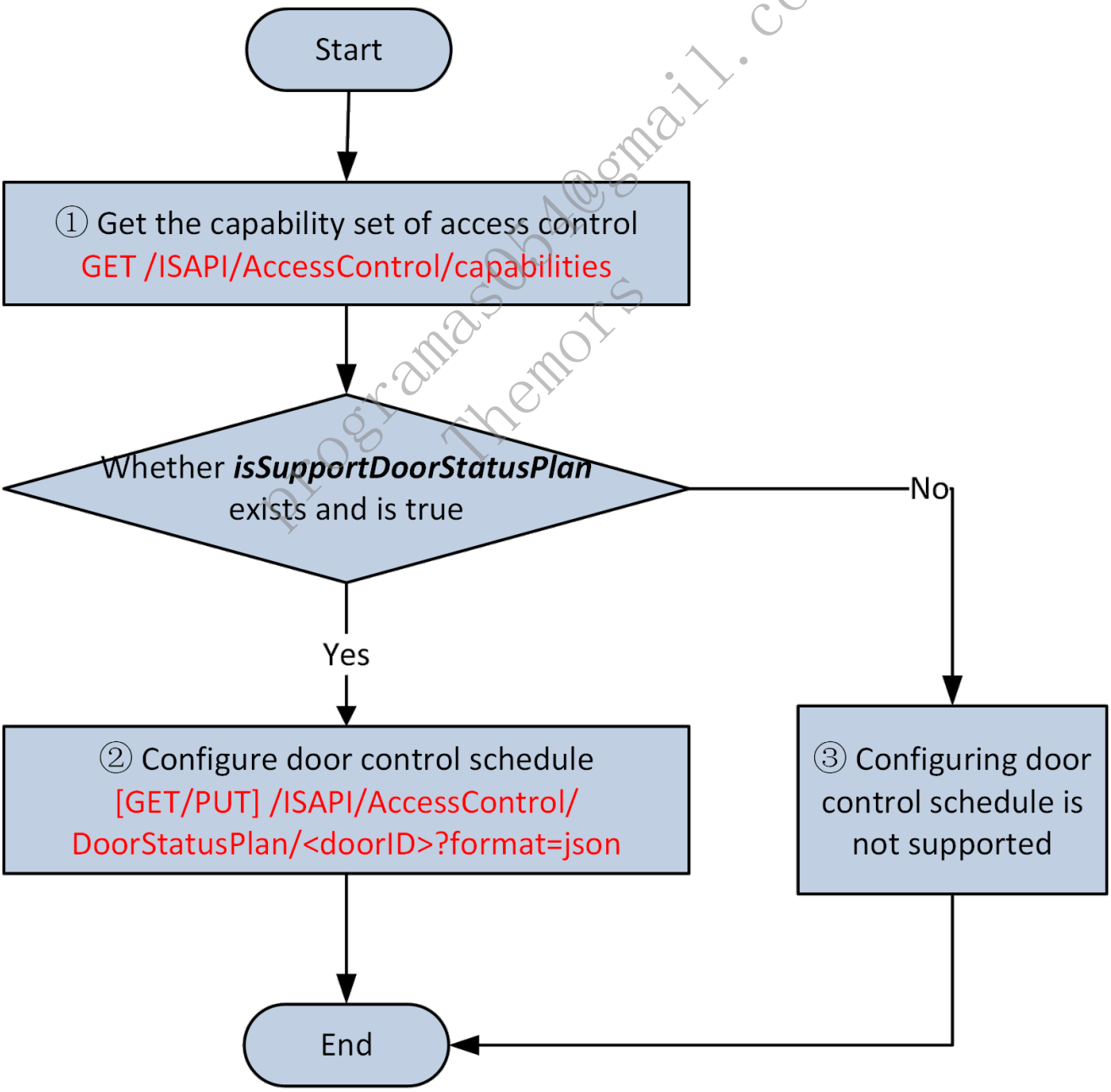
Each schedule template can be linked to one week schedule and four holiday group schedules. Holiday schedule priority is higher than that of weekly schedule. Weekly schedule can be configured with time periods from Monday to Sunday, and 8 different time periods are supported each day. Holiday group schedule can be linked to 16 different holiday schedules. Each holiday schedule can be configured with one start and end date of the holiday, and the access time period is the same for each day (up to 8 different time periods can be configured). This schedule template is configured to manage access control permission.



Door (i) should be linked with the corresponding door status schedule template (i) only, and so forth.

7.6.2 API Calling Flow

7.6.2.1 Configure Door Control Schedule



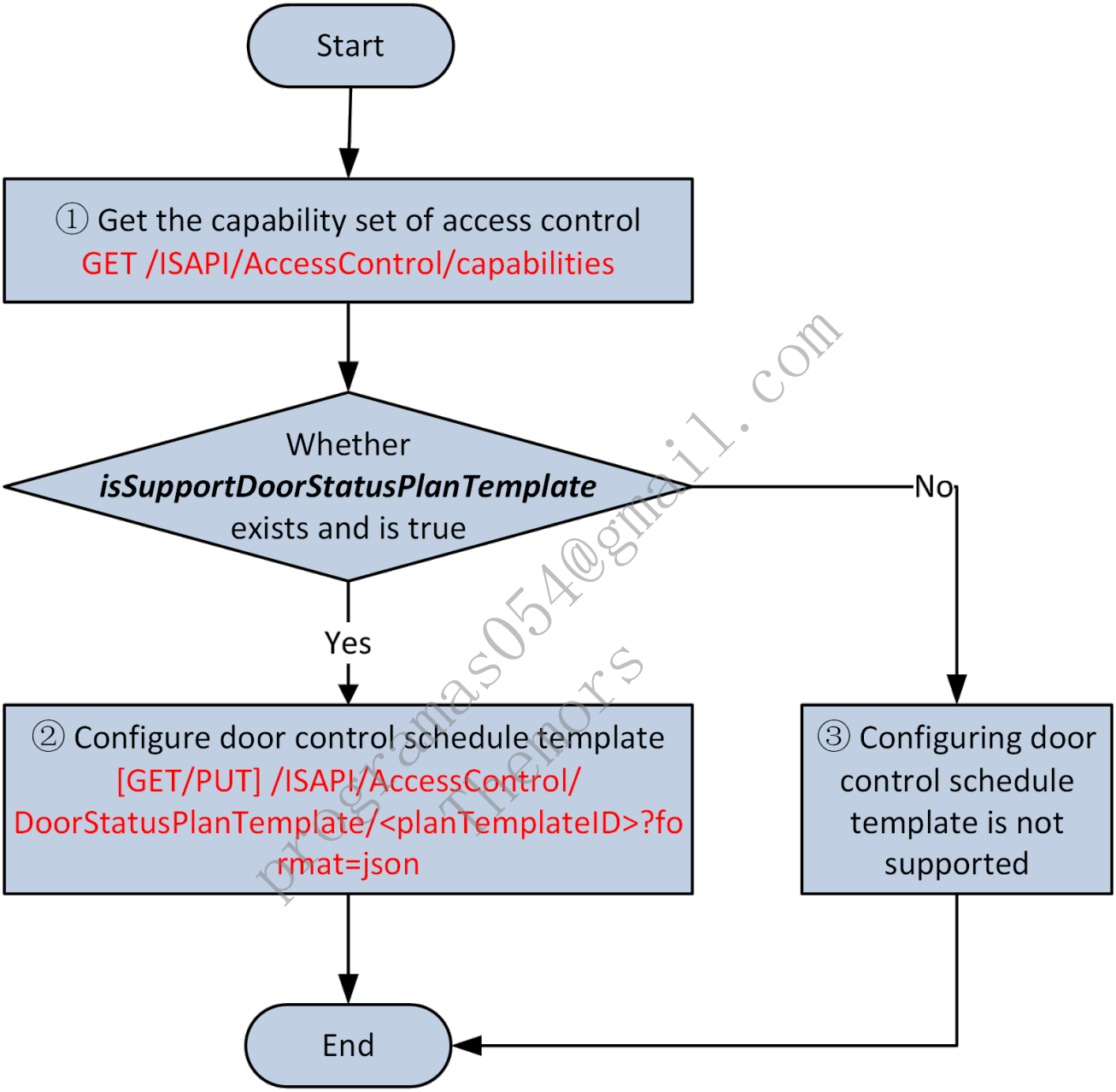
The API calling flow is as follow:

1. Check whether the device supports configuring door control schedule: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportDoorStatusPlan is returned and its value is "true", it indicates that the device supports

configuring door control schedule (and the device also supports configuring schedule template of door control).

2. Set door control schedule: `[GET/PUT] /ISAPI/AccessControl/DoorStatusPlan/<doorID>?format=json`.
3. If the node isSupportDoorStatusPlan is returned and its value is "false", it indicates that the device does not support configuring door control schedule.

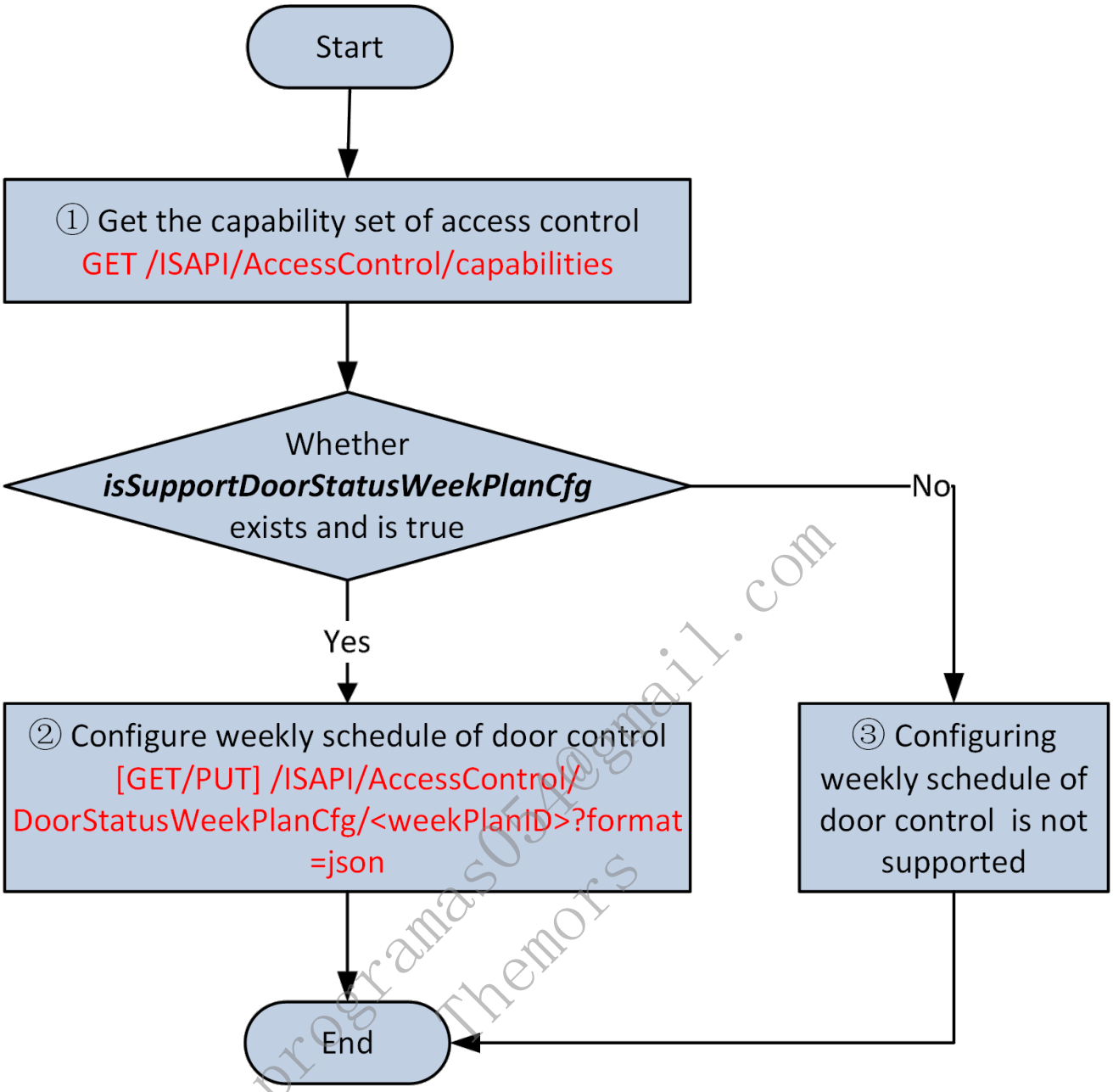
7.6.2.2 Configure Door Control Schedule Template



The API calling flow is as follow:

1. Check whether the device supports configuring door control schedule template:`GET /ISAPI/AccessControl/capabilities`; if the node isSupportDoorStatusPlanTemplate is returned and its value is "true", it indicates that the device supports configuring door control schedule template (and the device should also supports configuring door status schedule template).
2. Get and set parameters of door control schedule template: `[GET/PUT] /ISAPI/AccessControl/DoorStatusPlanTemplate/<planTemplateID>?format=json`.
3. If the node isSupportDoorStatusPlanTemplate is returned and its value is "false", it indicates that the device does not support configuring door control schedule template.

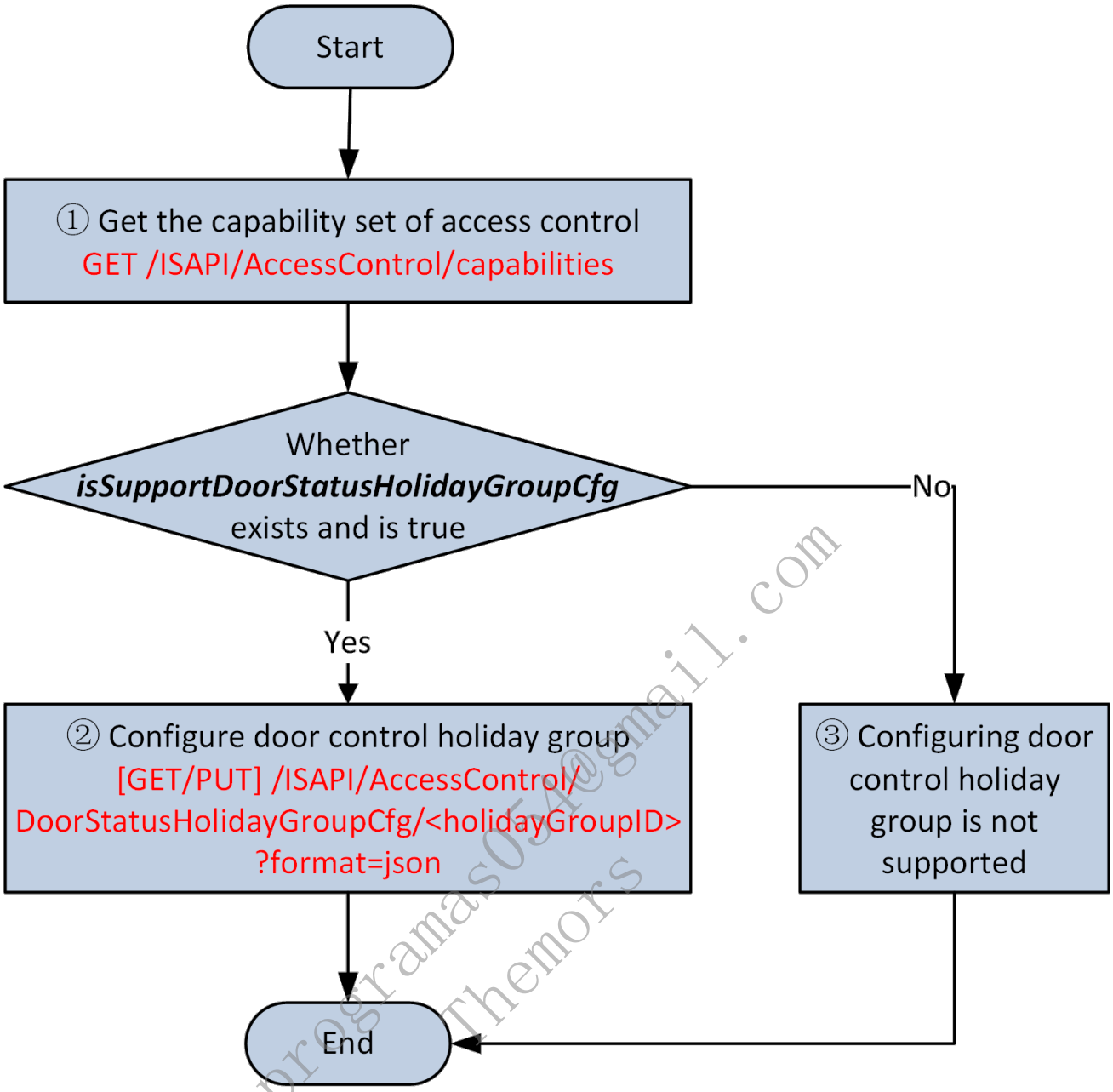
7.6.2.3 Configure Door Control Weekly Schedule



The API calling flow is as follow:

1. Check whether the device supports configuring door control weekly schedule: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportDoorStatusWeekPlanCfg is returned and its value is "true", it indicates that the device supports this function.
2. Set parameters of door control weekly schedule: `[GET/PUT] /ISAPI/AccessControl/DoorStatusWeekPlanCfg/<weekPlanID>?format=json`.
3. If the node isSupportDoorStatusWeekPlanCfg is returned and its value is "false", it indicates that the device does not support configuring this function.

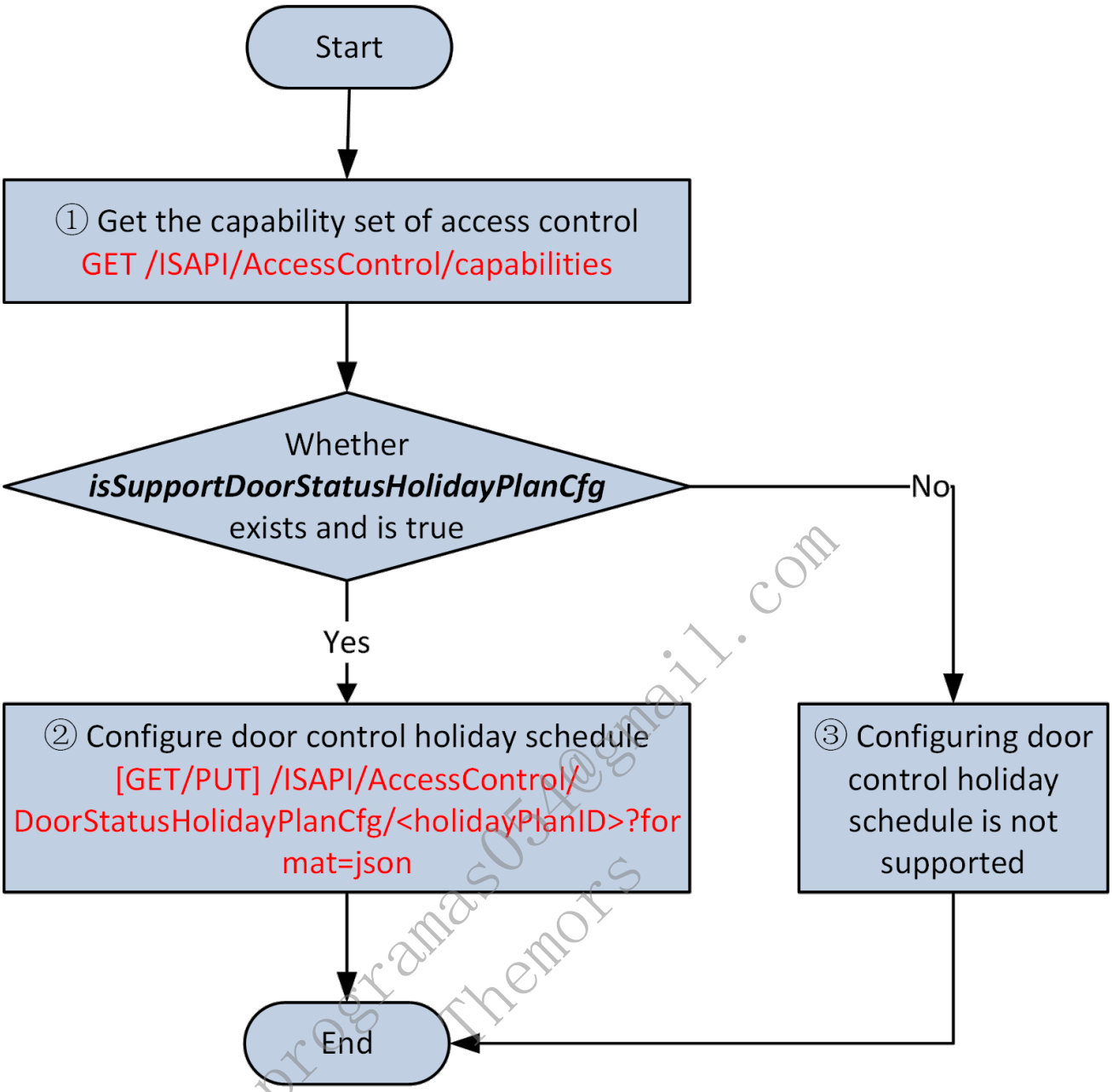
7.6.2.4 Configure Door Control Holiday Group



The API calling flow is as follow:

1. Check whether the device supports configuring door control holiday group: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportDoorStatusHolidayGroupCfg is returned and its value is "true", it indicates that the device supports configuring door control holiday group (and the device also supports configuring door control holiday schedule).
2. Set the holiday group configuration parameters of the door control schedule: `[GET/PUT] /ISAPI/AccessControl/DoorStatusHolidayGroupCfg/<holidayGroupID>?format=json`.
3. If the node isSupportDoorStatusHolidayGroupCfg is returned and its value is "false", it indicates that the device does not support this function.

7.6.2.5 Configure Door Control Holiday Schedule



The API calling flow is as follow:

1. Check whether the device supports configuring door control holiday schedule: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportDoorStatusHolidayPlanCfg is returned and its value is "true", it indicates that the device supports this function.
2. Set parameters of door control holiday schedule: `[GET/PUT] /ISAPI/AccessControl/DoorStatusHolidayPlanCfg/<holidayPlanID>?format=json`.
3. If the node isSupportDoorStatusHolidayPlanCfg is returned and its value is "false", it indicates that the device does not support this function.

7.7 Event and Card Linkage Parameters

7.7.1 Introduction to the Function

The device supports linking specific actions when access control events triggered. The linkage actions include event, card No., MAC address, and employee No. Take event for example, if a person authenticated by the device, the door will open for the person.

7.7.2 API Calling Flow

7.7.2.1 Configure Parameters of Event and Card Linkage

1. Get the functional capability of access control: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportEventCardLinkageCfg is returned and its value is "true", it indicates that the device supports configuring the parameters of event and card linkage.
2. Get the configuration capability of the event and card linkage: `GET /ISAPI/AccessControl/EventCardLinkageCfg/capabilities?format=json`.
3. Get and set the parameters of event and card linkage: `GET|PUT /ISAPI/AccessControl/EventCardLinkageCfg/<ACEID>?format=json`.

7.7.2.2 Search for Parameters of Event and Card Linkage

1. Get the functional capability of access control: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportEventCardLinkageCfgSearch is returned and its value is "true", it indicates that the device supports searching for the parameters of event and card linkage.
2. Get the capability of searching for parameters of event and card linkage: `GET /ISAPI/AccessControl/EventCardLinkageCfg/search/capabilities?format=json`.
3. Search for parameters of event and card linkage: `POST /ISAPI/AccessControl/EventCardLinkageCfg/search?format=json`.

7.7.2.3 Delete Parameters of Specific Event and Card Linkage

1. Get the functional capability of access control: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportEventCardLinkageCfgDelete is returned and its value is "true", it indicates that the device supports deleting parameters of event and card linkage.
2. Delete parameters of specific event and card linkage: `PUT /ISAPI/AccessControl/EventCardLinkageCfgDelete?format=json`.

7.7.2.4 Get List of Event and Card Linkage ID

1. Call the functional capability of access control: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportEventCardNoList is returned and its value is true, it indicates that the device supports getting events and the card linkage ID list.
2. Get the capability of the list of event and card linkage ID: `GET /ISAPI/AccessControl/EventCardNoList/capabilities?format=json`.
3. Get the list of event and card linkage ID: `GET /ISAPI/AccessControl/EventCardNoList?format=json`.

7.7.2.5 Optimize Event

1. Get the functional capability of access control: `GET /ISAPI/AccessControl/capabilities`; if the node iisSupportEventOptimizationCfg is returned and its value is "true", it indicates that the device supports event optimization.
2. Get the configuration capability of event optimization: `GET /ISAPI/AccessControl/EventOptimizationCfg/capabilities?format=json`.
3. Get the event optimization configuration parameters: `GET|PUT /ISAPI/AccessControl/EventOptimizationCfg?format=json`.

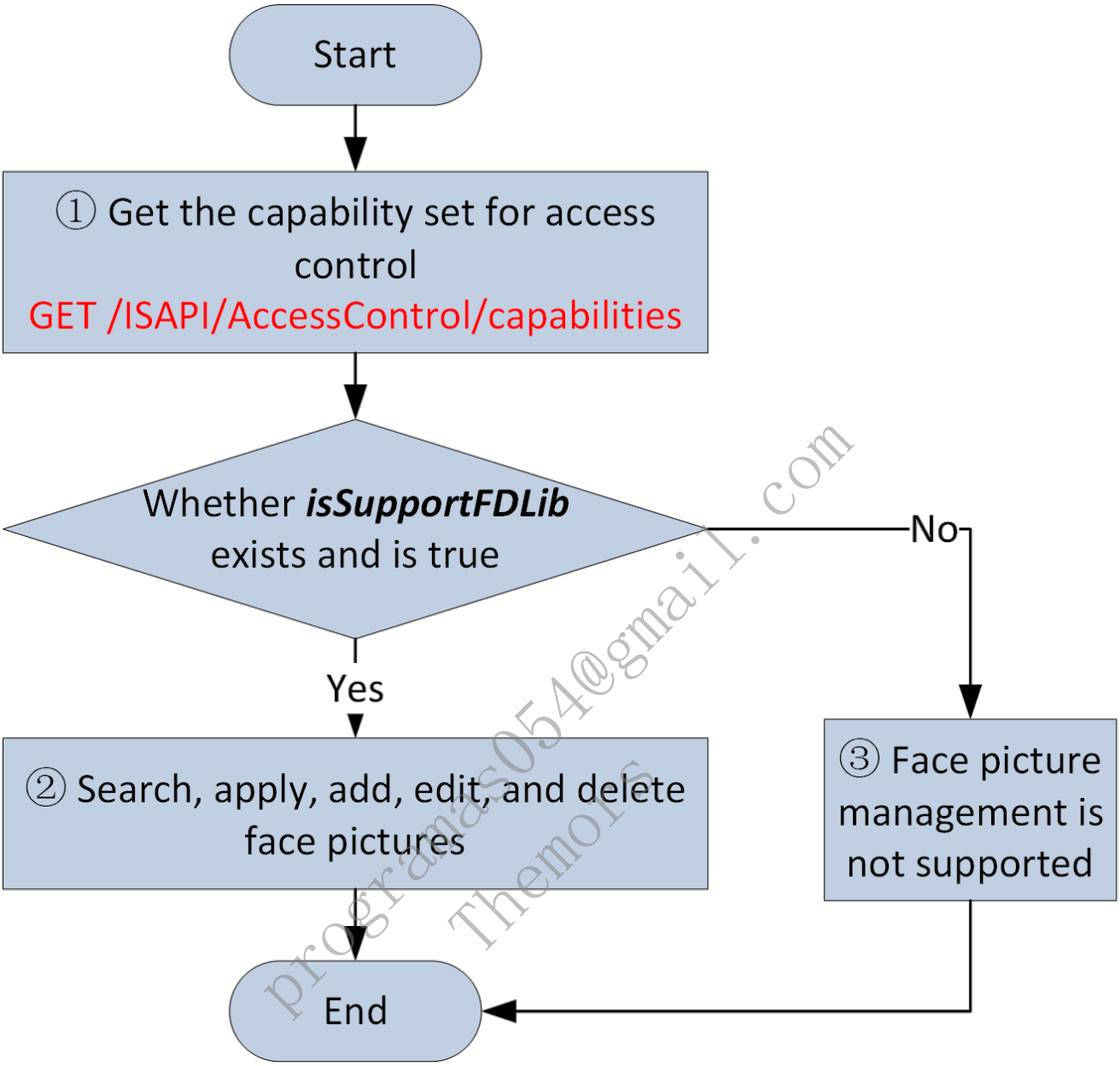
7.8 Face Picture Management

7.8.1 Introduction to the Function

Face picture management includes searching, applying, adding, editing, deleting, and collecting face pictures.

7.8.2 API Calling Flow

7.8.2.1 Check Whether the Device Supports Face Picture Management



Before calling the API for face picture management, make sure that the device supports face picture management.

1. Check whether the device supports face picture management: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportFDLib is returned and its value is "true", it indicates that the device supports face picture management.
2. Search, apply, add, edit, and delete face pictures.
3. If the node isSupportFDLib is returned and its value is "false", it indicates that the device does not support face picture management.

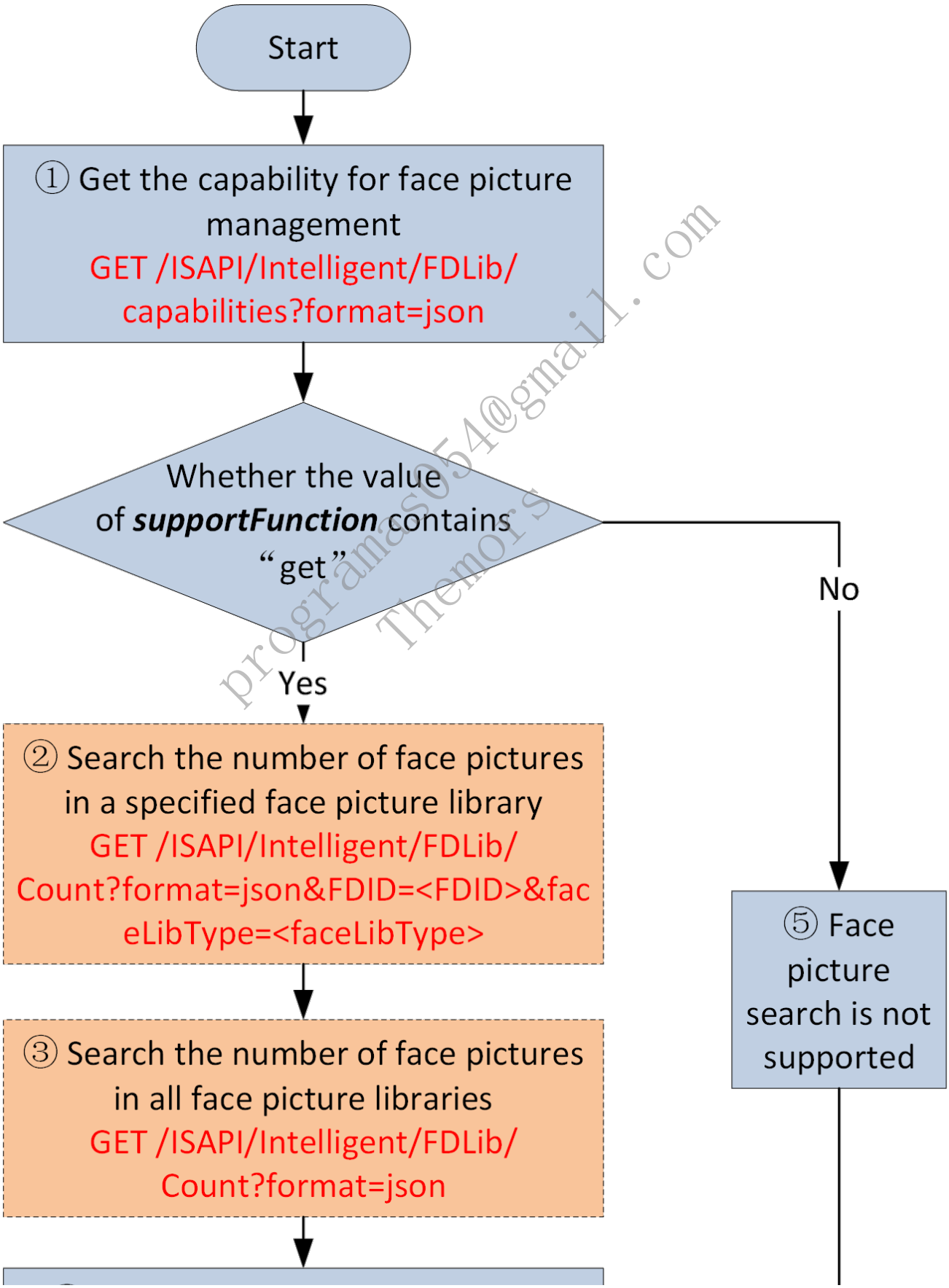
Note:

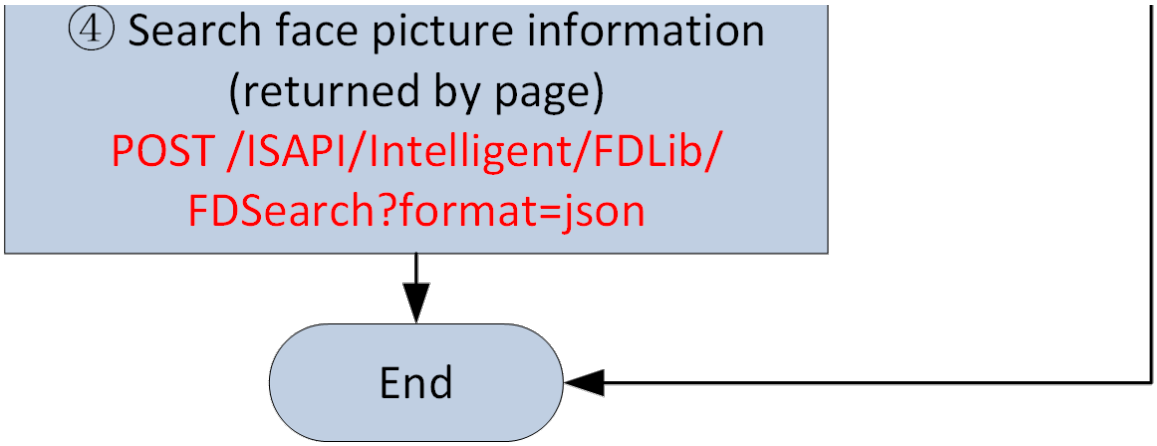
- Before applying, adding, or editing face picture information on the device, make sure that the related person information linked to the person ID has been applied to the device, and make sure that the device has its face picture library by calling `GET /ISAPI/Intelligent/FDLib?format=json` (if the device has no face picture library, then create the face picture library by calling `POST /ISAPI/Intelligent/FDLib?format=json`), and the ID of the library of face pictures captured in visible light (FDID) is 1.
- If the value of the node mode returned by calling `GET /ISAPI/AccessControl/FaceRecognizeMode/capabilities?format=json` contains "deepMode", it indicates that the device supports the deep mode, which compares face

pictures captured in infrared light. For devices which support the deep mode, if the face picture library ID (FDID) is 2, face pictures captured in infrared light will be applied to the face picture library and be used for face picture comparison; if the face picture library ID (FDID) is 1, face pictures captured in visible light will be applied to the face picture library and be displayed on the device.

- Switch between the deep mode and normal mode: `[GET/PUT] /ISAPI/AccessControl/FaceRecognizeMode?format=json`; the modes can be switched via the node mode.

7.8.2.2 Face Picture Search





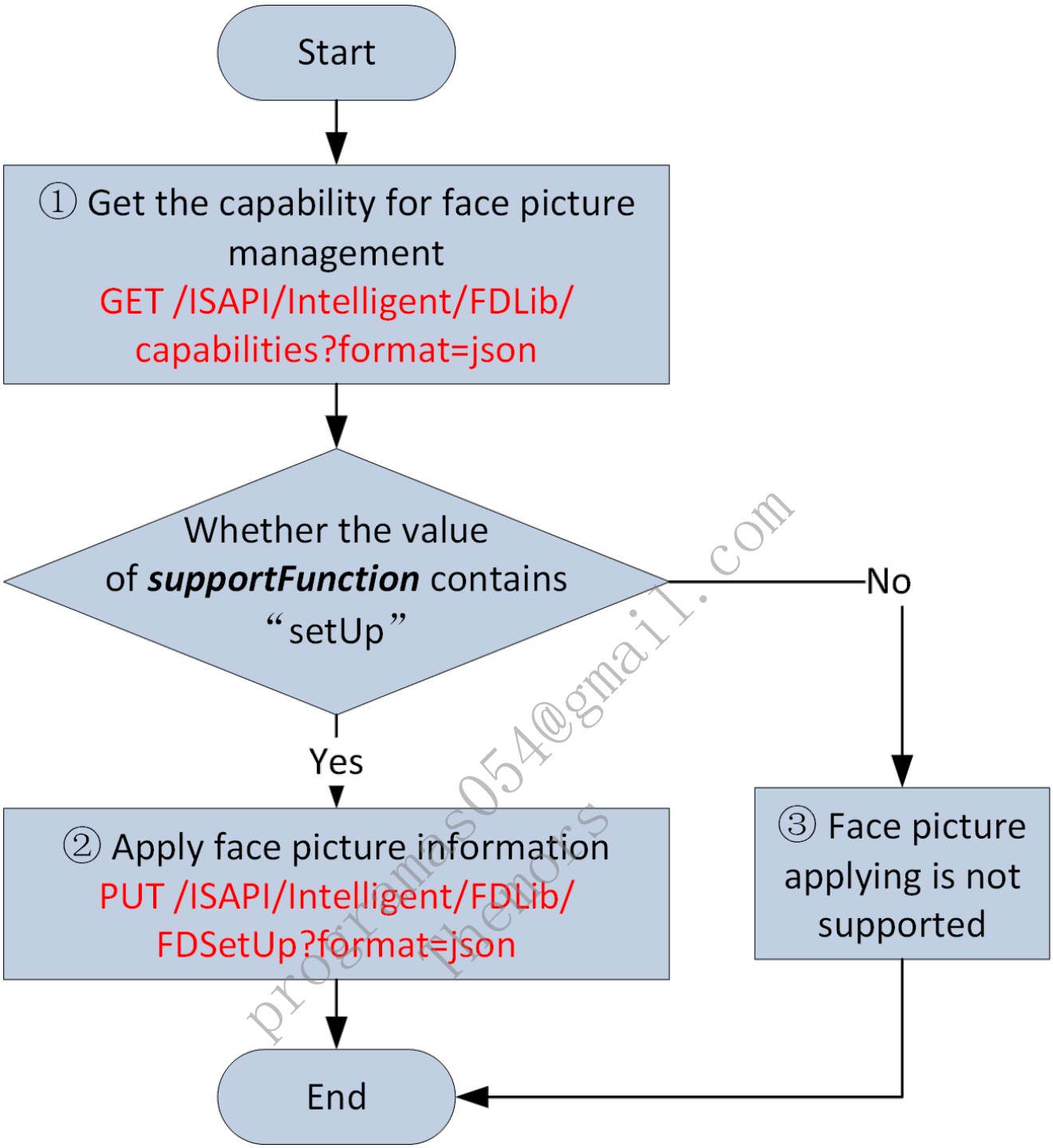
The face picture search function is for searching the number of face pictures and face picture information added to the device.

1. Check whether the device supports face picture search: `GET /ISAPI/Intelligent/FDLib/capabilities?format=json`; if the value of the node supportFunction contains "get", it indicates that the device supports face picture search.
2. Search the number of face pictures in the specified face picture libraries: `GET /ISAPI/Intelligent/FDLib/Count?format=json&FDID=<FDID>&faceLibType=<faceLibType>`; the returned value of the node recordDataNumber is the number of the added face pictures of the specified face picture libraries.
3. Search the number of face pictures in all face picture libraries: `GET /ISAPI/Intelligent/FDLib/Count?format=json`; the returned value of the node recordDataNumber is the number of face pictures in all face picture libraries.
4. Search face picture information: `POST /ISAPI/Intelligent/FDLib/FDSearch?format=json`; the face picture information is returned by page.
5. If the value of the node supportFunction does not contain "get", it indicates that the device does not support face picture search.

Note:

The value of the node FDRecordDataMaxNum returned by calling `GET /ISAPI/Intelligent/FDLib/capabilities?format=json` is the maximum number of face pictures supported by the device.

7.8.2.3 Face Picture Applying



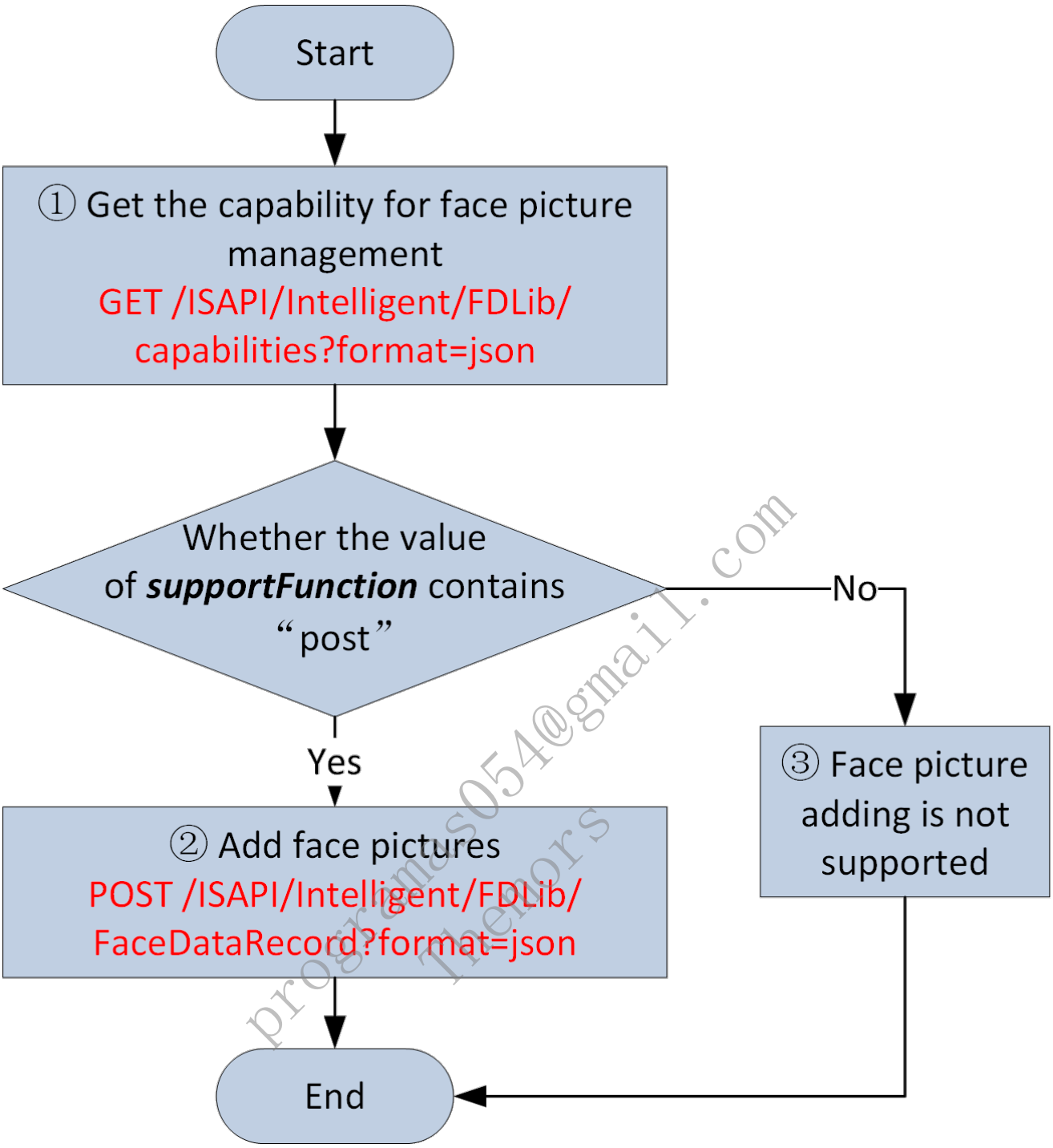
Face picture information can be applied to the device via the face picture applying function. If the face picture has been added to the device, the face picture information will be edited; if the face picture has not been added to the device, the face picture information will be added to the device.

1. Check whether the device supports face picture applying: `GET /ISAPI/Intelligent/FDLib/capabilities?format=json`; if the value of the node supportFunction contains "setUp", it indicates that the device supports face picture applying.
2. Apply face picture information: `PUT /ISAPI/Intelligent/FDLib/FDSetUp?format=json`.
3. If the value of the node supportFunction does not contain "setUp", it indicates that the device does not support face picture applying.

Note:

Check whether the face picture has been added to the device via the node FPID returned by calling the API for face picture applying, and link the face picture to the person information via the node FPID in face picture management and the node employeeNo in person management.

7.8.2.4 Face Picture Adding



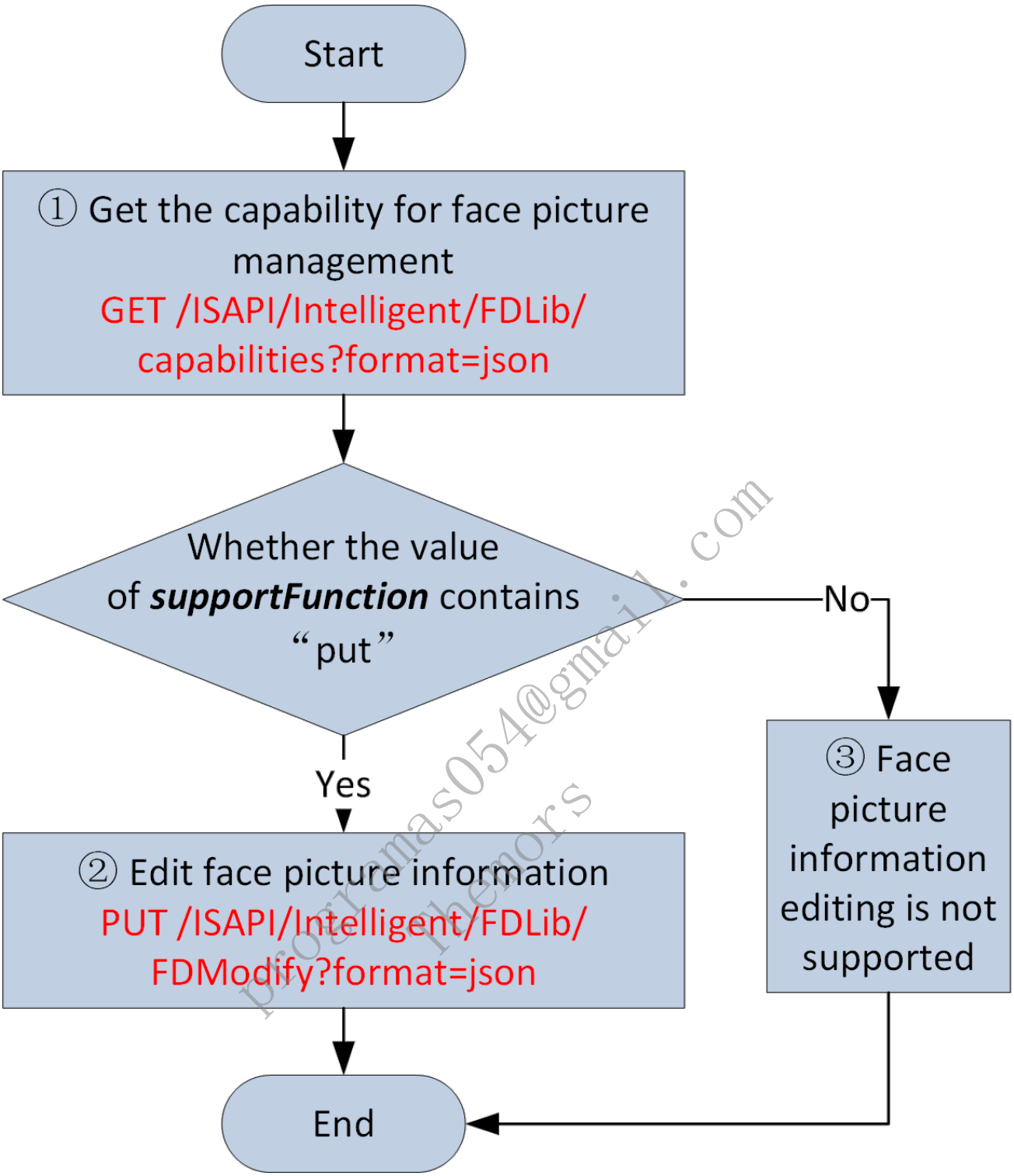
Face picture information can be added to the device via the face picture adding function. If the face picture has been added to the device, the device will report an error; if the face picture has not been added to the device, the face picture information will be added to the device.

1. Check whether the device supports face picture adding: `GET /ISAPI/Intelligent/FDLib/capabilities?format=json`; if the value of the node supportFunction contains "post", it indicates that the device supports face picture adding.
2. Add face picture information: `POST /ISAPI/Intelligent/FDLib/FaceDataRecord?format=json`.
3. If the value of the node supportFunction does not contain "post", it indicates that the device does not support face picture adding.

Note:

Check whether the face picture has been added to the device via the node FPID returned by calling the API for face picture adding, and link the face picture to the person information via the node FPID in face picture management and the node employeeNo in person management.

7.8.2.5 Face Picture Information Editing



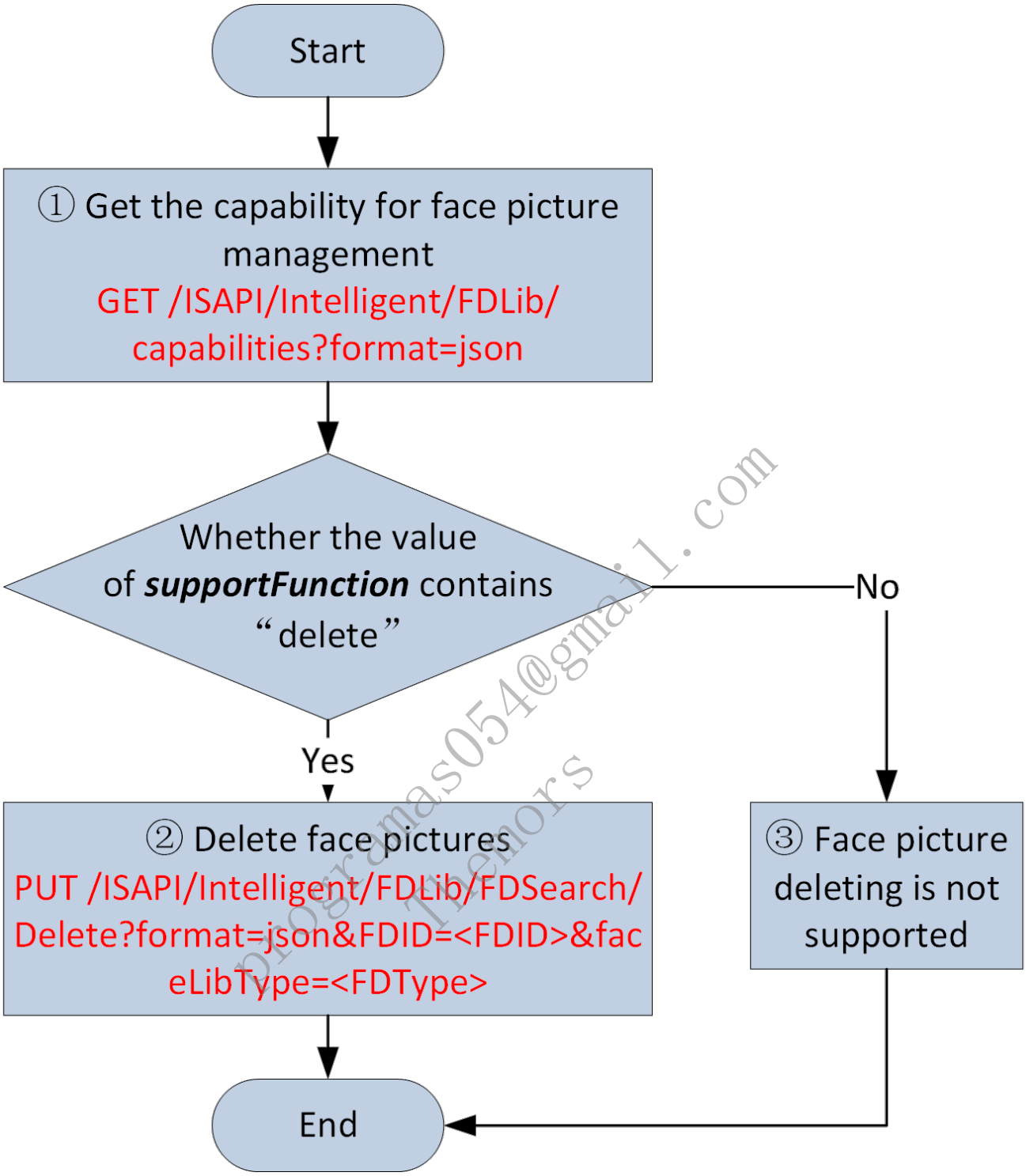
Face picture information on the device can be edited via the face picture information editing function. If the face picture has been added to the device, the face picture information will be edited; if the face picture has not been added to the device, the device will report an error.

1. Check whether the device supports face picture information editing: `GET /ISAPI/Intelligent/FDLib/capabilities?format=json`; if the value of the node supportFunction contains "put", it indicates that the device supports face picture information editing.
2. Edit face picture information: `PUT /ISAPI/Intelligent/FDLib/FDModify?format=json`.
3. If the value of the node supportFunction does not contain "put", it indicates that the device does not support face picture information editing.

Note:

Check whether the face picture has been added to the device via the node FPID returned by calling the API for face picture information editing, and link the face picture to the person information via the node FPID in face picture management and the node employeeNo in person management.

7.8.2.6 Face Picture Deleting



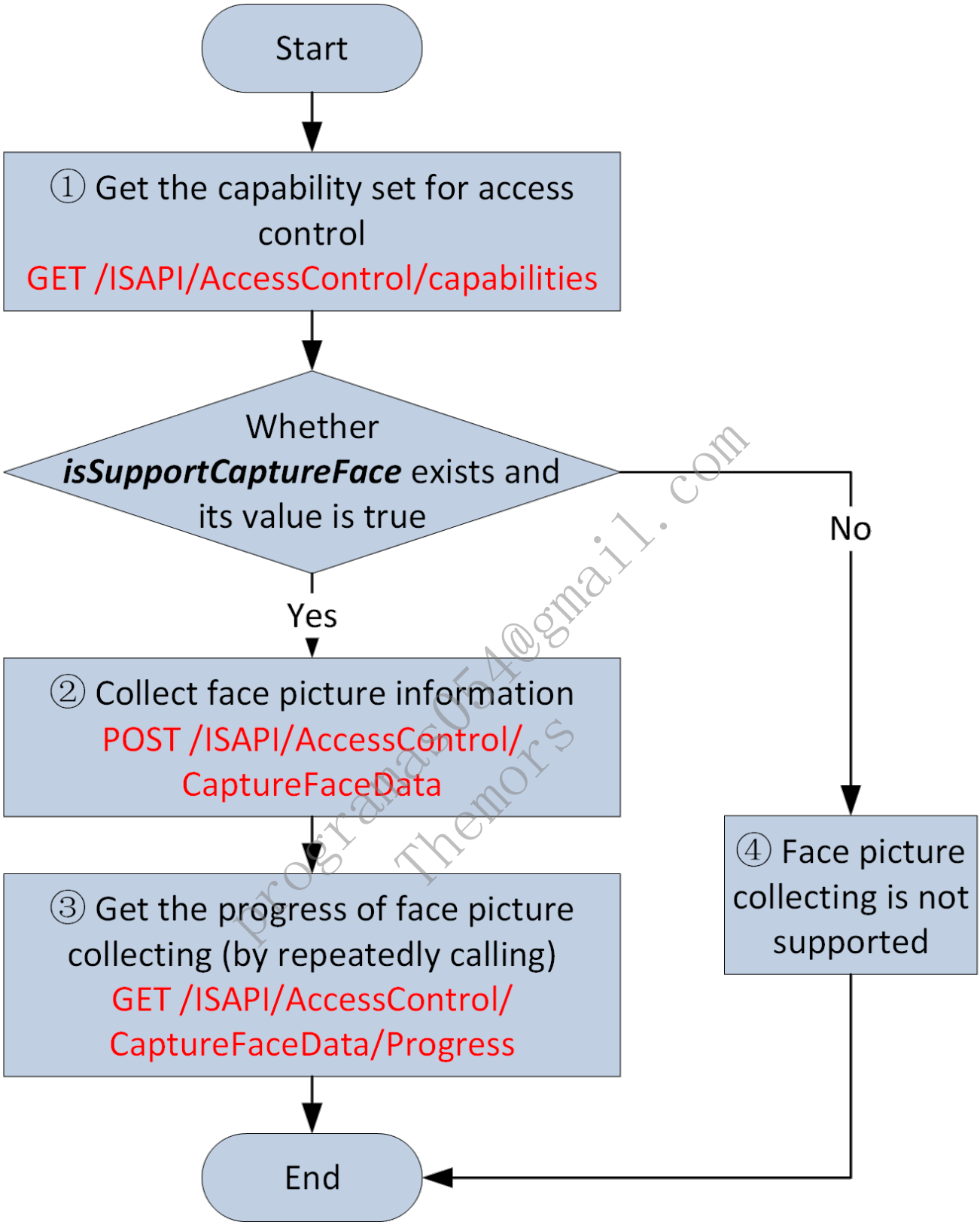
The face picture information on the device can be deleted via the face picture deleting function. The device will not report an error if the face picture to be deleted is not added to the device.

1. Check whether the device supports face picture deleting: `GET /ISAPI/Intelligent/FDLib/capabilities?format=json`; if the value of the node supportFunction contains "delete", it indicates that the device supports face picture deleting.
2. Delete face pictures: `PUT /ISAPI/Intelligent/FDLib/FDSearch/Delete?format=json&FDID=<FDID>&faceLibType=<FDType>`; if calling succeeded, it indicates that the device has deleted the face pictures.
3. If the value of the node supportFunction does not contain "delete", it indicates that the device does not support face picture deleting.

Note:

All the face picture libraries and the face picture information in the libraries on the device can be deleted by calling `DELETE /ISAPI/Intelligent/FDLib?format=json`.

7.8.2.7 Face Picture Collecting



Face picture data, face picture quality grades, etc., can be collected via the face picture collecting function.

1. Check whether the device supports face picture collecting: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportCaptureFace is returned and its value is "true", it indicates that the device supports face picture (captured in visible light) collecting. If the node isSupportCaptureInfraredFace is returned and its value is "true", it indicates that the device supports face picture (captured in infrared light) collecting.
2. Collect face picture information: `POST /ISAPI/AccessControl/CaptureFaceData`.

- If the node captureProgress is returned, and the value is 100, it indicates that the face picture has been collected, and the binary data or URL of the collected face picture will be parsed.
- If the node captureProgress is returned and the value is 0, it indicates that the face picture has not been collected,

and you need to get the progress of face picture collecting.

3. Get the progress of face picture collecting: `GET /ISAPI/AccessControl/CaptureFaceData/Progress`; repeatedly call this API to get the progress of face picture collecting.

- Repeatedly call this API until the node captureProgress is returned and its value is 100, which indicates that the face picture has been collected and the binary data and URL of the face picture will be parsed.
- If the value of the node captureProgress is 0 and the value of the isCurRequestOver is true, which indicates that the face picture collecting failed, stop calling the API.

4. If the node isSupportCaptureFace is returned and its value is "false", it indicates that the device does not support face picture collecting.

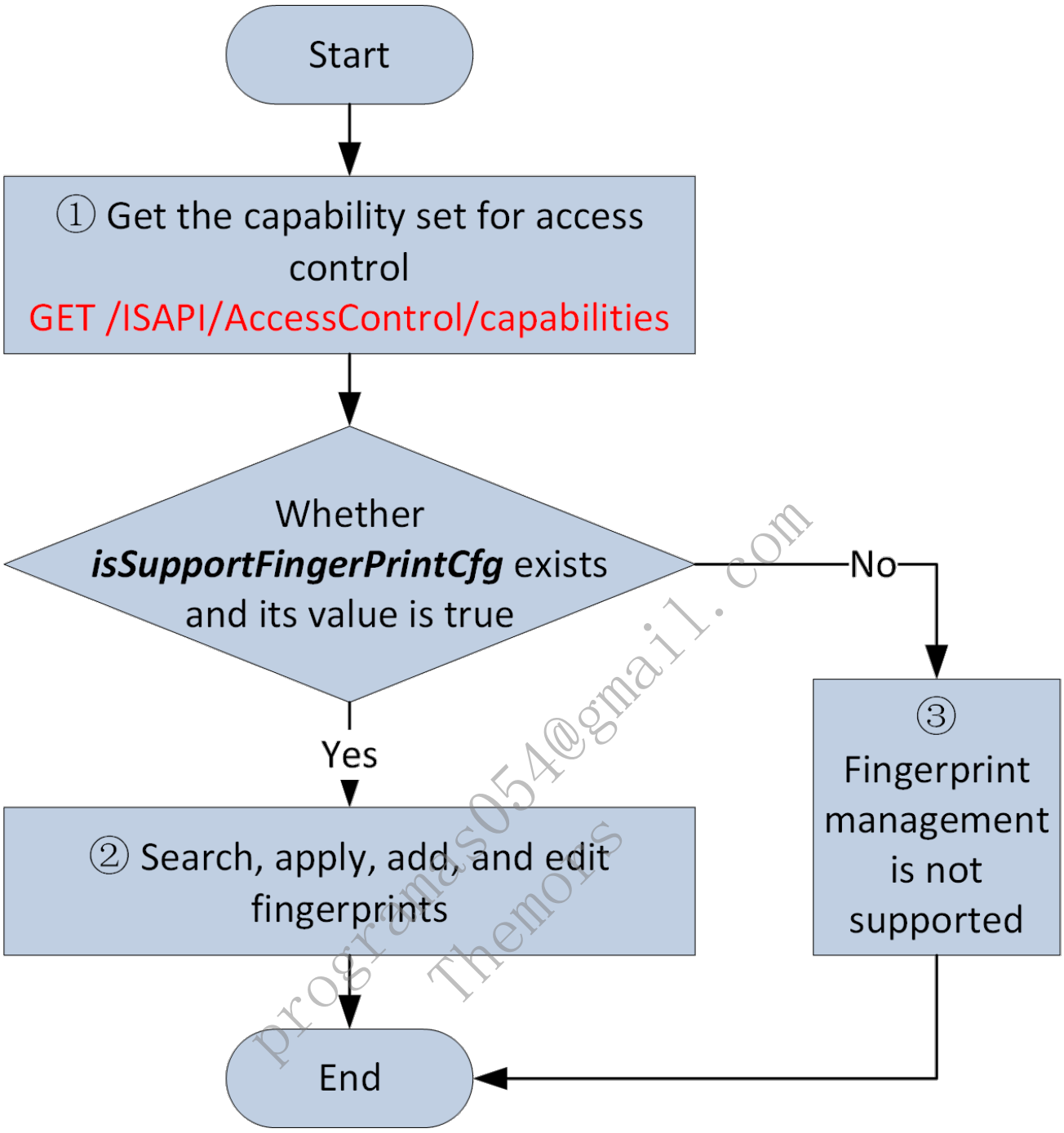
7.9 Fingerprint Management

7.9.1 Introduction to the Function

Fingerprint management includes searching, applying, deleting, and collecting fingerprints.

7.9.2 API Calling Flow

7.9.2.1 Check Whether the Device Supports Fingerprint Management



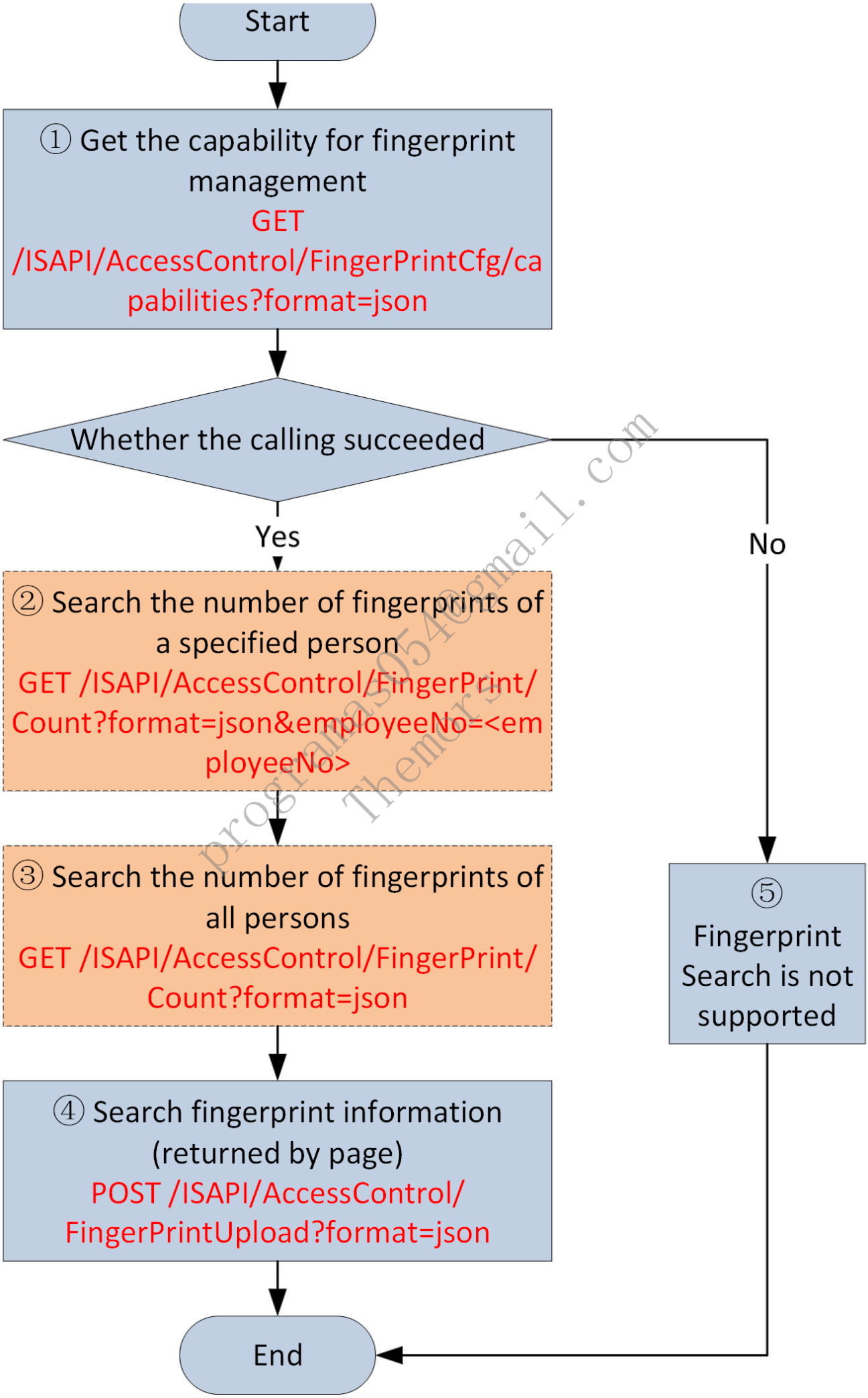
Before calling the API for fingerprint management, make sure that the device supports fingerprint management.

1. Check whether the device supports fingerprint management: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportFingerPrintCfg is returned and its value is "true", it indicates that the device supports fingerprint management.
2. Search, apply, add, and edit fingerprints.
3. If the node isSupportFingerPrintCfg is returned and its value is "false", it indicates that the device does not support fingerprint management.

Note:

- Before applying the fingerprint information to the device, make sure that the related person information linked to the person ID has been applied to the device.
- The maximum number of fingerprints that can be applied to the device per person is 10 (the 10 fingerprints of a person).

7.9.2.2 Fingerprint Search


Start
① Get the capability for fingerprint management
GET /ISAPI/AccessControl/FingerPrintCfg/capabilities?format=json
Whether the calling succeeded
Yes
No
② Search the number of fingerprints of a specified person
GET /ISAPI/AccessControl/FingerPrint/Count?format=json&employeeNo=<employeeNo>
③ Search the number of fingerprints of all persons
GET /ISAPI/AccessControl/FingerPrint/Count?format=json
④ Search fingerprint information (returned by page)
POST /ISAPI/AccessControl/FingerPrintUpload?format=json
⑤ Fingerprint Search is not supported
End

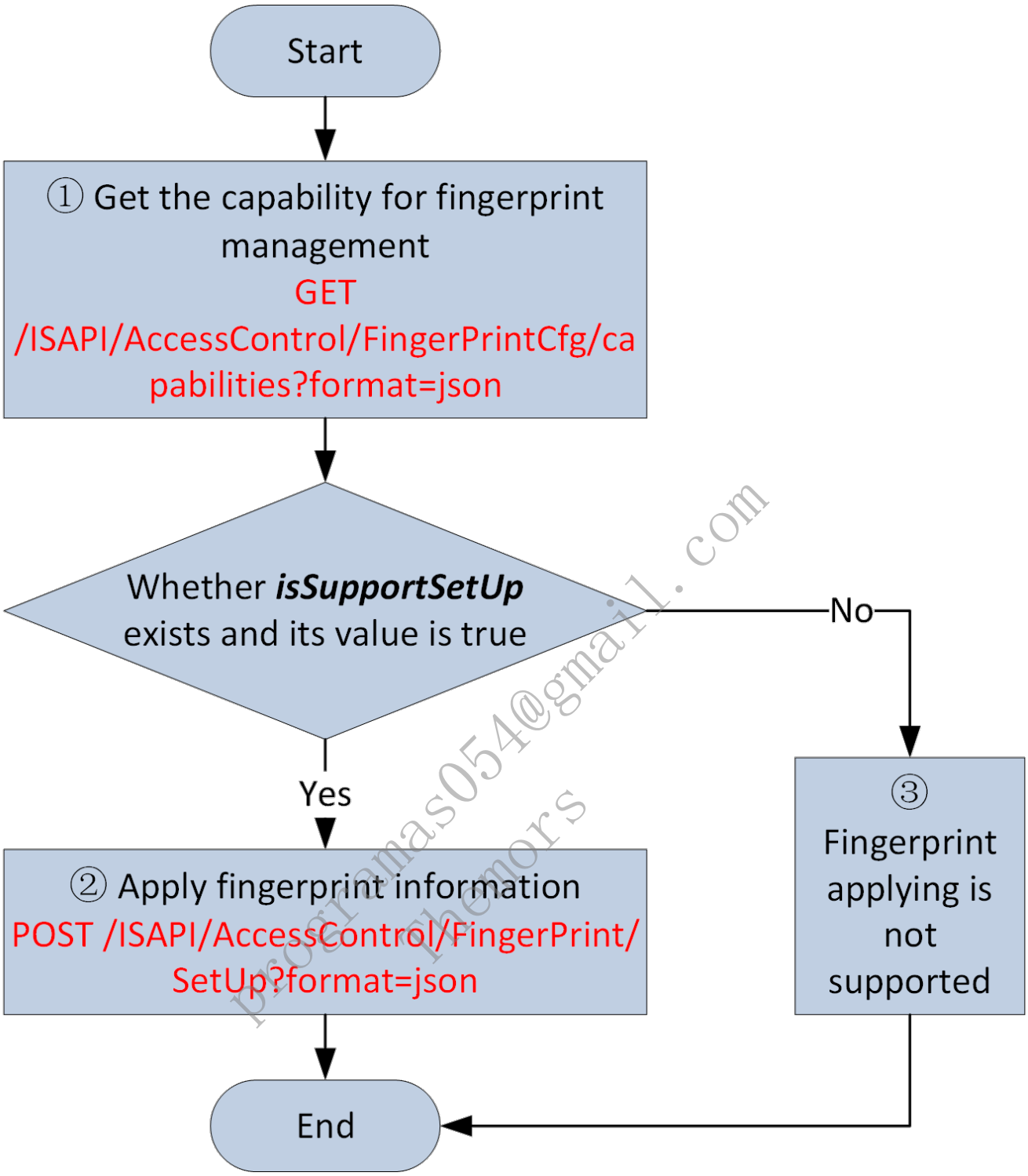
The fingerprint search function is for searching the number of fingerprints and fingerprint information added to the device.

1. Check whether the device supports fingerprint search: `GET /ISAPI/AccessControl/FingerPrintCfg/capabilities?format=json`; if calling succeeded, it indicates that the device supports fingerprint search.
2. Search the number of the specified persons' fingerprints: `GET /ISAPI/AccessControl/FingerPrint/Count?format=json&employeeNo=<employeeNo>`; the returned value of the node numberOfFP is the number of the added fingerprints of the specified persons.
3. Search the number of fingerprints of all persons: `GET /ISAPI/AccessControl/FingerPrint/Count?format=json`; the returned value of the node numberOfFP is the number of the added fingerprints of all persons.
4. Search fingerprint information: `POST /ISAPI/AccessControl/FingerPrintUpload?format=json`; the fingerprint information is returned by page. If the value of the child node status of the node FingerPrintInfo is "NoFP", it indicates that all fingerprint information are returned.
5. If calling failed, it indicates that the device does not support fingerprint search.

Note:

- The value of the node fingerPrintCapacity returned by calling `GET /ISAPI/AccessControl/CardReaderCfg/<cardReaderID>?format=json` is the maximum number of fingerprints supported by the card reader.
- The value of the node fingerPrintNum returned by calling `GET /ISAPI/AccessControl/CardReaderCfg/<cardReaderID>?format=json` is the number of fingerprints added to the card reader.

7.9.2.3 Fingerprint Applying



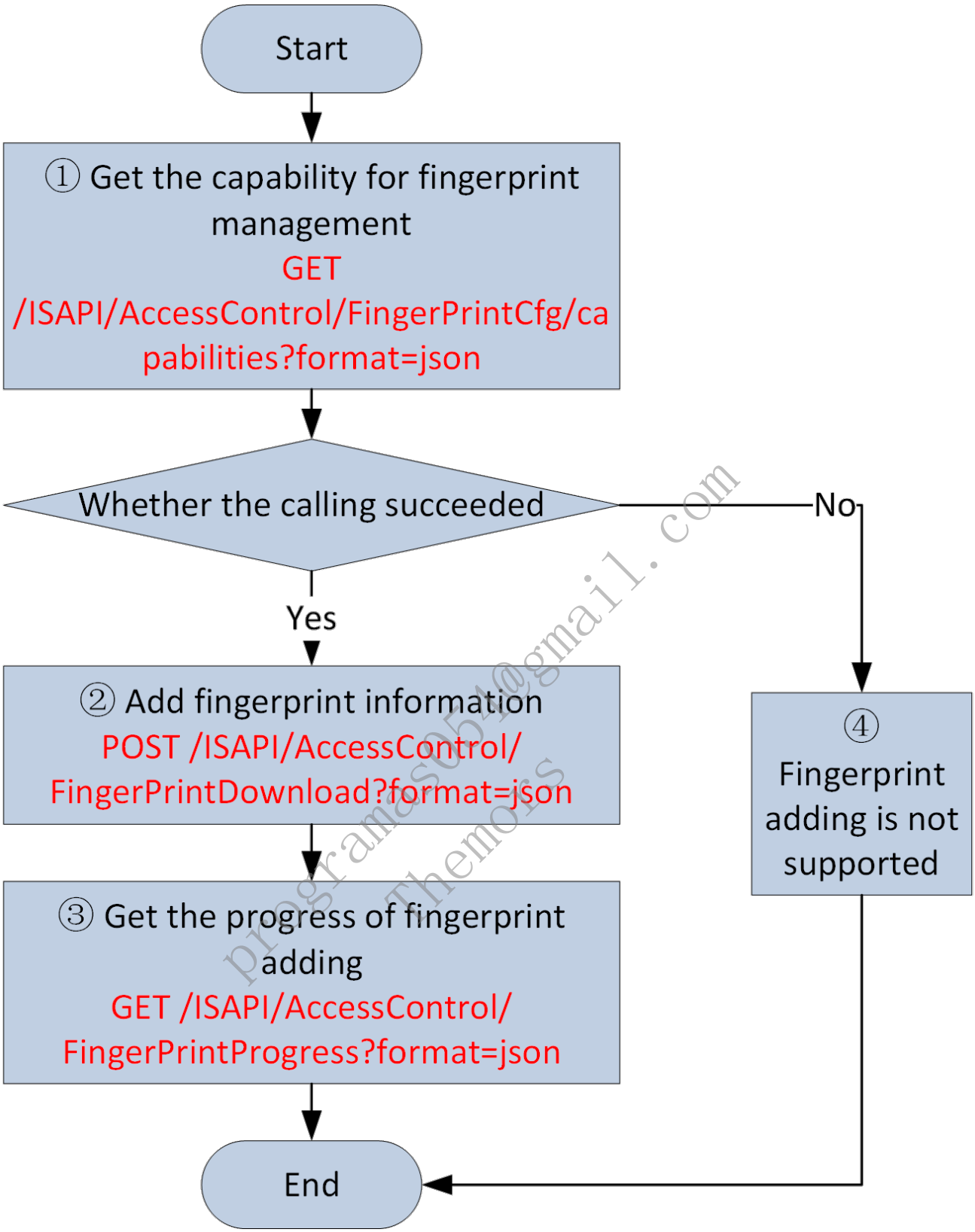
Fingerprint information can be applied to the device via the fingerprint applying function. If the fingerprint has been added to the device, the fingerprint information will be edited; if the fingerprint has not been added to the device, the fingerprint will be added to the device.

1. Check whether the device supports fingerprint applying: `GET /ISAPI/AccessControl/FingerPrintCfg/capabilities?format=json`; if the node isSupportSetUp is returned and its value is "true", it indicates that the device supports fingerprint applying.
2. Apply fingerprint information: `POST /ISAPI/AccessControl/FingerPrint/SetUp?format=json`.
3. If the node isSupportSetUp is returned and its value is false, it indicates that the device does not support fingerprint applying.

Note:

Check whether the fingerprint has been added to the device via the nodes employeeNo and fingerPrintID returned after calling the API for fingerprint applying.

7.9.2.4 Fingerprint Adding



Fingerprint information can be added to the device via the fingerprint adding function. If the fingerprint has been added to the device, the device will report an error; if the fingerprint has not been added to the device, the fingerprint will be added to the device.

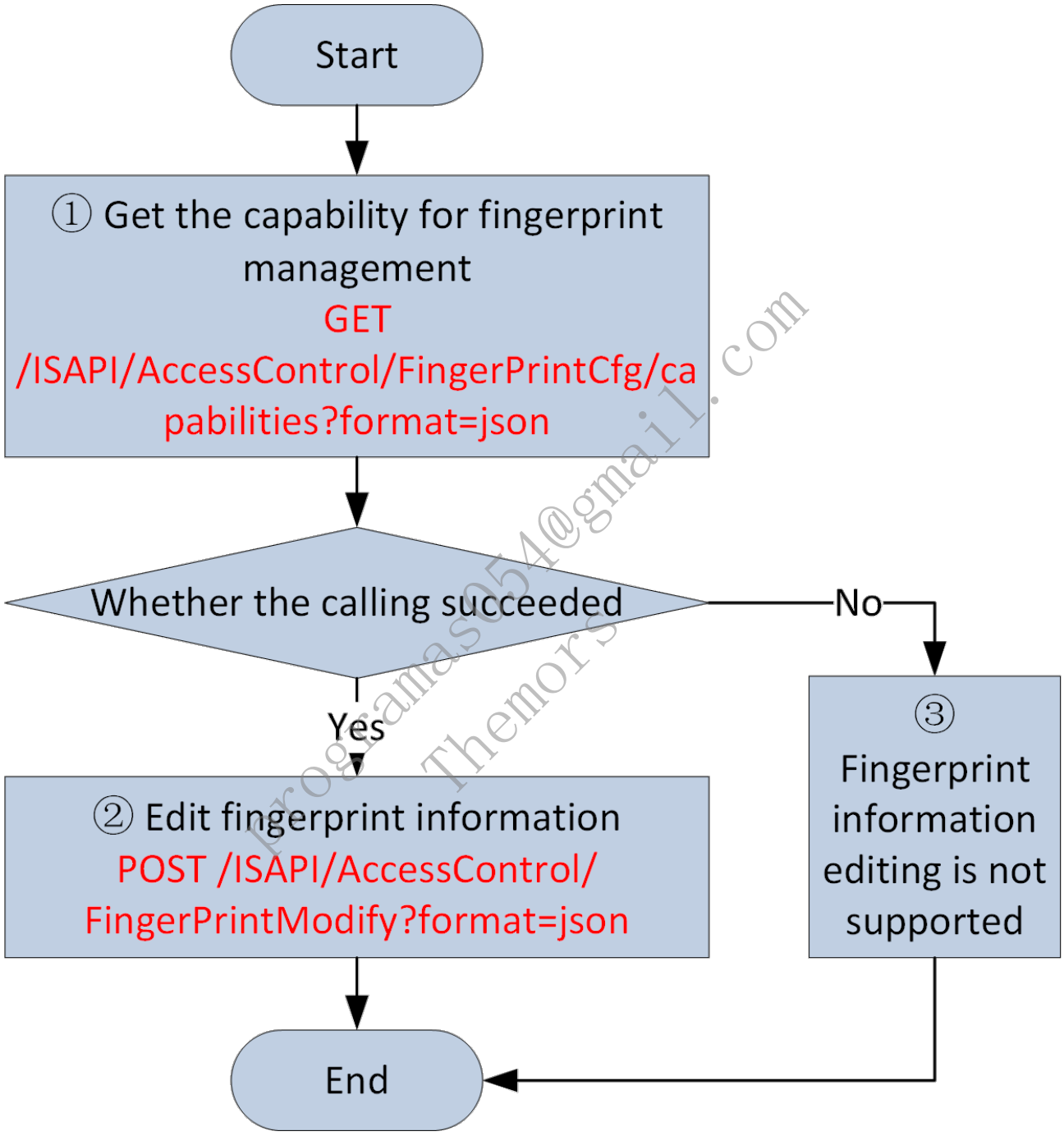
1. Check whether the device supports fingerprint adding: `GET /ISAPI/AccessControl/FingerPrintCfg/capabilities?format=json`; if calling succeeded, it indicates that the device supports fingerprint adding.
2. Add fingerprint information: `POST /ISAPI/AccessControl/FingerPrintDownload?format=json`; if calling succeeded, it indicates that the device has started to execute fingerprint adding, but it does not indicate that the device has added the fingerprint.
3. Get the progress of fingerprint adding: `GET /ISAPI/AccessControl/FingerPrintProgress?format=json`; repeatedly call this API to get the progress of fingerprint adding.

4. If calling failed, it indicates that the device does not support fingerprint adding.

Note:

Check whether the fingerprint has been added to the device via the nodes employeeNo and fingerPrintID returned after calling the API for fingerprint adding.

7.9.2.5 Fingerprint Information Editing



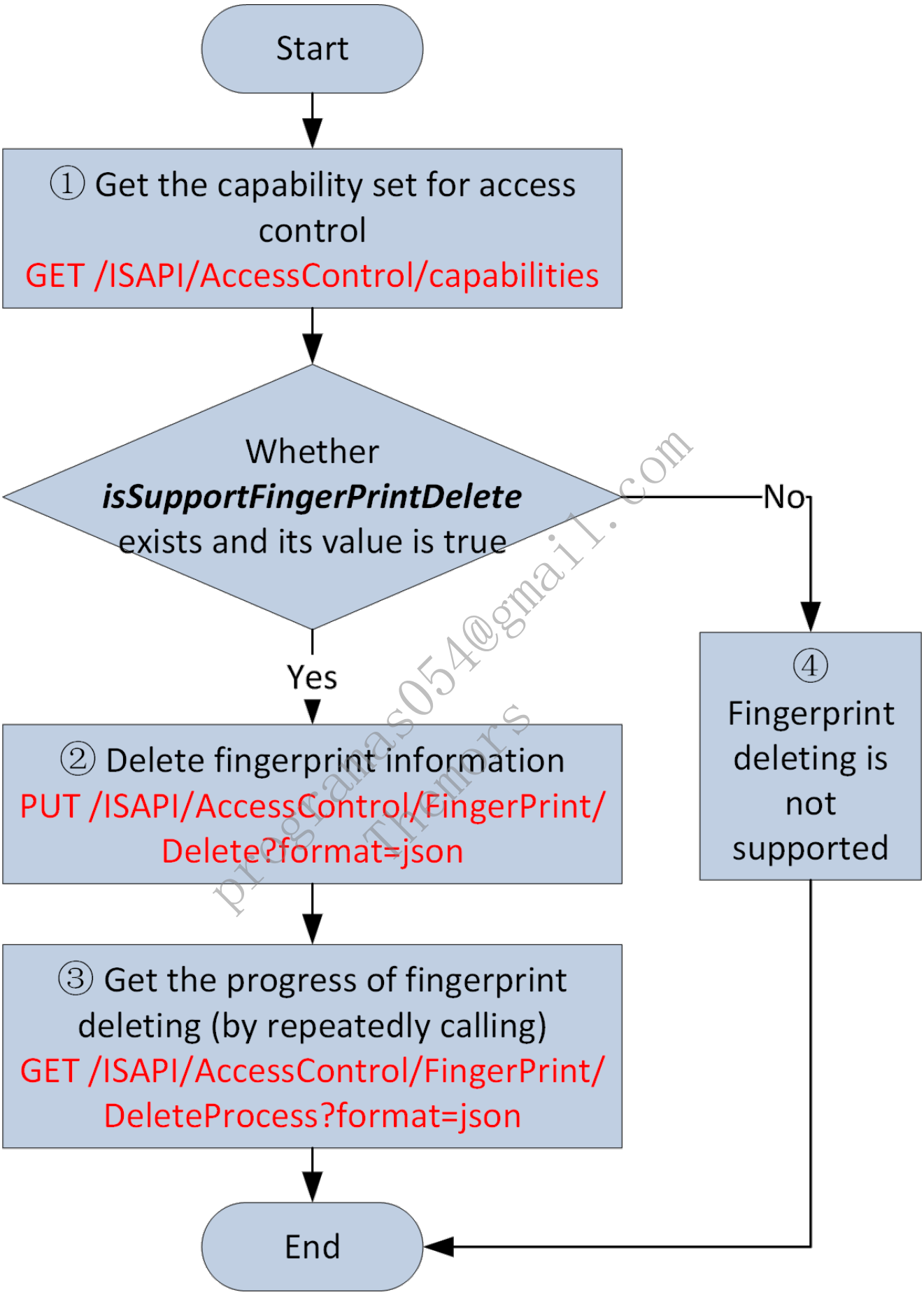
The fingerprint information on the device can be edited via the fingerprint information editing function. If the fingerprint has been added to the device, the fingerprint information will be edited; if the fingerprint has not been added to the device, the device will report an error.

1. Check whether the device supports fingerprint information editing: `GET /ISAPI/AccessControl/FingerPrintCfg/capabilities?format=json`; if calling succeeded, it indicates that the device supports fingerprint information editing.
2. Edit fingerprint information: `POST /ISAPI/AccessControl/FingerPrintModify?format=json`.
3. If calling failed, it indicates that the device does not support fingerprint information editing.

Note:

- Check whether the fingerprint has been added to the device via the nodes employeeNo and fingerPrintID returned after calling the API for fingerprint information editing.
- When the fingerprint information is edited, only the fingerprint parameters will be edited; the fingerprint data will not be edited.

7.9.2.6 Fingerprint Deleting

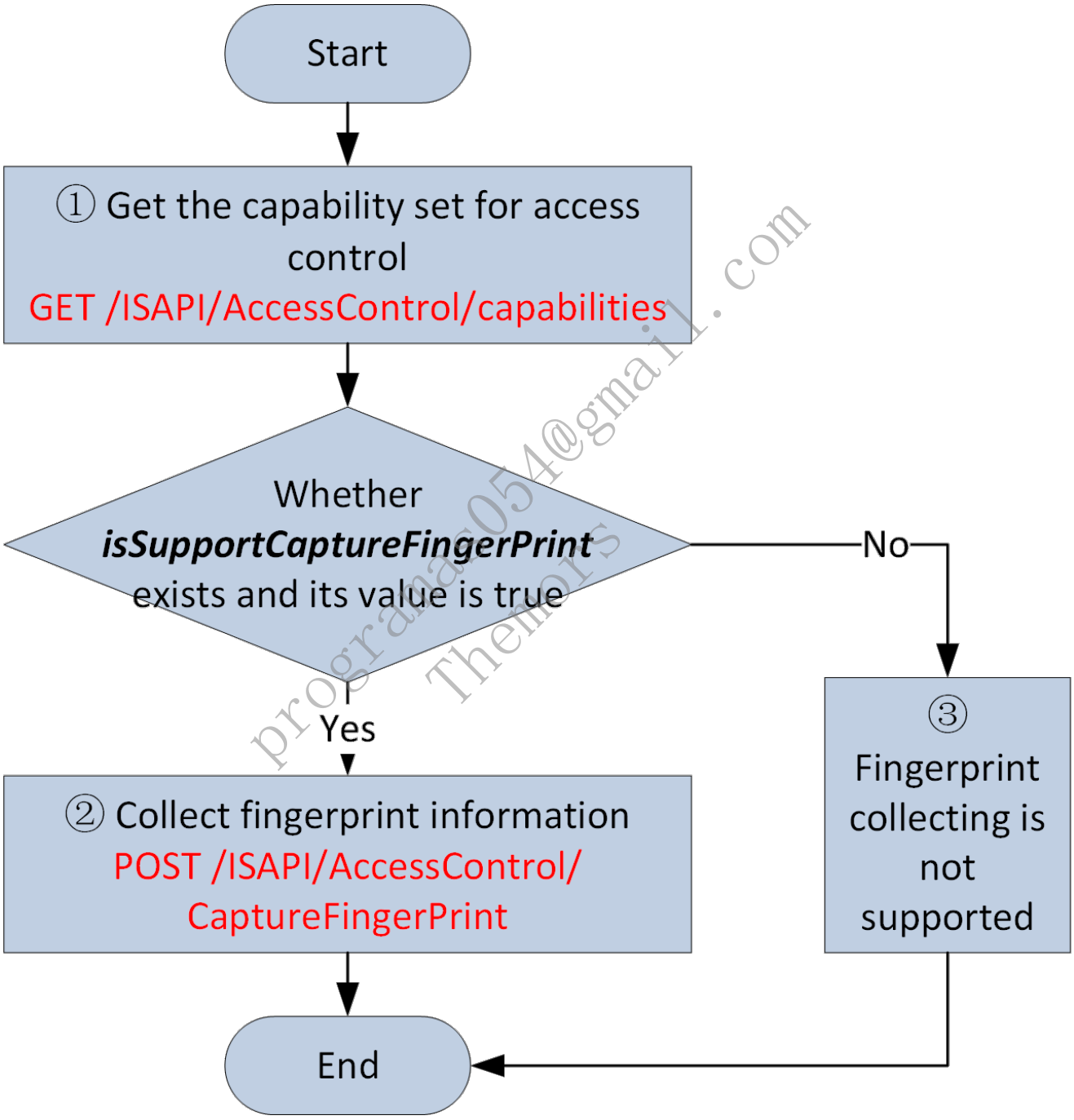


The fingerprint information on the device can be deleted via the fingerprint deleting function. The device will not report an error if the fingerprint information to be deleted is not added to the device.

1. Check whether the device supports fingerprint deleting: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportFingerPrintDelete is returned and its value is "true", it indicates that the device supports fingerprint deleting.

2. Delete fingerprint information: `PUT /ISAPI/AccessControl/FingerPrint/Delete?format=json`; if calling succeeded, it indicates that the device has started to execute fingerprint deleting, but it does not indicate that the device has deleted the fingerprints.
3. Get the progress of fingerprint deleting: `GET /ISAPI/AccessControl/FingerPrint/DeleteProcess?format=json`; repeatedly call this API to get the progress of fingerprint deleting.
4. If the node isSupportFingerPrintDelete is returned and its value is "false", it indicates that the device does not support fingerprint deleting.

7.9.2.7 Fingerprint Collecting



The fingerprint collecting function is for collecting the fingerprint data, fingerprint quality, etc.

1. Check whether the device supports fingerprint collecting: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportCaptureFingerPrint is returned and its value is "true", it indicates that the device supports fingerprint collecting.
2. Collect fingerprint information: `POST /ISAPI/AccessControl/CaptureFingerPrint`.
3. If the node isSupportCaptureFingerPrint is returned and its value is "false", it indicates that the device does not support fingerprint collecting.

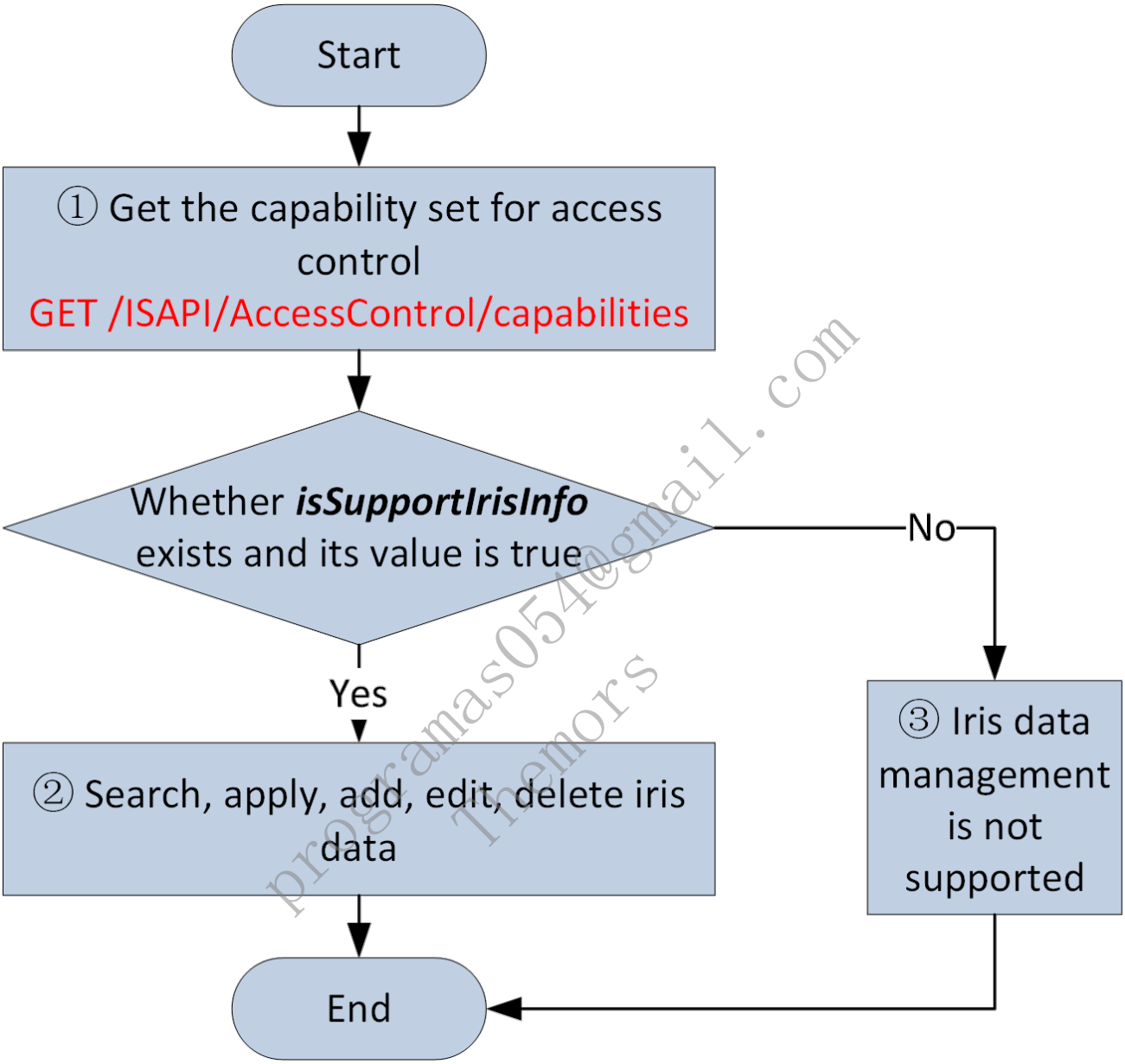
7.10 Iris Data Management

7.10.1 Introduction to the Function

Iris data management includes searching, applying, adding, editing, deleting, and collecting iris data.

7.10.2 API Calling Flow

7.10.2.1 Check Whether the Device Supports Iris Data Management



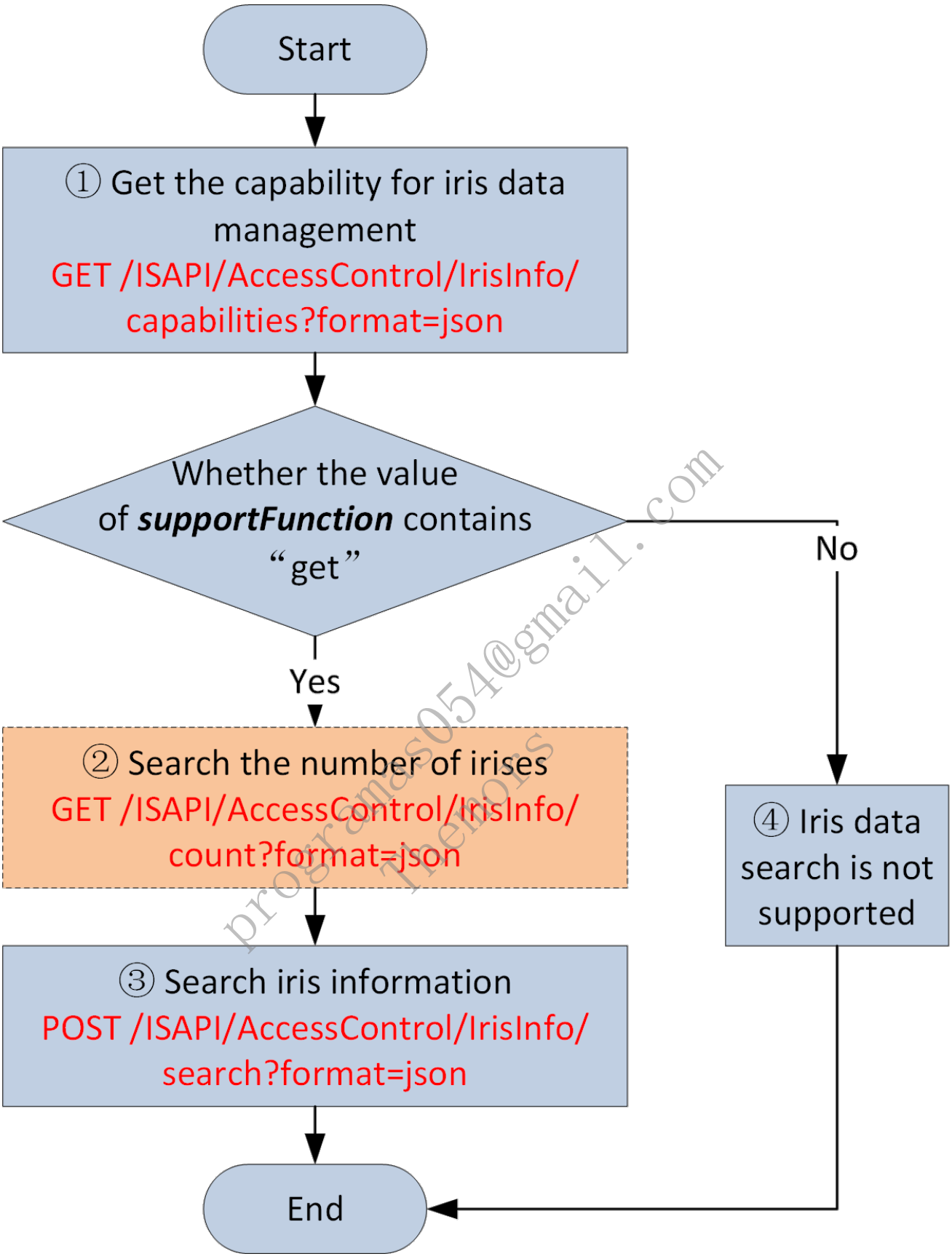
Before calling the API for iris data management, make sure that the device supports iris data management:

1. Check whether the device supports iris data management: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportIrisInfo is returned and its value is true, it indicates that the device supports iris data management.
2. Search, apply, add, edit, and delete iris data.
3. If the node isSupportIrisInfo is returned and its value is false, it indicates that the device does not support iris data management.

Note:

- Before applying, adding, and editing the iris data on the device, make sure that the related person information linked to the person ID has been applied to the device.
- Data for up to two irises of each person (two eyes of each person) can be applied to the device.

7.10.2.2 Iris Data Search



The iris data search function is for searching the number of irises and iris information added to the device.

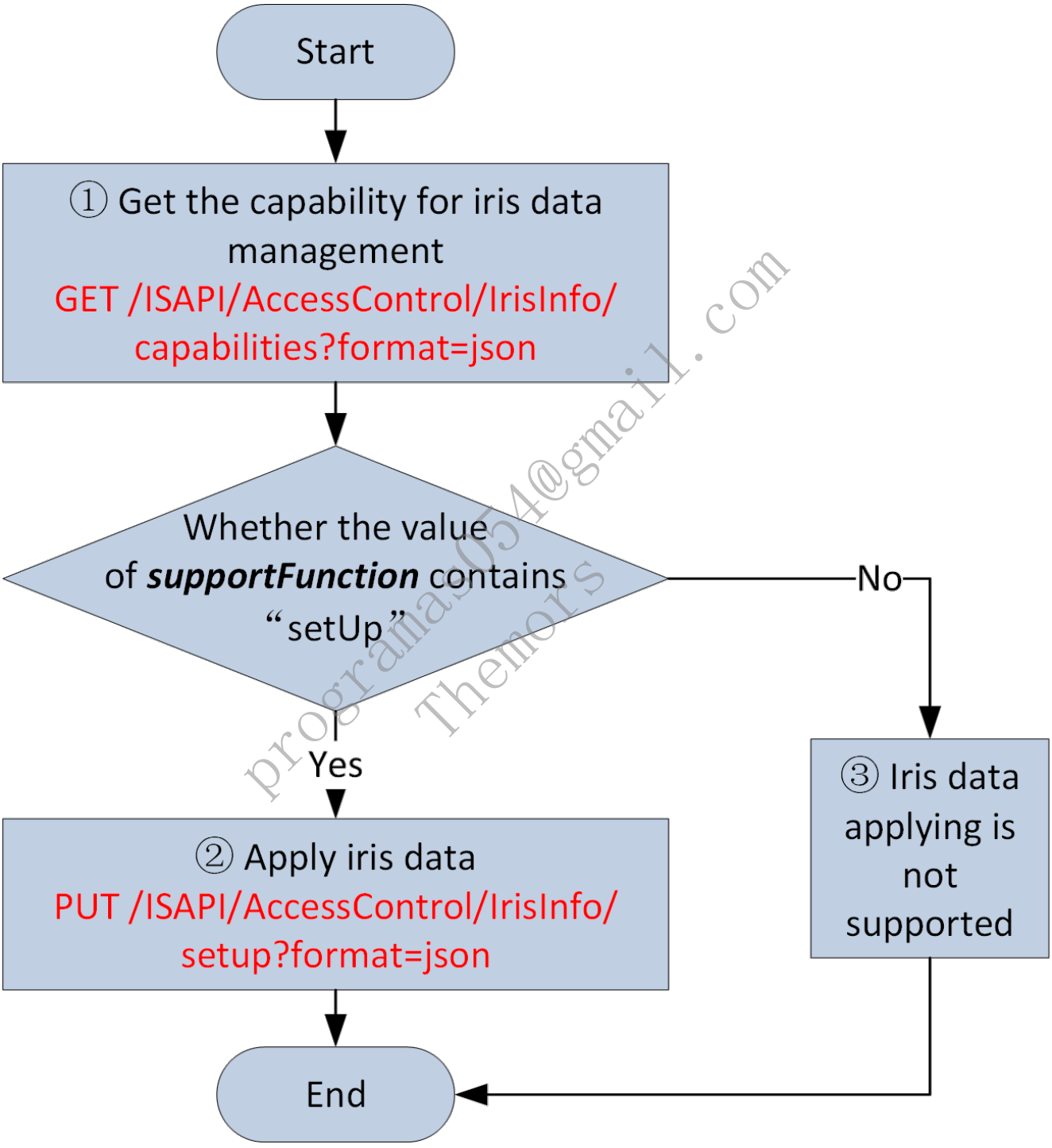
1. Check whether the device supports iris data search: `GET /ISAPI/AccessControl/IrisInfo/capabilities?format=json`; if the node supportFunction is returned, and the value contains "get", it indicates that the device supports iris data search.
2. Search the number of irises: `GET /ISAPI/AccessControl/IrisInfo/count?format=json`; the value of the node IrisNumber is the number of added irises on the device.
3. Search iris information: `POST /ISAPI/AccessControl/IrisInfo/search?format=json`; the searched iris information will be returned by page.

4. If the value of the node supportFunction does not contain “get”, it indicates that the device does not support iris data search.

Note:

The value of the node maxRecordNum returned by calling `GET /ISAPI/AccessControl/IrisInfo/capabilities?format=json` is the maximum number of irises supported by the device.

7.10.2.3 Iris Data Applying



The iris data applying function is for applying iris information to the device. If the iris data has already been applied to the device, the information about the iris will be edited; if the iris data has not been applied to the device, the iris information will be added to the device.

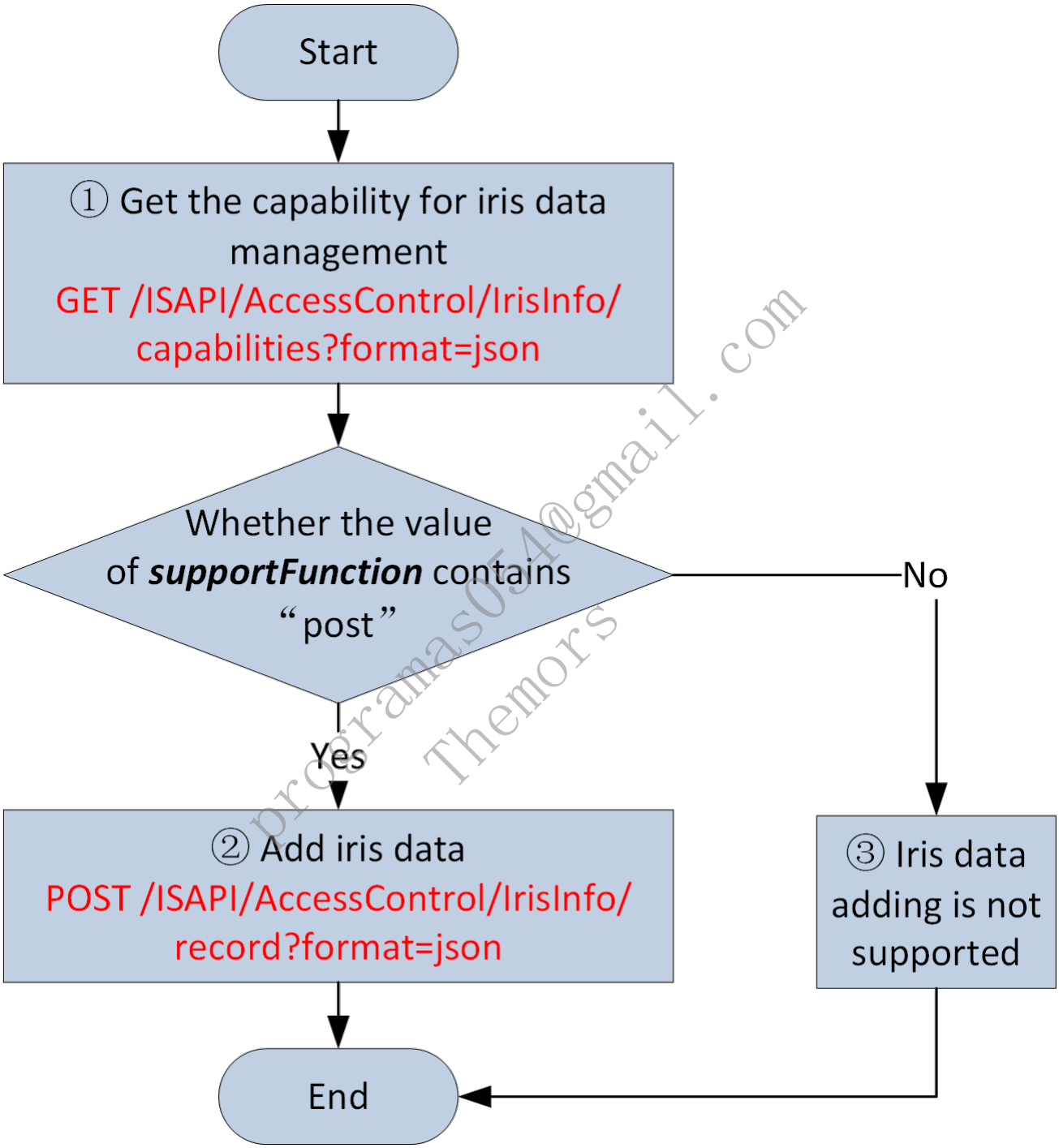
1. Check whether the device supports iris data applying: `GET /ISAPI/AccessControl/IrisInfo/capabilities?format=json`; if the node supportFunction is returned and the value contains “setUp”, it indicates that the device supports iris data applying.
2. Iris Data Applying: `PUT /ISAPI/AccessControl/IrisInfo/setup?format=json`.
3. If the value of the node supportFunction does not contain “setUp”, it indicates that the device does not support iris

data applying.

Note:

Check whether the iris data have been applied to the device via the nodes employeeNo and id returned by calling the API for iris data applying.

7.10.2.4 Iris Data Adding



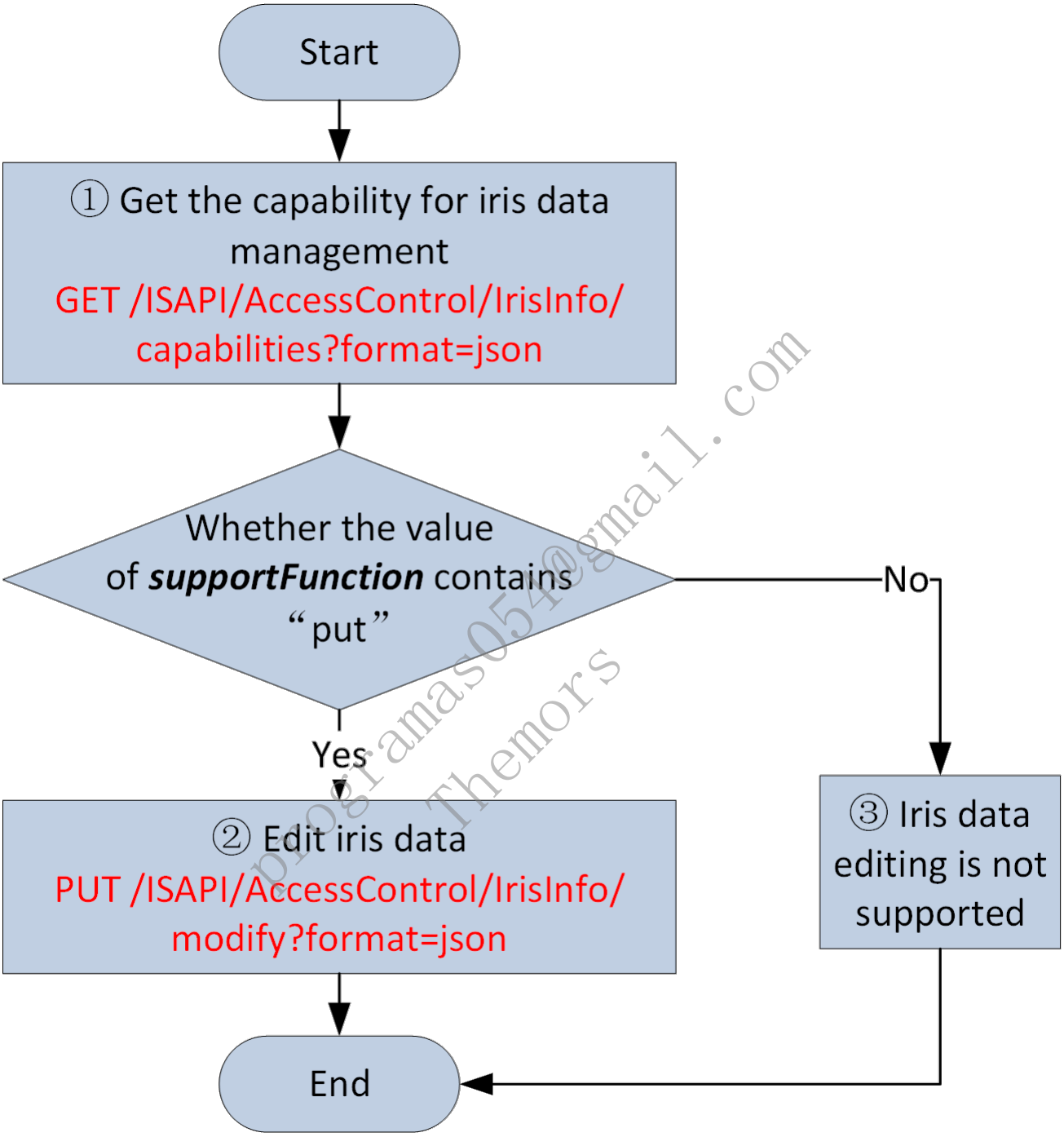
The iris data adding function is for adding the iris data to the device. If the iris data have already been added to the device, the device will report an error; if the iris data have not been added to the device, the iris data will be added to the device.

1. Check whether the device supports iris data adding: `GET /ISAPI/AccessControl/IrisInfo/capabilities?format=json`; if the value of the node supportFunction contains "post", it indicates that the device supports iris data applying.
2. Add iris data: `POST /ISAPI/AccessControl/IrisInfo/record?format=json`.
3. If the value of the node supportFunction does not contain "post", it indicates that the device does not support iris data applying.

Note:

Check whether the iris data have been applied to the device via the nodes employeeNo and id returned by calling the API for iris data adding.

7.10.2.5 Iris Data Editing



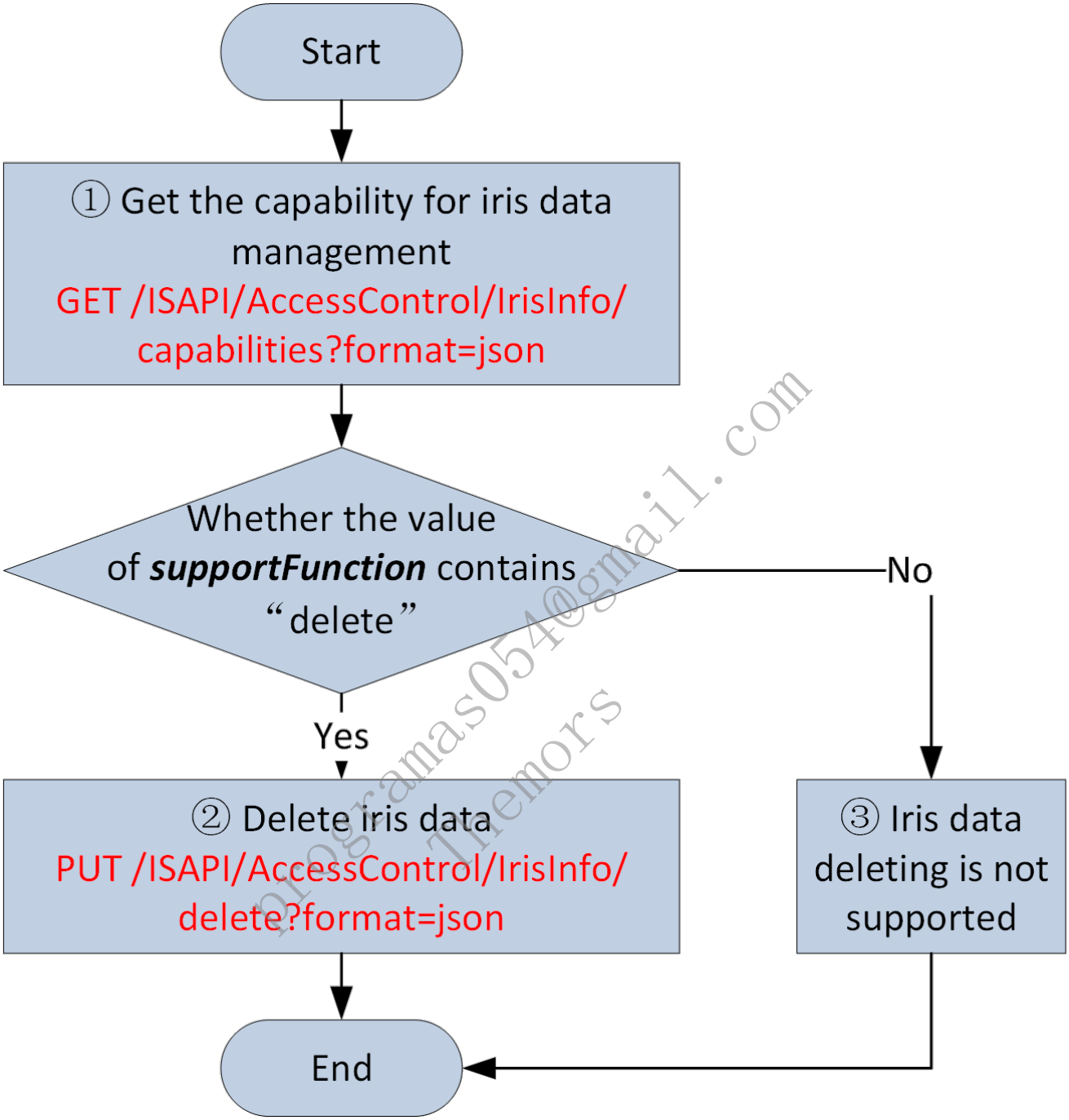
The iris data editing function is for editing the applied iris information on the device. If the iris data have already been added to the device, the iris information will be edited; if the iris data have not been added to the device, the device will report an error.

1. Check whether the device supports iris data editing: `GET /ISAPI/AccessControl/IrisInfo/capabilities?format=json`; if the value of the node supportFunction contains "put", it indicates that the device supports iris data editing.
2. Edit iris information: `PUT /ISAPI/AccessControl/IrisInfo/modify?format=json`.
3. If the value of the node supportFunction does not contain "put", it indicates that the device does not support iris data editing.

Note:

Check whether the iris data have been applied to the device via the nodes employeeNo and id returned by calling the API for iris data editing.

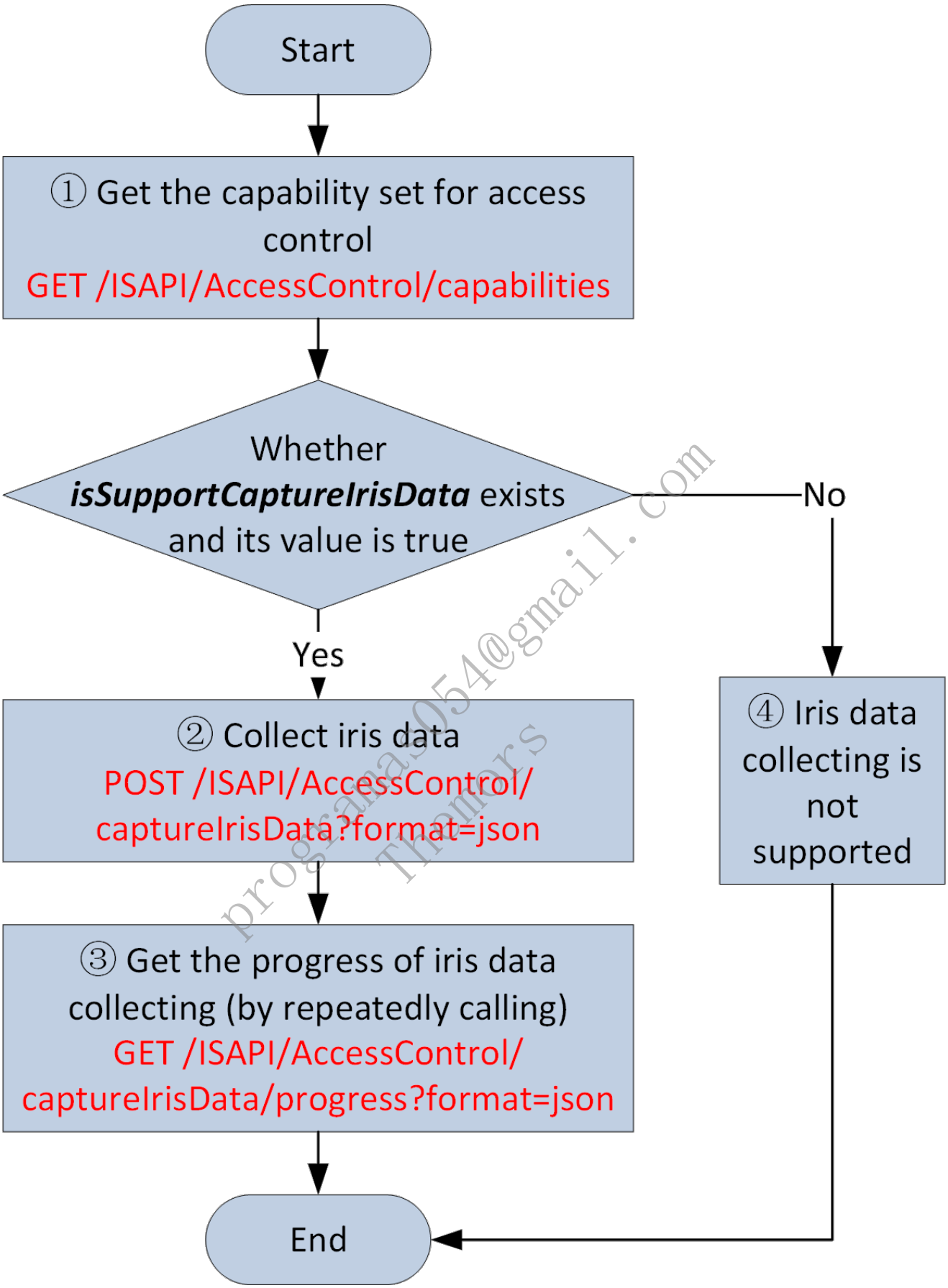
7.10.2.6 Iris Data Deleting



The iris data deleting function is for deleting the applied iris information on the device. If the iris data to be deleted have not been applied to the device, the device will not report an error.

1. Check whether the device supports iris data deleting: `GET /ISAPI/AccessControl/IrisInfo/capabilities?format=json`; if the value of the node supportFunction contains "delete", it indicates that the device supports iris data deleting.
2. Delete iris information: `PUT /ISAPI/AccessControl/IrisInfo/delete?format=json`; if calling succeeded, it indicates that the iris information has been deleted.
3. If the value of the node supportFunction does not contain "delete", it indicates that the device does not support iris data deleting.

7.10.2.7 Iris Data Collecting



The iris data collecting function is for collecting iris data and information.

1. Check whether the device supports iris data collecting: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportCaptureIrisData is returned and its value is "true", it indicates that the device supports iris data collecting.
2. Collect iris information: `POST /ISAPI/AccessControl/captureIrisData?format=json`; if calling succeeded, it indicates that the device has started to execute the collection.
3. Get the progress of iris data collecting: `GET /ISAPI/AccessControl/captureIrisData/progress?format=json`; repeatedly call this API until the value of captureProgress is returned and is 100, which indicates that the collecting

completed.

4. If the node isSupportCaptureIrisData is returned and its value is "false", it indicates that the device does not support iris data collecting.

7.11 Multi-Factor Authentication

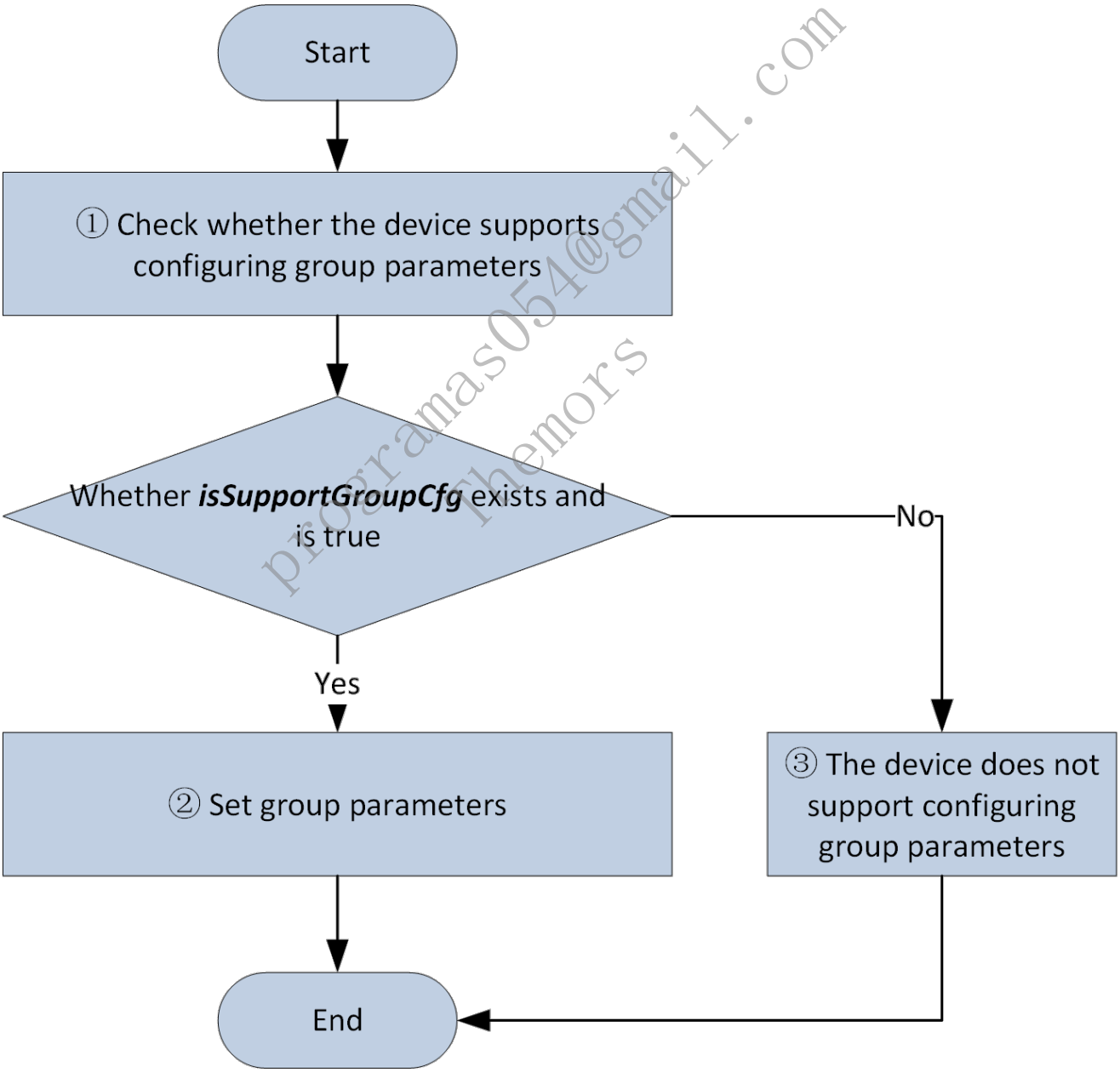
7.11.1 Introduction to the Function

In some scenarios with higher security levels, you can set rules that the door will only open when different persons authenticate in the access control point during fixed time.

For example, in a bank, a door will only open after two or more persons are authenticated (such as swiping card, authenticated by fingerprint, face picture, iris, etc.). If a door is configured multi-factor authentication, the number authentication persons, authentication order, and authentication methods should follow the rules.

7.11.2 API Calling Flow

7.11.2.1 Group Parameter Configuration



The API calling flow is as follow:

1. Check whether the device supports configuring group parameters: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportGroupCfg is returned and its value is "true", it indicates that the device supports configuring group parameters.
2. Set group parameters: `[GET/PUT] /ISAPI/AccessControl/GroupCfg/<groupID>?format=json`.

3. If the node isSupportGroupCfg is returned and its value is "false", it indicates that the device does not support configuring group parameters.

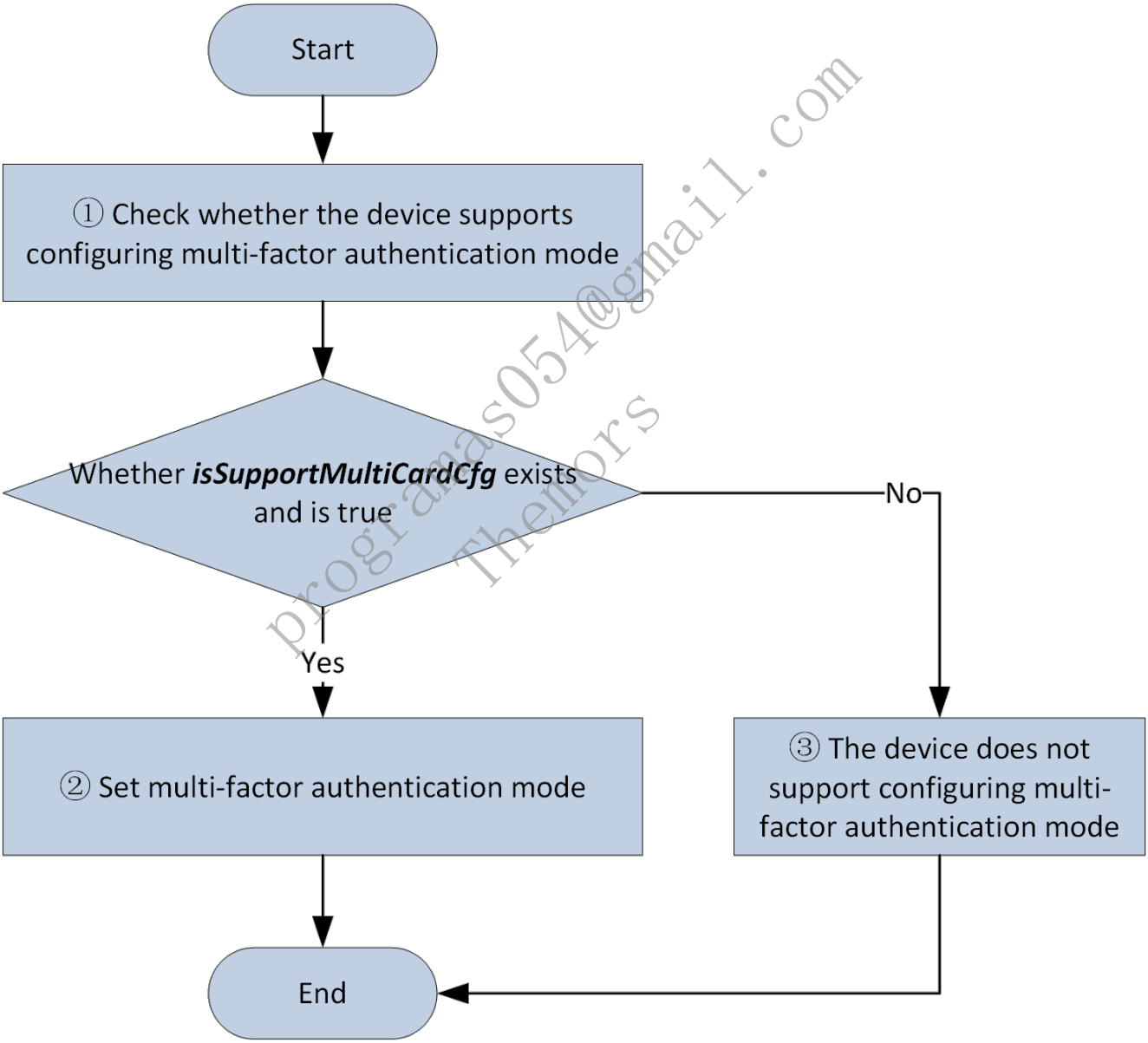
7.11.2.2 Add Person to Group

After configuring the group parameters, the person and group will be linked through ****person management in the process of person and credential management integration (through the belongGroup field). The corresponding interface is as follows:

1. Set person information: `PUT /ISAPI/AccessControl/UserInfo/SetUp?format=json`
2. Add person information: `POST /ISAPI/AccessControl/UserInfo/Record?format=json`
3. Edit person information: `PUT /ISAPI/AccessControl/UserInfo/Modify?format=json`

No more than 4 groups can be linked to one person.

7.11.2.3 Multi-Factor Authentication Mode Configuration



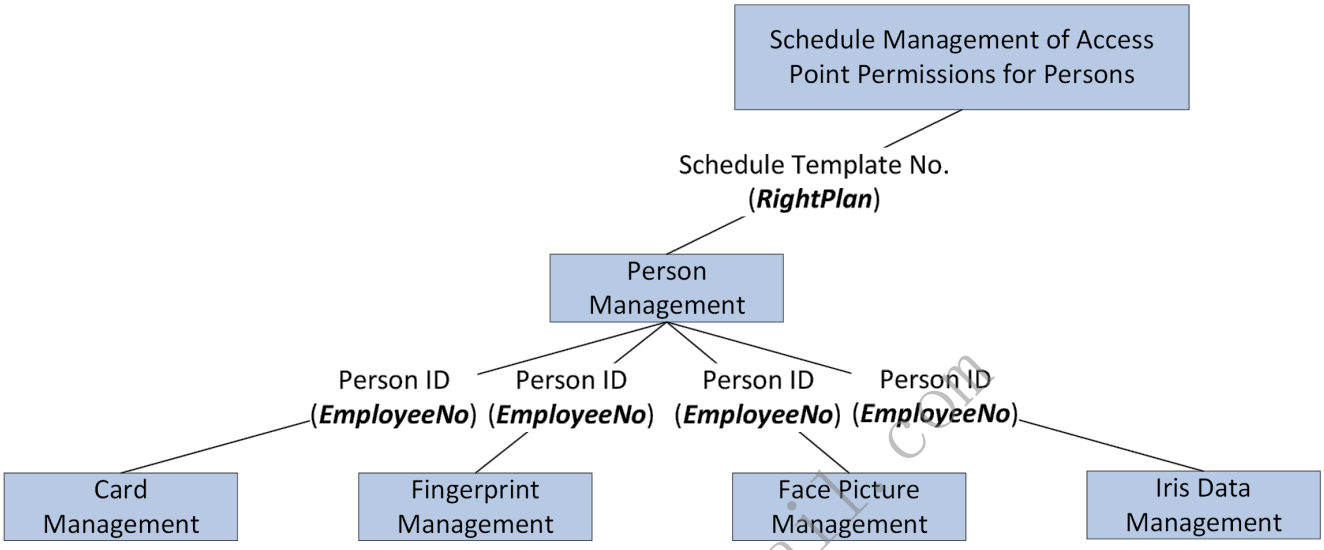
The API calling flow is as follow:

1. Check whether the device supports configuring multi-factor authentication mode: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportMultiCardCfg is returned and its value is "true", it indicates that the device supports configuring multi-factor authentication mode (and the device also supports configuring group parameters).
2. Set multi-factor authentication mode: `[GET/PUT] /ISAPI/AccessControl/MultiCardCfg/<doorID>?format=json`
3. If the node isSupportVerifyWeekPlanCfg is returned and its value is "false", it indicates that the device does not support configuring multi-factor authentication mode.

7.12 Person and Credential Management

7.12.1 Introduction to the Function

The person and credential management function is person-based, and is for managing persons, credentials (cards, fingerprints, face pictures, and iris data), and permission schedules which control the permissions for persons to enter and exit the controlled areas. Its architecture is shown below.



This document mainly introduces the calling flows for person management and credential management (card, fingerprint, face picture, iris data management). For details about the calling flow for permission schedule management, refer to the "Management of Permission Schedules for Persons and Access Points".

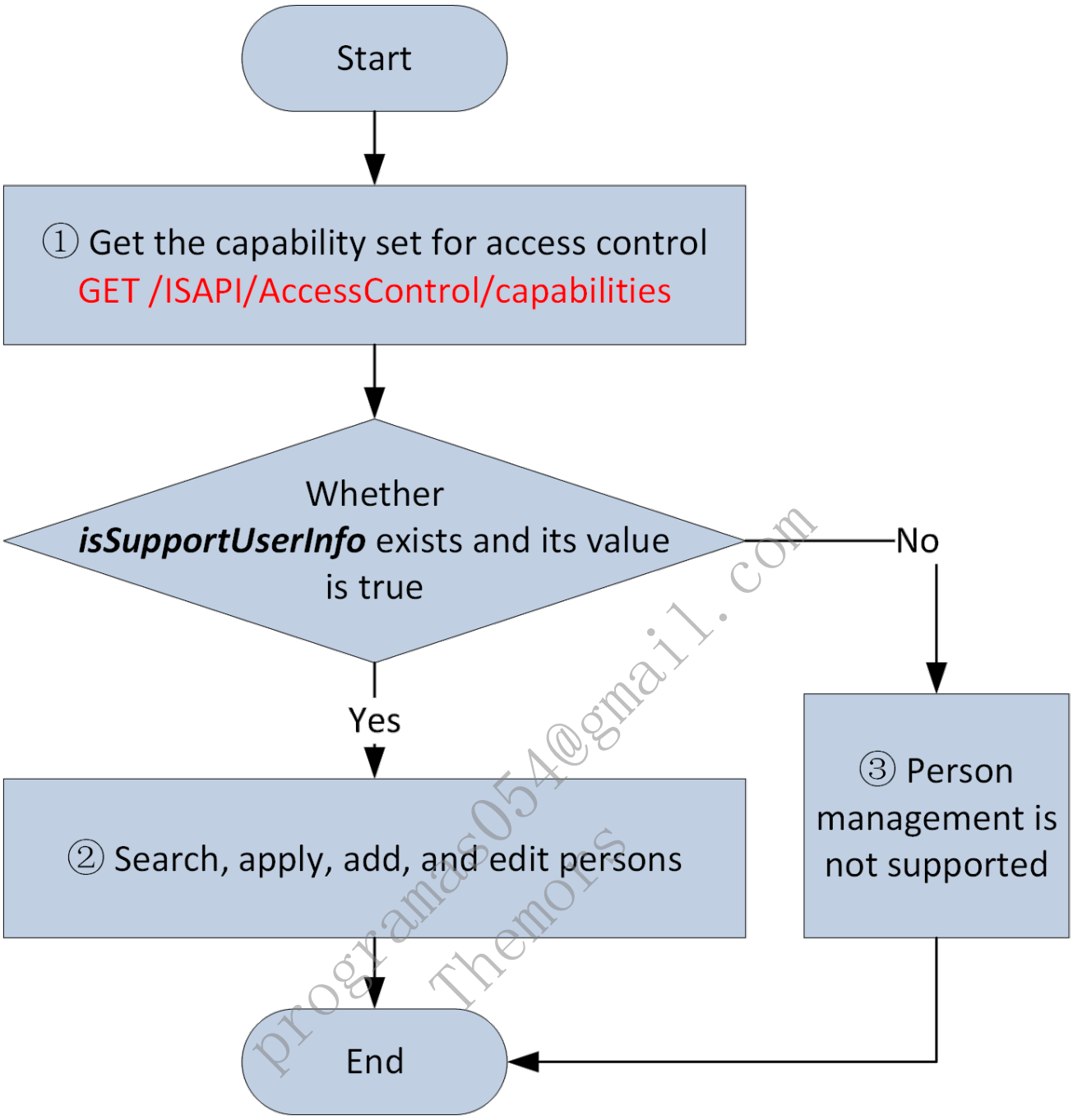
7.13 Person Management

7.13.1 Introduction to the Function

Person management includes searching, applying, adding, editing, and deleting persons.

7.13.2 API Calling Flow

7.13.2.1 Check Whether the Device Supports Person Management



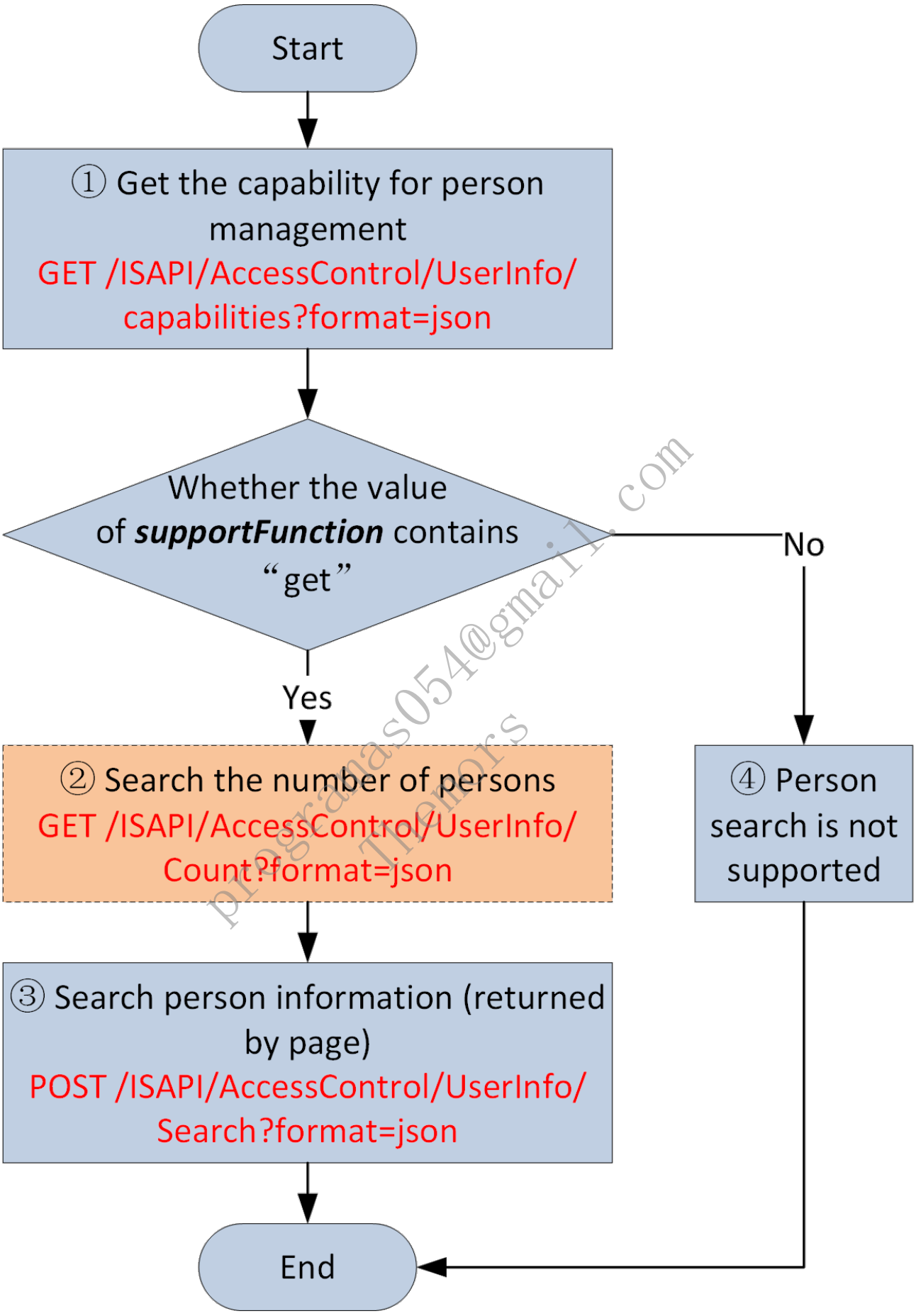
Before calling the API for person management, make sure that the device supports person management.

1. Check whether the device supports person management: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportUserInfo is returned and its value is true, it indicates that the device supports person management.
2. Search, apply, add, and edit persons.
3. If the node isSupportUserInfo is returned and its value is false, it indicates that the device does not support person management.

Note:

The person ID (EmployeeNo) is the unique identifier for person and credential management. After calling `GET /ISAPI/AccessControl/capabilities`, through the child nodes of EmployeeNoInfo which are employeeNo, characterType, and isSupportCompress, the maximum string length and character types of the person ID supported by the device can be checked. Generally, devices support up to 32 bytes and any type of characters. But for access controllers and distribution-type access control devices, check through the child nodes mentioned above.

7.13.2.2 Person Search



The person search function is for searching the number of persons and person information added to the device.

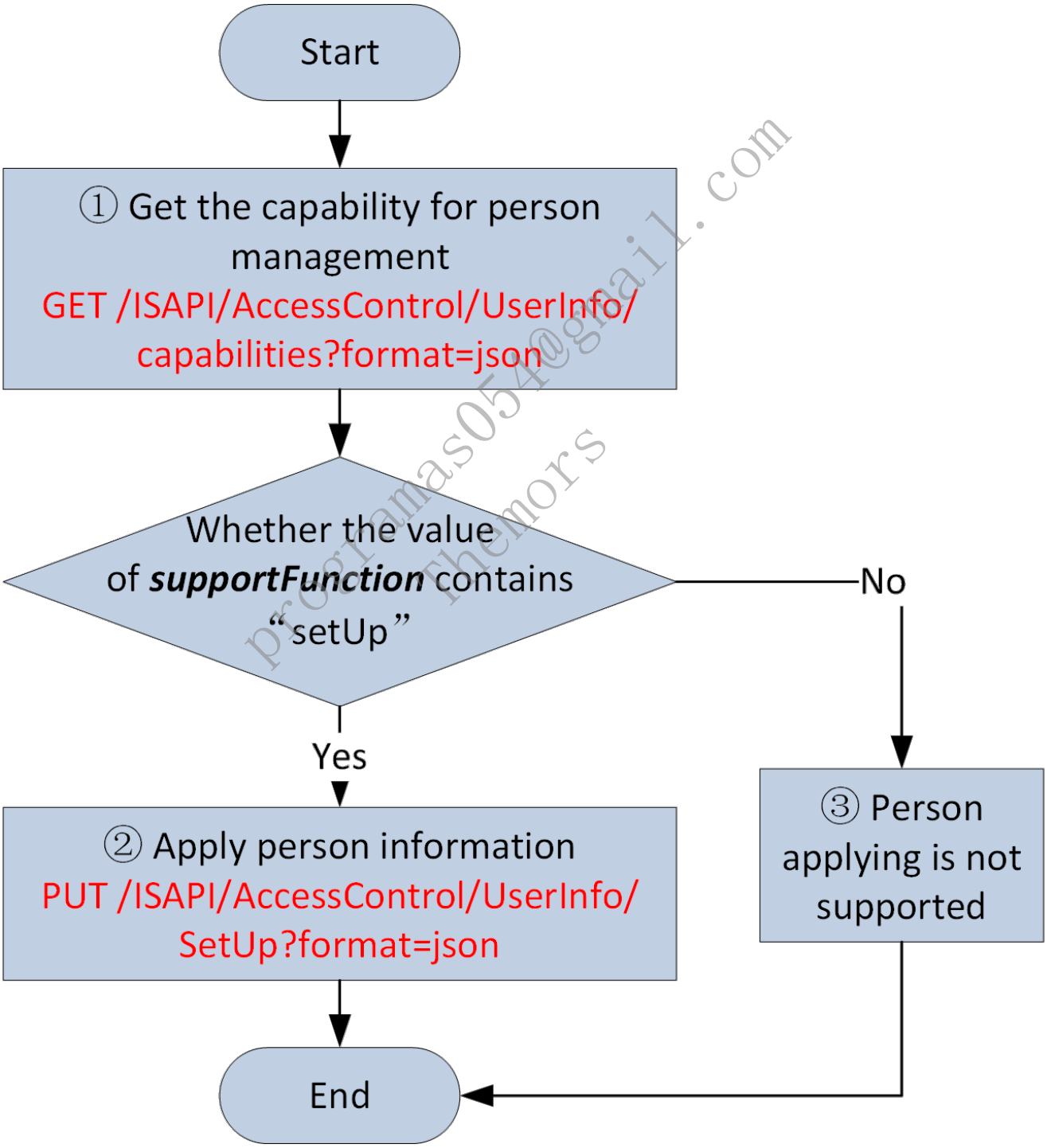
1. Check whether the device supports person search: `GET /ISAPI/AccessControl/UserInfo/capabilities?format=json`; if the value of the node supportFunction contains "get", it indicates that the device supports person search.

2. Search the number of persons: `GET /ISAPI/AccessControl/UserInfo/Count?format=json`; the returned value of the node userNumber is the number of the persons added to the device.
3. Search person information: `POST /ISAPI/AccessControl/UserInfo/Search?format=json`; the person information is returned by page.
4. If the value of the node supportFunction does not contain "get", it indicates that the device does not support person search.

Note:

The value of the node maxRecordNum returned by calling `GET /ISAPI/AccessControl/UserInfo/capabilities?format=json` is the maximum number of persons supported by the device.

7.13.2.3 Person Applying



Person information can be applied to the device via the person applying function. If the person has been added to the device, the person information will be edited; if the person has not been added to the device, the person information will be applied to the device.

1. Check whether the device supports person applying: `GET /ISAPI/AccessControl/UserInfo/capabilities?`

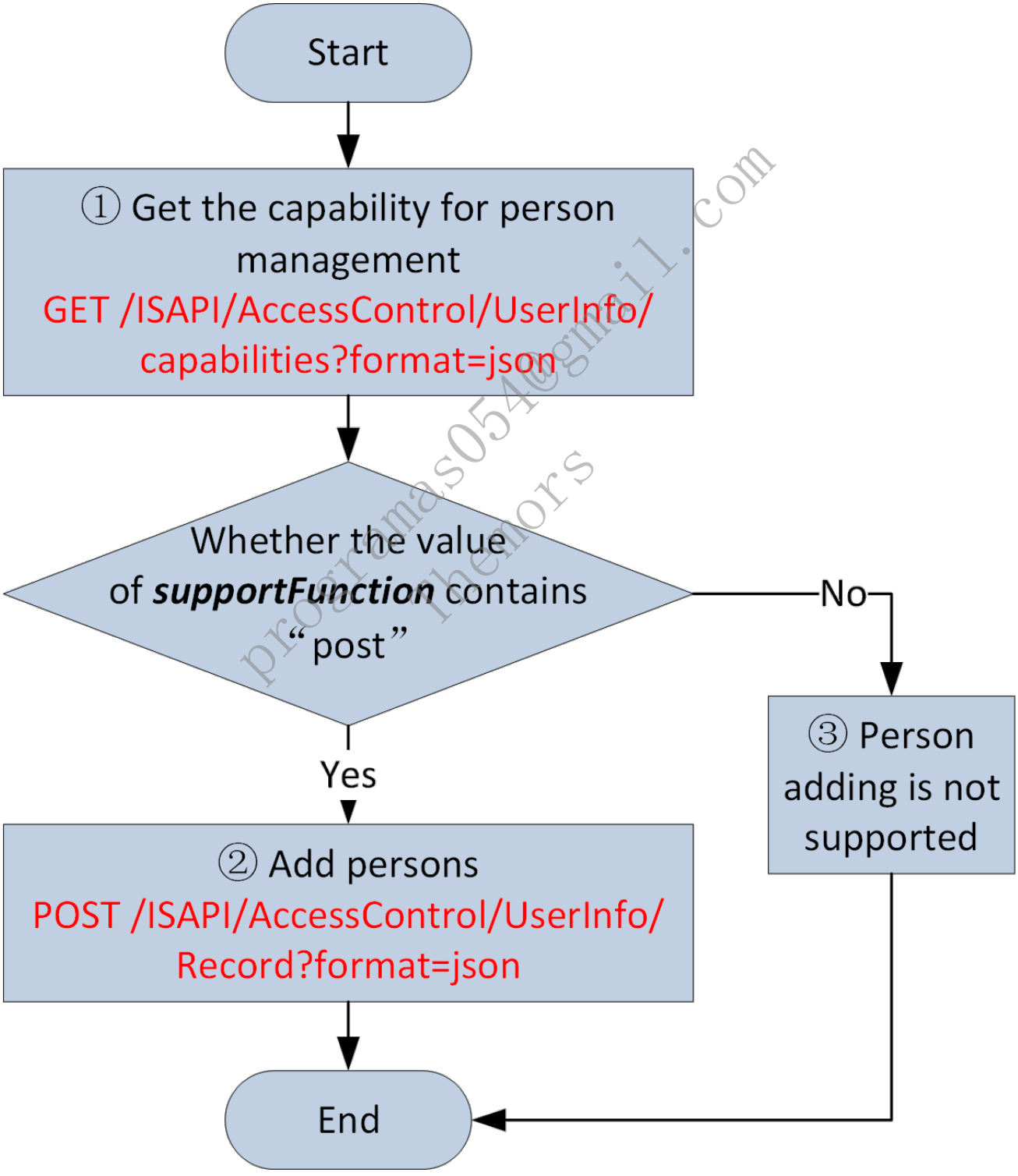
`format=json`; if the value of the node supportFunction contains "setUp", it indicates that the device supports person applying.

2. Apply person information: `PUT /ISAPI/AccessControl/UserInfo/SetUp?format=json`.
3. If the value of the node supportFunction does not contain setUp, it indicates that the device does not support person applying.

Note:

Check whether the person has been added to the device via the node employeeNo returned after calling the API for person applying.

7.13.2.4 Person Adding



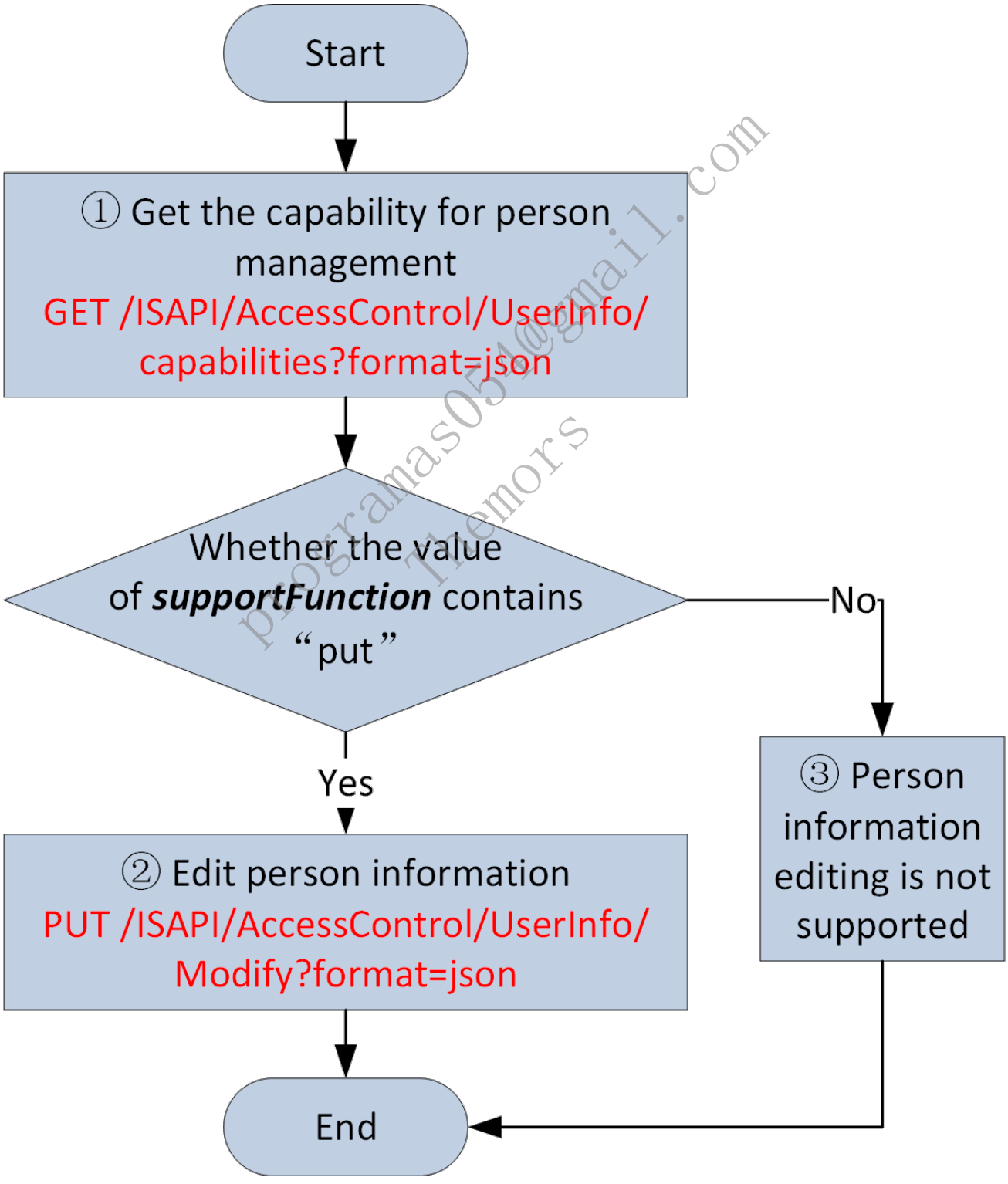
Person can be added to the device via the person adding function. If the person has been added to the device, the device will report an error; if the person has not been added to the device, the person will be added to the device.

1. Check whether the device supports person adding: `GET /ISAPI/AccessControl/UserInfo/capabilities?format=json`; if the value of the node supportFunction contains "post", it indicates that the device supports person adding.
2. Add persons: `POST /ISAPI/AccessControl/UserInfo/Record?format=json`.
3. If the value of the node supportFunction does not contain "post", it indicates that the device does not support person adding.

Note:

Check whether the person has been added to the device via the node employeeNo returned after calling the API for person adding.

7.13.2.5 Person Information Editing



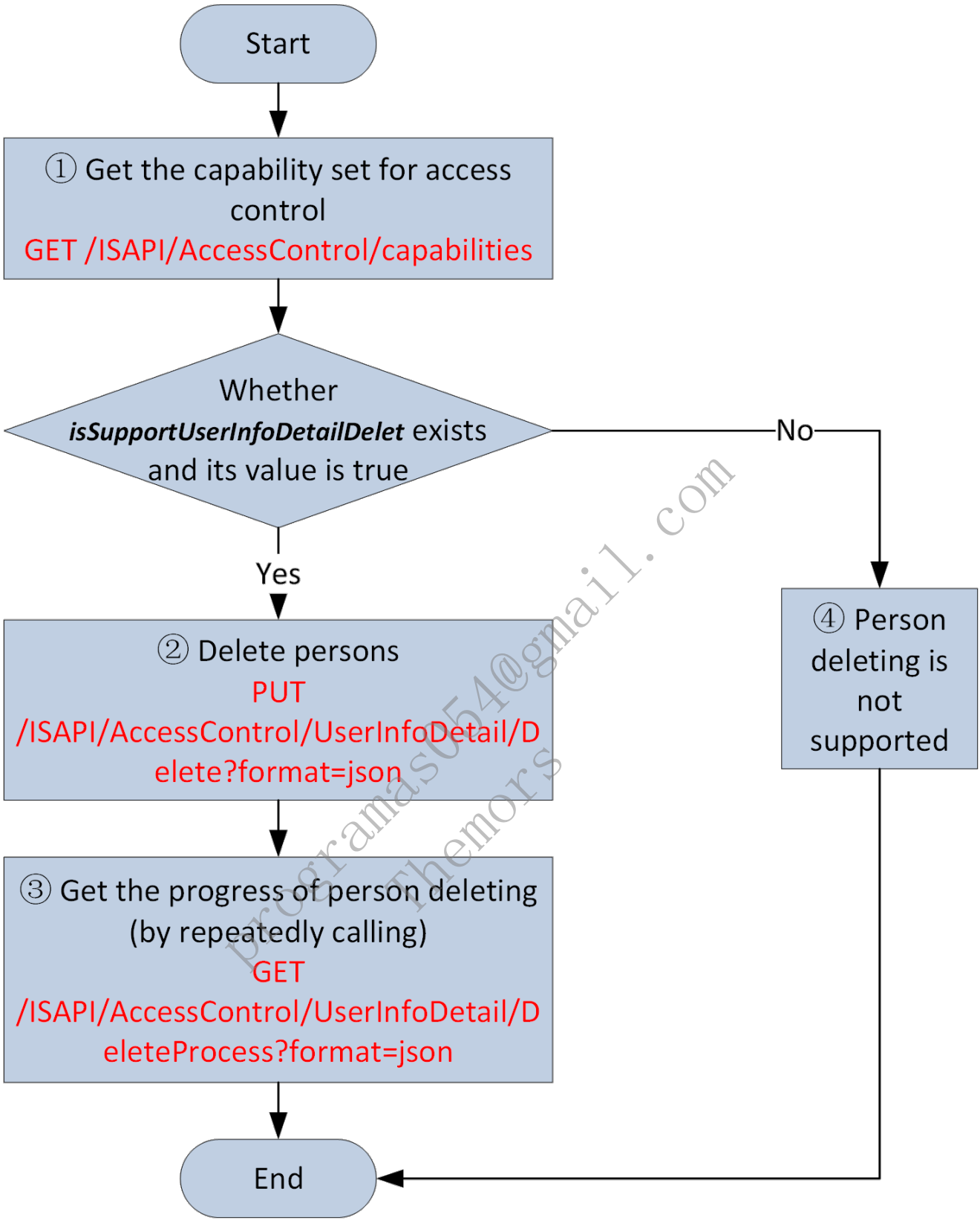
Person information added to the device can be edited via the person information editing function. If the person has been added to the device, the person information will be edited; if the person has not been added to the device, the device will report an error.

1. Check whether the device supports person information editing: `GET /ISAPI/AccessControl/UserInfo/capabilities?format=json`; if the value of the node supportFunction contains "put", it indicates that the device supports person information editing.
2. Edit Person Information: `PUT /ISAPI/AccessControl/UserInfo/Modify?format=json`.
3. If the value of the node supportFunction does not contain "put", it indicates that the device does not support person information editing.

Note:

Check whether the person has been added to the device via the node employeeNo returned after calling the API for person information editing.

7.13.2.6 Person Deleting



The person added to the device can be deleted via the person deleting function. The device will not report an error if the person to be deleted is not added to the device.

1. Check whether the device supports person deleting: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportUserInfoDetailDelete is returned and its value is "true", it indicates that the device supports person deleting.
2. Delete persons: `PUT /ISAPI/AccessControl/UserInfoDetail/Delete?format=json`; if calling succeeded, it indicates that the device has started to execute person deleting, but it does not indicate that the device has deleted the person.
3. Get the progress of deleting person information: `GET /ISAPI/AccessControl/UserInfoDetail/DeleteProcess`; repeatedly call this API to get the progress of person deleting.
4. If the node isSupportUserInfoDetailDelete is returned and its value is "false", it indicates that the device does not

support person deleting.

Note:

When the person is deleted, the information on the credentials (the card, fingerprint, face picture, and iris data) linked via the person ID will also be deleted.

7.14 Reset Anti-Passback Rule (Additional Function)

7.14.1 Introduction to the Function

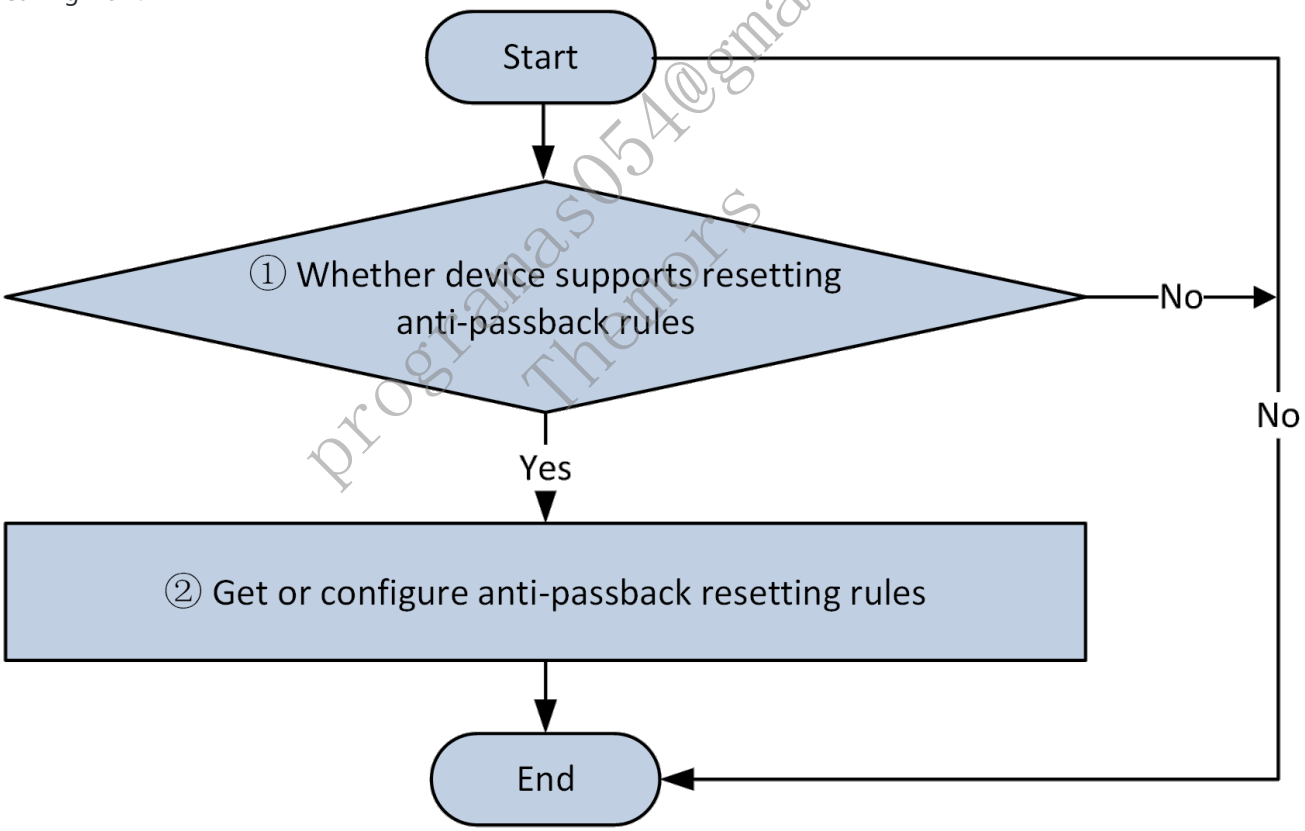
Anti-passback rules which can be reset:

1. Reset by authentication interval. This function will take effect in specific time period after the anti-passbak is triggered. If the user trigger the function by swiping a card by route, the anti-passback flag will be reset in certain time.
2. Reset by time. The anti-passback flag will be reset automatically in certain time.
3. Invalid mode. The resetting rule is disabled.

Application scenarios: The anti-passback function will be help to reduce the cost of manual monitoring. Anti-passback by time period and by time cannot set at the same time.

7.14.2 API Calling Flow

Calling Flow:



ISAPI Protocol Calling Flow:

1. Get the capability of access control: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportAntiPassbackResetRules is returned and its value is "true", it indicates that the device supports resetting rules of anti-passback.
2. Get resetting rules of anti-passback: `GET /ISAPI/AccessControl/AntiPassback/resetRules?format=json`; configure resetting rules of anti-passback: `PUT /ISAPI/AccessControl/AntiPassback/resetRules?format=json`.

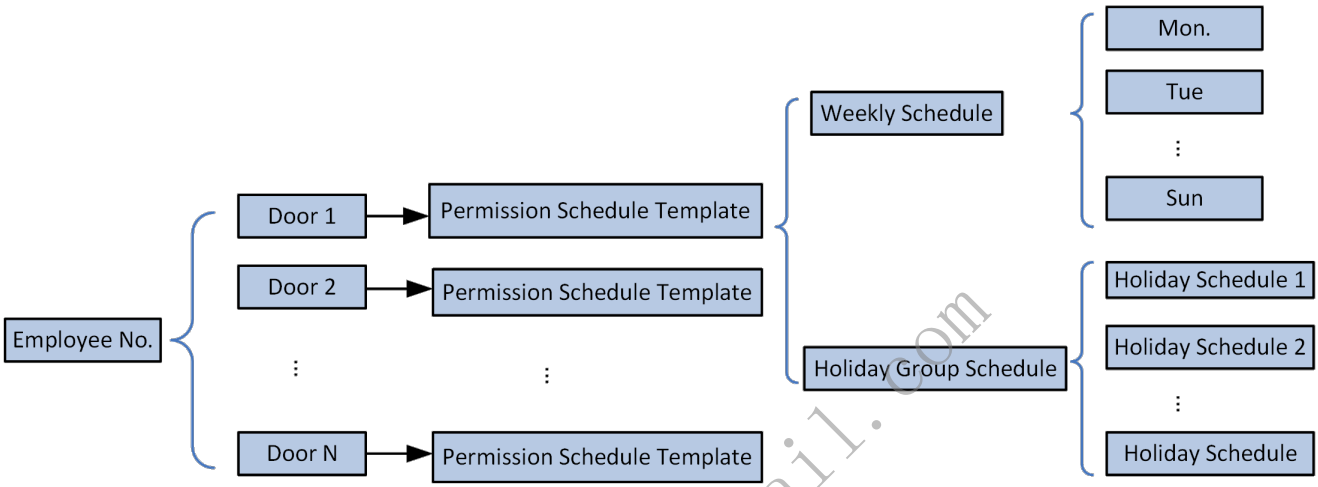
7.15 Schedules Management of Persons' Access Permission

7.15.1 Introduction to the Function

It is required to connect to door permission and schedule template of access permission related to each door before

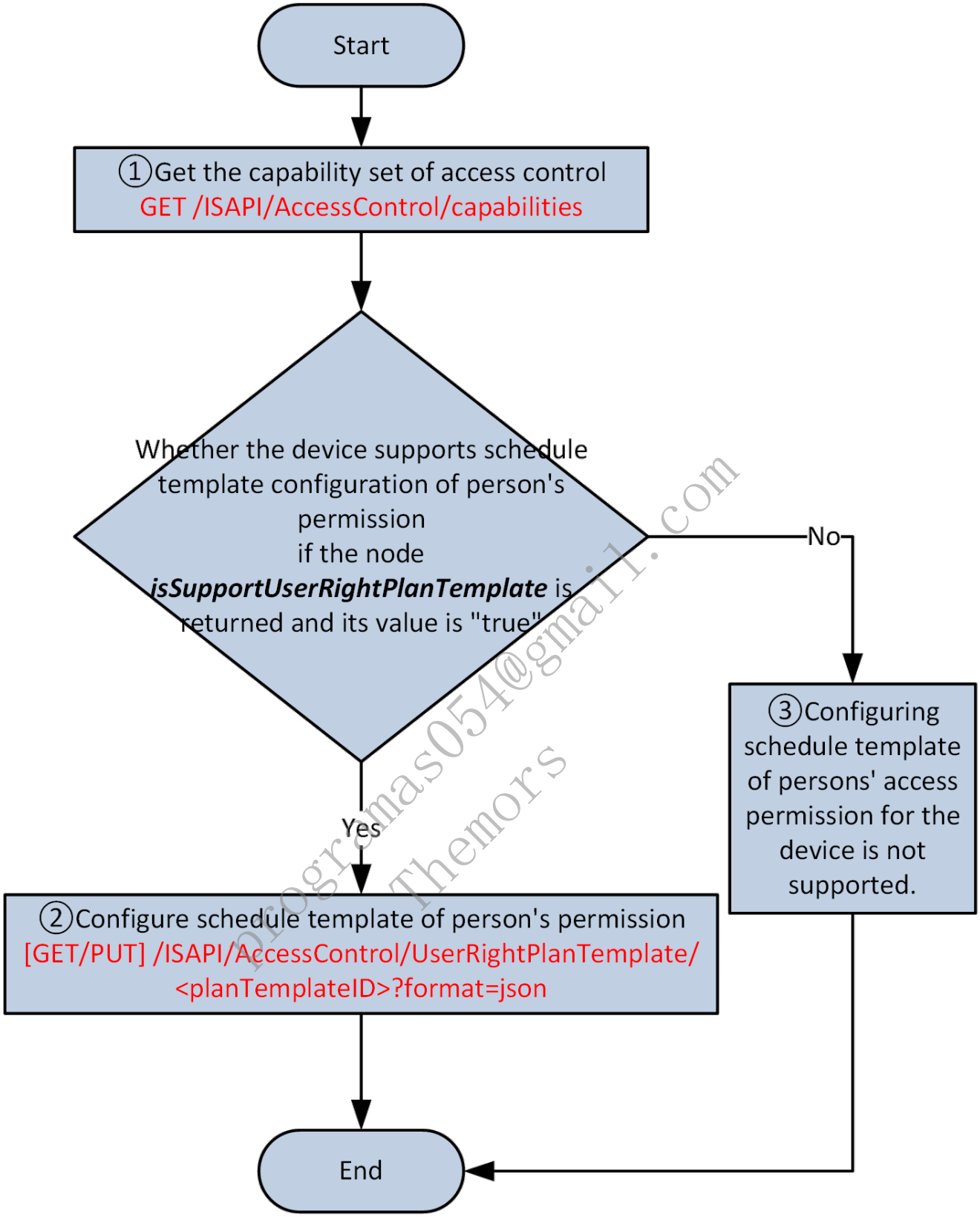
applying permissions to persons. For applying permissions to persons, see calling flow of *Person Management* of *Person and Credential Management*. Configuring schedules of persons' access permission is required, or the related persons cannot access.

1 weekly schedule and 4 holiday groups can be added in each schedule template. The priority of holiday schedule is higher than that of weekly schedule. Weekly schedule can be configured by day of a week and 8 different time period of a day. 16 holiday schedules can be added to a holiday group schedule. Each holiday schedule has its start and end day, and the time period is same in the holiday range (8 time periods can be added). The access control can follow the schedule template to manage person's permissions by time.



7.15.2 API Calling Flow

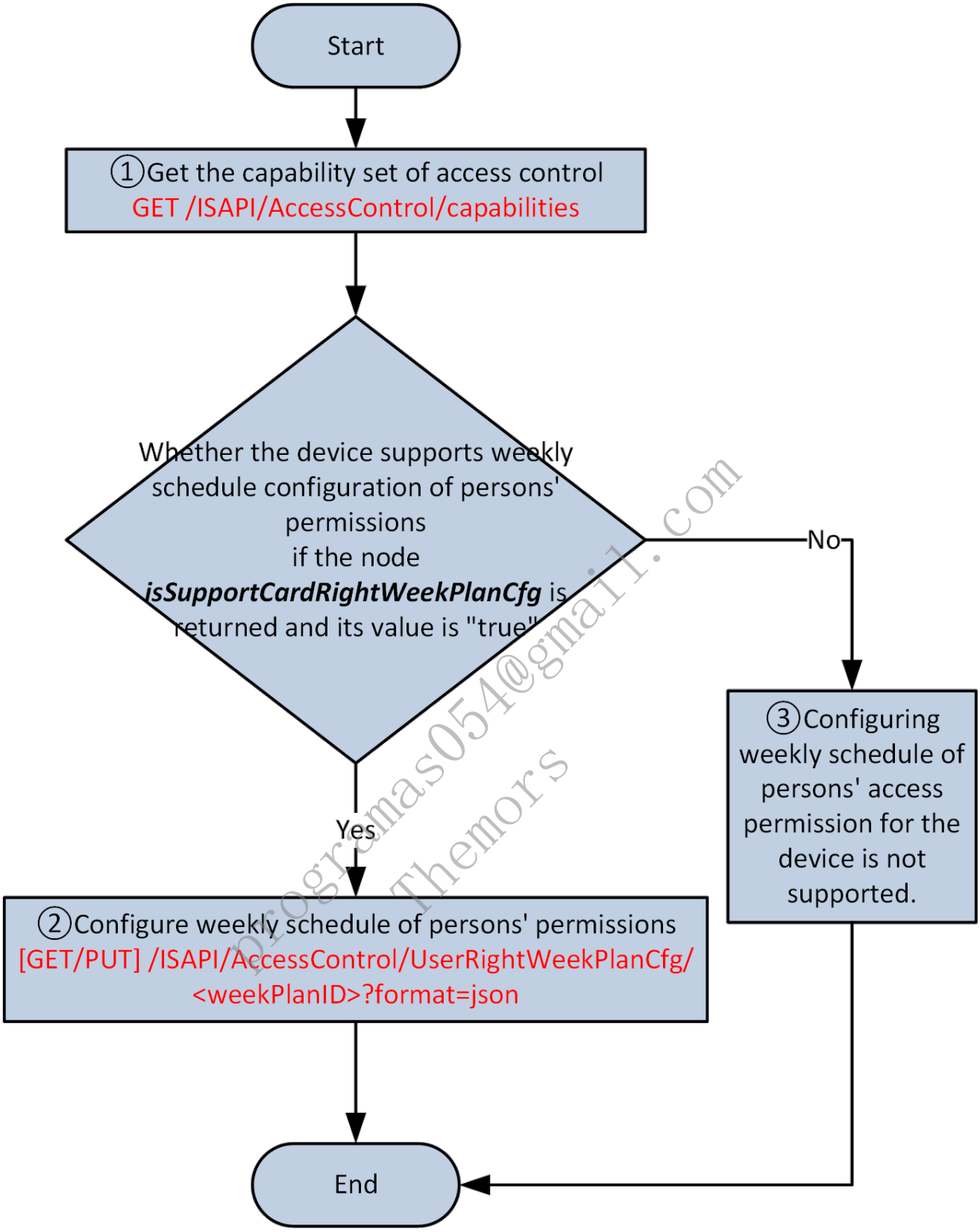
7.15.2.1 Schedule Template of Persons' Access Permission



Calling Flow:

1. Check whether the device supports schedule template configuration of person's permission: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportUserRightPlanTemplate is returned and its value is "true", it indicates that the device supports schedule template configuration of person's permission (if it supports, it also supports weekly schedule configuration of persons' permission).
2. Schedule template configuration of persons' permission: `[GET/PUT] /ISAPI/AccessControl/UserRightPlanTemplate/<planTemplateID>?format=json`.
3. Configuring schedule template of persons' access permission control for the device is not supported.

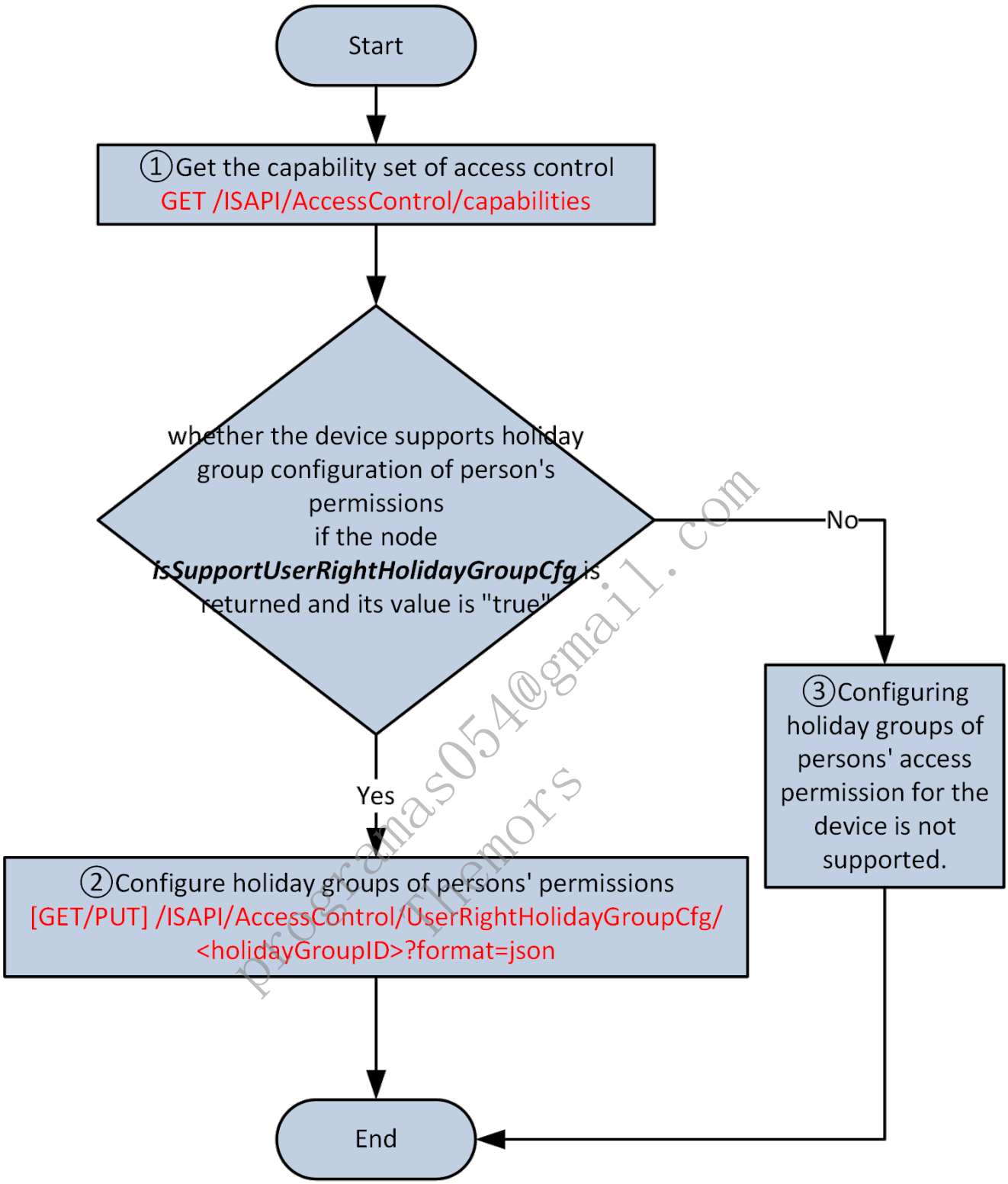
7.15.2.2 Weekly Schedule of Persons' Access Permission



Calling Flow:

1. Check whether the device supports weekly schedule configuration of persons' permissions: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportCardRightWeekPlanCfg is returned and its value is "true", it indicates that the device supports weekly schedule configuration of person's permissions.
2. Weekly schedule configuration of persons' permissions:`[GET/PUT] /ISAPI/AccessControl/UserRightWeekPlanCfg/<weekPlanID>?format=json`.
3. Configuring weekly schedule of persons' access permission control for the device is not supported.

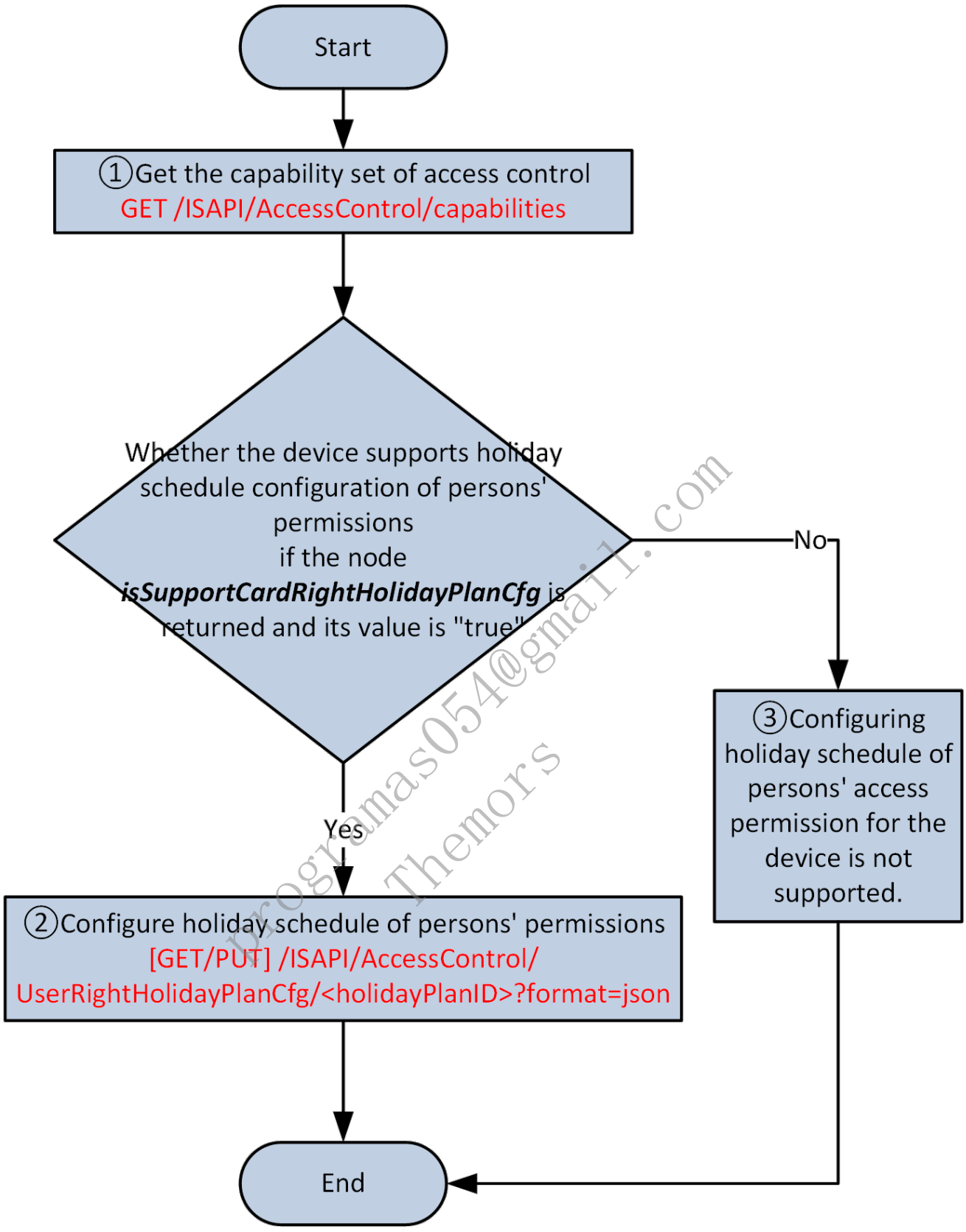
7.15.2.3 Holiday Groups of Persons' Access Permission



Calling Flow:

1. Check whether the device supports holiday group configuration of person's permissions: `GET /ISAPI/AccessControl/capabilities`; if the nodeisSupportUserRightHolidayGroupCfg is returned and its value is "true", it indicates that the device supports holiday group configuration of person's permission (if it supports, it also supports holiday schedule configuration of persons' permissions).
2. Holiday group configuration of persons' permissions:`[GET/PUT] /ISAPI/AccessControl/UserRightHolidayGroupCfg/<holidayGroupID>?format=json`.
3. Configuring holiday groups of persons' access permission control for the device is not supported.

7.15.2.4 Holiday Schedule of Persons' Access Permission



Calling Flow:

1. Check whether the device supports holiday schedule configuration of persons' permissions: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportCardRightHolidayPlanCfg is returned and its value is "true", it indicates that the device supports holiday schedule configuration of person's permissions.
2. Holiday schedule configuration of persons' permissions: `[GET/PUT] /ISAPI/AccessControl/UserRightHolidayPlanCfg/<holidayPlanID>?format=json`.
3. Configuring holiday schedule of persons' access permission control for the device is not supported.

7.16 Store and Search for Access Control Event

7.16.1 Introduction to the Function

The device supports configuring storage parameters of access control event, searching for access control events, and searching for the amount of access control events. There are three storage modes: deleting old events periodically, deleting old events by specified time and overwriting.

7.16.2 API Calling Flow

7.16.2.1 Configure Storage Parameters of Access Control Events

1. Call the functional capability of access control: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportEventStorageCfg is returned and its value is "true", it indicates that the device supports configuring the storage parameters of access control events.
2. Get the configuration capability of storing access control events: `GET /ISAPI/AccessControl/AcsEvent/StorageCfg/capabilities?format=json`.
3. Get and set storage parameters of access control events: `GET|PUT /ISAPI/AccessControl/AcsEvent/StorageCfg?format=json`.

7.16.2.2 Search for Access Control Events

1. Call the functional capability of access control: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportAcsEvent is returned and its value is "true", it indicates that the device supports searching for access control events.
2. Get the capability of searching for access control events: `GET /ISAPI/AccessControl/AcsEvent/capabilities?format=json`.
3. Search for access control events: `POST /ISAPI/AccessControl/AcsEvent?format=json`.

7.16.2.3 Get Total Number of Access Control Events

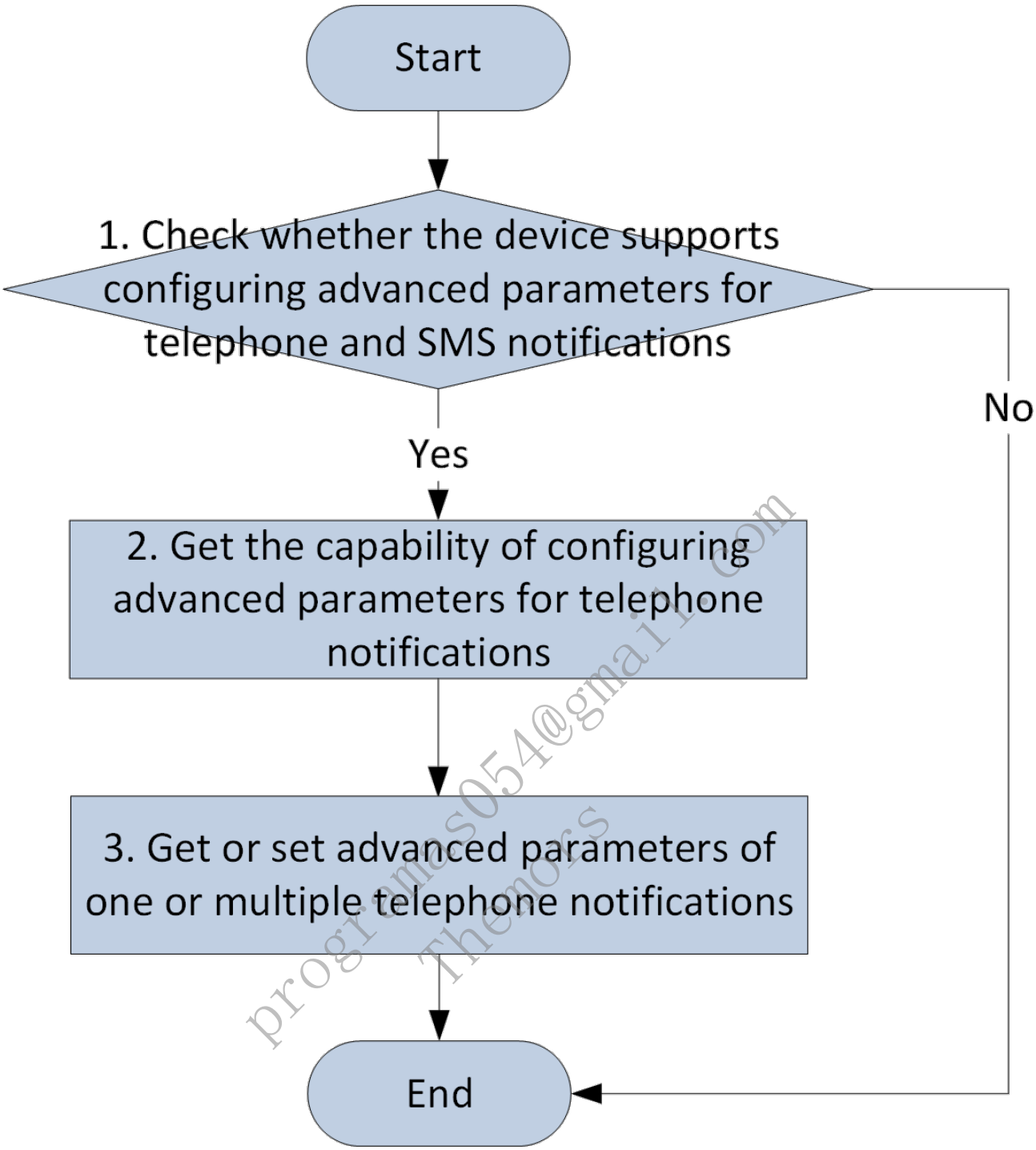
1. Call the functional capability of access control: `GET /ISAPI/AccessControl/capabilities`; if the node isSupportAcsEventTotalNum is returned and its value is "true", it indicates that the device supports getting the total number of access control events.
2. Get the capability of getting total number of access control events by specific conditions: `GET /ISAPI/AccessControl/AcsEventTotalNum/capabilities?format=json`.
3. Get the total number of access control events by specific conditions: `POST /ISAPI/AccessControl/AcsEventTotalNum?format=json`.

8 Security Control Device (General)

8.1 Advanced Configuration for Telephone Notification

You can configure advanced parameters for telephone notifications, including effective time period, arming / disarming / alarm clearing permission for each telephone number, and event/alarm type (emergency alarm, medical alarm, and gas alarm).

API Calling Flow:



API Calling Steps:

1. Get the configuration capability of the security control panel to check whether the device supports configuring advanced parameters for telephone and SMS notifications: `GET /ISAPI/SecurityCP/Configuration/capabilities?format=json`. If the returned value of the node isSptPhoneAnvanced is true, the device supports this function.

2. Get the capability of configuring advanced parameters for telephone notifications: `GET /ISAPI/SecurityCP/Configuration/messageSendPhoneAnvanced/capabilities?format=json`.

3. Get advanced parameters of all telephone notifications: `GET /ISAPI/SecurityCP/Configuration/messageSendPhoneAnvanced?format=json`; set advanced parameters of a single telephone notification: `PUT /ISAPI/SecurityCP/Configuration/messageSendPhoneAnvanced/<phoneID>?format=json`. The phoneID in the URL is the telephone number ID. Up to 8 telephone numbers are supported for a wireless security control panel.

8.2 ARC Management

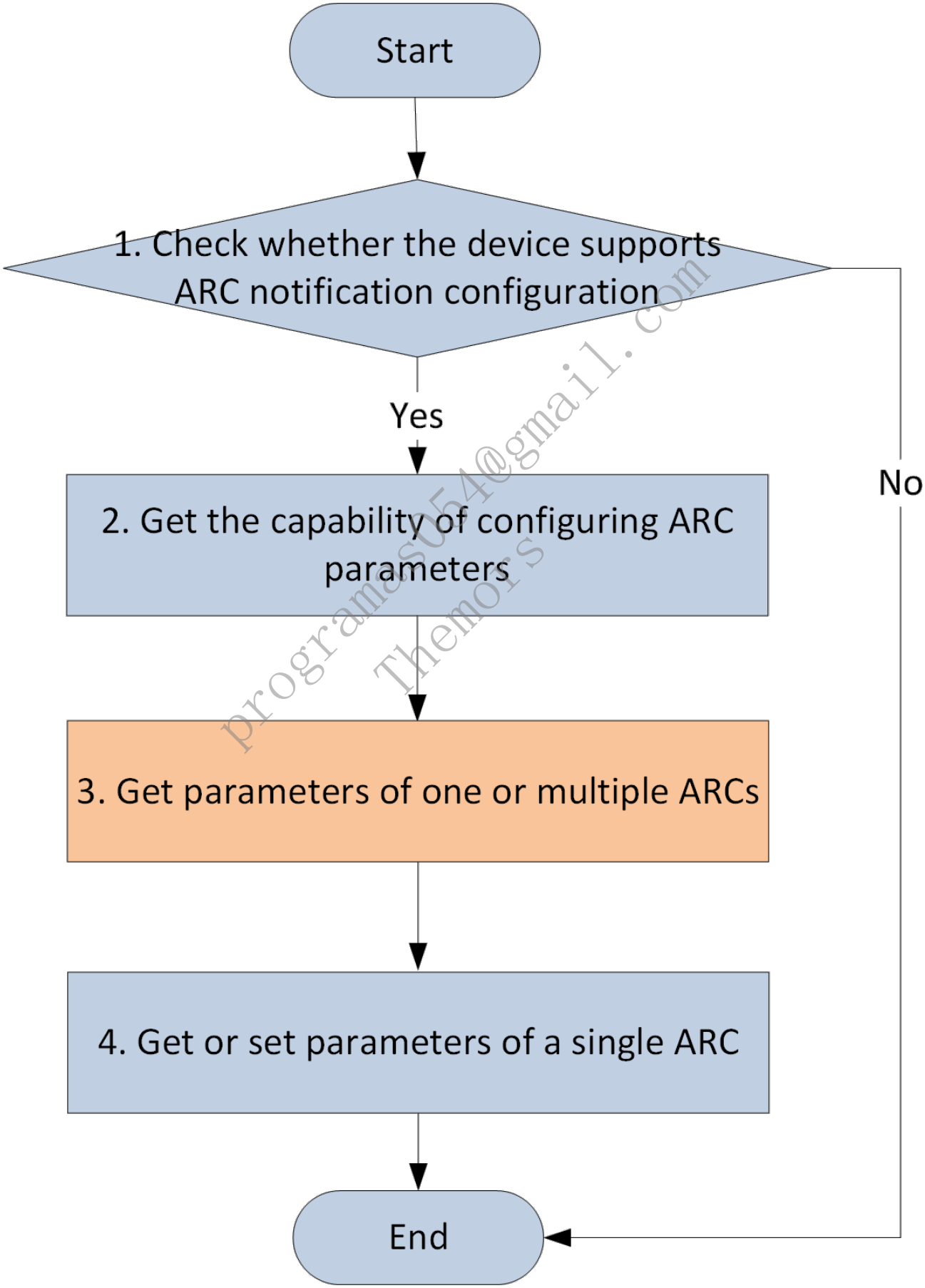
8.2.1 Introduction to the Function

If you have enabled the ARC function and configured events to be received, when the events occur, the corresponding

alarm messages will be pushed to the ARC platform. Currently up to 4 ARCs are supported. By default, No. 1 and 3 are the main ARCs and No. 2 and 4 are the spare ARCs. Only when the main ARCs fail to push alarm messages, will the spare ARCs take the charge.

8.2.2 ARC Parameter Configuration

API Calling Flow:



API Calling Steps: 1. Get the configuration capability of the security control panel to check whether the device supports ARC notification configuration: `GET /ISAPI/SecurityCP/Configuration/capabilities?format=json`. If the returned value of the node isSptARC is true, the device supports this function.

2. Get the capability of configuring ARC parameters: `GET /ISAPI/SecurityCP/Configuration/ARC/capabilities?format=json`.

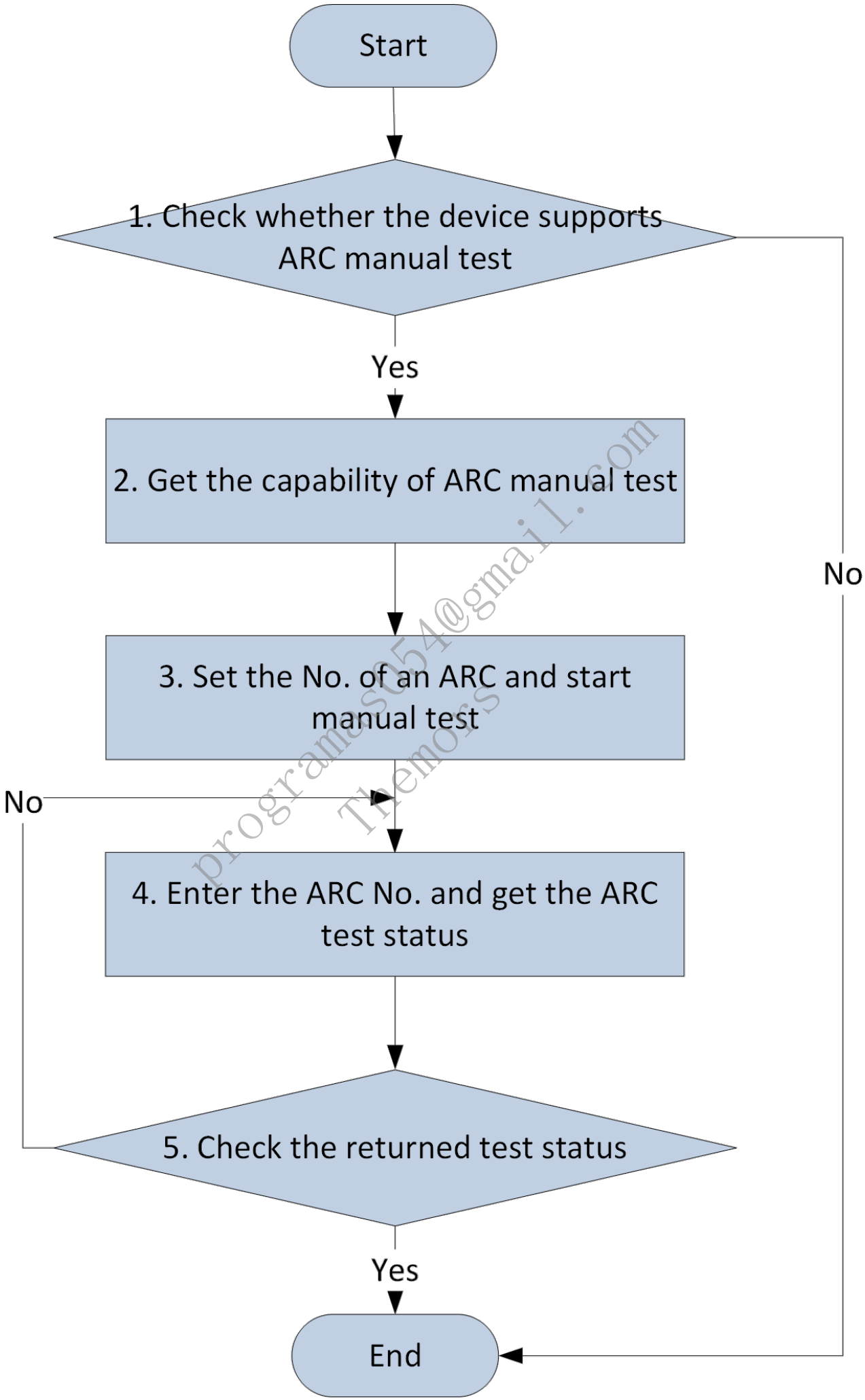
3. Get parameters of one or multiple ARCs: `GET /ISAPI/SecurityCP/Configuration/ARC?format=json&security=<security>&iv=<iv>`. The userName and password nodes should be encrypted.

4. Get or set parameters of a single ARC: `PUT /ISAPI/SecurityCP/Configuration/ARC/<indexID>?format=json&security=<security>&iv=<iv>`. The nodes userName and password should be encrypted. The indexID in the URL is the ARC No.

8.2.3 Manual Test of ARC

After configuring ARC parameters, you can check whether the parameter values are valid by testing the link.

API Calling Flow:


Start
1. Check whether the device supports ARC manual test
Yes
No
2. Get the capability of ARC manual test
3. Set the No. of an ARC and start manual test
No
4. Enter the ARC No. and get the ARC test status
5. Check the returned test status
Yes
End

API Calling Steps: 1. Get the configuration capability of the security control panel to check whether the device supports manual test of ARC: `GET /ISAPI/SecurityCP/Configuration/capabilities?format=json`. If the returned value of the node isSptARC is true, the device supports this function.

2. Get the capability of manual test of ARC: `GET /ISAPI/SecurityCP/Configuration/ARC/manualTest/capabilities?format=json`.

3. Set the No. of an ARC and start manual test: `PUT /ISAPI/SecurityCP/Configuration/ARC/manualTest?format=json`.

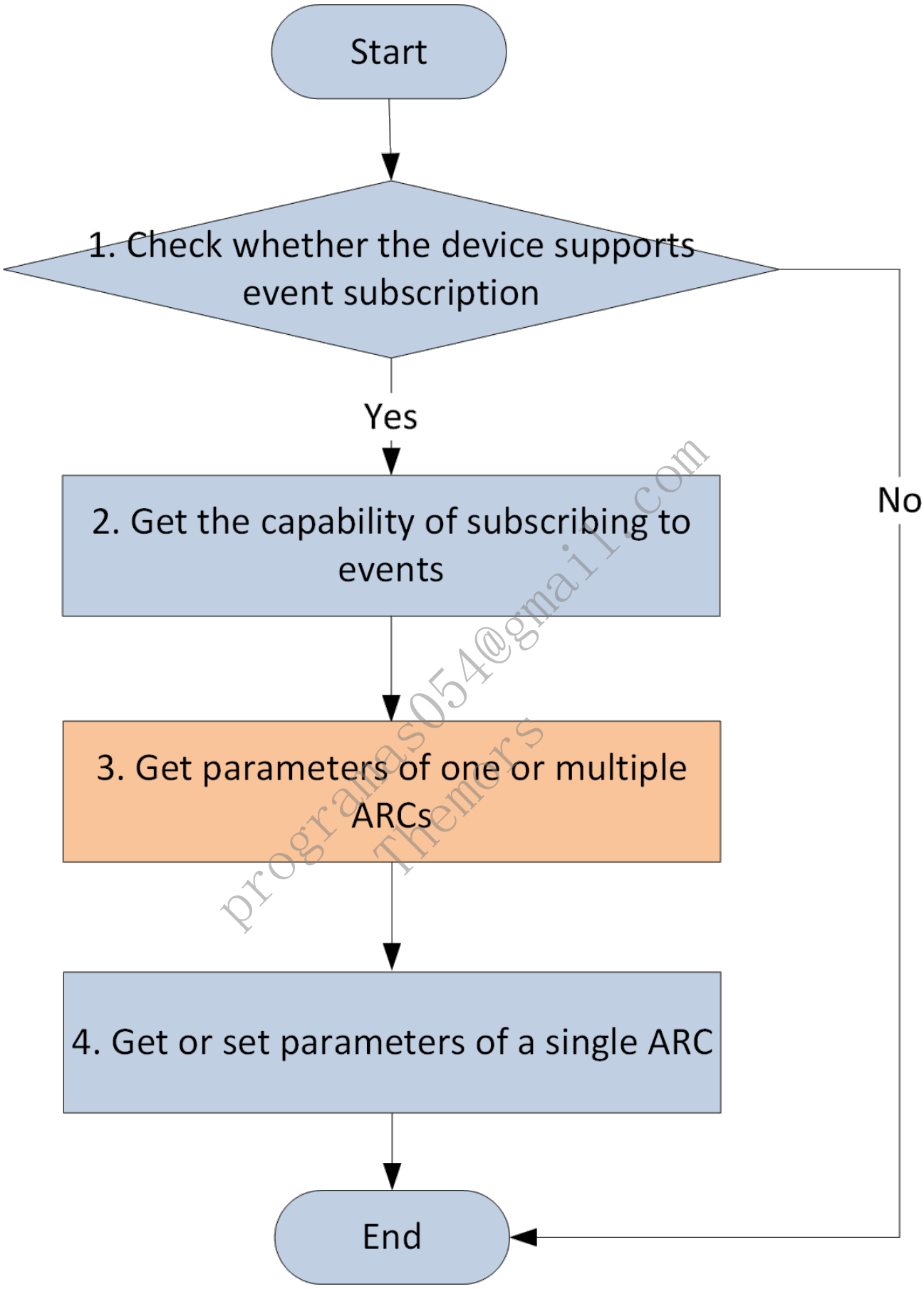
4. Enter the ARC No. and get the ARC test status: `POST /ISAPI/SecurityCP/Configuration/ARC/manualTest/status?format=json`.

5. Check the returned test status: `success` for success and `failed` for failure. If `processing` is returned within the configured timeout threshold, you need to call the API in the previous step again.

8.2.4 Event Subscription

Event/alarm types include tampering event, life security event, system status event, operation event, emergency alarm, medical alarm, and gas alarm. When the alarms are triggered or the events occur, the corresponding alarm/event details will be uploaded by the device.

API Calling Flow:



API Calling Steps:

1. Get the configuration capability of the security control panel to check whether the device supports event subscription: `GET /ISAPI/SecurityCP/Configuration/capabilities?format=json`. If the returned value of the node isSptARC is true, the device supports this function.

2. Get the capability of subscribing to events: `GET /ISAPI/SecurityCP/Configuration/messageSendARC/capabilities?format=json`.

3. Get parameters of one or multiple ARCs: `GET /ISAPI/SecurityCP/Configuration/ARC?format=json&security=<security>&iv=<iv>`.

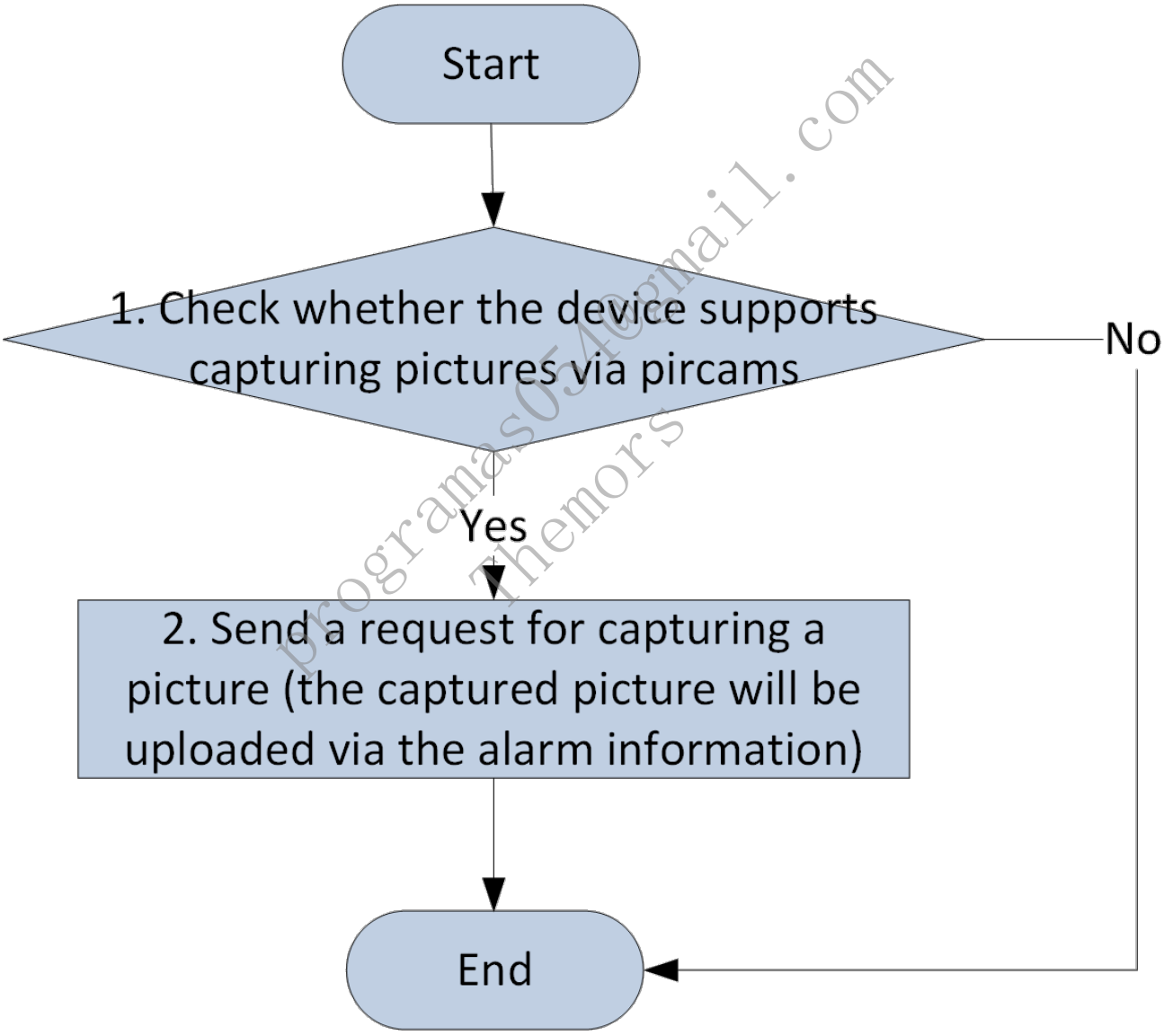
4. Get parameters of a single ARC: `GET /ISAPI/SecurityCP/Configuration/messageSendARC?id=<indexID>&format=json`; set parameters of a single ARC: `PUT /ISAPI/SecurityCP/Configuration/messageSendARC?id=<indexID>&format=json`. The indexID in the URL is the ARC No.

8.3 Capture Pictures of Specific Zones

There are two ways to get the picture captured by pircams: 1) send a request for capturing a picture and get the picture URL in the uploaded alarm (CID event) information; 2) send a request for capturing a picture and call another API to get the picture URL.

8.3.1 Upload Pictures by Pircams

After the pircam captures a picture, the picture will be uploaded via the alarm (CID event) information.

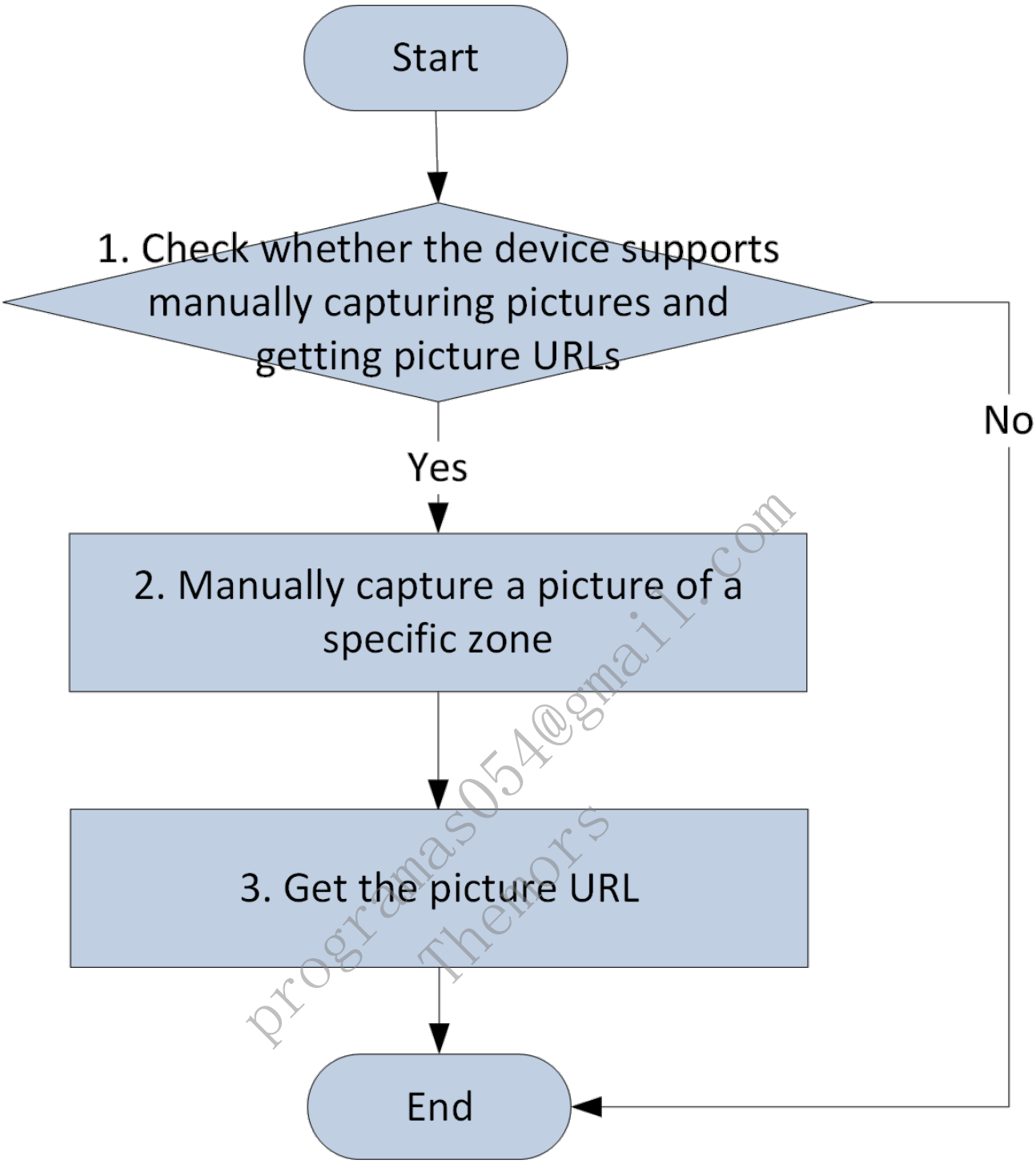


API Calling Steps:

1. Get the configuration capability of the security control panel to check whether the device supports capturing pictures via pircams: `GET /ISAPI/SecurityCP/Configuration/capabilities?format=json`. If the returned value of the node isSptPircamCapture is true, the device supports this function.

2. Send a request for capturing a picture (the captured picture will be uploaded via the alarm information): `GET /ISAPI/Streaming/channels/<zoneID>/picture/devicePush`. The zoneID in the URL refers to zone No.

8.3.2 Get Captured Pictures in Asynchronous Mode



API Calling Steps:

1. Get the configuration capability of the security control panel to check whether the device supports manually capturing pictures and getting picture URLs: `GET /ISAPI/SecurityCP/capabilities?format=json`. If the returned values of the node isSptManualControlCapture and isSptGetPictureByURL is true, the device supports the functions above.

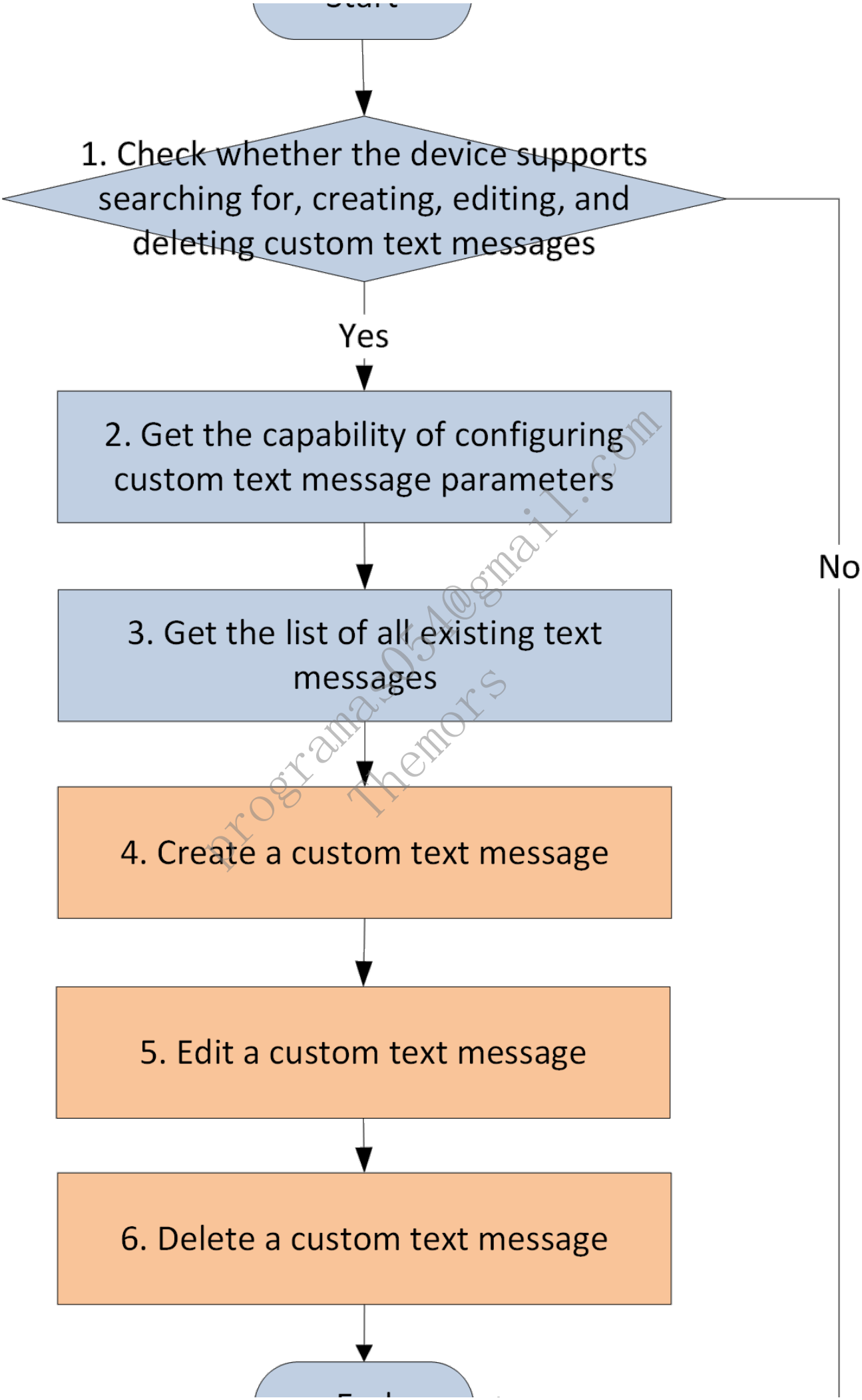
2. Manually capture a picture of a specific zone: `PUT /ISAPI/SecurityCP/Zone/<zoneID>/Capture/ManualControlCapture?format=json`. The zoneID in the URL refers to zone No.

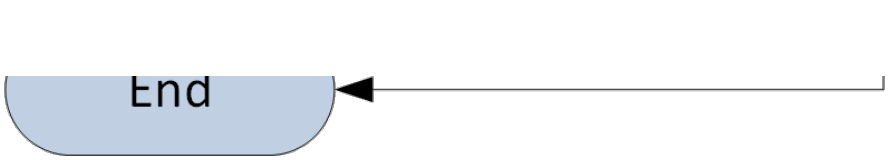
3. Get the picture URL: `GET /ISAPI/SecurityCP/Zone/<zoneID>/Capture/GetPictureByURL?format=json`.

8.4 Custom Text Message Management

Telephone Notifications support both custom audios and custom message texts.

API Calling Flow:





API Calling Steps:

1. Get the configuration capability of the security control panel to check whether the device supports searching for, creating, editing, and deleting custom text messages: `GET /ISAPI/SecurityCP/Configuration/capabilities?format=json`. If the returned values of isSupportSearchCustomAudioList, isSupportAddCustomMessage, isSupportModifyCustomMessage, and isSupportDeleteCustomMessage are all true, the device supports all the functions above.

2. Get the capability of configuring custom text message parameters: `GET /ISAPI/SecurityCP/customMessage/capabilities?format=json`.

3. Get the list of all existing text messages: `GET /ISAPI/SecurityCP/customMessage/searchCustomMessageList?format=json&security=<security>&iv=<iv>`. The node customMessageContent should be encrypted.

4. (Optional) Create a custom text message: `POST /ISAPI/SecurityCP/customMessage/addCustomMessage?format=json&security=<security>&iv=<iv>`.

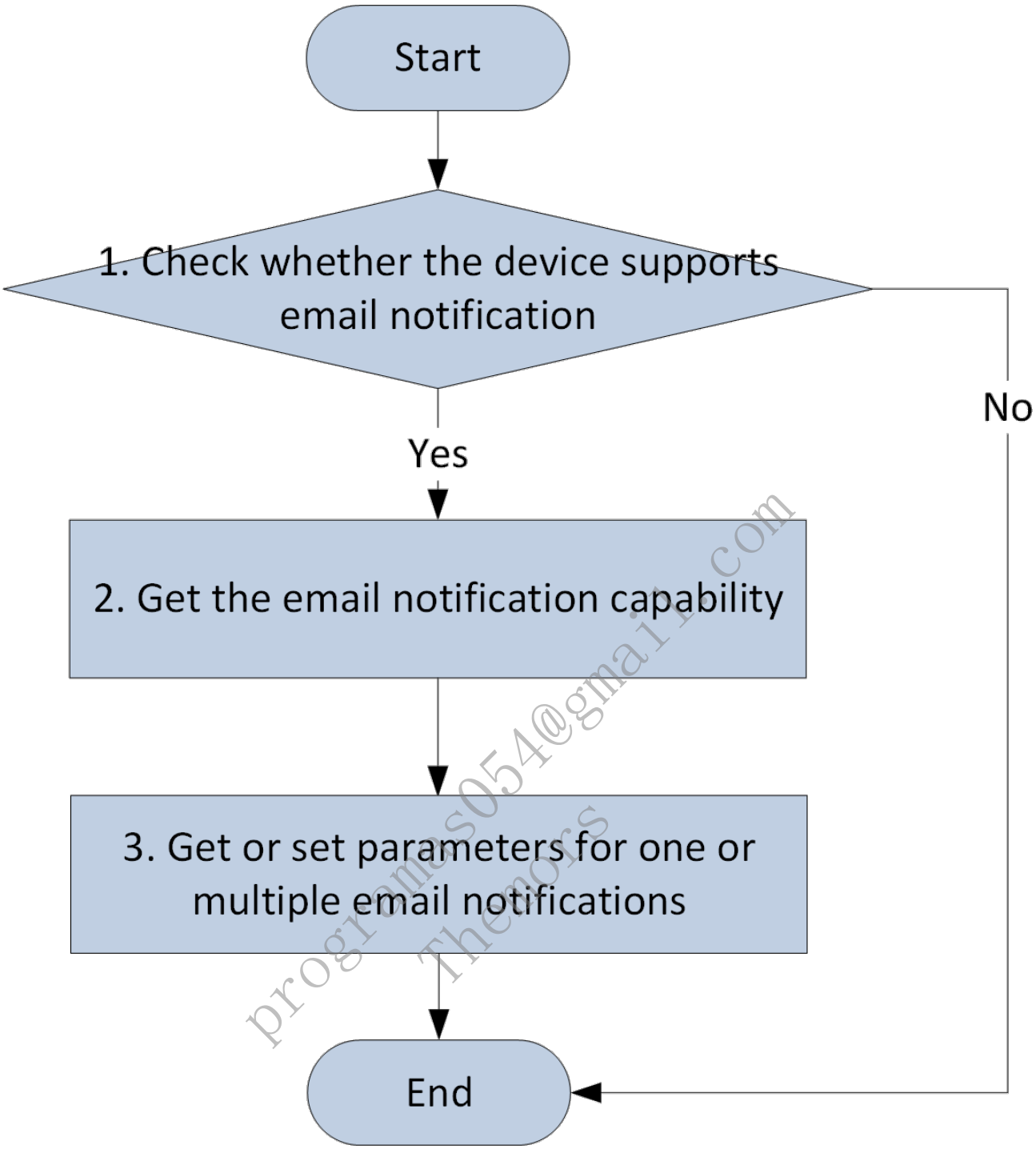
5. (Optional) Edit a custom text message: `PUT /ISAPI/SecurityCP/customMessage/modifyCustomMessage?format=json&security=<security>&iv=<iv>`.

6. (Optional) Delete a custom text message: `POST /ISAPI/SecurityCP/customMessage/deleteCustomMessage?format=json`.

8.5 Email Notification Management

If the device supports email notification, you can enable the event notification and configure related email notification parameters.

API Calling Flow:



API Calling Steps:

1. Get the configuration capability of the security control panel to check whether the device supports email notification: `GET /ISAPI/SecurityCP/Configuration/capabilities?format=json`. If the returned value of the node isSptMail is true, the device supports this function.

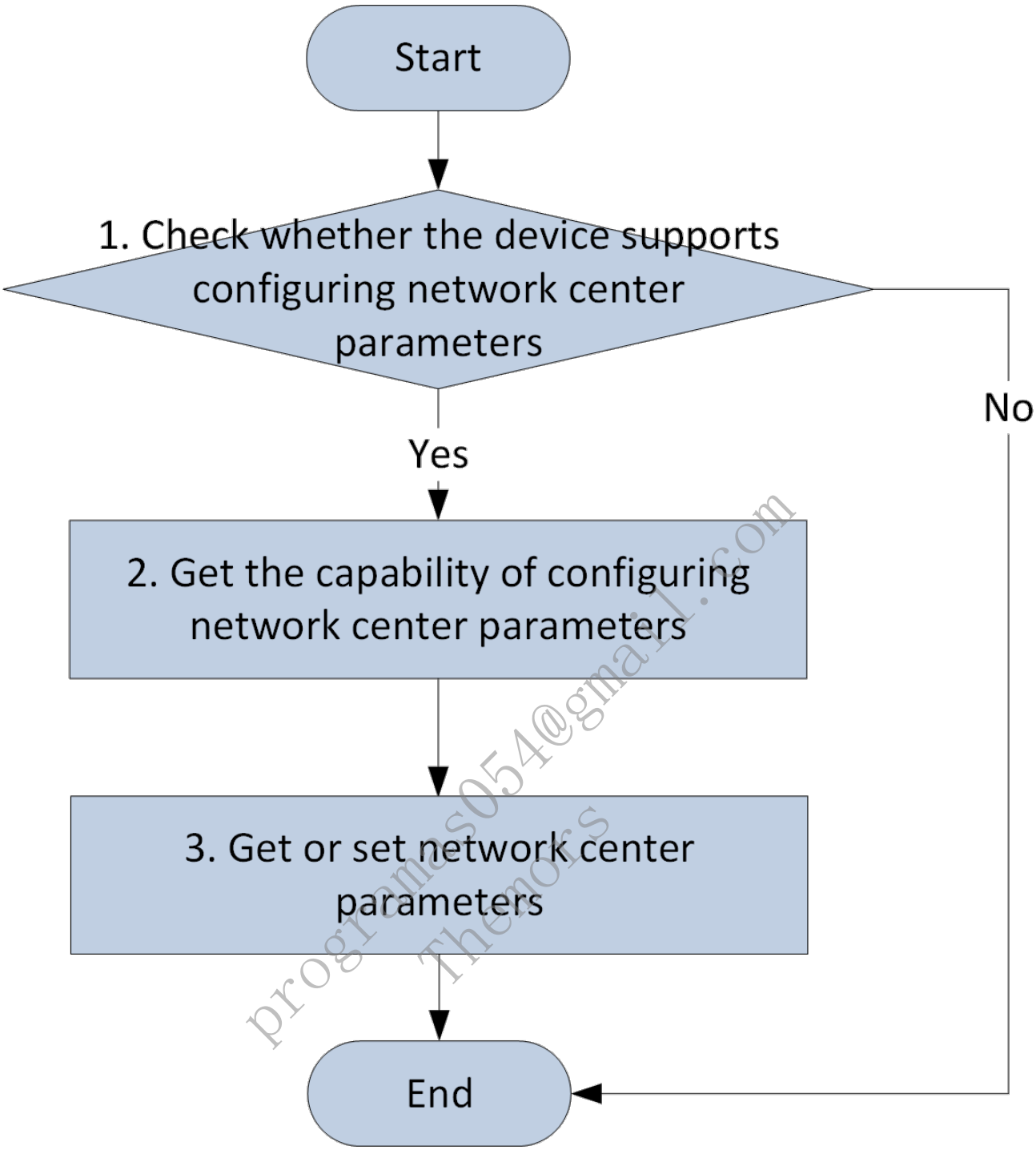
2. Get the email notification capability: `GET /ISAPI/SecurityCP/Configuration/messageSendMail/capabilities?format=json`.

3. Get parameters for multiple email notifications: `GET /ISAPI/SecurityCP/Configuration/messageSendMail?format=json`; set parameters for a single email notification: `PUT /ISAPI/SecurityCP/Configuration/messageSendMail/<mailID>?format=json`. The mailID in the URL is the email ID.

8.6 Network Center Parameter Configuration

For each NIC of the device, you can configure parameters of one or multiple network centers (one NIC can correspond to multiple network centers). Private protocol, NAL2300 protocol, and ISUP are supported currently.

API Calling Flow:



API Calling Steps:

1. Get the configuration capability of the security control panel to check whether the device supports configuring network center parameters: `GET /ISAPI/SecurityCP/Configuration/capabilities?format=json`. If the returned value of the node isSptNetCfg is true, the device supports this function.

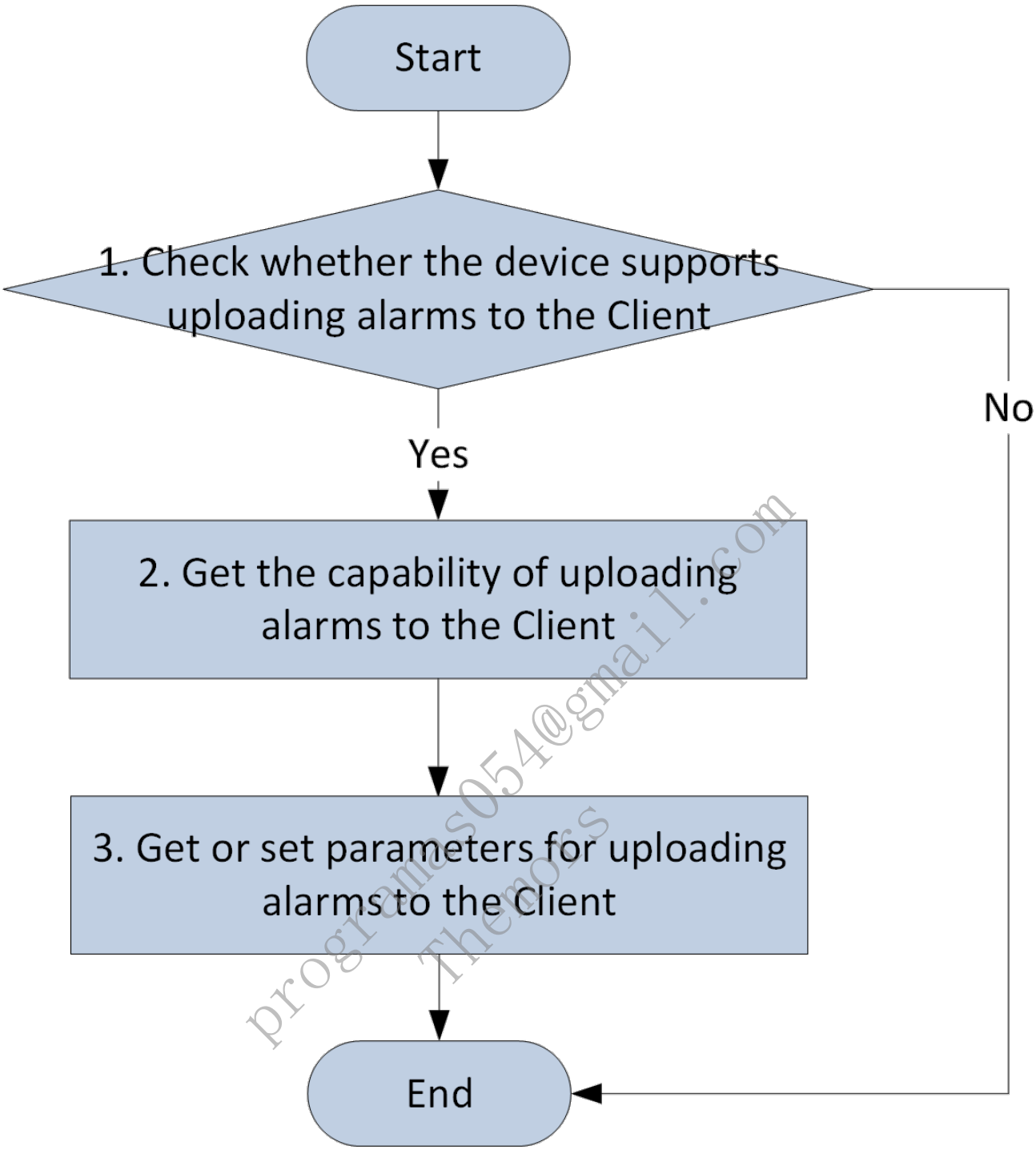
2. Get the capability of configuring network center parameters: `GET /ISAPI/SecurityCP/NetCfg/capabilities?format=json`.

3. Get network center parameters: `GET /ISAPI/SecurityCP/NetCfg/<interfaceID>?format=json`; configure network center parameters: `PUT /ISAPI/SecurityCP/NetCfg/<interfaceID>?format=json`. The interfaceID in the URL refers to the NIC No.: 1 for the main NIC and 2 for the extended NIC.

8.7 Upload Alarms to Client

Events/alarms that can be uploaded to the Client (such as iVMS-4200) include tampering event, life security event, system status event, operation event, emergency alarm, medical alarm, and gas alarm.

API Calling Flow:



API Calling Steps:

1. Get the configuration capability of the security control panel to check whether the device supports uploading alarms to the Client: `GET /ISAPI/SecurityCP/Configuration/capabilities?format=json`. If the returned value of the node isSptDirect is true, the device supports this function.

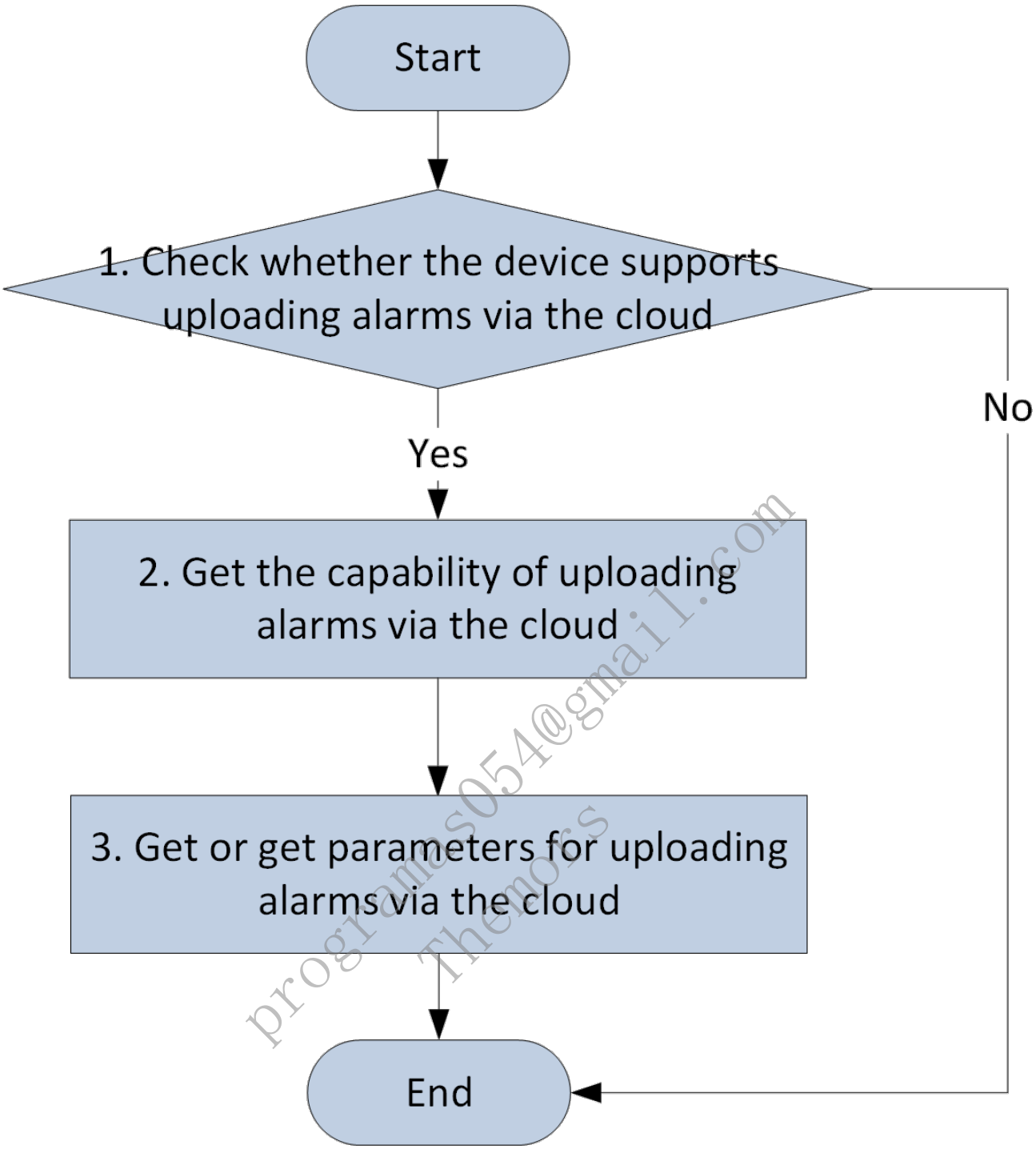
2. Get the capability of uploading alarms to the Client: `GET /ISAPI/SecurityCP/Configuration/messageSendDirect/capabilities?format=json`.

3. Get parameters for uploading alarms to the Client: `GET /ISAPI/SecurityCP/Configuration/messageSendDirect?format=json`; set parameters: `PUT /ISAPI/SecurityCP/Configuration/messageSendDirect?format=json`.

8.8 Upload Alarms via Cloud

Events/alarms that can be uploaded to apps via the cloud include tampering event, life security event, system status event, operation event, emergency alarm, medical alarm, and gas alarm.

API Calling Flow:



API Calling Steps:

1. Get the configuration capability of the security control panel to check whether the device supports uploading alarms via the cloud: `GET /ISAPI/SecurityCP/Configuration/capabilities?format=json`. If the returned value of the node isSptCloud is true, the device supports this function.

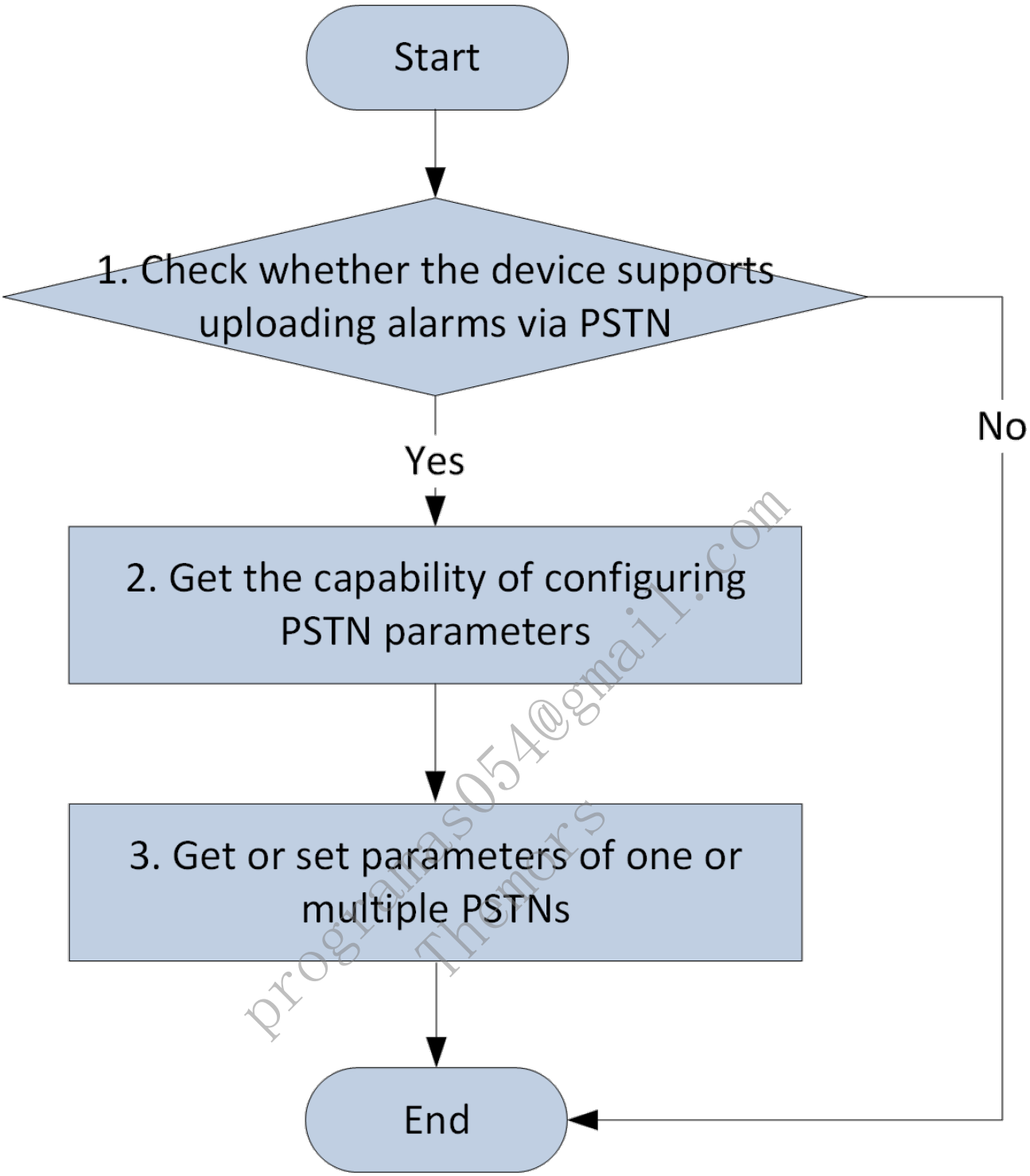
2. Get the capability of uploading alarms via the cloud: `GET /ISAPI/SecurityCP/Configuration/messageSendCloud/capabilities?format=json`.

3. Get parameters for uploading alarms via the cloud: `GET /ISAPI/SecurityCP/Configuration/messageSendCloud?format=json`; set parameters: `PUT /ISAPI/SecurityCP/Configuration/messageSendCloud?format=json`.

8.9 Upload Alarms via PSTN

If the device supports Public Switched Telephone Network (PSTN), the alarms can be uploaded by dial-up and the dial-up target can be an ARC or a person.

API Calling Flow:



API Calling Steps:

1. Get the configuration capability of the security control panel to check whether the device supports uploading alarms via PSTN: `GET /ISAPI/SecurityCP/Configuration/capabilities?format=json`. If the returned value of the node isSptPSTNCfg is true, the device supports this function.

2. Get the capability of configuring PSTN parameters: `GET /ISAPI/SecurityCP/Configuration/PSTNCfg/capabilities?format=json`.

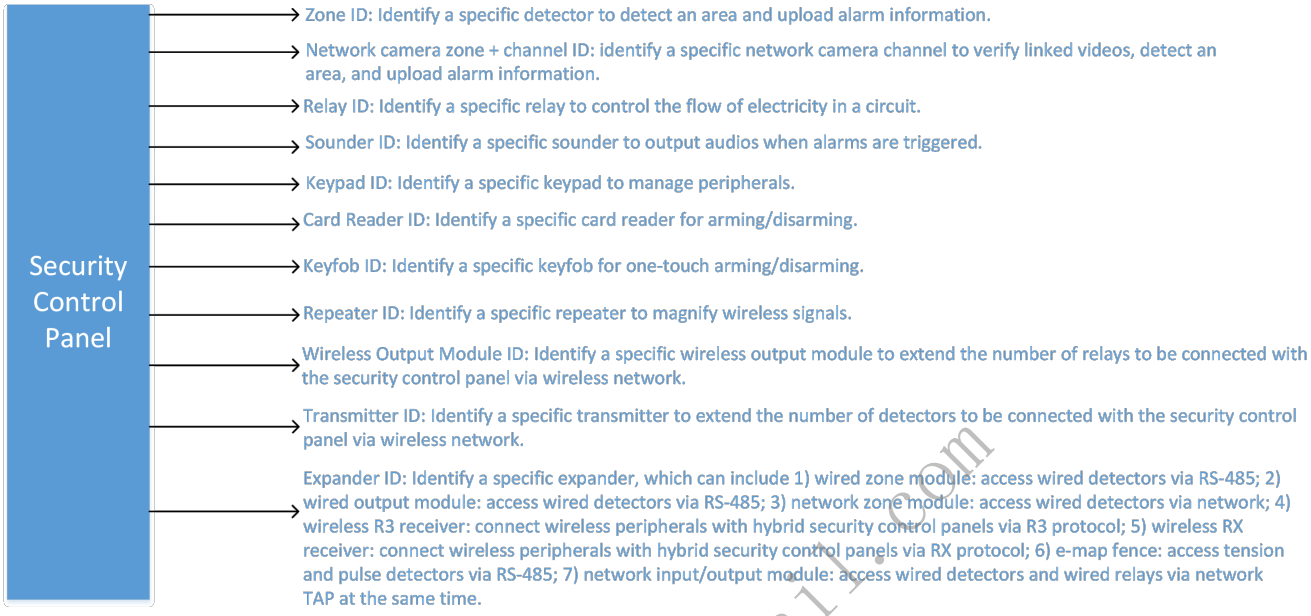
3. Get parameters of all PSTNs: `GET /ISAPI/SecurityCP/Configuration/PSTNCfg?format=json`; set parameters of a single PSTN: `PUT /ISAPI/SecurityCP/Configuration/PSTNCfg/<indexID>?format=json`. Available parameters include whether to enable PSTN link, telephone of the target, communication protocol (CID), and transmission method (0#DTMF 5/S, 1#DTMF 10/S). The indexID in the URL is the PSTN No.

9 Zone Alarm

9.1 Peripheral Management

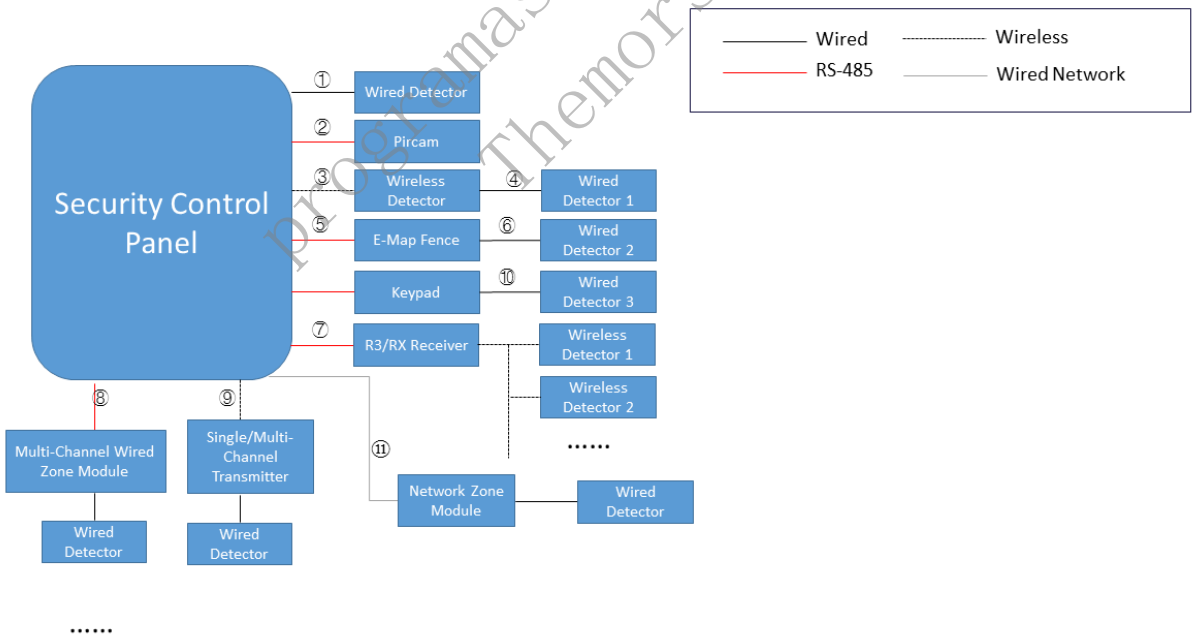
9.1.1 Peripheral No.

The peripheral No. is used to specify a certain peripheral (such as detector, network camera, relay, and sounder) when you perform operations including configuring parameters and upgrading devices. For details about rules, see the picture below:



9.2 Topology of Detectors and Peripherals

9.2.1 Topology of Detectors



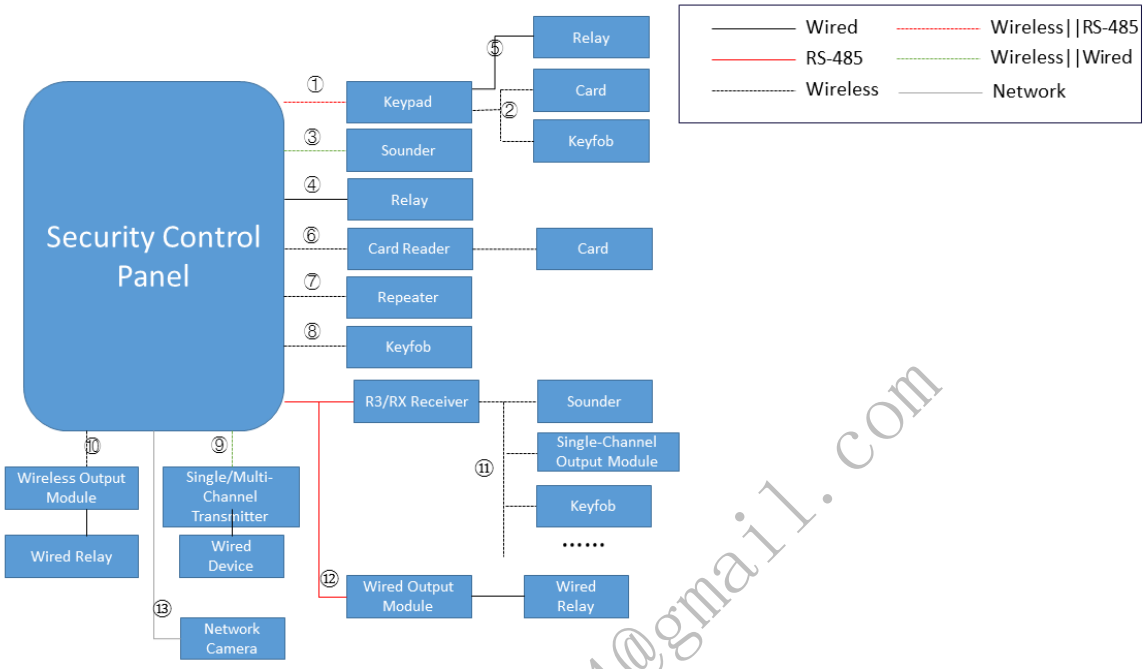
The hybrid security control panel can access detectors in the following ways:

1 (access wired detectors via onboard zones), 2 (access pircams via the RS-485 bus), 5 (access 1 to 2 detectors by connecting the RS-485 bus with e-map fences), 6 (access 1 detector by enabling the function of extending zones for e-map fences), 7 (access wireless detectors by connecting the RS-485 bus with R3/RX receivers), 8 (access wired detectors via multi-channel wired zone modules), 10 (access wired detectors via keypads), 11 (access wired detectors via network zone modules)

The wireless security control panel can access detectors in the following ways:

1 (access wired detectors via onboard transmitters), 3 (access wireless detectors via internal wireless receivers), 4 (access 1 to 2 wired detectors by enabling the function of extending zones for some magnetic contact detectors), 9 (access wired detectors by connecting wireless receivers with single-channel / multi-channel transmitters)

9.2.2 Topology of Peripherals



The hybrid security control panel can access peripherals in the following ways:

1 (access keypads via the RS-485 bus), 2 (access wireless cards and keyfobs via keypads), 3 (access wired sounders via onboard sounder modules), 4 (access wired relays via onboard relay modules), 5 (access wired relays via keypads), 11 (access wireless sounders, single-channel output modules (wireless relays), and keyfobs via the receivers connected to the RS-485 bus), 12 (access wired relays via the wired output module connected to the RS-485 bus), 13 (access network cameras by login via the network)

The wireless security control panel can access peripherals in the following ways:

1 (access keypas via wireless network), 3 (access wireless sounders via internal wireless receivers), 4 (access wired relays via onboard transmitters), 6 (access card readers via wireless network), 7 (access repeaters via wireless network), 8 (access keyfobs via wireless network), 9 (access wired relays by connecting wireless receivers with single-channel / multi-channel transmitters), 10 (access wired relays by connecting wireless receivers with wireless output modules), 13 (access network cameras by login via the network)

10 API Reference

10.1 Device (General)

10.1.1 Event Subscription

10.1.1.1 Get the alarm/event subscription capability

Request URL

GET /ISAPI/Event/notification/subscribeEventCap

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>
<SubscribeEventCap xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, picture uploading modes of all events which contain pictures, attr:version{req, string, protocolVersion}-->
  <heartbeat min="1" max="180">
    <!--ro, opt, int, heartbeat interval time, range:[1,180], unit:s, attr:min{req, int},max{req, int}-->1
  </heartbeat>
  <channelMode opt="all,list">
    <!--ro, opt, enum, channel subscription mode, subType:string, attr:opt{req, string}, desc:"all" (subscribe to all channels), "list" (subscribe to
channels according to channel list)-->list
  </channelMode>
  <EventList>
    <!--ro, opt, array, event type list for subscription, subType:object, desc:this node is valid when eventMode is "list"-->
    <Event>
      <!--ro, opt, object, subscription of a specified alarm/event-->
      <type>
        <!--ro, req, enum, event type, subType:string, desc:refer to event type list (eventType): "ADAS"(advanced driving assistance system), "ADASAlarm"
(advanced driving assistance alarm), "AID"(traffic incident detection), "ANPR"(automatic number plate recognition), "AccessControllerEvent" (access
controller event), "CDsStatus" (CD burning status), "DBD"(driving behavior detection) "GPSUpload" (GPS information upload), "HFPD"(frequently appeared
person detection), "IO"(I/O alarm), "IOTD" (IoT device detection), "LES" (logistics scanning event), "LFPD"(rarely appeared person detection), "PALMismatch"
(video standard mismatch), "PIR", "PeopleCounting" (people counting), "PeopleNumChange" (people number change detection), "Standup"(standing up detection),
"TMA"(thermometry alarm), "TMPA"(temperature measurement pre-alarm), "VMD"(motion detection), "abnormalAcceleration", "abnormalDriving", "advReachHeight",
"alarmResult", "attendance", "attendedBaggage", "audioAbnormal", "audioexception", "behaviorResult"(abnormal event detection), "blindSpotDetection"(blind
spot detection alarm), "cardMatch", "changedStatus", "collision", "containerDetection", "crowdSituationAnalysis", "databaseException", "defocus"(defocus
detection), "diskUnformat"(disk unformatted), "diskerror", "diskfull", "driverConditionMonitor"(driver status monitoring alarm); "emergencyAlarm",
"faceCapture", "faceSnapModeling", "facedetection", "failDown"(People Falling Down), "faultAlarm", "fielddetection"(intrusion detection), "fireDetection",
"fireEscapeDetection", "flowOverrun", "framesPeopleCounting", "getUp"(getting up detection), "group" (people gathering), "hdBadBlock"(HDD bad sector
detection event), "hdImpact"(HDD impact detection event), "heatmap"(heat map alarm), "highHDTemperature"(HDD high temperature detection event),
"highTempAlarm"(HDD high temperature alarm), "hotSpare"(hot spare exception), "illaccess"(invalid access), "ipcTransferAbnormal", "ipconflict"(IP address
conflicts), "keyPersonGetUp"(key person getting up detection), "leavePosition"(absence detection), "linedetection"(line crossing detection),
"listSyncException"(list synchronization exception), "loitering"(loitering detection), "lowHDTemperature"(HDD low temperature detection event),
"mixedTargetDetection"(multi-target-type detection), "modelError", "nicbroken"(network disconnected), "nodeOffline"(node disconnected),
"nonPoliceIntrusion", "overSpeed"(overspeed alarm), "overtimeTarry"(staying overtime detection), "parking"(parking detection), "peopleNumChange",
"peopleNumCounting", "personAbnormalAlarm"(person ID exception alarm), "personDensityDetection", "personQueueCounting", "personQueueDetection",
"personQueueRealTime"(real-time data of people queuing-up detection), "personQueueTime"(waiting time detection), "playCellphone"(playing mobile phone
detection), "pocException"(video exception), "poe"(POE power exception), "policeUser", "radarAlarm", "radarFieldDetection", "radarLineDetection",
"radarPerimeterRule"(radar rule data), "radarTargetDetection", "radarVideoDetection"(radar-assisted target detection), "raidException", "rapidMove",
"reachHeight"(climbing detection), "recordCycleAbnormal"(insufficient recording period), "recordException", "regionEntrance", "regionExiting", "retention"
(people overstay detection), "rollover", "running"(people running), "safetyHelmetDetection"(hard hat detection), "scenechangedetection", "sensorAlarm"
(angular acceleration alarm), "severeHDFailure"(HDD major fault detection), "shelteralarm"(video tampering alarm), "shipsDetection", "sitQuietly"(sitting
detection), "smokeAndFireDetection", "smokeDetection", "softIO", "spacingChange"(distance exception), "sysStorFull"(storaging full alarm of cluster system),
"takingElevatorDetection"(elevator electric moped detection), "targetCapture", "temperature"(temperature difference alarm), "thermometry"(temperature
alarm), "thirdPartyException", "toiletTarry"(in-toilet overtime detection), "tollCodeInfo"(QR code information report), "tossing"(thrown object detection),
"unattendedBaggage", "vehicleMatchResult"(uploading list alarm), "vehicleRcogResult", "versionAbnormal"(cluster version exception), "videoException",
"videoloss", "violationAlarm", "violentMotion"(violent motion detection), "yardTarry"(playground overstay detection), "AccessControllerEvent",
"IDCardInfoEvent", "FaceTemperatureMeasurementEvent", "QRCodeEvent"(QR code event of access control), "CertificateCaptureEvent"(person ID capture comparison
event), "UncertificateCompareEvent", "ConsumptionAndTransactionRecordEvent", "ConsumptionEvent", "TFS" (traffic enforcement event),
"TransactionRecordEvent", "HealthInfoSyncQuery" (health information search event), "SetMealQuery"(searching consumption set meals), "ConsumptionStatusQuery"
(searching the consumption status), "certificateRevocation" (certificate expiry), "humanBodyComparison" (human body comparison),
"regionTargetNumberCounting" (regional target statistics)-->mixedTargetDetection
      </type>
      <minorAlarm opt="0x400,0x401,0x402,0x403">
        <!--ro, opt, string, minor alarm type, attr:opt{req, string}, desc:"IDCardInfoEvent" is required when the type of event is "AccessControllerEvent"--
>0x400,0x401
      </minorAlarm>
      <minorException opt="0x400,0x401,0x402,0x403">
        <!--ro, opt, string, minor exception type, attr:opt{req, string}, desc:"IDCardInfoEvent" is required when the type of event is
"AccessControllerEvent"-->0x400,0x401
      </minorException>
      <minorOperation opt="0x400,0x401,0x402,0x403">
        <!--ro, opt, string, minor operation type, attr:opt{req, string}, desc:"IDCardInfoEvent" is required when the type of event is
"AccessControllerEvent"-->0x400,0x401
      </minorOperation>
      <minorEvent opt="0x01,0x02,0x03,0x04">
        <!--ro, opt, string, minor event type, attr:opt{req, string}, desc:"IDCardInfoEvent" is required when the type of event is "AccessControllerEvent"--
>0x400,0x401
      </minorEvent>
    </Event>
  </EventList>
</SubscribeEventCap>
```


10.1.1.2 Event subscription heartbeat

EventType:heartBeat

```
<?xml version="1.0" encoding="UTF-8"?>

<EventNotificationAlert xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, opt, object, alarm message, attr:version{opt, string, protocolVersion}-->
  <ipAddress>
    <!--ro, req, string, IPv4 address of the device that triggers the alarm-->172.6.64.7
  </ipAddress>
  <ipv6Address>
    <!--ro, opt, string, IPv6 address of the device that triggers the alarm-->1080:0:0:0:8:800:200C:417A
  </ipv6Address>
  <portNo>
    <!--ro, opt, int, communication port No. of the device that triggers the alarm-->80
  </portNo>
  <protocol>
    <!--ro, opt, enum, transmission communication protocol type, subType:string, desc:when ISAPI protocol is transmitted via HCNetSDK, the channel No. is
the video channel No. of private protocol. When ISAPI protocol is transmitted via EZ protocol, the channel No. is the video channel No. of EZ protocol. When
ISAPI protocol is transmitted via ISUP, the channel No. is the video channel No. of ISUP-->HTTP
  </protocol>
  <macAddress>
    <!--ro, opt, string, MAC address-->01:17:24:45:D9:F4
  </macAddress>
  <channelID>
    <!--ro, opt, int, channel No. of the device that triggers the alarm, desc:video channel No. that triggers the alarm-->1
  </channelID>
  <dateTime>
    <!--ro, req, datetime, alarm trigger time-->2004-05-03T17:30:08+08:00
  </dateTime>
  <activePostCount>
    <!--ro, opt, int, times that the same alarm has been uploaded, desc:event triggering frequency-->1
  </activePostCount>
  <eventType>
    <!--ro, req, string, event type-->heartBeat
  </eventType>
  <eventState>
    <!--ro, req, enum, event status, subType:string, desc:for durative event: "active" (valid), "inactive" (invalid)-->active
  </eventState>
  <eventDescription>
    <!--ro, req, string, event description-->heartBeat
  </eventDescription>
  <channelName>
    <!--ro, opt, string, channel name, range:[1,64]-->test
  </channelName>
  <deviceID>
    <!--ro, opt, string, device ID, desc:it should be returned for ISUP alarms, e.g., test0123 (Ehome2.0, Ehome4.0, and ISUP5.0)-->12345
  </deviceID>
</EventNotificationAlert>
```

10.1.2 Port Settings

10.1.2.1 Get the capability of a specific serial port

Request URL

GET /ISAPI/System/Serial/ports/<portID>/capabilities

Query Parameter

Parameter Name	Parameter Type	Description
portID	string	--

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>
<SerialPort xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, opt, object, port No., attr:version{req, string, protocolVersion}-->
  <id>
    <!--ro, req, int, ID-->0
  </id>
  <serialPortType opt="RS485,RS422,RS232">
    <!--ro, opt, string, serial port type, attr:opt{req, string}-->RS485
  </serialPortType>
  <serialAddress min="0" max="10">
    <!--ro, opt, int, serial port address, attr:min{req, int},max{req, int}, desc:serial port address-->1
  </serialAddress>
  <duplexMode opt="half,full">
    <!--ro, opt, string, duplex mode of the serial port, attr:opt{req, string}-->half
  </duplexMode>
  <baudRate opt="1200,2400,4800,9600,19200,38400,57600,115200">
    <!--ro, opt, int, attr:opt{req, string}-->1200
  </baudRate>
  <dataBits opt="6,7,8">
    <!--ro, opt, int, attr:opt{req, string}-->6
  </dataBits>
  <parityType opt="none,even,odd,mark,space">
    <!--ro, opt, string, attr:opt{req, string}-->none
  </parityType>
  <stopBits opt="1,1.5,2">
    <!--ro, opt, string, stop bit, attr:opt{req, string}-->1
  </stopBits>
  <workMode opt="console,transparent,audiomixe,screenCtrl,ptzCtrl,keyboard,matrix,audioMixers">
    <!--ro, opt, string, working mode, attr:opt{req, string}-->console
  </workMode>
  <flowCtrl opt="none,software,hardware">
    <!--ro, opt, string, flowCtrl, attr:opt{req, string}-->none
  </flowCtrl>
  <mode opt="readerMode,clientMode,externMode,stairsControl,accessControlHost,disabled,custom,cardReceiver,QRCodeReader">
    <!--ro, opt, string, working mode, attr:opt{req, string}-->readerMode
  </mode>
  <outputDataType opt="cardNo,employeeNo,auto">
    <!--ro, opt, string, output data type, attr:opt{req, string}, desc:data type output from the door station to the elevator controller: floorNumber (floor
No.,default),cardNo (card No.)-->cardNo
  </outputDataType>
</SerialPort>
```

10.1.2.2 Set the parameters of a specific serial port supported by the device

Request URL

PUT /ISAPI/System/Serial/ports/<portID>?permissionController=<indexID>&childDevID=<childDevID>&deviceIndex=<deviceIndex>

Query Parameter

Parameter Name	Parameter Type	Description
portID	string	--
indexID	string	--
childDevID	string	--
deviceIndex	string	--

Request Message

```
<?xml version="1.0" encoding="UTF-8"?>
<SerialPort xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--opt, object, port, attr:version{req, string, protocolVersion}-->
  <id>
    <!--req, int, serial port ID-->0
  </id>
  <serialPortType>
    <!--opt, enum, serial port type: "RS485","RS422","RS232", subType:string, desc:serial port type: "RS485","RS422","RS232"-->RS485
  </serialPortType>
  <serialAddress>
    <!--opt, int-->1
  </serialAddress>
  <duplexMode>
    <!--opt, enum, duplex mode of the serial port, subType:string, desc:"half", "full"”-->half
  </duplexMode>
  <baudRate>
    <!--opt, enum, subType:int-->2400
  </baudRate>
  <dataBits>
    <!--opt, int-->6
  </dataBits>
  <parityType>
    <!--opt, enum, parity type, subType:string, desc:"none, even, odd, mark, space”-->none
  </parityType>
  <stopBits>
    <!--opt, string, stop bit: "1,1.5,2"-->1
  </stopBits>
  <workMode>
    <!--opt, enum, work mode, subType:string, desc:working mode: "console","transparent","audiomixer","stairsControl"-elevator control,"cardReader"-card
reader,"disabled","custom". This node is required only when <serialPortType> is set to "RS232"-->console
  </workMode>
  <flowCtrl>
    <!--opt, enum, "none,software,hardware", subType:string, desc:"none, software, hardware”-->none
  </flowCtrl>
  <mode>
    <!--opt, enum, work mode, subType:string, desc:deq,working mode: "readerMode,clientMode,externMode,accessControlHost,disabled",this node is valid only
when <serialPortType> is "RS485"-->readerMode
  </mode>
  <outputDataType>
    <!--opt, enum, output data type, subType:string, dep:and,{$.SerialPort.mode,eq,accessControlHost}, desc:"cardNo,employeeNo", this node is valid when
<mode>is "accessControlHost”-->cardNo
  </outputDataType>
</SerialPort>
```

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>
<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response message, attr:version{ro, req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL-->null
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:int, desc:0 (OK), 1 (OK), 2 (Device Busy), 3 (Device Error), 4 (Invalid Operation), 5 (Invalid XML Format), 6
(Invalid XML Content), 7 (Reboot Required)-->0
  </statusCode>
  <statusString>
    <!--ro, req, enum, status description, subType:string, desc:“OK” (succeeded), “Device Busy”, “Device Error”, “Invalid Operation”, “Invalid XML Format”,
“Invalid XML Content”, “Reboot” (reboot device)-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, error reason description in detail, desc:error reason description in detail-->OK
  </subStatusCode>
</ResponseStatus>
```

10.1.2.3 Get the parameters of a specific port supported by the device

Request URL

GET /ISAPI/System/Serial/ports/<portID>?permissionController=<indexID>&childDevID=<childDevID>&deviceIndex=<deviceIndex>

Query Parameter

Parameter Name	Parameter Type	Description
portID	string	--
indexID	string	--
childDevID	string	--
deviceIndex	string	--

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>
<SerialPort xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, opt, object, port, attr:version{req, string, protocolVersion}-->
  <id>
    <!--ro, req, int, serial port ID-->0
  </id>
  <serialPortType>
    <!--ro, opt, enum, serial port type, subType:string, desc:"RS485", "RS422", "RS232"-->RS485
  </serialPortType>
  <serialAddress>
    <!--ro, opt, int-->1
  </serialAddress>
  <duplexMode>
    <!--ro, opt, enum, duplex mode of the serial port, subType:string, desc:"half", "full"-->half
  </duplexMode>
  <baudRate>
    <!--ro, opt, enum, subType:int-->2400
  </baudRate>
  <dataBits>
    <!--ro, opt, int-->6
  </dataBits>
  <parityType>
    <!--ro, opt, enum, parity type, subType:string, desc:parity type: "none,even,odd,mark,space"-->none
  </parityType>
  <stopBits>
    <!--ro, opt, enum, stop bit, subType:string, desc:stop bit-->1
  </stopBits>
  <workMode>
    <!--ro, opt, enum, working mode, subType:string, desc:"console", "transparent", "audiomixer", "screenCtrl", "ptzCtrl", "keyboard", "matrix",
"audioMixers"-->console
  </workMode>
  <flowCtrl>
    <!--ro, opt, enum, "none,software,hardware", subType:string, desc:"none", "software", "hardware"-->none
  </flowCtrl>
  <mode>
    <!--ro, opt, enum, working mode, subType:string, desc:deq,working mode: "readerMode,clientMode,externMode,accessControlHost,disabled",this node is valid
only when <serialPortType> is "RS485"-->readerMode
  </mode>
  <outputDataType>
    <!--ro, opt, enum, output data type, subType:string, dep:and,{$.SerialPort.mode,eq,accessControlHost}, desc:output data type: "cardNo,employeeNo",this
node is valid when <mode>is "accessControlHost"-->cardNo
  </outputDataType>
</SerialPort>
```


10.1.3 System Maintenance

10.1.3.1 Get the system security capability

Request URL

GET /ISAPI/Security/capabilities?username=<userName>&deviceIndex=<deviceIndex>

Query Parameter

Parameter Name	Parameter Type	Description
userName	string	user name
deviceIndex	string	--

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>
<SecurityCap xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, system security capability, attr:version{req, string, protocolVersion}-->
  <supportUserNums>
    <!--ro, opt, int, supported maximum number of users-->0
  </supportUserNums>
  <userBondIpNums>
    <!--ro, opt, int, supported maximum number of IP addresses that can be bound-->0
  </userBondIpNums>
  <userBondMacNums>
    <!--ro, opt, int, supported maximum number of MAC addresses that can be bound-->0
  </userBondMacNums>
  <issupIllegalLoginLock>
    <!--ro, opt, bool, whether the device supports locking login-->true
  </issupIllegalLoginLock>
  <isSupportOnlineUser>
    <!--ro, opt, bool, whether the device supports the online user configuration-->true
  </isSupportOnlineUser>
  <isSupportAnonymous>
    <!--ro, opt, bool, whether the device supports anonymous login-->true
  </isSupportAnonymous>
  <securityVersion opt="1,2">
    <!--ro, opt, int, encryption capability set, attr:opt{req, string}, desc:the encryption capability of each version consists of two parts: encryption
algorithm and the range of encrypted nodes currently 1 refers to AES128 encryption and 2 refers to AES256 encryption, the range of encrypted nodes is
described in each protocol-->1
  </securityVersion>
  <keyIterateNum>
    <!--ro, opt, int, secret key iteration times, dep:or,{$.SecurityCap.securityVersion,eq,1},{$.SecurityCap.securityVersion,eq,2}, desc:this node depends
on the node securityVersion, the range is between 100 and 1000-->100
  </keyIterateNum>
  <isSupportUserCheck>
    <!--ro, opt, bool, whether the device supports verifying the login password when editing (editing/adding/deleting) user parameters, dep:or,
{$.SecurityCap.securityVersion,eq,0},{$.SecurityCap.securityVersion,eq,1}, desc:it is an added capability, which indicates that whether supporting the login
password verification for editing/adding/deleting user parameters, this node depends on the node securityVersion, which means that it is only valid for the
versions that support encrypting the sensitive information-->true
  </isSupportUserCheck>
  <isIrreversible>
    <!--ro, opt, bool, whether the device supports irreversible password storage, desc:If this function is not supported, the plaintext password of the user
information will be stored in the device; otherwise, the password will be hashed for storage in the device.-->true
  </isIrreversible>
  <salt>
    <!--ro, opt, string, the specific salt used by the user to log in-->test
  </salt>
</SecurityCap>
```


10.1.3.2 Get the device's system capability

Request URL

GET /ISAPI/System/capabilities

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>
<DeviceCap xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, system capability of the device, attr:version{opt, string, protocolVersion}-->
  <SysCap>
    <!--ro, opt, object, system capability-->
    <isSupportDst>
      <!--ro, opt, bool, whether the device supports DST (Daylight Saving Time)-->true
    </isSupportDst>
    <SerialCap>
      <!--ro, opt, object, range of RS-485 serial port numbers supported by the device-->
    </SerialCap>
    <isSupportExternalDevice>
      <!--ro, opt, bool, whether the device supports connecting to peripherals: true (support), desc:related URI: /ISAPI/System/externalDevice/capabilities-
->true
    </isSupportExternalDevice>
    <isSupportSubscribeEvent>
      <!--ro, opt, bool, whether the device supports subscribing to events, desc:related URI: /ISAPI/Event/notification/subscribeEventCap-->true
    </isSupportSubscribeEvent>
  </SysCap>
  <isSupportSnapshot>
    <!--ro, opt, bool, whether the device supports capturing pictures-->true
  </isSupportSnapshot>
  <isSupportGIS>
    <!--ro, opt, bool, whether the device supports GIS, desc:related URI: /ISAPI/GIS/channels-->true
  </isSupportGIS>
  <isSupportCompass>
    <!--ro, opt, bool, whether the device supports compass configuration, desc:related URI: /ISAPI/Compass/channels/<ID>/capabilities-->true
  </isSupportCompass>
  <isSupportRoadInfoOverlays>
    <!--ro, opt, bool, whether the device supports overlaying the lane information-->true
  </isSupportRoadInfoOverlays>
  <isSupportFaceCaptureStatistics>
    <!--ro, opt, bool, whether the device supports face capture statistics, desc:related URI: /ISAPI/Intelligent/channels/<ID>/faceCaptureStatistics/search-
->true
  </isSupportFaceCaptureStatistics>
  <isSupportElectronicsEnlarge>
    <!--ro, opt, bool, whether the device supports electronics enlarge-->true
  </isSupportElectronicsEnlarge>
  <isSupportCloud>
    <!--ro, opt, bool, whether the device supports cloud storage-->true
  </isSupportCloud>
  <isSupportRecordHost>
    <!--ro, opt, bool, whether the device supports configuring the education sharing server, desc:related URI: /ISAPI/RecordHost/capabilities-->true
  </isSupportRecordHost>
  <isSupportAcsUpdate>
    <!--ro, opt, bool, whether the device supports peripheral module upgrade (true, support), desc:related URI: /ISAPI/System/AcsUpdate/capabilities-->true
  </isSupportAcsUpdate>
  <isSupportAccessControlCap>
    <!--ro, opt, bool, access control capability, desc:related URI: /ISAPI/AccessControl/capabilities-->true
  </isSupportAccessControlCap>
</DeviceCap>
```


10.1.3.3 Get the languages supported by the device

Request URL

GET /ISAPI/System/DeviceLanguage

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<DeviceLanguage xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, languages supported by the device, attr:version{req, string, protocolVersion}-->
  <language>
    <!--ro, req, enum, language, subType:string, desc:"SimChinese" (simplified Chinese), "TraChinese" (traditional Chinese), "English", "Russian",
"Bulgarian", "Hungarian", "Greek", "German", "Italian", "Czech", "Slovakia", "French", "Polish", "Dutch", "Portuguese", "Spanish", "Romanian", "Turkish",
"Japanese", "Danish", "Swedish", "Norwegian", "Finnish", "Korean", "Thai", "Estonia", "Vietnamese", "Hebrew", "Latvian", "Arabic", "Sovenian"-Slovenian,
"Croatian", "Lithuanian", "Serbian", "BrazilianPortuguese"-Brazilian Portuguese, "Indonesian", "Ukrainian", "EURSpanish", "Sovenian", "Uzbek", "Kazak",
"Kirghiz", "Farsi", "Azerbaidzhan", "Burmese", "Mongolian"-->SimChinese
  </language>
</DeviceLanguage>
```

10.1.3.4 Set device language parameters

Request URL

PUT /ISAPI/System/DeviceLanguage

Query Parameter

None

Request Message

```
<?xml version="1.0" encoding="UTF-8"?>

<DeviceLanguage xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--req, object, languages supported by the device, attr:version{req, string, protocolVersion}-->
  <language>
    <!--req, enum, language, subType:string, desc:"SimChinese" (simplified Chinese), "TraChinese" (traditional Chinese), "English", "Russian", "Bulgarian",
"Hungarian", "Greek", "German", "Italian", "Czech", "Slovakia", "French", "Polish", "Dutch", "Portuguese", "Spanish", "Romanian", "Turkish", "Japanese",
"Danish", "Swedish", "Norwegian", "Finnish", "Korean", "Thai", "Estonia", "Vietnamese", "Hebrew", "Latvian", "Arabic", "Sovenian" (Slovenian), "Croatian",
"Lithuanian", "Serbian", "BrazilianPortuguese" (Brazilian Portuguese), "Indonesian", "Ukrainian", "EURSpanish"-->SimChinese
  </language>
</DeviceLanguage>
```

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response message, attr:version{ro, req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL-->null
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:int, desc:0 (OK), 1 (OK), 2 (Device Busy), 3 (Device Error), 4 (Invalid Operation), 5 (Invalid XML Format), 6
(Invalid XML Content), 7 (Reboot Required)-->0
  </statusCode>
  <statusString>
    <!--ro, req, enum, status information, subType:string, desc:"OK" (succeeded), "Device Busy", "Device Error", "Invalid Operation", "Invalid XML Format",
"Invalid XML Content", "Reboot" (reboot device)-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, sub status code, which describes the error in details, desc:sub status code, which describes the error in details-->OK
  </subStatusCode>
</ResponseStatus>
```


10.1.3.5 Get the capability of configuring the device language

Request URL

GET /ISAPI/System/DeviceLanguage/capabilities

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<DeviceLanguage xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, opt, object, device language configuration, attr:version{req, string, protocolVersion}-->
  <language
opt="SimChinese,TraChinese,English,Russian,Bulgarian,Hungarian,Greek,German,Italian,Czech,Slovakia,French,Polish,Dutch,Portuguese,Spanish,Romanian,Turkish,J
apanese,Danish,Swedish,Norwegian,Finnish,Korean,Thai,Estonia,Vietnamese,Hebrew,Latvian,Arabic,Sovenian,Croatian,Lithuanian,Serbian,BrazilianPortuguese,Indon
esian,Ukrainian,EURSpanish,Uzbek,Kazak,Kirghiz,Farsi,Azerbaidzhan,Burmese,Mongolian,Anglicism,Estonian">
    <!--ro, req, enum, language, subType:string, attr:opt{req, string}, desc:"SimChinese" (simplified Chinese), "TraChinese" (traditional Chinese),
"English", "Russian", "Bulgarian", "Hungarian", "Greek", "German", "Italian", "Czech", "Slovakia", "French", "Polish", "Dutch", "Portuguese", "Spanish",
"Romanian", "Turkish", "Japanese", "Danish", "Swedish", "Norwegian", "Finnish", "Korean", "Thai", "Estonia", "Vietnamese", "Hebrew", "Latvian", "Arabic",
"Sovenian"-Slovenian, "Croatian", "Lithuanian", "Serbian", "BrazilianPortuguese"-Brazilian Portuguese, "Indonesian", "Ukrainian", "EURSpanish", "Sovenian",
"Uzbek", "Kazak", "Kirghiz", "Farsi", "Azerbaidzhan", "Burmese", "Mongolian"-->SimChinese
  </language>
  <upgradeFirmWareEnabled>
    <!--ro, opt, bool, whether to enable upgrading the firmware-->true
  </upgradeFirmWareEnabled>
</DeviceLanguage>
```

10.1.3.6 Get the network service capability

Request URL

GET /ISAPI/System/Network/capabilities

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>
<NetworkCap xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, get the network service capability, attr:version{opt, string, protocolVersion}-->
  <isSupportWireless>
    <!--ro, req, bool, whether the device supports wireless network-->true
  </isSupportWireless>
  <isSupportNtp>
    <!--ro, opt, bool, whether the device supports NTP (Network Time Protocol)-->true
  </isSupportNtp>
  <isSupportSSH>
    <!--ro, opt, bool, whether the device supports SSH-->true
  </isSupportSSH>
  <isSupportEhome>
    <!--ro, opt, bool, whether the device supports EHome (ISUP) protocol-->true
  </isSupportEhome>
  <isSupportWirelessServer>
    <!--ro, opt, bool, whether the device supports wireless server-->true
  </isSupportWirelessServer>
</NetworkCap>
```

10.1.3.7 Set SSH parameters

Request URL

PUT /ISAPI/System/Network/ssh?readerID=<readerID>&type=<type>&SOCChipID=<SOCChipID>

Query Parameter

Parameter Name	Parameter Type	Description
readerID	string	--
type	enum	--
SOCChipID	string	--

Request Message

```
<?xml version="1.0" encoding="UTF-8"?>

<SSH xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--wo, opt, object, attr:version{opt, string, protocolVersion}-->
  <enabled>
    <!--wo, req, bool, whether to enable the function-->true
  </enabled>
  <port>
    <!--wo, opt, int-->22
  </port>
</SSH>
```

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response message, attr:version{ro, req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL-->null
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:int, desc:0 (OK), 1 (OK), 2 (Device Busy), 3 (Device Error), 4 (Invalid Operation), 5 (Invalid XML Format), 6
(Invalid XML Content), 7 (Reboot Required)-->0
  </statusCode>
  <statusString>
    <!--ro, req, enum, read-only,status description: OK,Device Busy,Device Error,Invalid Operation,Invalid XML Format,Invalid XML Content,Reboot,Additional
Error, subType:string, desc:"OK" (succeeded), "Device Busy", "Device Error", "Invalid Operation", "Invalid XML Format", "Invalid XML Content", "Reboot"
(reboot device)-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, sub status code, which describes the error in details, desc:sub status code, which describes the error in details-->OK
  </subStatusCode>
</ResponseStatus>
```

10.1.3.8 Reboot device

Request URL

PUT /ISAPI/System/reboot?childDevID=<devIndex>&module=<module>&loginPassword=<loginPassword>

Query Parameter

Parameter Name	Parameter Type	Description
devIndex	string	--
module	enum	--
loginPassword	string	--

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response message, attr:version{ro, req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL-->null
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:int, desc:0 (OK), 1 (OK), 2 (Device Busy), 3 (Device Error), 4 (Invalid Operation), 5 (Invalid XML Format), 6
(Invalid XML Content), 7 (Reboot Required)-->0
  </statusCode>
  <statusString>
    <!--ro, req, enum, status description, subType:string, desc:“OK” (succeeded), “Device Busy”, “Device Error”, “Invalid Operation”, “Invalid XML Format”,
“Invalid XML Content”, “Reboot” (reboot device)-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, sub status code, desc:sub status code-->OK
  </subStatusCode>
</ResponseStatus>
```

10.1.4 System Upgrade

10.1.4.1 Get the capabilities of peripheral module upgrade

Request URL

GET /ISAPI/System/AcsUpdate/capabilities

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<AcsUpdate xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, capabilities of peripheral module upgrade, attr:version{req, string, protocolVersion}-->
  <type
opt="cardReader,FPModule,securityModule,extendModule,channelController,IRModule,lampModule,elevatorController,FPAlgorithmProgram,uboot,keypad,wirelessRecv,w
iredZone,sirenIndoor,sirenOutdoor,sirenAudio,repeater,bluetoothModule,single,wallSwitch,smartPlug,detector,transmitter,remoteCtrl,subModule,dispModule,netRe
ader,lockControlBoard,networkZoneModule,userInterfaceBoard,subPermissionController,electricLock,heatingModule,RS485Module,QRCodeModule,R3WirelessRecv,RXWire
lessRecv,wiredOutput,electricGenie,zigbeeModule,PMRModule,networkInputAndOutputModule,infoModule,keypadAndCardReader,socialSecurityCardModule,localControlle
r">
    <!--ro, opt, enum, upgrade type: upgrade peripheral, subType:string, attr:opt{req, string}, desc:"cardReader" (485 card reader), "FPModule" (fingerprint
module), "securityModule" (security module), "extendModule" (I/O extended module), "channelController" (lane controller), "IRModule" (infrared module),
"lampModule" (indicator module), "elevatorController" (sub elevator controller), "FPAlgorithmProgram" (fingerprint algorithm programma of card reader),
"uboot" (uboot upgrade), "keypad" (keypad), "wirelessRecv" (wireless receiver module), "wiredZone" (wired zone module), "sirenIndoor" (indoor sounder),
"sirenOutdoor" (outdoor sounder), "sirenAudio" (sounder two-way audio), "repeater" (repeater), "bluetoothModule" (bluetooth module), "single" (single output
module), "wallSwitch" (wall mounted switch), "smartPlug" (smart plug), "detector" (detector), "transmitter" (transmitter peripheral), "remoteCtrl" (keyfob),
"subModule" (sub module), "dispModule" (display module), "netReader" (network card reader), "faceModule" (face module), "touchScreenModule" (touch screen
module), "temperatureModule" (temperature measurement module), "lockControlBoard" (lock control board), "networkZoneModule" (network zone module),
"userInterfaceBoard" (interface board), "subPermissionController" (sub permission controller), "electricLock" (electric lock), "heatingModule" (heating
module), "RS485Module" (RS-485 module), "QRCodeModule" (QR code module), "electricGenie" (electric Genie), "zigbeeModule" (zigbee module), "PMRModule" (PMR
module), "networkInputAndOutputModule" (network input and output module via network TAP)-->cardReader
  </type>
  <batchType opt="electricGenie,wiredOutput,networkZoneModule,networkInputAndOutputModule">
    <!--ro, opt, enum, batch upgrade type (peripheral type), subType:string, attr:opt{req, string},
desc:peripheral types which support batch upgrading;
if this node is supported, the following operations are supported: 1. Upgrade devices: /ISAPI/System/updateFirmware?type=<type>; 2. Start upgrading the
specified analysis unit (device of the cluster): /ISAPI/System/upgradeStatus/startUpgrade?format=json; 3. Get the device upgrade progress:
/ISAPI/System/upgradeStatus?format=json-->electricGenie
  </batchType>
  <cardReaderNo min="1" max="10" opt="1,4">
    <!--ro, opt, int, card reader No. range, it is valid when the returned value of type consists cardReader, attr:min{opt, int, step:1},max{opt, int,
step:1},opt{opt, string}-->1
  </cardReaderNo>
  <FPModuleNo min="1" max="10">
    <!--ro, opt, int, fingerprint module No. range, it is valid when the returned value of type consists FPModule, attr:min{req, int},max{req, int}-->1
  </FPModuleNo>
  <securityModuleNo min="1" max="10">
    <!--ro, opt, int, security module No. range, it is valid when the returned value of type consists securityModule, attr:min{req, int},max{req, int}-->1
  </securityModuleNo>
  <extendModuleNo min="1" max="10">
    <!--ro, opt, int, No. range of I/O extended module, it is valid when the returned value of type consists extendModule, attr:min{req, int},max{req, int}-
->1
  </extendModuleNo>
  <channelControllerNo min="1" max="10">
    <!--ro, opt, int, lane controller No. range, it is valid when the returned value of type consists channelController, attr:min{req, int},max{req, int}--
>1
  </channelControllerNo>
  <IRModuleNo min="1" max="10">
    <!--ro, opt, int, infrared module No. range, it is valid when the returned value of type consists IRModule, attr:min{req, int},max{req, int}-->1
  </IRModuleNo>
  <lampModuleNo min="1" max="10">
    <!--ro, opt, int, indicator module No. range, it is valid when the returned value of type consists lampModule, attr:min{req, int},max{req, int}-->1
  </lampModuleNo>
  <elevatorControllerNo min="1" max="10">
    <!--ro, opt, int, sub elevator controller No. range, it is valid when the returned value of type consists elevatorController, attr:min{req,
int},max{req, int}-->1
  </elevatorControllerNo>
  <FPAlgorithmProgramNo min="1" max="10">
    <!--ro, opt, int, No. range of fingerprint algorithm programma of card reader, it is valid when the returned value of type consists FPAlgorithmProgram,
attr:min{req, int},max{req, int}-->1
  </FPAlgorithmProgramNo>
  <faceModuleNo min="1" max="10">
    <!--ro, opt, int, face module No. range, it is valid when the returned value of type consists faceModule, attr:min{req, int},max{req, int}-->1
  </faceModuleNo>
  <touchScreenModuleNo min="1" max="10">
    <!--ro, opt, int, touch screen module No. range, it is valid when the returned value of type consists touchScreenModule, attr:min{req, int},max{req,
int}-->1
  </touchScreenModuleNo>
  <temperatureModuleNo min="1" max="10">
    <!--ro, opt, int, No. range of temperature measurement module, it is valid when the returned value of type consists temperatureModule, attr:min{req,
int},max{req, int}-->1
  </temperatureModuleNo>
  <keypadAddress opt="1,3,5">
    <!--ro, opt, int, keypad address range, it is valid when the returned value of type consists keypad, attr:opt{req, string}-->1
  </keypadAddress>
  <wirelessRecvAddress opt="1,3,5">
    <!--ro, opt, int, No. range of wireless receiver module, it is valid when the returned value of type consists wirelessRecv, attr:opt{req, string}-->1
  </wirelessRecvAddress>
  <wiredZoneAddress opt="1,3,5">
    <!--ro, opt, int, No. range of wired zone module, it is valid when the returned value of type consists wiredZone, attr:opt{req, string}-->1
  </wiredZoneAddress>
  <sirenIndoorNo opt="1,3,5">
    <!--ro, opt, int, indoor sounder No. range, it is valid when the returned value of type consists sirenIndoor, attr:opt{req, string}-->1
  </sirenIndoorNo>
  <sirenOutdoorNo opt="1,3,5">
    <!--ro, opt, int, outdoor sounder No. range, it is valid when the returned value of type consists sirenOutdoor, attr:opt{req, string}-->1
  </sirenOutdoorNo>
  <repeaterNo opt="1,3,5">
    <!--ro, opt, int, repeater No. range, it is valid when the returned value of type consists repeater, attr:opt{req, string}-->1
  </repeaterNo>
```

```
  <subModuleNo opt="1,3,5">
    <!--ro, opt, int, sub module No. range, it is valid when the returned value of type consists subModule, attr:opt{req, string}-->1
  </subModuleNo>
  <dispModuleNo opt="1,3,5">
    <!--ro, opt, int, display module No. range, it is valid when the returned value of type consists dispModule, attr:opt{req, string}-->1
  </dispModuleNo>
  <single opt="1,3,5">
    <!--ro, opt, int, No. range of single output module, it is valid when the returned value of type consists single, attr:opt{req, string}-->1
  </single>
  <wallSwitch opt="1,3,5">
    <!--ro, opt, int, No. range of wall mounted switch, it is valid when the returned value of type consists wallSwitch, attr:opt{req, string}-->1
  </wallSwitch>
  <smartPlug opt="1,3,5">
    <!--ro, opt, int, smart plug No. range, it is valid when the returned value of type consists smartPlug, attr:opt{req, string}-->1
  </smartPlug>
  <zoneNo opt="1,3,5">
    <!--ro, opt, int, zone  No. range, it is valid when the returned value of type consists detector, attr:opt{req, string}, desc:get the detector via the
zone No.-->1
  </zoneNo>
  <transmitterNo opt="1,3,5">
    <!--ro, opt, int, transmitter No. range, it is valid when the returned value of type consists transmitter, attr:opt{req, string}-->1
  </transmitterNo>
  <remoteCtrlNo opt="1,3,5">
    <!--ro, opt, int, keyfob No. range, it is valid when the returned value of type consists remoteCtrl, attr:opt{req, string}-->1
  </remoteCtrlNo>
  <netReaderNo min="1" max="10">
    <!--ro, opt, int, No. range of network card reader, it is valid when the returned value of type consists netReader, attr:min{req, int},max{req, int}-->1
  </netReaderNo>
  <lockControlBoardLNo opt="1,2,3,4,5,6,7,8">
    <!--ro, opt, int, No. range of the left part of the bracket, it is valid when the value of type returns, attr:opt{req, string}-->1
  </lockControlBoardLNo>
  <lockControlBoardRNo opt="1,2,3,4,5,6,7,8">
    <!--ro, opt, int, No. range of the right part of the bracket, it is valid when the value of type returns, attr:opt{req, string}-->1
  </lockControlBoardRNo>
  <networkZoneModuleNo opt="1,3,5">
    <!--ro, opt, int, No. range of network zone module, it is valid when the returned value of type consists networkZoneModule, attr:opt{req, string}-->1
  </networkZoneModuleNo>
  <userInterfaceBoardNo opt="0,1">
    <!--ro, opt, enum, interface board No. range, it is valid when the returned value of type consists userInterfaceBoard, subType:string, attr:opt{req,
string}, desc:0-main user extended interface board, 1-sub user extended interface board-->0
  </userInterfaceBoardNo>
  <electricLockNo opt="1,3,5">
    <!--ro, opt, int, electric lock No. range, it is valid when the returned value of type consists electricLock, attr:opt{req, string}-->1
  </electricLockNo>
  <heatingModuleNo opt="1,3,5">
    <!--ro, opt, int, heating module No. range, it is valid when the returned value of type consists heatingModule, attr:opt{req, string}, desc:the heating
module starts to work to increase the device temperature when the devices runs under low temperature-->1
  </heatingModuleNo>
  <sirenAudioNo opt="1,3,5">
    <!--ro, opt, int, sounder (two-way audio) No. range, it is valid when the returned value of type consists sirenAudio, attr:opt{req, string}-->1
  </sirenAudioNo>
  <QRCodeModuleNo min="1" max="10">
    <!--ro, opt, int, No. range of QR code module, it is valid when the returned value of type consists QRCodeModule, attr:min{req, int},max{req, int}-->1
  </QRCodeModuleNo>
  <R3WirelessRecvNo opt="1,2,3,4">
    <!--ro, opt, int, No. range of R3 wireless receiver module, it is valid when the returned value of type consists R3WirelessRecv, attr:opt{req, string}--
>1
  </R3WirelessRecvNo>
  <RXWirelessRecvNo opt="1,2,3,4">
    <!--ro, opt, int, No. range of RX wireless receiver module, it is valid when the returned value of type consists RXWirelessRecv, attr:opt{req, string}--
>1
  </RXWirelessRecvNo>
  <wiredOutputNo opt="1,2,3,4">
    <!--ro, opt, int, No. range of wired output module , it is valid when the returned value of type consists wiredOutput, attr:opt{req, string}-->1
  </wiredOutputNo>
  <electricGenieNo opt="1,2,3,4">
    <!--ro, opt, int, No. range of electric Genie, it is valid when the returned value of type or batchType consists electricGenie, attr:opt{req, string}--
>1
  </electricGenieNo>
  <networkInputAndOutputModuleNo opt="1,2,3,4">
    <!--ro, opt, int, No. range of network input and output module, it is valid when the returned value of type consists networkInputAndOutputModule,
attr:opt{req, string}-->1
  </networkInputAndOutputModuleNo>
  <infoModuleNo opt="1,2,3,4">
    <!--ro, opt, int, dep:or,{$.AcsUpdate.type,eq,infoModule}, attr:opt{req, string}-->1
  </infoModuleNo>
  <keypadAndCardReaderModuleNo opt="1,2,3,4">
    <!--ro, opt, int, dep:or,{$.AcsUpdate.type,eq,keypadAndCardReaderModule}, attr:opt{req, string}-->1
  </keypadAndCardReaderModuleNo>
  <threeDimensionalStructuredLightModuleNo opt="1,2,3,4">
    <!--ro, opt, int, dep:or,{$.AcsUpdate.type,eq,threeDimensionalStructuredLightModule}, attr:opt{req, string}-->1
  </threeDimensionalStructuredLightModuleNo>
  <socialSecurityCardModuleNo opt="1">
    <!--ro, opt, int, dep:or,{$.AcsUpdate.type,eq,socialSecurityCardModule}, attr:opt{req, string}-->1
  </socialSecurityCardModuleNo>
  <localControllerNo min="1" max="62">
    <!--ro, opt, int, attr:min{req, int},max{req, int}-->1
  </localControllerNo>
</AcsUpdate>
```

10.1.4.2 Upgrade devices

Request URL

POST /ISAPI/System/updateFirmware?type=<type>&moduleAddress=<moduleAddress>&id=<indexID>&childDevID=<devIndex>&isUpdateAuxDevice=<isUpdateAuxDevice>

Query Parameter

Parameter Name	Parameter Type	Description
type	enum	device type 1.alarm device: "keypad" (keypad), "wirelessRecv" (wireless receiver module), "wiredZone" (wired zone module), "sirenIndoor" (indoor sounder), "sirenOutdoor" (outdoor sounder), "sirenAudio" (sounder two-way audio), "repeater" (repeater), "bluetoothModule" (bluetooth module), "single" (single output module), "wallSwitch" (wall mounted switch), "smartPlug" (smart plug), "detector" (detector), "transmitter" (transmitter peripheral), "remoteCtrl" (keyfob), "subModule" (sub module), "dispModule" (display module), "netReader" (network card reader), "faceModule" (face module), "touchScreenModule" (touch screen module), "temperatureModule" (temperature measurement module), "lockControlBoard" (lock control board), "networkZoneModule" (network zone module), "userInterfaceBoard" (interface board), "subPermissionController" (sub permission controller), "electricLock" (electric lock), "heatingModule" (heating module), "RS485Module" (RS-485 module), "QRCodeModule" (QR code module), "electricGenie" (electric Genie), "zigbeeModule" (zigbee module), "PMRModule" (PMR module), "networkInputAndOutputModule" (network input and output module via network TAP) 2. access control device: "cardReader" (485 card reader), "FPModule" (fingerprint module), "securityModule" (security module), "extendModule" (I/O extended module), "channelController" (lane controller), "IRModule" (infrared module), "lampModule" (indicator module), "elevatorController" (sub elevator controller), "FPAlgorithmProgram" (fingerprint algorithm program of card reader), "bluetoothModule" (bluetooth module), "MCU"-MCU upgrade, "temperatureModule" (temperature measurement module), "faceModule" (face module), "touchScreenModule" (touch screen module), "netReader" (network card reader), "userInterfaceBoard" (interface board), "subPermissionController" (sub permission controller), "QRCodeModule" (QR code module) 3. intelligent cabinet: "lockControlBoard" (lock control board) 4. "ledReceiverCard"-receiving card; 5. "subsystem"-sub system; 6. radar-assisted device: "warningScreenFirmware"-firmware upgrade of pre-alarm screen, "warningScreenFont"-font library upgrade of pre-alarm screen, "warningScreenVoice"-audio upgrade of pre-alarm screen; 7. "electricGenie"-electric Genie
moduleAddress	string	module address is an exclusive node for Ax Hybrid example: keypad: GET /ISAPI/SecurityCP/Configuration/outputs?format=json; relay: GET /ISAPI/SecurityCP/Configuration/outputs?format=json; sounder: GET /ISAPI/SecurityCP/Configuration/wirelessSiren?format=json; output module: GET /ISAPI/SecurityCP/Configuration/outputModules?format=json; extension module: GET /ISAPI/SecurityCP/Configuration/extensionModule?format=json; zone: GET /ISAPI/SecurityCP/Configuration/zones?format=json
indexID	string	device No. 1. AX Hybrid Pro and wireless security control upgrade by ID; 2. access control devices upgrade by ID; 3. intelligent cabinet upgrade by IS
devIndex	string	--
isUpdateAuxDevice	string	--

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response message, attr:version{ro, req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL, range:[0,1024]-->null
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:int, desc:0 (OK), 1 (OK), 2 (Device Busy), 3 (Device Error), 4 (Invalid Operation), 5 (Invalid XML Format), 6
(Invalid XML Content), 7 (Reboot Required)-->0
  </statusCode>
  <statusString>
    <!--ro, req, enum, status information, subType:string, desc:"OK" (succeeded), "Device Busy", "Device Error", "Invalid Operation", "Invalid XML Format",
"Invalid XML Content", "Reboot" (reboot device)-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, sub status code, desc:it describes the error in details-->OK
  </subStatusCode>
  <description>
    <!--ro, opt, string, custom error information, range:[0,1024], desc:this node is used for debugging-->badXmlFormat
  </description>
</ResponseStatus>
```

10.1.4.3 Get the device upgrade progress

Request URL

GET /ISAPI/System/upgradeStatus?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "requestURL":  "/ISAPI/Streaming/channels/1",
    /*ro, opt, string, request URL*/
    "statusCode":  "test",
    /*ro, req, string, status code*/
    "statusString":  "test",
    /*ro, req, string, status description*/
    "subStatusCode":  "test",
    /*ro, req, string, sub status code*/
    "errorCode":  1,
    /*ro, opt, int, This field is required when the value of statusCode is not 1, and it corresponds to subStatusCode.*/
    "errorMsg":  "ok",
    /*ro, opt, string, This field is required when the value of statusCode is not 1. Detailed error description of a certain parameter can be provided*/
    "upgrading":  "TRUE",
    /*ro, opt, string, whether the device is upgrading: “TRUE" (upgrading), "FALSE" (not in upgrading)*/
    "percent":  22,
    /*ro, opt, int, upgrade progress (% complete)*/
    "idList": [
    /*ro, opt, array, ID list, subType:object*/
        {
            "id":  "test",
            /*ro, req, string, analysis unit ID*/
            "percent":  22,
            /*ro, opt, int, upgrade progress (% complete)*/
            "status":  "test"
            /*ro, opt, string, "backingUp" (backing up upgrade)*/
        }
    ]
}
```

10.1.4.4 Get the device upgrading status and progress

Request URL

GET /ISAPI/System/upgradeStatus?type=<Type>&childDevID=<devIndex>

Query Parameter

Parameter Name	Parameter Type	Description
Type	string	--
devIndex	string	--

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<upgradeStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, upgrade status and result, attr:version{req, string, protocolVersion}-->
  <upgrading>
    <!--ro, req, bool, upgrade status-->true
  </upgrading>
  <percent>
    <!--ro, req, int, upgrade progress (% complete), range:[0,100]-->1
  </percent>
</upgradeStatus>
```

10.1.5 Time Settings

10.1.5.1 Get device time synchronization management parameters

Request URL

GET /ISAPI/System/time

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<Time xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, opt, object, time management, attr:version{opt, string, protocolVersion}-->
  <timeMode>
    <!--ro, req, enum, time synchronization mode, subType:string, desc:“NTP” (NTP time synchronization), “manual” (manual time synchronization), “satellite” (satellite time synchronization), “platform” (platform time synchronization), “NONE” (time synchronization is not allowed or no time synchronization source), “GB28181” (GB28181 time synchronization)-->NTP
  </timeMode>
  <localTime>
    <!--ro, opt, string, local time, range:[0,256], dep:and,{$.Time.timeMode,eq,manual}-->2019-02-28T10:50:44+08:00
  </localTime>
  <timeZone>
    <!--ro, opt, string, time zone, range:[0,256], dep:and,{$.Time.timeMode,eq,manual},{$.Time.timeMode,eq,NTP}-->CST-8:00:00DST00:30:00,M4.1.0/02:00:00,M10.5.0/02:00:00
  </timeZone>
  <satelliteInterval>
    <!--ro, opt, int, satellite time synchronization interval, step:1, unit:min, desc:unit: minute-->60
  </satelliteInterval>
  <isSummerTime>
    <!--ro, opt, bool, whether the device time returned currently is in DST (Daylight Saving Time) system-->true
  </isSummerTime>
  <platformType>
    <!--ro, opt, enum, platform type, subType:string, dep:and,{$.Time.timeMode,eq,platform}, desc:exists only when the timeMode is selected as platform-->EZVIZ
  </platformType>
  <platformNo>
    <!--ro, opt, int, platform No., range:[1,2], dep:and,{$.Time.timeMode,eq,GB28181}, desc:it is the only ID, which is configured via platformNo in GB28181List, related URI: /ISAPI/System/Network/SIP/<SIPServerID>-->1
  </platformNo>
</Time>
```

10.1.5.2 Set device time synchronization management parameters

Request URL

PUT /ISAPI/System/time

Query Parameter

None

Request Message

```
<?xml version="1.0" encoding="UTF-8"?>

<Time xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--req, object, time management, attr:version{opt, string, protocolVersion}-->
  <timeMode>
    <!--req, enum, time synchronization mode, subType:string, desc:"NTP" (NTP time synchronization), "manual" (manual time synchronization), "satellite"
(satellite time synchronization), "platform" (platform time synchronization), "NONE" (time synchronization is not allowed or no time synchronization
source), "GB28181" (GB28181 time synchronization)-->NTP
  </timeMode>
  <localTime>
    <!--opt, string, local time, range:[0,256], dep:and,{$.Time.timeMode,eq,manual}-->2019-02-28T10:50:44+08:00
  </localTime>
  <timeZone>
    <!--opt, string, time zone, range:[0,256], dep:and,{$.Time.timeMode,eq,manual},{$.Time.timeMode,eq,NTP}-->CST-
8:00:00DST00:30:00,M4.1.0/02:00:00,M10.5.0/02:00:00
  </timeZone>
  <satelliteInterval>
    <!--opt, int, satellite time synchronization interval, step:1, unit:min, desc:unit: minute-->60
  </satelliteInterval>
  <isSummerTime>
    <!--opt, bool, whether the time returned by the current device is that in the DST (daylight saving time)-->true
  </isSummerTime>
  <platformType>
    <!--opt, enum, platform type, subType:string, dep:and,{$.Time.timeMode,eq,platform}, desc:exists only when the timeMode is selected as platform, related
URI: /ISAPI/System/Network/EZVIZ-->EZVIZ
  </platformType>
  <platformNo>
    <!--opt, int, platform No., range:[1,2], dep:and,{$.Time.timeMode,eq,GB28181}, desc:it is the only ID, which is configured via platformNo in
GB28181List, related URI: /ISAPI/System/Network/SIP/<SIPServerID>-->1
  </platformNo>
</Time>
```

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response message, attr:version{ro, req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL-->null
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:int, desc:0 (OK), 1 (OK), 2 (Device Busy), 3 (Device Error), 4 (Invalid Operation), 5 (Invalid XML Format), 6
(Invalid XML Content), 7 (Reboot Required)-->0
  </statusCode>
  <statusString>
    <!--ro, req, enum, status description, subType:string, desc:"OK" (succeeded), "Device Busy", "Device Error", "Invalid Operation", "Invalid XML Format",
"Invalid XML Content", "Reboot" (reboot device)-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, sub status code, desc:sub status code-->OK
  </subStatusCode>
  <FailedNodeInfoList>
    <!--ro, opt, object, information list of failed nodes, desc:for the manual time synchronization of central analysis cluster, this field is returned if
time synchronization failed-->
    <FailedNodeInfo>
      <!--ro, opt, object, information of failed nodes-->
      <nodeID>
        <!--ro, req, string, node ID, range:[0,64]-->test
      </nodeID>
      <nodeIP>
        <!--ro, req, string, node IP, range:[0,20]-->test
      </nodeIP>
      <reason>
        <!--ro, opt, string, reason why the node failed to synchronize time, range:[0,128]-->test
      </reason>
    </FailedNodeInfo>
  </FailedNodeInfoList>
</ResponseStatus>
```


10.1.5.3 Get the capability of device time synchronization management

Request URL

GET /ISAPI/System/time/capabilities

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<Time xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, opt, object, time management capability set, attr:version{opt, string, protocolVersion}-->
  <timeMode opt="NTP,manual,satellite,SDK,28181,ONVIF,ALL(任何支持的校时方式都允许校时),NONE(不允校时或无校时源),platform">
    <!--ro, req, enum, time synchronization mode, subType:string, attr:opt{opt, string}, desc:“NTP” (NTP time synchronization), “manual” (manual time
synchronization), “satellite” (satellite time synchronization), “platform” (platform time synchronization), “NONE” (time synchronization is not allowed or
no time synchronization source), “GB28181” (GB28181 time synchronization)-->NTP
  </timeMode>
  <localTime min="0" max="256">
    <!--ro, opt, string, local time, range:[0,256], attr:min{opt, string},max{opt, string}-->test
  </localTime>
  <timeZone min="0" max="256">
    <!--ro, opt, string, time zone, range:[0,256], attr:min{opt, string},max{opt, string}-->test
  </timeZone>
  <satelliteInterval min="0" max="3600">
    <!--ro, opt, int, satellite time synchronization interval, step:1, unit:min, attr:min{opt, string},max{opt, string}, desc:unit: minute-->60
  </satelliteInterval>
  <timeType opt="local,UTC">
    <!--ro, opt, enum, time type, subType:string, attr:opt{opt, string}, desc:“local” (local time), “UTC” (UTC time)-->local
  </timeType>
  <platformType opt="EZVIZ">
    <!--ro, opt, enum, platform type, subType:string, dep:and,{$.Time.timeMode,eq,platform}, attr:opt{opt, string}, desc:platform type-->EZVIZ
  </platformType>
  <platformNo min="1" max="2">
    <!--ro, opt, int, platform No., range:[1,2], dep:and,{$.Time.timeMode,eq,GB28181}, attr:min{req, int},max{req, int}, desc:it is the only ID, which is
configured via platformNo in GB28181List, related URI: /ISAPI/System/Network/SIP/<SIPServerID>-->1
  </platformNo>
  <isSupportHistoryTime>
    <!--ro, opt, bool, supported capability of the historical time synchronization list, desc:related URI: /ISAPI/System/time/historyInfo?format=json-->true
  </isSupportHistoryTime>
  <isSupportTimeFilter>
    <!--ro, opt, bool, supported capability of filtering time synchronization, desc:related URI: /ISAPI/System/time/filter/capabilities?format=json-->true
  </isSupportTimeFilter>
  <displayFormat opt="MM/dd/yyyy hh:mm,mm,dd-MM-yyyy hh,MM-dd-yyyy hh:mm,yyyy-MM-dd hh:mm">
    <!--ro, opt, enum, time display format, subType:string, attr:opt{req, string}, desc:if this node is returned, it indicates that the device supports
configuring time display format, related URI: /ISAPI/System/time/timeType?format=json-->MM/dd/yyyy hh:mm
  </displayFormat>
  <isSupportSyncDeviceNTPInfoToCamera>
    <!--ro, opt, bool, the capability of synchronizing device's NTP service information with the camera, desc:related URI:
/ISAPI/System/time/SyncDeviceNTPInfoToCamera/capabilities?format=json-->true
  </isSupportSyncDeviceNTPInfoToCamera>
  <isSupportNTPService>
    <!--ro, opt, bool-->true
  </isSupportNTPService>
</Time>
```


10.1.5.4 Set device local time

Request URL

PUT /ISAPI/System/time/localTime

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response message, attr:version{ro, req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL-->null
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:int, desc:0 (OK), 1 (OK), 2 (Device Busy), 3 (Device Error), 4 (Invalid Operation), 5 (Invalid XML Format), 6
(Invalid XML Content), 7 (Reboot Required)-->0
  </statusCode>
  <statusString>
    <!--ro, req, enum, status information, subType:string, desc:"OK" (succeeded), "Device Busy", "Device Error", "Invalid Operation", "Invalid XML Format",
"Invalid XML Content", "Reboot" (reboot device)-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, sub status code, desc:sub status code description-->OK
  </subStatusCode>
</ResponseStatus>
```

10.1.5.5 Set parameters of all NTP servers

Request URL

PUT /ISAPI/System/time/ntpServers

Query Parameter

None

Request Message

```
<?xml version="1.0" encoding="UTF-8"?>

<NTPServerList xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--req, array, NTP server information list, subType:object, attr:version{opt, string, protocolVersion}-->
  <NTPServer>
    <!--opt, object, NTP server information-->
    <id>
      <!--req, string, ID-->1
    </id>
    <addressingFormatType>
      <!--req, enum, NTP server address type, subType:string, desc:"ipaddress" (IP address), "hostname" (domain name)-->hostname
    </addressingFormatType>
    <hostName>
      <!--opt, string, NTP server domain name, range:[1,64]-->12345
    </hostName>
    <ipAddress>
      <!--opt, string, IPv4 address, range:[1,32]-->192.168.1.112
    </ipAddress>
    <ipv6Address>
      <!--opt, string, IPv6 address, range:[1,128]-->1030:C9B4:FF12:48AA:1A2B
    </ipv6Address>
    <portNo>
      <!--opt, int, port No., range:[1,65535], desc:the default port No. is 123-->123
    </portNo>
    <synchronizeInterval>
      <!--opt, int, time synchronization interval, range:[1,10800], unit:min-->1440
    </synchronizeInterval>
    <enabled>
      <!--opt, bool, whether to enable, desc:disabled (by default)-->false
    </enabled>
  </NTPServer>
</NTPServerList>
```

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response message, attr:version{ro, req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL-->/ISAPI/xxxx
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:int, desc:0 (OK), 1 (OK), 2 (Device Busy), 3 (Device Error), 4 (Invalid Operation), 5 (Invalid XML Format), 6
(Invalid XML Content), 7 (Reboot Required)-->0
  </statusCode>
  <statusString>
    <!--ro, req, enum, status description, subType:string, desc:"OK" (succeeded), "Device Busy", "Device Error", "Invalid Operation", "Invalid XML Format",
"Invalid XML Content", "Reboot" (reboot device)-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, sub status code, desc:sub status code-->OK
  </subStatusCode>
</ResponseStatus>
```

10.1.5.6 Get the parameters of a specific NTP (Network Time Protocol) server

Request URL

GET /ISAPI/System/time/ntpServers

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<NTPServerList xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, array, NTP server information list, subType:object, attr:version{req, string, protocolVersion}-->
  <NTPServer>
    <!--ro, opt, object, NTP server information-->
    <id>
      <!--ro, req, string, ID-->1
    </id>
    <addressingFormatType>
      <!--ro, req, enum, NTP server address type, subType:string, desc:"ipaddress" (IP address), "hostname" (domain name)-->hostname
    </addressingFormatType>
    <hostName>
      <!--ro, opt, string, NTP server domain name, range:[1,64]-->12345
    </hostName>
    <ipAddress>
      <!--ro, opt, string, IPv4 address, range:[1,32]-->192.168.1.112
    </ipAddress>
    <ipv6Address>
      <!--ro, opt, string, IPv6 address, range:[1,128]-->1030:C9B4:FF12:48AA:1A2B
    </ipv6Address>
    <portNo>
      <!--ro, opt, int, port No., range:[1,65535], desc:the default port No. is 123-->123
    </portNo>
    <synchronizeInterval>
      <!--ro, opt, int, time synchronization interval, range:[1,10800], unit:min-->1440
    </synchronizeInterval>
    <enabled>
      <!--ro, opt, bool, whether to enable, desc:disabled (by default)-->false
    </enabled>
  </NTPServer>
</NTPServerList>
```

10.1.5.7 Set the parameters of a NTP server

Request URL

PUT /ISAPI/System/time/ntpServers/<NTPServerID>

Query Parameter

Parameter Name	Parameter Type	Description
NTPServerID	string	--

Request Message

```
<?xml version="1.0" encoding="UTF-8"?>

<NTPServer xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--req, object, NTP server information, attr:version{req, string, protocolVersion}-->
  <id>
    <!--req, string, ID-->1
  </id>
  <addressingFormatType>
    <!--req, enum, IP address type of NTP server, subType:string, desc:"ipaddress" (IP address), "hostname" (domain name)-->hostname
  </addressingFormatType>
  <hostName>
    <!--opt, string, NTP server domain name地址去掉, range:[1,64]-->12345
  </hostName>
  <ipAddress>
    <!--opt, string, IPv4 address, range:[1,32], desc:IPv4 address-->192.168.1.112
  </ipAddress>
  <ipv6Address>
    <!--opt, string, IPv6 address, range:[1,128], desc:IPv6 address-->1030:C9B4:FF12:48AA:1A2B
  </ipv6Address>
  <portNo>
    <!--opt, int, port No., range:[1,65535], step:1, desc:port No.-->1
  </portNo>
  <synchronizeInterval>
    <!--opt, int, time synchronization interval, range:[1,10800], step:1, unit:min, desc:NTP time synchronization interval, unit: minute-->1440
  </synchronizeInterval>
  <enabled>
    <!--opt, bool-->false
  </enabled>
</NTPServer>
```

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response message, attr:version{ro, req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL-->null
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:int, desc:0 (OK), 1 (OK), 2 (Device Busy), 3 (Device Error), 4 (Invalid Operation), 5 (Invalid XML Format), 6
(Invalid XML Content), 7 (Reboot Required)-->0
  </statusCode>
  <statusString>
    <!--ro, req, enum, status description, subType:string, desc:"OK" (succeeded), "Device Busy", "Device Error", "Invalid Operation", "Invalid XML Format",
"Invalid XML Content", "Reboot" (reboot device)-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, sub status code, desc:sub status code-->OK
  </subStatusCode>
</ResponseStatus>
```


10.1.5.8 Get the parameters of a NTP server

Request URL

GET /ISAPI/System/time/ntpServers/<NTPServerID>

Query Parameter

Parameter Name	Parameter Type	Description
NTPServerID	string	--

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<NTPServer xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, NTP server information, attr:version{req, string, protocolVersion}-->
  <id>
    <!--ro, req, string, ID-->1
  </id>
  <addressingFormatType>
    <!--ro, req, enum, IP address type of NTP server, subType:string, desc:"ipaddress" (IP address), "hostname" (domain name)-->hostname
  </addressingFormatType>
  <hostName>
    <!--ro, opt, string, NTP server domain name, range:[1,64]-->12345
  </hostName>
  <ipAddress>
    <!--ro, opt, string, IPv4 address, range:[1,32], desc:IPv4 address-->192.168.1.112
  </ipAddress>
  <ipv6Address>
    <!--ro, opt, string, IPv6 address, range:[1,128], desc:IPv6 address-->1030:C9B4:FF12:48AA:1A2B
  </ipv6Address>
  <portNo>
    <!--ro, opt, int, port No., range:[1,65535], step:1, desc:port No.-->1
  </portNo>
  <synchronizeInterval>
    <!--ro, opt, int, time synchronization interval, range:[1,10800], step:1, unit:min, desc:NTP time synchronization interval, unit: minute-->1440
  </synchronizeInterval>
  <enabled>
    <!--ro, opt, bool-->false
  </enabled>
</NTPServer>
```

10.1.5.9 Get time zone

Request URL

GET /ISAPI/System/time/timeZone

Query Parameter

None

Request Message

None

Response Message

None

10.1.5.10 Set time zone

Request URL

PUT /ISAPI/System/time/timeZone

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response message, attr:version{ro, req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL-->null
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:int, desc:0 (OK), 1 (OK), 2 (Device Busy), 3 (Device Error), 4 (Invalid Operation), 5 (Invalid XML Format), 6
(Invalid XML Content), 7 (Reboot Required)-->0
  </statusCode>
  <statusString>
    <!--ro, req, enum, status description, subType:string, desc:“OK” (succeeded), “Device Busy”, “Device Error”, “Invalid Operation”, “Invalid XML Format”,
“Invalid XML Content”, “Reboot” (reboot device)-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, sub status code, desc:sub status code-->OK
  </subStatusCode>
</ResponseStatus>
```

10.2 Network Settings

10.2.1 HTTP Listening

10.2.1.1 Delete receiving server(s)

Request URL

DELETE /ISAPI/Event/notification/httpHosts

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response message, attr:version{ro, req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL-->null
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:int, desc:0 (OK), 1 (OK), 2 (Device Busy), 3 (Device Error), 4 (Invalid Operation), 5 (Invalid XML Format), 6
(Invalid XML Content), 7 (Reboot Required)-->0
  </statusCode>
  <statusString>
    <!--ro, req, enum, status information, subType:string, desc:"OK" (succeeded), "Device Busy", "Device Error", "Invalid Operation", "Invalid XML Format",
"Invalid XML Content", "Reboot" (reboot device)-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, sub status code, which describes the error in details, desc:sub status code, which describes the error in details-->OK
  </subStatusCode>
</ResponseStatus>
```

10.2.1.2 Delete a HTTP listening server

Request URL

DELETE /ISAPI/Event/notification/httpHosts/<hostID>

Query Parameter

Parameter Name	Parameter Type	Description
hostID	string	--

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response message, attr:version{ro, req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL-->null
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:int, desc:0-OK, 1-OK, 2-Device Busy, 3-Device Error, 4-Invalid Operation, 5-Invalid XML Format, 6-Invalid XML
Content, 7-Reboot Required-->0
  </statusCode>
  <statusString>
    <!--ro, req, enum, status description, subType:string, desc:“OK” (succeeded), “Device Busy”, “Device Error”, “Invalid Operation”, “Invalid XML Format”,
“Invalid XML Content”, “Reboot” (reboot device)-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, sub status code, desc:sub status code-->OK
  </subStatusCode>
</ResponseStatus>
```

10.2.1.3 Set the parameters of a listening host

Request URL

PUT /ISAPI/Event/notification/httpHosts/<hostID>

Query Parameter

Parameter Name	Parameter Type	Description
hostID	string	--

Request Message

```
<?xml version="1.0" encoding="UTF-8"?>
<HttpHostNotification xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--req, object, listening host, attr:version{req, string, protocolVersion}-->
  <id>
    <!--req, string, listening host ID, range:[1,10]-->test
  </id>
  <url>
    <!--req, string, URL-->test
  </url>
  <protocolType>
    <!--req, enum, protocol type, subType:string, desc:"HTTP", "HTTPS", "EHome" (ISUP)-->HTTP
  </protocolType>
  <parameterFormatType>
    <!--req, enum, parameter format type, subType:string, desc:"JSON", "XML"-->JSON
  </parameterFormatType>
  <addressingFormatType>
    <!--req, enum, address type, subType:string, desc:"hostname", "ipaddress"-->hostname
  </addressingFormatType>
  <ipAddress>
    <!--opt, string, IP address, dep:or,{$.HttpHostNotification.addressingFormatType,eq,ipAddress}, desc:IP address-->test
  </ipAddress>
  <portNo>
    <!--opt, int, port number-->1
  </portNo>
  <httpAuthenticationMethod>
    <!--req, enum, authentication method, subType:string, desc:"MD5digest"(MD5), "none", "base64"-->MD5digest
  </httpAuthenticationMethod>
</HttpHostNotification>
```

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response message, attr:version{ro, req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL, range:[0,1024]-->null
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:int, desc:0 (OK), 1 (OK), 2 (Device Busy), 3 (Device Error), 4 (Invalid Operation), 5 (Invalid XML Format), 6
(Invalid XML Content), 7 (Reboot Required)-->0
  </statusCode>
  <statusString>
    <!--ro, req, enum, status description, subType:string, desc:"OK"(succeeded), "Device Busy", "Device Error", "Invalid Operation", "Invalid XML Format",
"Invalid XML Content", "Reboot"(reboot device)-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, sub status code, desc:sub status code-->OK
  </subStatusCode>
  <description>
    <!--ro, opt, string, range:[0,1024]-->badXmlFormat
  </description>
</ResponseStatus>
```


10.2.1.4 Get the parameters of a listening host

Request URL

GET /ISAPI/Event/notification/httpHosts/<hostID>

Query Parameter

Parameter Name	Parameter Type	Description
hostID	string	--

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<HttpHostNotification xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, listening host, attr:version{req, string, protocolVersion}-->
  <id>
    <!--ro, req, string, listening host ID, range:[1,10]-->test
  </id>
  <url>
    <!--ro, req, string, URL-->test
  </url>
  <protocolType>
    <!--ro, req, enum, protocol type, subType:string, desc:"HTTP", "HTTPS", "EHome"-->HTTP
  </protocolType>
  <parameterFormatType>
    <!--ro, req, enum, parameter format type, subType:string, desc:"JSON", "XML"-->JSON
  </parameterFormatType>
  <addressingFormatType>
    <!--ro, req, enum, address type, subType:string, desc:"hostname", "ipaddress"-->hostname
  </addressingFormatType>
  <hostName>
    <!--ro, opt, string, host name, dep:and,{$.HttpHostNotification.addressingFormatType,eq,hostName}-->test
  </hostName>
  <ipAddress>
    <!--ro, opt, string, IP address, dep:or,{$.HttpHostNotification.addressingFormatType,eq,ipAddress}, desc:IP address-->test
  </ipAddress>
  <ipv6Address>
    <!--ro, opt, string, IPv6 address, dep:or,{$.HttpHostNotification.addressingFormatType,eq,ipAddress}, desc:IPv6 address-->test
  </ipv6Address>
  <portNo>
    <!--ro, opt, int, port number-->1
  </portNo>
  <userName>
    <!--ro, opt, string, user name, dep:and,{$.HttpHostNotification.httpAuthenticationMethod,eq,MD5digest}-->test
  </userName>
  <httpAuthenticationMethod>
    <!--ro, req, enum, authentication method, subType:string, desc:"MD5digest" (MD5), "none", "base64"-->MD5digest
  </httpAuthenticationMethod>
  <ANPR>
    <!--ro, opt, object, ANPR-->
    <detectionUpLoadPicturesType>
      <!--ro, opt, enum, uploaded pictures type, subType:string, desc:"all", "licensePlatePicture" (license plate picture), "detectionPicture" (detected
picture)-->all
    </detectionUpLoadPicturesType>
  </ANPR>
  <Extensions>
    <!--ro, opt, object, range-->
    <intervalBetweenEvents>
      <!--ro, opt, int, event interval-->1
    </intervalBetweenEvents>
  </Extensions>
  <uploadImagesDataType>
    <!--ro, opt, enum, picture data type, subType:string, desc:"URL", "binary" (default value). For cloud storage, only "URL" is supported-->URL
  </uploadImagesDataType>
  <httpBroken>
    <!--ro, opt, bool, whether to enable the automatic network replenishment, desc:if the ANR function is enabled, it will be applied to all events-->true
  </httpBroken>
  <SubscribeEvent>
    <!--ro, opt, object, picture uploading modes of all events which contain pictures-->
    <heartbeat>
      <!--ro, opt, int, heartbeat interval time-->30
    </heartbeat>
    <eventMode>
      <!--ro, req, enum, event mode, subType:string, desc:"all" (all alarms need to be reported), "list" (only listed alarms need to be reported)-->all
    </eventMode>
    <EventList>
      <!--ro, opt, array, event list, subType:object-->
      <Event>
        <!--ro, opt, object, channel information linked to event-->
        <type>
          <!--ro, req, enum, event type, subType:string, desc:refer to event type list (eventType): "ADAS"(advanced driving assistance system), "ADASAlarm"
(advanced driving assistance alarm), "AID"(traffic incident detection), "ANPR"(automatic number plate recognition), "AccessControllerEvent" (access
controller event), "CDsStatus" (CD burning status), "DBD"(driving behavior detection) "GPSUpload" (GPS information upload), "HFPD"(frequently appeared
person detection), "IO"(I/O alarm), "IOTD" (IoT device detection), "LES" (logistics scanning event), "LFPD"(rarely appeared person detection), "PALMismatch"
(video standard mismatch), "PIR", "PeopleCounting" (people counting), "PeopleNumChange" (people number change detection), "Standup"(standing up detection),
"TMA"(thermometry alarm), "TMPA"(temperature measurement pre-alarm), "VMD"(motion detection), "abnormalAcceleration", "abnormalDriving", "advReachHeight",
"alarmResult", "attendance", "attendedBaggage", "audioAbnormal", "audioexception", "behaviorResult"(abnormal event detection), "blindSpotDetection"(blind
spot detection alarm), "cardMatch", "changedStatus", "collision", "containerDetection", "crowdSituationAnalysis", "databaseException", "defocus"(defocus
detection), "diskUnformat"(disk unformatted), "diskerror", "diskfull", "driverConditionMonitor"(driver status monitoring alarm); "emergencyAlarm",
"faceCapture", "faceSnapModeling", "facedetection", "failDown"(People Falling Down), "faultAlarm", "fielddetection"(intrusion detection), "fireDetection",
"fireEscapeDetection", "flowOverrun", "framesPeopleCounting", "getUp"(getting up detection), "group" (people gathering), "hdBadBlock"(HDD bad sector
detection event), "hdImpact"(HDD impact detection event), "heatmap"(heat map alarm), "highHDTemperature"(HDD high temperature detection event),
"highTempAlarm"(HDD high temperature alarm), "hotSpare"(hot spare exception), "illaccess"(invalid access), "ipcTransferAbnormal", "ipconflict"(IP address
conflicts), "keyPersonGetUp"(key person getting up detection), "leavePosition"(absence detection), "linedetection"(line crossing detection),
"listSyncException"(list synchronization exception), "loitering"(loitering detection), "lowHDTemperature"(HDD low temperature detection event),
"mixedTargetDetection"(multi-target-type detection), "modelError", "nicbroken"(network disconnected), "nodeOffline"(node disconnected),
"nonPoliceIntrusion", "overSpeed"(overspeed alarm), "overtimeTarry"(staying overtime detection), "parking"(parking detection), "peopleNumChange"
```

```
"peopleNumCounting", "personAbnormalAlarm"(person ID exception alarm), "personDensityDetection", "personQueueCounting", "personQueueDetection",
"personQueueRealTime"(real-time data of people queuing-up detection), "personQueueTime"(waiting time detection), "playCellphone"(playing mobile phone
detection), "pocException"(video exception), "poe"(POE power exception), "policeAbsent", "radarAlarm", "radarFieldDetection", "radarLineDetection",
"radarPerimeterRule"(radar rule data), "radarTargetDetection", "radarVideoDetection"(radar-assisted target detection), "raidException", "rapidMove",
"reachHeight"(climbing detection), "recordCycleAbnormal"(insufficient recording period), "recordException", "regionEntrance", "regionExiting", "retention"
(people overstay detection), "rollover", "running"(people running), "safetyHelmetDetection"(hard hat detection), "scenechangedetection", "sensorAlarm"
(angular acceleration alarm), "severeHDFailure"(HDD major fault detection), "shelteralarm"(video tampering alarm), "shipsDetection", "sitQuietly"(sitting
detection), "smokeAndFireDetection", "smokeDetection", "softIO", "spacingChange"(distance exception), "sysStorFull"(storaging full alarm of cluster system),
"takingElevatorDetection"(elevator electric moped detection), "targetCapture", "temperature"(temperature difference alarm), "thermometry"(temperature
alarm), "thirdPartyException", "toiletTarry"(in-toilet overtime detection), "tollCodeInfo"(QR code information report), "tossing"(thrown object detection),
"unattendedBaggage", "vehicleMatchResult"(uploading list alarms), "vehicleRcogResult", "versionAbnormal"(cluster version exception), "videoException",
"videoloss", "violationAlarm", "violentMotion"(violent motion detection), "yardTarry"(playground overstay detection), "AccessControllerEvent",
"IDCardInfoEvent", "FaceTemperatureMeasurementEvent", "QRCodeEvent"(QR code event of access control), "CertificateCaptureEvent"(person ID capture comparison
event), "UncertificateCompareEvent", "ConsumptionAndTransactionRecordEvent", "ConsumptionEvent", "TransactionRecordEvent", "SetMealQuery"(searching
consumption set meals), "ConsumptionStatusQuery"(searching the consumption status), "humanBodyComparison" (human body comparison),
"regionTargetNumberCounting" (regional target statistics)-->mixedTargetDetection
        </type>
        <minorAlarm>
          <!--ro, opt, string, alarm sub type, desc:refer to the macro definition of uploaded events. "IDCardInfoEvent" is required when the type of event
is "AccessControllerEvent"-->0x400,0x401,0x402,0x403
        </minorAlarm>
        <minorException>
          <!--ro, opt, string, minor exception type, desc:refer to the macro definition of uploaded events. "IDCardInfoEvent" is required when the type of
event is "AccessControllerEvent"-->0x400,0x401,0x402,0x403
        </minorException>
        <minorOperation>
          <!--ro, opt, string, minor operation type, desc:refer to the macro definition of uploaded events. "IDCardInfoEvent" is required when the type of
event is "AccessControllerEvent"-->0x400,0x401,0x402,0x403
        </minorOperation>
        <minorEvent>
          <!--ro, opt, string, minor event type, desc:refer to the macro definition of uploaded events. "IDCardInfoEvent" is required when the type of event
is "AccessControllerEvent"-->0x01,0x02,0x03,0x04
        </minorEvent>
        <pictureURLType>
          <!--ro, opt, enum, alarm picture format of the specified event, subType:string, desc:"binary", "localURL" (local URL), "cloudStorageURL" (cloud
storage URL), "EZVIZURL" (EZ URL)-->binary
        </pictureURLType>
        <channels>
          <!--ro, opt, string, listen to the events on the specified channel No. list, desc:if all channels are being listened to, the node shall not be
applied. If some channels are being listened to, the channel No. shall be listed and separated by commas-->1,2,3,4
        </channels>
      </Event>
    </EventList>
    <channels>
      <!--ro, opt, string, listen to the specified channel No. list, desc:if all channels are being listened to, the node shall not be applied. If some
channels are being listened to, the channel No. shall be listed and separated by commas-->1,2,3,4
    </channels>
    <pictureURLType>
      <!--ro, opt, enum, alarm picture format, subType:string, desc:"binary", "localURL" (local URL), "cloudStorageURL" (cloud storage URL), "EZVIZURL" (EZ
URL). The node indicates the upload mode of all event pictures. If the node is applied, <pictureURLType> of <Event> will be invalid. If the node is not
applied, the pictures are uploaded in the default mode. The default data type of uploaded pictures for front-end devices is binary, and for back-end devices
is local URL of the device-->binary
    </pictureURLType>
    <ChangedUploadSub>
      <!--ro, opt, object, subscribe to messages-->
      <interval>
        <!--ro, opt, int, the lifecycle of arming GUID, desc:within the interval, if the client software does not reconnect to the device, a new GUID will
be generated by the device-->5
      </interval>
      <StatusSub>
        <!--ro, opt, object, sub status-->
        <all>
          <!--ro, opt, bool, whether to subscribe to all-->true
        </all>
        <channel>
          <!--ro, opt, bool, channel subscription status (whether the channel is subscribed), desc:it is not required if the value of <all> is true-->true
        </channel>
        <hd>
          <!--ro, opt, bool, HDD subscription status (whether the HDD is subscribed), desc:it is not required if the value of <all> is true-->true
        </hd>
        <capability>
          <!--ro, opt, bool, subscription status of capability set change (whether the capability set change is subscribed), desc:it is not required if the
value of <all> is true-->true
        </capability>
      </StatusSub>
    </ChangedUploadSub>
  </SubscribeEvent>
  <PackingSpaceRecognition>
    <!--ro, opt, object, current control parameters of event listened by parking space detector on the security control panel, desc:related event:
PackingSpaceRecognition-->
    <upLoadPicturesType>
      <!--ro, req, enum, uploaded picture type, subType:string, desc:"all", "picturesTypes"(upload specified types of pictures), "notUpload"(not upload
pictures)-->all
    </upLoadPicturesType>
    <PicturesTypes>
      <!--ro, opt, array, specified list of uploaded picture types, subType:object, dep:and,
{$.HttpHostNotification.PackingSpaceRecognition.upLoadPicturesType,eq,picturesTypes}-->
      <picturesType>
        <!--ro, opt, enum, uploaded picture type, subType:string, desc:"backgroundImage"(captured background picture), "plateImage"(license plate
thumbnail)-->backgroundImage
      </picturesType>
    </PicturesTypes>
  </PackingSpaceRecognition>
```

```
    <enabled>
      <!--ro, opt, bool-->true
    </enabled>
    <netWork>
      <!--ro, opt, enum, subType:int-->1
    </netWork>
  </HttpHostNotification>
```

10.2.1.5 Get the capabilities of listening hosts parameters

Request URL

GET /ISAPI/Event/notification/httpHosts/capabilities?type=<type>

Query Parameter

Parameter Name	Parameter Type	Description
type	enum	--

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>
<HttpHostNotificationCap xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, capabilities of subscribing to event types and event channels, attr:version{req, string, protocolVersion}-->
  <hostNumber>
    <!--ro, req, int, the number of listening hosts-->0
  </hostNumber>
  <urlLen max="10">
    <!--ro, req, string, URL length, attr:max{req, int}-->test
  </urlLen>
  <protocolType opt="HTTP,HTTPS,EHome">
    <!--ro, req, enum, protocol type, subType:string, attr:opt{req, string}, desc:"HTTP", "HTTPS", "EHome"-->HTTP
  </protocolType>
  <parameterFormatType opt="XML,querystring,JSON">
    <!--ro, req, enum, alarm parameter format type, subType:string, attr:opt{req, string}, desc:"xml" (XML format), "querystring", "json" (JSON format)--
>XML
  </parameterFormatType>
  <addressingFormatType opt="ipaddress,hostname">
    <!--ro, req, enum, address format type, subType:string, attr:opt{req, string}, desc:"ipaddress", "hostname"-->ipaddress
  </addressingFormatType>
  <ipAddress opt="ipv4,ipv6">
    <!--ro, opt, string, IP address, attr:opt{req, string}, desc:it is valid when addressingFormatType is "ipaddress"-->test
  </ipAddress>
  <portNo min="0" max="10">
    <!--ro, opt, string, port number, range:[0,65535], attr:min{req, int},max{req, int}-->1
  </portNo>
  <SubscribeEventCap>
    <!--ro, opt, string, subscribe to changing status of capability set-->test
    <heartbeat min="0" max="10">
      <!--ro, opt, string, heartbeat interval time, attr:min{req, int},max{req, int}-->1
    </heartbeat>
    <channelMode opt="all,list">
      <!--ro, opt, enum, channel mode, subType:string, attr:opt{req, string}, desc:"all" (the device does not support subscribing to event channels
separately, "list" (the device support subscribing to event channels separately). If both channelMode and eventMode return all, the device does not support
subscribing to event types and event channels.-->all
    </channelMode>
    <eventMode opt="all,list">
      <!--ro, opt, enum, event mode, subType:string, attr:opt{req, string}, desc:"all" (subscribe to all channels and events on the device), "list"
(subscribe to all channels and events on the device)-->all
    </eventMode>
    <EventList>
      <!--ro, opt, object, event list-->
      <Event>
        <!--ro, opt, string, event subscription information-->test
        <type>
          <!--ro, req, enum, see details in event type list, subType:string, desc:"AccessControllerEvent", "IDCardInfoEvent",
"FaceTemperatureMeasurementEvent", "QRCodeEvent", "CertificateCaptureEvent", "UncertificateCompareEvent", "ConsumptionAndTransactionRecordEvent",
"ConsumptionEvent", "TransactionRecordEvent", "HealthInfoSyncQuery", "SetMealQuery", "ConsumptionStatusQuery", "humanBodyComparison",
"regionTargetNumberCounting"-->AccessControllerEvent
        </type>
        <minorAlarm opt="0x400,0x401,0x402,0x403">
          <!--ro, opt, string, minor alarm type, attr:opt{req, string}, desc:"IDCardInfoEvent" is required when the type of event is
"AccessControllerEvent"-->0x400,0x401
        </minorAlarm>
        <minorException opt="0x400,0x401,0x402,0x403">
          <!--ro, opt, string, minor exception type, attr:opt{req, string}, desc:"IDCardInfoEvent" is required when the type of event is
"AccessControllerEvent"-->0x400,0x401
        </minorException>
        <minorOperation opt="0x400,0x401,0x402,0x403">
          <!--ro, opt, string, minor operation type, attr:opt{req, string}, desc:"IDCardInfoEvent" is required when the type of event is
"AccessControllerEvent"-->0x400,0x401
        </minorOperation>
        <minorEvent opt="0x01,0x02,0x03,0x04">
          <!--ro, opt, string, minor event type, attr:opt{req, string}, desc:"IDCardInfoEvent" is required when the type of event is
"AccessControllerEvent"-->0x400,0x401
        </minorEvent>
      </Event>
    </EventList>
  </SubscribeEventCap>
</HttpHostNotificationCap>
```


10.2.1.6 Set IP address of receiving server(s)

Request URL

PUT /ISAPI/Event/notification/httpHosts

Query Parameter

None

Request Message

```
<?xml version="1.0" encoding="UTF-8"?>
<HttpHostNotificationList xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--req, array, listening host list, subType:object, attr:version{req, string, protocolVersion}-->
  <HttpHostNotification>
    <!--opt, object, listening host-->
    <id>
      <!--req, string, ID-->test
    </id>
    <url>
      <!--req, string, URL-->test
    </url>
    <protocolType>
      <!--req, enum, protocol type, subType:string, desc:"HTTP", "HTTPS", "EHome"-->HTTP
    </protocolType>
    <parameterFormatType>
      <!--req, enum, parameter format type, subType:string, desc:"JSON", "XML"-->JSON
    </parameterFormatType>
    <addressingFormatType>
      <!--req, enum, address types, subType:string, desc:"hostname" (host name), "ipaddress" (ip address)-->hostname
    </addressingFormatType>
    <ipAddress>
      <!--opt, string, IP address-->test
    </ipAddress>
    <portNo>
      <!--opt, int, port No.-->1
    </portNo>
    <userName>
      <!--opt, string, user name, dep:and,{$.HttpHostNotification.httpAuthenticationMethod,eq,MD5digest}-->test
    </userName>
    <httpAuthenticationMethod>
      <!--req, enum, authentication method, subType:string, desc:"MD5digest" (MD5), "none"-->MD5digest
    </httpAuthenticationMethod>
    <SubscribeEvent>
      <!--opt, object, picture uploading modes of all events which contain pictures-->
      <eventMode>
        <!--req, enum, event mode, subType:string, desc:event mode-->all
      </eventMode>
    </SubscribeEvent>
  </HttpHostNotification>
</HttpHostNotificationList>
```

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response message, attr:version{ro, req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL, range:[0,1024]-->null
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:int, desc:0 (OK), 1 (OK), 2 (Device Busy), 3 (Device Error), 4 (Invalid Operation), 5 (Invalid XML Format), 6
(Invalid XML Content), 7 (Reboot Required)-->0
  </statusCode>
  <statusString>
    <!--ro, req, enum, status description, subType:string, desc:"OK" (succeeded), "Device Busy", "Device Error", "Invalid Operation", "Invalid XML Format",
"Invalid XML Content", "Reboot" (reboot device)-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, sub status code, which describes the error in details, desc:sub status code, which describes the error in details-->OK
  </subStatusCode>
  <description>
    <!--ro, opt, string, range:[0,1024]-->badXmlFormat
  </description>
</ResponseStatus>
```

10.2.1.7 Get parameters of all listening hosts

Request URL

GET /ISAPI/Event/notification/httpHosts

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<HttpHostNotificationList xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, array, listening host list, subType:object, attr:version{req, string, protocolVersion}-->
  <HttpHostNotification>
    <!--ro, opt, object, listening host-->
    <id>
      <!--ro, req, string, subscribe to ID, range:[1,10]-->test
    </id>
    <url>
      <!--ro, req, string, URL-->test
    </url>
    <protocolType>
      <!--ro, req, enum, protocol type, subType:string, desc:"HTTP", "HTTPS", "EHome" (ISUP)-->HTTP
    </protocolType>
    <parameterFormatType>
      <!--ro, req, enum, parameter format type, subType:string, desc:"JSON", "XML"-->JSON
    </parameterFormatType>
    <addressingFormatType>
      <!--ro, req, enum, address type, subType:string, desc:"hostname", "ipaddress"-->hostname
    </addressingFormatType>
    <hostName>
      <!--ro, opt, string, host name, dep:and,{$.HttpHostNotification.addressingFormatType,eq,hostName}-->test
    </hostName>
    <ipAddress>
      <!--ro, opt, string, IP address, dep:or,{$.HttpHostNotification.addressingFormatType,eq,ipAddress}, desc:IP address-->test
    </ipAddress>
    <ipv6Address>
      <!--ro, opt, string, IPv6 Address, dep:or,{$.HttpHostNotification.addressingFormatType,eq,ipAddress}, desc:IPv6 Address-->test
    </ipv6Address>
    <portNo>
      <!--ro, opt, int, port number-->1
    </portNo>
    <userName>
      <!--ro, opt, string, user name, dep:and,{$.HttpHostNotification.httpAuthenticationMethod,eq,MD5digest}-->test
    </userName>
    <httpAuthenticationMethod>
      <!--ro, req, enum, authentication method, subType:string, desc:"MD5digest"(MD5), "none", "base64"-->MD5digest
    </httpAuthenticationMethod>
    <ANPR>
      <!--ro, opt, object, ANPR-->
      <detectionUpLoadPicturesType>
        <!--ro, opt, enum, uploaded pictures type, subType:string, desc:"all", "licensePlatePicture", "detectionPicture"(detected picture)-->all
      </detectionUpLoadPicturesType>
    </ANPR>
    <Extensions>
      <!--ro, opt, object, range-->
      <intervalBetweenEvents>
        <!--ro, opt, int, event interval-->1
      </intervalBetweenEvents>
    </Extensions>
    <uploadImagesDataType>
      <!--ro, opt, enum, picture data type, subType:string, desc:"URL", "binary" (default value). For cloud storage, only "URL" is supported-->URL
    </uploadImagesDataType>
    <httpBroken>
      <!--ro, opt, bool, whether to enable the automatic network replenishment, desc:if the ANR function is enabled, it will be applied to all events-->true
    </httpBroken>
    <SubscribeEvent>
      <!--ro, opt, object, picture uploading modes of all events which contain pictures-->
      <heartbeat>
        <!--ro, opt, int, heartbeat interval time-->30
      </heartbeat>
      <eventMode>
        <!--ro, req, enum, event mode, subType:string, desc:"all" (all alarms need to be reported), "list" (only listed alarms need to be reported)-->all
      </eventMode>
      <EventList>
        <!--ro, opt, array, event list, subType:object-->
        <Event>
          <!--ro, opt, object, channel information linked to event-->
          <type>
            <!--ro, req, enum, event type, subType:string, desc:refer to event type list (eventType): "ADAS"(advanced driving assistance system),
"ADASAlarm"(advanced driving assistance alarm), "AID"(traffic incident detection), "ANPR"(automatic number plate recognition), "AccessControllerEvent"
(access controller event), "CDsStatus" (CD burning status), "DBD"(driving behavior detection) "GPSUpload" (GPS information upload), "HFPD"(frequently
appeared person detection), "IO"(I/O alarm), "IOTD" (IoT device detection), "LES" (logistics scanning event), "LFPD"(rarely appeared person detection),
"PALMismatch" (video standard mismatch), "PIR", "PeopleCounting" (people counting), "PeopleNumChange" (people number change detection), "Standup"(standing
up detection), "TMA"(thermometry alarm), "TMPA"(temperature measurement pre-alarm), "VMD"(motion detection), "abnormalAcceleration", "abnormalDriving",
"advReachHeight", "alarmResult", "attendance", "attendedBaggage", "audioAbnormal", "audioexception", "behaviorResult"(abnormal event detection),
"blindSpotDetection"(blind spot detection alarm), "cardMatch", "changedStatus", "collision", "containerDetection", "crowdSituationAnalysis",
"databaseException", "defocus"(defocus detection), "diskUnformat"(disk unformatted), "diskerror", "diskfull", "driverConditionMonitor"(driver status
monitoring alarm); "emergencyAlarm", "faceCapture", "faceSnapModeling", "facedetection", "failDown"(People Falling Down), "faultAlarm", "fielddetection"
(intrusion detection), "fireDetection", "fireEscapeDetection", "flowOverrun", "framesPeopleCounting", "getUp"(getting up detection), "group" (people
gathering), "hdBadBlock"(HDD bad sector detection event), "hdImpact"(HDD impact detection event), "heatmap"(heat map alarm), "highHDTemperature"(HDD high
temperature detection event), "highTempAlarm"(HDD high temperature alarm), "hotSpare"(hot spare exception), "illaccess"(invalid access),
"ipcTransferAbnormal", "ipconflict"(IP address conflicts), "keyPersonGetUp"(key person getting up detection), "LeavePosition"(absence detection),
"linedetection"(line crossing detection), "listSyncException"(list synchronization exception), "loitering"(loitering detection), "lowHDTemperature"(HDD low
temperature detection event), "mixedTargetDetection"(multi-target-type detection), "modelError", "nicbroken"(network disconnected), "nodeOffline"(node
disconnected), "nonPoliceIntrusion", "overSpeed"(overspeed alarm), "overtimeTarry"(staying overtime detection), "parking"(parking detection),
"peopleNumChange", "peopleNumCounting", "personAbnormalAlarm"(person ID exception alarm), "personDensityDetection", "personQueueCounting",
"personQueueDetection", "personQueueRealTime"(real-time data of people queuing-up detection), "personQueueTime"(waiting time detection), "playCellphone"
(playing mobile phone detection), "pocException"(video exception), "poe"(POE power exception), "policeAbsent", "radarAlarm", "radarFieldDetection",
"radarLineDetection", "radarPerimeterRule"(radar rule data), "radarTargetDetection", "radarVideoDetection"(radar-assisted target detection),
"raidException", "rapidMove", "reachHeight"(climbing detection), "recordCycleAbnormal"(insufficient recording period), "recordException", "regionEntrance",
"regionExiting", "retention"(people overstay detection), "rollover", "running"(people running), "safetyHelmetDetection"(hard hat detection),
"scenechangedetection", "sensorAlarm"(angular acceleration alarm), "severeHDFailure"(HDD major fault detection), "shelteralarm"(video tampering alarm),
"shipsDetection", "sitQuietly"(sitting detection), "smokeAndFireDetection", "smokeDetection", "softIO", "spacingChange"(distance exception), "sysStorFull"
(storaging full alarm of cluster system), "takingElevatorDetection"(elevator electric moped detection), "targetCapture", "temperature"(temperature
```

```
difference alarm), "thermometry"(temperature alarm), "thirdPartyException", "toiletTarry"(in-toilet overtime detection), "tollCodeInfo"(QR code information
report), "tossing"(thrown object detection), "unattendedBaggage", "vehicleMatchResult"(uploading list alarms), "vehicleRcogResult", "versionAbnormal"
(cluster version exception), "videoException", "videoloss", "violationAlarm", "violentMotion"(violent motion detection), "yardTarry"(playground overstay
detection), "AccessControllerEvent", "IDCardInfoEvent", "FaceTemperatureMeasurementEvent", "QRCodeEvent"(QR code event of access control),
"CertificateCaptureEvent"(person ID capture comparison event), "UncertificateCompareEvent", "ConsumptionAndTransactionRecordEvent", "ConsumptionEvent",
"TransactionRecordEvent", "SetMealQuery"(searching consumption set meals), "ConsumptionStatusQuery"(searching the consumption status), "humanBodyComparison"
(human body comparison), "regionTargetNumberCounting" (regional target statistics)-->mixedTargetDetection
          </type>
          <minorAlarm>
            <!--ro, opt, string, alarm sub type, desc:refer to the macro definition of uploaded events. "IDCardInfoEvent" is required when the type of event
is "AccessControllerEvent"-->0x400,0x401,0x402,0x403
          </minorAlarm>
          <minorException>
            <!--ro, opt, string, exception sub type, desc:refer to the macro definition of uploaded events. "IDCardInfoEvent" is required when the type of
event is "AccessControllerEvent"-->0x400,0x401,0x402,0x403
          </minorException>
          <minorOperation>
            <!--ro, opt, string, operation sub type, desc:refer to the macro definition of uploaded events. "IDCardInfoEvent" is required when the type of
event is "AccessControllerEvent"-->0x400,0x401,0x402,0x403
          </minorOperation>
          <minorEvent>
            <!--ro, opt, string, event sub type, desc:refer to the macro definition of uploaded events. "IDCardInfoEvent" is required when the type of event
is "AccessControllerEvent"-->0x01,0x02,0x03,0x04
          </minorEvent>
          <pictureURLType>
            <!--ro, opt, enum, alarm picture format of the specified event, subType:string, desc:alarm picture format of the specified event-->binary
          </pictureURLType>
          <channels>
            <!--ro, opt, string, listen to the events on the specified channel No. list, desc:if all channels are being listened to, the node shall not be
applied. if some channels are being listened to, the channel No. shall be listed and separated by commas-->1,2,3,4
          </channels>
        </Event>
      </EventList>
      <channels>
        <!--ro, opt, string, listen to the specified channel No. list, desc:if all channels are being listened to, the node shall not be applied. if some
channels are being listened to, the channel No. shall be listed and separated by commas-->1,2,3,4
      </channels>
      <pictureURLType>
        <!--ro, opt, enum, alarm picture format, subType:string, desc:the node indicates the picture types of all events which contain pictures to be
uploaded. if the node is applied, the pictureURLType of the Event will not take effect. if the node is not applied, the pictures reported from the device by
default mode: the default data type of uploaded pictures captured from front-end devices is binary. the default data type of uploaded pictures reported from
storage devices is local URL of the device.-->binary
      </pictureURLType>
      <ChangedUploadSub>
        <!--ro, opt, object, subscribe to messages-->
        <interval>
          <!--ro, opt, int, the lifecycle of arming GUID, desc:if GUID is not reconnected in the internal, a new GUID will be generated since the device
start a new arm period-->5
        </interval>
        <StatusSub>
          <!--ro, opt, object, sub status-->
          <all>
            <!--ro, opt, bool, subscribe to all?-->true
          </all>
          <channel>
            <!--ro, opt, bool, subscribe to channel status, desc:reporting is not required if all is true-->true
          </channel>
          <hd>
            <!--ro, opt, bool, subscribe to HDD status, desc:reporting is not required if all is true-->true
          </hd>
          <capability>
            <!--ro, opt, bool, subscribe to changing status of capability set, desc:reporting is not required if all is true-->true
          </capability>
        </StatusSub>
      </ChangedUploadSub>
    </SubscribeEvent>
    <PackingSpaceRecognition>
      <!--ro, opt, object, current control parameters of event listened by parking space detector on the security control panel, desc:related event:
PackingSpaceRecognition-->
      <upLoadPicturesType>
        <!--ro, opt, enum, uploaded picture type, subType:string, desc:"all", "picturesTypes" (upload specified types of pictures), "notUpload" (not upload
pictures)-->all
      </upLoadPicturesType>
      <PicturesTypes>
        <!--ro, opt, array, specified list of uploaded picture type, subType:object, dep:and,
{$.HttpHostNotificationList[*].HttpHostNotification.PackingSpaceRecognition.upLoadPicturesType,eq,picturesTypes}-->
        <picturesType>
          <!--ro, opt, enum, uploaded picture type, subType:string, desc:"backgroundImage" (captured background picture), "plateImage" (license plate
thumbnail)-->backgroundImage
        </picturesType>
      </PicturesTypes>
    </PackingSpaceRecognition>
  </HttpHostNotification>
  <enabled>
    <!--ro, opt, bool-->true
  </enabled>
  <netWork>
    <!--ro, opt, enum, subType:int-->1
  </netWork>
</HttpHostNotificationList>
```

10.3 Access Control (General)

10.3.1 Access Control Event Management

10.3.1.1 Get the capability of searching for access control events

Request URL

GET /ISAPI/AccessControl/AcsEvent/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "AcsEvent": {
    /*ro, req, object, access control events*/
        "AcsEventCond": {
        /*ro, opt, object, search conditions*/
            "searchID": {
            /*ro, req, object, search ID, it is used to check whether the current search requester is the same as the previous one. If they are the same,
the search record will be stored in the device to speed up the next search*/
                "@min":  1,
                /*ro, req, int, the minimum value*/
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
            "searchResultPosition": {
            /*ro, req, object, the start position of the search result in the result list*/
                "@min":  1,
                /*ro, req, int, the minimum value*/
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
            "maxResults": {
            /*ro, req, object, the maximum number of search results that can be obtained by calling this URL*/
                "@min":  1,
                /*ro, req, int, the minimum value*/
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
            "major": {
            /*ro, opt, object, major alarm type (the type value should be transformed to the decimal number)*/
                "@opt":  "0,1,2,3,5"
                /*ro, req, string, major type*/
            },
            "minorAlarm": {
            /*ro, opt, object, minor alarm type (the type value should be transformed to the decimal number)*/
                "@opt":  "1024,1025,1026,1027…"
                /*ro, req, string, minor alarm type*/
            },
            "minorException": {
            /*ro, opt, object, minor exception type (the type value should be transformed to the decimal number)*/
                "@opt":  "39,58,59,1024…"
                /*ro, req, string, minor exception type*/
            },
            "minorOperation": {
            /*ro, opt, object, minor operation type (the type value should be transformed to the decimal number)*/
                "@opt":  "80,90,112,113…"
                /*ro, req, string, minor operation type*/
            },
            "minorEvent": {
            /*ro, opt, object, minor event type (the type value should be transformed to the decimal number)*/
                "@opt":  "1,2,3,4…"
                /*ro, req, string, minor event type*/
            },
            "startTime": {
            /*ro, opt, object, start time*/
                "@min":  0,
                /*ro, req, int, the minimum value, range:[0,32]*/
                "@max":  32
                /*ro, req, int, the maximum value, range:[0,32]*/
            },
            "endTime": {
            /*ro, opt, object, end time*/
                "@min":  0,
                /*ro, req, int, the minimum value, range:[0,32]*/
                "@max":  32
                /*ro, req, int, the maximum value, range:[0,32]*/
            },
            "cardNo": {
            /*ro, opt, object, card No.*/
```

```
                "@min":  1,
                /*ro, req, int, the minimum value*/
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
            "beginSerialNo": {
            /*ro, opt, object, start serial No.*/
                "@min":  1,
                /*ro, req, int, the minimum value*/
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
            "endSerialNo": {
            /*ro, opt, object, end serial No.*/
                "@min":  1,
                /*ro, req, int, the minimum value*/
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
            "employeeNoString": {
            /*ro, opt, object, employee No.*/
                "@min":  1,
                /*ro, req, int, the minimum value*/
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
        },
        "InfoList": {
        /*ro, opt, object, information list*/
            "maxSize":  10,
            /*ro, opt, int, the maximum value*/
            "time": {
            /*ro, opt, object, time (UTC time)*/
                "@min":  0,
                /*ro, req, int, the minimum value, range:[0,32]*/
                "@max":  32
                /*ro, req, int, the maximum value, range:[0,32]*/
            },
            "netUser": {
            /*ro, opt, object, user name*/
                "@min":  1,
                /*ro, req, int, the minimum value*/
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
            "remoteHostAddr": {
            /*ro, opt, object, remote host address*/
                "@min":  1,
                /*ro, req, int, the minimum value*/
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
            "cardNo": {
            /*ro, opt, object, card No.*/
                "@min":  1,
                /*ro, req, int, the minimum value*/
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
            "cardType": {
            /*ro, opt, object, card type*/
                "@opt":  "1,2,3,4,5,6,7,8"
                /*ro, req, string, 1 (normal card), 2 (disability card), 3 (blocklist card), 4 (patrol card), 5 (duress card), 6 (super card), 7 (visitor
card), 8 (dismiss card)*/
            },
            "reportChannel": {
            /*ro, opt, object, channel type for uploading alarms/events*/
                "@opt":  "1,2,3"
                /*ro, req, string, "1" (for uploading arming information), "2" (for uploading by central group 1), "3" (for uploading by central group 2)*/
            },
            "cardReaderKind": {
            /*ro, opt, object, card reader type*/
                "@opt":  "1,2,3,4"
                /*ro, req, string, "1" (IC card reader), "2" (ID card reader), "3" (QR code scanner), "4" (fingerprint module)*/
            },
            "cardReaderNo": {
            /*ro, opt, object, card reader No.*/
                "@min":  1,
                /*ro, req, int, the minimum value*/
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
            "doorNo": {
            /*ro, opt, object, door (floor) No.*/
                "@min":  1,
                /*ro, req, int, the minimum value*/
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
            "verifyNo": {
            /*ro, opt, object, multiple authentication No.*/
                "@min":  1,
                /*ro, req, int, the minimum value*/
```

```
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
            "alarmOutNo": {
            /*ro, opt, object, alarm output No.*/
                "@min":  1,
                /*ro, req, int, the minimum value*/
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
            "caseSensorNo": {
            /*ro, opt, object, event trigger No.*/
                "@min":  1,
                /*ro, req, int, the minimum value*/
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
            "deviceNo": {
            /*ro, opt, object, device No.*/
                "@min":  1,
                /*ro, req, int, the minimum value*/
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
            "MACAddr": {
            /*ro, opt, object, MAC Address*/
                "@min":  1,
                /*ro, req, int, the minimum value*/
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
            "serialNo": {
            /*ro, opt, object, event serial No.*/
                "@min":  1,
                /*ro, req, int, the minimum value*/
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
            "userType": {
            /*ro, opt, object, person types*/
                "@opt":  "normal,visitor,blackList,administrators"
                /*ro, req, string, "normal" (normal person (household)), "visitor" (visitor), "blacklist" (person in blocklist), "administrators"
(administrator)*/
            },
            "currentVerifyMode": {
            /*ro, opt, object, current authentication mode of the card reader*/
                "@opt":
"cardAndPw,card,cardOrPw,fp,fpAndPw,fpOrCard,fpAndCard,fpAndCardAndPw,faceOrFpOrCardOrPw,faceAndFp,faceAndPw,faceAndCard,face,employeeNoAndPw,fpOrPw,employe
eNoAndFp,employeeNoAndFpAndPw,faceAndFpAndCard,faceAndPwAndFp,employeeNoAndFace,faceOrfaceAndCard,fpOrface,cardOrfaceOrPw,iris,faceOrFpOrCardOrPwOrIris,face
OrCardOrPwOrIris"
                /*ro, req, string, "cardAndPw"-card+password, "card", "cardOrPw"-card or password, "fp"-fingerprint, "fpAndPw"-fingerprint+password,
"fpOrCard"-fingerprint or card, "fpAndCard"-fingerprint+card, "fpAndCardAndPw"-fingerprint+card+password, "faceOrFpOrCardOrPw"-face or fingerprint or card
or password, "faceAndFp"-face+fingerprint, "faceAndPw"-face+password, "faceAndCard"-face+card, "face", "employeeNoAndPw"-employee No.+password, "fpOrPw"-
fingerprint or password, "employeeNoAndFp"-employee No.+fingerprint, "employeeNoAndFpAndPw"-employee No.+fingerprint+password, "faceAndFpAndCard"-
face+fingerprint+card, "faceAndPwAndFp"-face+password+fingerprint, "employeeNoAndFace"-employee No.+face, "faceOrfaceAndCard"-face or face+card, "fpOrface"-
fingerprint or face, "cardOrfaceOrPw"-card or face or password, "cardOrFpOrPw"-card or fingerprint or password*/
            },
            "attendanceStatus": {
            /*ro, opt, object, attendance status, desc:"undefined", "checkIn" (check-in), "checkOut" (check-out), "breakOut" (start of break), "breakIn"
(end of break), "overtimeIn" (start of overtime), "overTimeOut" (end of overtime)*/
                "@opt":  "undefined,checkIn,checkOut,breakOut,breakIn,overtimeIn,overtimeOut"
                /*ro, req, string, options*/
            },
            "statusValue": {
            /*ro, opt, object, status value*/
                "@min":  1,
                /*ro, req, int, the minimum value*/
                "@max":  1
                /*ro, req, int, the maximum value*/
            },
        }
    }
}
```


10.3.1.2 Search for access control events

Request URL

POST /ISAPI/AccessControl/AcsEvent?format=json

Query Parameter

None

Request Message

```
{
    "AcsEventCond": {
    /*req, object, access control events*/
        "searchID":  "test",
        /*req, string, search ID, desc:it is used to check whether the current search requester is the same as the previous one. If they are the same, the search record will be stored in the device to speed up the next search*/
        "searchResultPosition":  0,
        /*req, int, the start position of the search result in the result list:, desc:in a single search, if you cannot get all the records in the result list, you can mark the end position and get the following records after the marked position in the next search. If the maximum number of totalMatches supported by the device is M and the number of totalMatches stored in the device now is N (N<=M), the valid range of this node is 0 to N-1*/
        "maxResults":  30,
        /*req, int, the maximum number of search results, which is defined by the device capability, will be returned if the value of maxResults reaches the limit, desc:if maxResults exceeds the range returned by the device capability, the device will return the maximum number of search results according to the device capability and will not return error message*/
        "major":  1,
        /*req, int, major type, desc:the type value should be transformed to the decimal number; see Access Control Event Types for details*/
        "minor":  1024,
        /*req, int, minor type, desc:the type value should be transformed to the decimal number; see Access Control Event Types for details*/
        "startTime":  "1970-01-01T00:00:00+08:00",
        /*opt, datetime, start time (UTC time)*/
        "endTime":  "1970-01-01T00:00:00+08:00",
        /*opt, datetime, end time (UTC time)*/
        "cardNo":  "test",
        /*opt, string, card No.*/
        "name":  "test",
        /*opt, string, name of the card holder*/
        "videoChannel":  1,
        /*opt, int, video channel No., range:[1,86400], desc:this node is newly added to DeepinMind devices for attendance*/
        "picEnable":  true,
        /*opt, bool, whether to upload the picture along with the event information, desc:false (no), true (yes, default value); (1. all matched events will be uploaded without pictures; 2. all matched events will be uploaded with pictures if there are any; 3. if this node is not configured, the default value is true)*/
        "beginSerialNo":  1,
        /*opt, int, start serial No.*/
        "endSerialNo":  1,
        /*opt, int, end serial No.*/
        "employeeNoString":  "test",
        /*opt, string, employee No. (person ID)*/
        "timeReverseOrder":  true,
        /*opt, bool, whether to return events in descending order of time (later events will be returned first), desc:true (yes), false or this node is not returned (no)*/
        "isAbnomalTemperature":  true,
        /*opt, bool, whether the skin-surface temperature is abnormal*/
        "temperatureSearchCond":  "all",
        /*opt, enum, temperature search condition, subType:string, desc:when this node and isAbnormalTemperature both exist, isAbnormalTemperature is invalid; "all" (event with temperature), "normal" (event with normal temperature), "abnormal" (event with abnormal temperature)*/
        "isAttendanceInfo":  true,
        /*opt, bool, whether it contains attendance records, desc:this node is newly added to HEOP protocol; if this node is true, main type, minor type, employee No., name, and time will be returned*/
        "hasRecordInfo":  true
        /*opt, bool*/
    }
}
```

Response Message

```
{
    "AcsEvent": {
    /*ro, req, object, access control events*/
        "searchID":  "test",
        /*ro, req, string, search ID, it is used to check whether the current search requester is the same as the previous one. If they are the same, the search record will be stored in the device to speed up the next search*/
        "responseStatusStrg":  "OK",
        /*ro, req, string, searching status description*/
        "numOfMatches":  1,
        /*ro, req, int, number of results returned this time*/
        "totalMatches":  1,
        /*ro, req, int, total number of matched results*/
        "InfoList": [
        /*ro, opt, array, information list, subType:object*/
            {
                "major":  1,
                /*ro, req, int, major alarm type*/
                "minor":  1,
                /*ro, req, int, minor alarm type*/
                "time":  "2016-12-12T17:30:08+08:00",
                /*ro, req, string, time (UTC time)*/
                "netUser":  "test",
                /*ro, opt, string, user name*/
                "remoteHostAddr":  "test",
                /*ro, opt, string, remote host address*/
                "videoChannel":  1,
                /*ro, opt, int, video channel No., range:[1,86400], desc:this node is newly added to DeepinMind devices for attendance*/
                "cardNo":  "test",
                /*ro, opt, string, card No.*/
                "cardType":  1,
                /*ro, opt, enum, card type, subType:int, desc:1 (normal card), 2 (disability card), 3 (blocklist card), 4 (patrol card), 5 (duress card), 6 (super card), 7 (visitor card), 8 (dismiss card)*/
                "whiteListNo":  1,
```

```
                /*ro, opt, int, allowlist No.*/
                "reportChannel": 1,
                /*ro, opt, int, channel type for uploading alarm/event*/
                "cardReaderKind": 1,
                /*ro, opt, int, card reader type: 1 (IC card reader)*/
                "cardReaderNo": 1,
                /*ro, opt, int, card reader No.*/
                "doorNo": 1,
                /*ro, opt, int, door or floor No.*/
                "verifyNo": 1,
                /*ro, opt, int, multi-factor authentication No.*/
                "alarmInNo": 1,
                /*ro, opt, int, alarm input No.*/
                "alarmOutNo": 1,
                /*ro, opt, int, alarm output No.*/
                "caseSensorNo": 1,
                /*ro, opt, int, event trigger No.*/
                "RS485No": 1,
                /*ro, opt, int, RS-485 channel No.*/
                "multiCardGroupNo": 1,
                /*ro, opt, int, group No.*/
                "accessChannel": 1,
                /*ro, opt, int, RS-485 channel No.*/
                "deviceNo": 1,
                /*ro, opt, int, device No.*/
                "distractControlNo": 1,
                /*ro, opt, int, distributed controller No.*/
                "employeeNoString": "test",
                /*ro, opt, string, employee No. (person ID)*/
                "localControllerID": 1,
                /*ro, opt, int, distributed controller No.*/
                "InternetAccess": 1,
                /*ro, opt, int, network interface No.*/
                "type": 1,
                /*ro, opt, int, zone type, desc:0 (instant alarm zone), 1 (24-hour zone), 2 (delayed zone), 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 zone), 11 (emergency door closed zone), 255 (none)*/
                "MACAddr": "test",
                /*ro, opt, string, MAC address*/
                "swipeCardType": 1,
                /*ro, opt, enum, card swiping type, subType:int, desc:0 (invalid), 1 (QR code)*/
                "serialNo": 1,
                /*ro, opt, int, event serial No.*/
                "channelControllerID": 1,
                /*ro, opt, int, lane controller ID*/
                "channelControllerLampID": 1,
                /*ro, opt, int, light board ID of lane controller, range:[1,255]*/
                "channelControllerIRAdaptorID": 1,
                /*ro, opt, int, IR adapter ID of lane controller, range:[1,255]*/
                "channelControllerIREmitterID": 1,
                /*ro, opt, int, active infrared intrusion detector No. of lane controller, range:[1,255]*/
                "userType": "normal",
                /*ro, opt, string, person type*/
                "currentVerifyMode": "cardAndPw",
                /*ro, opt, enum, current authentication mode of the card reader, subType:string, desc:"cardAndPw" (card + password); "card", "cardOrPw" (card or password), "fp" (fingerprint), "fpAndPw" (fingerprint + password), "fpOrCard" "fingerprint or card", "fpAndCard" (fingerprint + card), "fpAndCardAndPw" (fingerprint + card + password), "faceOrFpOrCardOrPw" (face or fingerprint or card or password), "faceAndFp" (face + fingerprint), "faceAndPw" (face + password), "faceAndCard" (face + card), "face", "employeeNoAndPw" (emplyee No. +password), "fpOrPw" (fingerprint or password), "employeeNoAndFp" (employee No. + fingerprint), "employeeNoAndFpAndPw" (employee No. + fingerprint + password), "faceAndFpAndCard" (face + fingerprint + card), "faceAndPwAndFp" (face + password + fingerprint), "employeeNoAndFace" (employee No. + face), "faceOrfaceAndCard" (face or face + card), "fpOrface" (fingerprint or face), "cardOrfaceOrPw" (card or face or password), "faceOrFpOrCardOrPwOrIris" (face or fingerprint or card or password or iris), "faceOrCardOrPwOrIris" (face or card or password or iris), "sleep", "invalid"*/
                "QRCodeInfo": "test",
                /*ro, opt, string, QR code information*/
                "thermometryUnit": "celsius",
                /*ro, opt, enum, temperature unit, subType:string, desc:"celsius" (Celsius, default value), "fahrenheit" (Fahrenheit), "kelvin" (Kelvin)*/
                "currTemperature": 36.5,
                /*ro, opt, float, skin-surface temperature, which is accurate to one decimal place*/
                "isAbnomalTemperature": true,
                /*ro, opt, bool, whether the skin-surface temperature is abnormal (true-yes)*/
                "RegionCoordinates": {
                /*ro, opt, object, coordinates of the skin-surface temperature*/
                    "positionX": 254,
                    /*ro, opt, int, normalized X-coordinate which is between 0 and 1000*/
                    "positionY": 133
                    /*ro, opt, int, normalized Y-coordinate which is between 0 and 1000*/
                },
                "mask": "unknown",
                /*ro, opt, enum, whether the person wears a mask, subType:string, desc:"unknown"*/
                "pictureURL": "test",
                /*ro, opt, string, picture URL*/
                "filename": "picture1",
                /*ro, opt, string, file name, desc:if multiple pictures are returned at a time, filename of each picture should be unique*/
                "attendanceStatus": "undefined",
                /*ro, opt, enum, attendance status, subType:string, desc:"undefined", "checkIn" (check-in), "checkOut" (check-out), "breakOut" (start of break), "breakIn" (end of break), "overtimeIn" (start of overtime), "overTimeOut" (end of overtime)*/
                "label": "test",
                /*ro, opt, string, custom attendance name*/
                "statusValue": 1,
                /*ro, opt, int, status value*/
                "helmet": "unknown",
                /*ro, opt, enum, whether the person wears a hard hat, subType:string, desc:"unknown", "yes", "no"*/
                "visibleLightPicUrl": "test",
                /*ro, opt, string, visible light picture URL*/
```

```
                /*ro, opt, string, visible light picture URL*/
                "thermalPicUrl":  "test",
                /*ro, opt, string, URL of the thermal imaging picture*/
                "appType":  "attendance",
                /*ro, opt, enum, application type, subType:string, desc:"attendance" (Time & Attendance module), "signIn" (Check-In module, which is only
used for FocSign products)*/
                "HealthInfo": {
                /*ro, opt, object, health information*/
                    "healthCode":  1,
                    /*ro, opt, enum, health code status, subType:int, desc: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)*/
                    "NADCode":  1,
                    /*ro, opt, enum, nucleic acid test result, subType:int, desc:0 (no result), 1 (negative, which means normal), 2 (positive, which means
diagnosed), 3 (the result has expired)*/
                    "travelCode":  1,
                    /*ro, opt, enum, trip code, subType:int, desc:0 (no trip in the past 14 days), 1 (has left the current area in the past 14 days), 2 (has
been to the high-risk area in the past 14 days), 3 (other)*/
                    "travelInfo":  "test",
                    /*ro, opt, string*/
                    "vaccineStatus":  1,
                    /*ro, opt, enum, whether the person is vaccinated, subType:int, desc:0 (not vaccinated), 1 (vaccinated)*/
                    "vaccineNum":  1
                    /*ro, opt, int, step:1*/
                },
                "meetingID":  "test",
                /*ro, opt, string, meeting ID*/
                "PersonInfoExtends": [
                /*ro, opt, array, additional person information, subType:object, desc:this node displays additional person information on the device*/
                    {
                        "id":  1,
                        /*ro, opt, int, extended ID of the additional person information, range:[1,32], desc:related URL:
/ISAPI/AccessControl/personInfoExtendName?format=json; this node is used for displaying the name of value; if ID does not exists, it starts from 1*/
                        "value":  "test"
                        /*ro, opt, string, extended content of the additional person information*/
                    }
                ],
                "name":  "test",
                /*ro, opt, string, name, desc:person name*/
                "FaceRect": {
                /*ro, opt, object, rectangle frame for human face, desc:the origin is the upper-left corner of the screen*/
                    "height":  1.000,
                    /*ro, req, float, height, range:[0.000,1.000]*/
                    "width":  1.000,
                    /*ro, req, float, width, range:[0.000,1.000]*/
                    "x":  0.000,
                    /*ro, req, float, X-coordinate of the upper-left corner of the frame, range:[0.000,1.000]*/
                    "y":  0.000
                    /*ro, req, float, Y-coordinate of the upper-left corner of the frame, range:[0.000,1.000]*/
                },
                "RecordInfo": {
                /*ro, opt, object*/
                    "startTime":  "1970-01-01T00:00:00+08:00",
                    /*ro, opt, datetime, recording start time*/
                    "endTime":  "1970-01-01T00:00:00+08:00",
                    /*ro, opt, datetime, recording end time*/
                    "playbackURL":  "rtsp://10.65.130.168:554/ISAPI/Streaming/tracks/201/?starttime=20190213T091134Z&amp;endtime=20190213T092116Z"
                    /*ro, opt, string, range:[0,256]*/
                },
                "currentAuthenticationTimes":  1,
                /*ro, opt, int, range:[0,255], step:1*/
                "allowAuthenticationTimes":  1
                /*ro, opt, int, range:[0,255], step:1*/
            }
        ]
    }
}
```

10.3.1.3 Get the capability of getting total number of access control events by specific conditions

Request URL

GET /ISAPI/AccessControl/AcsEventTotalNum/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "AcsEvent": {
    /*ro, opt, object*/
        "AcsEventTotalNumCond": {
        /*ro, opt, object, search conditions*/
            "major": {
```

```
            /*ro, req, object, major alarm type*/
                "@opt":  "0,1,2,3,5"
                /*ro, opt, string, major alarm type*/
            },
            "minorAlarm": {
            /*ro, req, object, minor alarm type*/
                "@opt":  "1024,1025,1026,1027"
                /*ro, opt, string, minor alarm type*/
            },
            "minorException": {
            /*ro, req, object, minor exception type*/
                "@opt":  "39,58,59,1024"
                /*ro, opt, string, minor exception type*/
            },
            "minorOperation": {
            /*ro, req, object, minor operation type*/
                "@opt":  "80,90,112,113"
                /*ro, opt, string, minor operation type*/
            },
            "minorEvent": {
            /*ro, opt, object, minor event type*/
                "@opt":  "1,2,3,4"
                /*ro, opt, string, minor event type*/
            },
            "startTime": {
            /*ro, opt, object, start time*/
                "@min":  1,
                /*ro, opt, int, start time (UTC time)*/
                "@max":  1
                /*ro, opt, int, end time (UTC time)*/
            },
            "endTime": {
            /*ro, opt, object, end time*/
                "@min":  1,
                /*ro, opt, int, start time (UTC time)*/
                "@max":  1
                /*ro, opt, int, end time (UTC time)*/
            },
            "cardNo": {
            /*ro, opt, object, card No.*/
                "@min":  1,
                /*ro, opt, int, card No.*/
                "@max":  32
                /*ro, opt, int, card No.*/
            },
            "name": {
            /*ro, opt, object, name of the card holder*/
                "@min":  1,
                /*ro, opt, int, name of the card holder*/
                "@max":  32
                /*ro, opt, int, name of the card holder*/
            },
            "picEnable":  "true,false",
            /*ro, opt, string*/
            "beginSerialNo": {
            /*ro, opt, object, start serial No.*/
                "@min":  1,
                /*ro, opt, int, start serial No.*/
                "@max":  1
                /*ro, opt, int, start serial No.*/
            },
            "endSerialNo": {
            /*ro, opt, object, end serial No.*/
                "@min":  1,
                /*ro, opt, int, end serial No.*/
                "@max":  1
                /*ro, opt, int, end serial No.*/
            },
            "employeeNoString": {
            /*ro, opt, object, employee No. (person ID)*/
                "@min":  1,
                /*ro, opt, int, employee No. (person ID)*/
                "@max":  32
                /*ro, opt, int, employee No. (person ID)*/
            }
        },
        "totalNum": {
        /*ro, req, object*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  1
            /*ro, opt, int*/
        }
    }
}
```


10.3.1.4 Get the total number of access control events by specific conditions

Request URL

POST /ISAPI/AccessControl/AcsEventTotalNum?format=json

Query Parameter

None

Request Message

```
{
    "AcsEventTotalNumCond": {
    /*req, object*/
        "major": 1,
        /*req, int, major alarm type, desc:(the type value should be transformed to the decimal number), refer to Access Control Event Types for details*/
        "minor": 1024,
        /*req, int, sub type, step:1, desc:(the type value should be transformed to the decimal number),refer to Access Control Event Types for details*/
        "startTime": "1970-01-01+08:00",
        /*opt, date, start time (UTC time)*/
        "endTime": "1970-01-01+08:00",
        /*opt, date, end time (UTC time)*/
        "cardNo": "test",
        /*opt, string, card No.*/
        "name": "test",
        /*opt, string, name of the card holder*/
        "picEnable": true,
        /*opt, bool, whether to upload the picture along with the event information, desc:whether to contain pictures: "true"-yes,"false"-no*/
        "beginSerialNo": 1,
        /*opt, int, start serial No.*/
        "endSerialNo": 100,
        /*opt, int, end serial No.*/
        "employeeNoString": "test"
        /*opt, string, employee No. (person ID), range:[1,32]*/
    }
}
```

Response Message

```
{
    "AcsEventTotalNum": {
    /*ro, req, object*/
        "totalNum": 1,
        /*ro, req, int, total number of events that match the search conditions*/
        "existedEventNum": 1
        /*ro, opt, int*/
    }
}
```

10.3.1.5 Get the capability of clearing event and card linkage parameters

Request URL

GET /ISAPI/AccessControl/ClearEventCardLinkageCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "ClearEventCardLinkageCfg": {
    /*ro, opt, object, clear event and card linkage parameters*/
        "ClearFlags": {
        /*ro, opt, object*/
            "eventCardLinkage": "true,false"
            /*ro, req, string, event and card linkage parameters*/
        }
    }
}
```

10.3.1.6 Clear event card linkage configurations

Request URL

PUT /ISAPI/AccessControl/ClearEventCardLinkageCfg?format=json

Query Parameter

None

Request Message

```
{
    "ClearEventCardLinkageCfg": {
    /*req, object*/
        "ClearFlags": {
        /*opt, object*/
            "eventCardLinkage":  true
            /*req, bool, whether to clear event and card linkage parameters*/
        }
    }
}
```

Response Message

```
{
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded); it is required when an error occurred*/
    "statusString":  "ok",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "subStatusCode":  "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:when the value of statusCode is not 1, it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, opt, string, error information, desc:this node must be returned when the value of statusCode is not 1*/
}
```

10.3.1.7 Getting arming information

Request URL

GET /ISAPI/AccessControl/DeployInfo

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<DeployInfo xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, opt, object, arming information, attr:version{req, string, protocolVersion}-->
  <DeployList size="5">
    <!--ro, opt, array, arming list, subType:object, attr:size{req, int}-->
    <Content>
      <!--ro, opt, object, subscribe to messages-->
      <deployNo>
        <!--ro, req, int, arming No.-->1
      </deployNo>
      <deployType>
        <!--ro, req, enum, arming type, subType:int-->1
      </deployType>
      <protocolType>
        <!--ro, opt, enum, protocol type, subType:string, dep:or,{$.DeployInfo.DeployList[*].Content.deployType,eq,2},
{$.DeployInfo.DeployList[*].Content.deployType,eq,3}, desc:"HTTP", "HTTPS"-->HTTP
      </protocolType>
      <ipAddr>
        <!--ro, req, string, IP address-->test
      </ipAddr>
      <port>
        <!--ro, opt, int, port No., range:[1,65535]-->1
      </port>
      <eventType>
        <!--ro, opt, enum, subType:string-->AccessController
      </eventType>
    </Content>
  </DeployList>
</DeployInfo>
```

10.3.1.8 Getting arming information capability

Request URL

GET /ISAPI/AccessControl/DeployInfo/capabilities

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<DeployInfo xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, opt, object, arming Information, attr:version{req, string, protocolVersion}-->
  <DeployList size="5">
    <!--ro, opt, array, arming list, subType:object, attr:size{req, int}-->
    <Content>
      <!--ro, opt, object-->
      <deployNo min="1" max="10">
        <!--ro, req, int, arming No., attr:min{req, int},max{req, int}-->1
      </deployNo>
      <deployType opt="0,1,2,3">
        <!--ro, req, int, arming type, attr:opt{req, string}-->1
      </deployType>
      <protocolType opt="HTTP,HTTPS">
        <!--ro, opt, enum, protocol type, subType:string, attr:opt{req, string}, desc:"HTTP", "HTTPS"-->HTTP
      </protocolType>
      <ipAddr min="1" max="10">
        <!--ro, req, string, IP address, attr:min{req, int},max{req, int}-->test
      </ipAddr>
      <port min="0" max="10">
        <!--ro, opt, int, port No., range:[1,65535], attr:min{req, int},max{req, int}-->1
      </port>
      <eventType opt="AccessController,Consumer,AccessControllerAndConsumer">
        <!--ro, opt, enum, subType:string, attr:opt{req, string}-->AccessController
      </eventType>
    </Content>
  </DeployList>
</DeployInfo>
```

10.3.1.9 Set the event card linkage parameters

Request URL

PUT /ISAPI/AccessControl/EventCardLinkageCfg/<ACEID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
ACEID	string	--

Request Message

```
{
    "EventCardLinkageCfg": {
    /*req, object, event card linkage parameters*/
        "proMode":  "event",
        /*req, enum, linkage type, subType:string, desc:"event" (event linkage), "card" (card linkage), "mac" (MAC address linkage), "employee" (employee
No., i.e., person ID)*/
        "EventLinkageInfo": {
        /*opt, object, event linkage parameters, desc:it is valid when proMode is "event"*/
            "mainEventType":  0,
            /*opt, enum, major event type, subType:int, desc:0-device event, 1-alarm input event, 2-access control point event, 3-authentication unit (card
reader, fingerprint module) event*/
            "subEventType":  54
            /*opt, int, sub event type, desc:minor event type,refer to Event Linkage Types for details*/
        },
        "eventSourceID":  1,
        /*opt, int, event source ID, desc:it is valid when proMode is "event". For device event (mainEventType is 0), this field is invalid; for access
control point event (mainEventType is 2), this field refers to the access control point No.; for authentication unit event (mainEventType is 3), this field
refers to the authentication unit No.; for alarm input event (mainEventType is 1), this field refers to the zone alarm input ID or the event alarm input ID
65535-all*/
        "mainDevBuzzer":  true,
        /*opt, bool, whether to enable buzzer linkage of the access controller (start buzzing):, desc:false-no, true-yes*/
        "mainDevStopBuzzer":  true,
        /*opt, bool, whether to enable buzzer linkage of access controller (stop buzzing), desc:false-no, true-yes*/
    }
}
```

Response Message

```
{
    "requestURL":  "test",
    /*ro, opt, string, URI*/
    "statusCode":  "test",
    /*ro, opt, string, status code*/
    "statusString":  "test",
    /*ro, opt, string, status description*/
    "subStatusCode":  "test",
    /*ro, opt, string, sub status code*/
    "errorCode":  1,
    /*ro, req, int, error code*/
    "errorMsg":  "ok"
    /*ro, req, string, error description*/
}
```

10.3.1.10 Get the event and card linkage configuration parameters

Request URL

GET /ISAPI/AccessControl/EventCardLinkageCfg/<ACEID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
ACEID	string	--

Request Message

None

Response Message

```
{
    "EventCardLinkageCfg": {
    /*ro, req, object*/
        "proMode":  "event",
        /*ro, req, enum, linkage type, subType:string, desc:"event"-event linkage, "card"-card linkage, "mac"-MAC address linkage, "employee"-employee No.
(person ID)*/
        "EventLinkageInfo": {
        /*ro, opt, object, event linage parameters, desc:it is valid when proMode is "event"*/
            "mainEventType":  0,
            /*ro, opt, enum, major event type, subType:int, desc:0-device event,1-alarm input event,2-access control point event,3-authentication unit (card
reader, fingerprint module) event*/
            "subEventType":  54
            /*ro, opt, int, event sub type, desc:minor event type,refer to Event Linkage Types for details*/
        },
        "eventSourceID":  1,
        /*ro, opt, int, event source ID, desc:it is valid when proMode is "event",65535-all. For device event (mainEventType is 0),this field is invalid;
for access control point event (mainEventType is 2),this field refers to the access control point No.; for authentication unit event (mainEventType is
3,this field refers to the authentication unit No.; for alarm input event (mainEventType is 1),this field refers to the zone alarm input ID or the event
alarm input ID*/
        "mainDevBuzzer":  true,
        /*ro, opt, bool, whether to enable buzzer linkage of the access controller (start buzzing), desc:"false"-no, "true"-yes*/
        "mainDevStopBuzzer":  true,
        /*ro, opt, bool, whether to enable buzzer linkage of access controller (stop buzzing), desc:"false"-no,"true"-yes*/
    }
}
```

10.3.1.11 Get the configuration capability of the event and card linkage

Request URL

GET /ISAPI/AccessControl/EventCardLinkageCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "EventCardLinkageCfg": {
    /*ro, req, object, parameters of the event and card linkage*/
        "eventID": {
```

```
"eventID" : {
/*ro, opt, object, event ID*/
    "@min":  1,
    /*ro, opt, int*/
    "@max":  1
    /*ro, opt, int*/
},
"proMode": {
/*ro, req, object, linkage type*/
    "@opt":  "event,card,mac,employee"
    /*ro, opt, string, linkage method*/
},
"EventLinkageInfo": {
/*ro, opt, object, event linkage information*/
    "mainEventType": {
    /*ro, opt, object, event main type*/
        "@opt":  "0,1,2,3"
        /*ro, opt, string, event main type*/
    },
    "devSubEventType": {
    /*ro, opt, object, minor event type*/
        "@opt":  "0,1,2,3,54…"
        /*ro, opt, string, minor event type*/
    },
    "alarmSubEventType": {
    /*ro, opt, object, minor type of alarm input event*/
        "@opt":  "0,1,2,3,52…"
        /*ro, opt, string, minor type of alarm input event*/
    },
    "doorSubEventType": {
    /*ro, opt, object, minor type of access control point event*/
        "@opt":  "0,1,2,3…"
        /*ro, opt, string, minor type of access control point event*/
    },
    "cardReaderSubEventType": {
    /*ro, opt, object, minor type of authentication unit event*/
        "@opt":  "0,1,2,3…"
        /*ro, opt, string, minor type of authentication unit event*/
    }
},
"CardNoLinkageInfo": {
/*ro, opt, object, card linkage parameters*/
    "cardNo": {
    /*ro, opt, object, card No.*/
        "@min":  1,
        /*ro, opt, int*/
        "@max":  32
        /*ro, opt, int*/
    }
},
"EmployeeInfo": {
/*ro, opt, object, person ID*/
    "employeeNo": {
    /*ro, opt, object, person ID*/
        "@min":  1,
        /*ro, opt, int, employee No. (person ID)*/
        "@max":  32
        /*ro, opt, int*/
    }
},
"eventSourceID": {
/*ro, opt, object, event source ID*/
    "@min":  1,
    /*ro, opt, int*/
    "@max":  1
    /*ro, opt, int*/
},
"alarmout": {
/*ro, opt, object, linked alarm output No.*/
    "@min":  1,
    /*ro, opt, int*/
    "@max":  1
    /*ro, opt, int*/
},
"openDoor": {
/*ro, opt, object, linked door No. to open*/
    "@min":  1,
    /*ro, opt, int*/
    "@max":  1
    /*ro, opt, int*/
},
"closeDoor": {
/*ro, opt, object, linked door No. to close*/
    "@min":  1,
    /*ro, opt, int*/
    "@max":  1
    /*ro, opt, int*/
},
"alwaysOpen": {
/*ro, opt, object, array,linked door No. to remain unlocked*/
    "@min":  1,
    /*ro, opt, int*/
    "@max":  1
    /*ro, opt, int*/
```

```
        },
        "alwaysClose": {
        /*ro, opt, object, linked door No. to remain locked*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  1
            /*ro, opt, int*/
        },
        "mainDevBuzzer":  "true,false",
        /*ro, opt, string, buzzer linkage of the access controller*/
        "mainDevStopBuzzer":  "true,false",
        /*ro, opt, string, whether to enable buzzer linkage of the access controller (stop buzzing): "false"-no,"true"-yes*/
        "readerBuzzer": {
        /*ro, opt, object, linked buzzer*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  1,
            /*ro, opt, int*/
        },
        "alarmOutClose": {
        /*ro, opt, object, array,linked alarm output No.*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  1
            /*ro, opt, int*/
        },
        "readerStopBuzzer": {
        /*ro, opt, object, linked buzzer No. to stop buzzing*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  1,
            /*ro, opt, int*/
        },
    }
}
```

10.3.1.12 Get the capability of the list of event and card linkage ID

Request URL

GET /ISAPI/AccessControl/EventCardNoList/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "EventCardNoList": {
    /*ro, opt, object*/
        "id": {
        /*ro, opt, object, range of event ID that can be configured*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  1
            /*ro, opt, int*/
        }
    }
}
```

10.3.1.13 Get the list of event and card linkage ID

Request URL

GET /ISAPI/AccessControl/EventCardNoList?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "EventCardNoList": {
    /*ro, opt, object*/
        "id": [1, 2, 3]
        /*ro, req, array, list of configured event and card linkage ID, subType:int, desc:[1, 2, 3] indicates that the device is configured with event
linkage 1, 2, and 3*/
    }
}
```

10.3.1.14 Get the configuration capability of event optimization

Request URL

GET /ISAPI/AccessControl/EventOptimizationCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "EventOptimizationCfg": {
    /*ro, opt, object*/
        "enable":  "true,false",
        /*ro, opt, string, whether to enable event optimization*/
        "isCombinedLinkageEvents":  "true,false"
        /*ro, opt, string, whether to enable linked event combination*/
    }
}
```

10.3.1.15 Set the event optimization parameters

Request URL

PUT /ISAPI/AccessControl/EventOptimizationCfg?format=json

Query Parameter

None

Request Message

```
{
    "EventOptimizationCfg": {
    /*opt, object*/
        "enable":  true,
        /*opt, bool, whether to enable event optimization*/
        "isCombinedLinkageEvents":  true
        /*opt, bool, whether to enable linked event combination*/
    }
}
```

Response Message

```
{
    "requestURL":  "test",
    /*ro, opt, string, URI*/
    "statusCode":  "test",
    /*ro, opt, string, status code*/
    "statusString":  "test",
    /*ro, opt, string, status description*/
    "subStatusCode":  "test",
    /*ro, opt, string, sub status code*/
    "errorCode":  1,
    /*ro, req, int, error code*/
    "errorMsg":  "ok"
    /*ro, req, string, error details*/
}
```

10.3.1.16 Get the event optimization configuration parameters

Request URL

GET /ISAPI/AccessControl/EventOptimizationCfg?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "EventOptimizationCfg": {
    /*ro, opt, object*/
        "enable": true,
        /*ro, opt, bool, whether to enable event optimization*/
        "isCombinedLinkageEvents": true
        /*ro, opt, bool, whether to enable linked event combination*/
    }
}
```

10.3.1.17 Access control event

EventType:AccessControllerEvent

```
{
    "ipAddress": "172.6.64.7",
    /*ro, req, string, IPv4 address of the device that triggers the alarm*/
    "ipv6Address": "1080:0:0:0:8:800:200C:417A",
    /*ro, opt, string, IPv6 address of the device that triggers the alarm*/
    "portNo": 80,
    /*ro, opt, int, communication port No. of the device that triggers the alarm*/
    "protocol": "HTTP",
    /*ro, opt, enum, transmission communication protocol type, subType:string, desc:when ISAPI protocol is transmitted via HCNetSDK, the channel No. is the video channel No. of private protocol. When ISAPI protocol is transmitted via EZ protocol, the channel No. is the video channel No. of EZ protocol. When ISAPI protocol is transmitted via ISUP, the channel No. is the video channel No. of ISUP*/
    "macAddress": "01:17:24:45:D9:F4",
    /*ro, opt, string, MAC address*/
    "channelID": 1,
    /*ro, opt, int, channel No. of the device that triggers the alarm, desc:when ISAPI protocol is transmitted via HCNetSDK, the channel No. is the video channel No. of private protocol. When ISAPI protocol is transmitted via EZ protocol, the channel No. is the video channel No. of EZ protocol. When ISAPI protocol is transmitted via ISUP, the channel No. is the video channel No. of ISUP*/
    "dateTime": "2004-05-03T17:30:08+08:00",
    /*ro, req, datetime, alarm trigger time*/
    "activePostCount": 1,
    /*ro, req, int, times that the same alarm has been uploaded, desc:times that the same alarm has been uploaded*/
    "eventType": "AccessControllerEvent",
    /*ro, req, string, event type, desc:"AccessControllerEvent" (access control event)*/
    "eventState": "active",
    /*ro, req, enum, event status, subType:string, desc:for durative event: "active" (valid event or event starts), "inactive" (invalid event or the event ends). For the heartbeat, the node value indicates the heartbeat data, and it is uploaded every 10 seconds*/
    "eventDescription": "AccessControllerEvent",
    /*ro, req, string, event description, desc:"AccessControllerEvent" (access control event)*/
    "deviceID": "test0123",
    /*ro, opt, string, device ID (PUID), desc:this node must be returned when ISAPI event information is transmitted via ISUP*/
    "AccessControllerEvent": {
    /*ro, req, object, access control event information*/
        "deviceName": "test",
        /*ro, opt, string, device name*/
        "majorEventType": 1,
        /*ro, req, int, major alarm type, desc:the type value should be transformed to the decimal number; see Access Control Alarm Types for details*/
        "subEventType": 1,
        /*ro, req, int, minor alarm type, desc:the type value should be transformed to the decimal number; see Access Control Alarm Types for details*/
        "inductiveEventType": "authenticated",
        /*ro, opt, enum, inductive event type, subType:string, desc:this node is used by storage devices; for access control devices, this node is invalid; "authenticated", "authenticationFailed", "openingDoor", "closingDoor", "doorException", "remoteOperation", "timeSynchronization", "deviceException", "deviceRecovered", "alarmTriggered", "alarmRecovered" (arming restoring event), "callCenter"*/
        "netUser": "test",
        /*ro, opt, string, user name for network operations*/
        "remoteHostAddr": "test",
        /*ro, opt, string, remote host address*/
        "cardNo": "test",
        /*ro, opt, string, card No.*/
        "cardType": 1,
        /*ro, opt, enum, card type, subType:int, desc:1 (normal card), 2 (disability card), 3 (blocklist card), 4 (patrol card), 5 (duress card), 6 (super card), 7 (visitor card), 8 (dismiss card)*/
        "name": "test",
        /*ro, opt, string, person name*/
        "sex": "male",
        /*ro, opt, enum, subType:string, desc:"male", "female"*/
        "whiteListNo": 1,
        /*ro, opt, int, allowlist No.*/
        "reportChannel": 1,
        /*ro, opt, enum, channel type for uploading alarms/events, subType:int, desc:1 (uploading in arming mode), 2 (uploading by central group 1), 3 (uploading by central group 2)*/
        "cardReaderKind": 1,
```

```
        /*ro, opt, enum, card reader type, subType:int, desc:1 (IC card reader), 2 (ID card reader), 3 (QR code scanner), 4 (fingerprint module)*/
        "cardReaderNo": 1,
        /*ro, opt, int, card reader No., step:1, desc:card reader No.*/
        "doorNo": 1,
        /*ro, opt, int, door (floor) No.*/
        "verifyNo": 1,
        /*ro, opt, int, multiple authentication No.*/
        "alarmInNo": 1,
        /*ro, opt, int, alarm input No.*/
        "alarmOutNo": 1,
        /*ro, opt, int, alarm output No.*/
        "caseSensorNo": 1,
        /*ro, opt, int, event trigger No.*/
        "RS485No": 1,
        /*ro, opt, int, RS-485 channel No.*/
        "multiCardGroupNo": 1,
        /*ro, opt, int, group No.*/
        "accessChannel": 1,
        /*ro, opt, int, turnstile No.*/
        "deviceNo": 1,
        /*ro, opt, int, device No.*/
        "distractControlNo": 1,
        /*ro, opt, int, distributed access controller No.*/
        "employeeNo": 1,
        /*ro, opt, int, employee No. (person ID)*/
        "employeeNoString": "test",
        /*ro, opt, string, employee No. (person ID), desc:if the node employeeNo exists or the value of employeeNoString can be converted to that of employeeNo, this node is required. For the upper-layer platform or client software, the node employeeNoString will be parsed in priority; if employeeNoString is not configured, the node employeeNo will be parsed*/
        "employeeName": "test",
        /*ro, opt, string, person name, desc:this node is only used for FocSign products*/
        "localControllerID": 1,
        /*ro, opt, int, distributed access controller No., desc:0 (access controller), 1 to 64 (distributed access controller No. 1 to distributed access controller No. 64)*/
        "InternetAccess": 1,
        /*ro, opt, enum, network interface No., subType:int, desc:1 (upstream network interface No. 1), 2 (upstream network interface No. 2), 3 (downstream network interface No. 1)*/
        "type": 1,
        /*ro, opt, enum, zone type, subType:int, desc:0 (instant zone), 1 (24-hour zone), 2 (delayed zone), 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 zone), 11 (emergency door closed zone), 255 (none)*/
        "MACAddr": "test",
        /*ro, opt, string, MAC address*/
        "swipeCardType": 1,
        /*ro, opt, enum, card swiping types, subType:int, desc:0 (invalid), 1 (QR code)*/
        "serialNo": 1,
        /*ro, opt, int, event serial No., range:[1,100000], desc:it starts at 1 and each record increases by 1. It will be overwritten repeatedly when reaching the maximum value supported by the device*/
        "channelControllerID": 1,
        /*ro, opt, enum, lane controller ID, subType:int, desc:1 (main lane controller), 2 (sub-lane controller)*/
        "channelControllerLampID": 1,
        /*ro, opt, int, light board ID of lane controller, range:[1,255]*/
        "channelControllerIRAdaptorID": 1,
        /*ro, opt, int, IR adaptor ID of the lane controller, range:[1,255]*/
        "channelControllerIREmitterID": 1,
        /*ro, opt, int, active infrared intrusion detector No. of the lane controller, range:[1,255]*/
        "userType": "normal",
        /*ro, opt, enum, person type, subType:string, desc:"normal" (normal person (resident)), "visitor" (visitor), "blacklist" (person in the blocklist), "administrators" (administrator)*/
        "currentVerifyMode": "cardAndPw",
        /*ro, opt, enum, current authentication mode of the card reader, subType:string, desc:"cardAndPw" (card+password), "card" (card), "cardOrPw" (card or password), "fp" (fingerprint), "fpAndPw" (fingerprint+password), "fpOrCard" (fingerprint or card), "fpAndCard" (fingerprint+card), "fpAndCardAndPw" (fingerprint+card+password), "faceOrFpOrCardOrPw" (face or fingerprint or card or password), "faceAndFp" (face+fingerprint), "faceAndPw" (face+password), "faceAndCard" (face+card), "face" (face), "employeeNoAndPw" (employee No.+password), "fpOrPw" (fingerprint or password), "employeeNoAndFp" (employee No.+fingerprint), "employeeNoAndFpAndPw" (employee No.+fingerprint+password), "faceAndFpAndCard" (face+fingerprint+card), "faceAndPwAndFp" (face+password+fingerprint), "employeeNoAndFace" (employee No.+face), "faceOrfaceAndCard" (face or face+card), "fpOrface" (fingerprint or face), "cardOrfaceOrPw" (card or face or password), "iris" (iris), "faceOrFpOrCardOrPwOrIris" (face, fingerprint, card, password, or iris), "faceOrCardOrPwOrIris" (face, card, password, or iris)*/
        "currentEvent": true,
        /*ro, opt, bool, whether it is a real-time event*/
        "QRCodeInfo": "test",
        /*ro, opt, string, QR code information*/
        "thermometryResult": "success",
        /*ro, opt, enum, temperature screening result, subType:string, desc:"success", "fail"*/
        "thermometryUnit": "celsius",
        /*ro, opt, enum, temperature unit, subType:string, desc:"celsius" (Celsius, default value), "fahrenheit" (Fahrenheit), "kelvin" (Kelvin)*/
        "currTemperature": 36.1,
        /*ro, opt, float, skin-surface temperature, which is accurate to one decimal place*/
        "isAbnomalTemperature": true,
        /*ro, opt, bool, whether the skin-surface temperature is abnormal*/
        "RegionCoordinates": {
        /*ro, opt, object, coordinates of the skin-surface temperature*/
            "positionX": 0,
            /*ro, opt, int, normalized X-coordinate which is between 0 and 1000, range:[0,1000]*/
            "positionY": 0
            /*ro, opt, int, normalized Y-coordinate which is between 0 and 1000, range:[0,1000]*/
        },
        "remoteCheck": true,
        /*ro, opt, bool, whether remote verification is required: true-yes, false-no (default)*/
        "mask": "unknown",
        /*ro, opt, enum, whether the person wears a mask, subType:string, desc:"unknown", "yes", "no"*/
        "frontSerialNo": 1,
        /*ro, opt, int, serial No. of the previous event, desc:if this node does not exist, the platform will check whether the event loss occurred according to the node serialNo. If both the serialNo and frontSerialNo are returned, the platform will check whether the event loss occurred according to
```

```
according to the node serialNo. If both the serialNo and frontSerialNo are returned, the platform will check whether the event logs occurred according to
both nodes. It is mainly used to solve the problem that the serialNo is inconsistent after subscribing events or alarms*/
        "attendanceStatus":  "checkIn",
        /*ro, opt, enum, attendance status, subType:string, desc:"checkIn" (check-in), "checkOut" (check-out), "breakOut" (start of break), "breakIn" (end
of break), "overtimeIn" (start of overtime), "overTimeOut" (end of overtime)*/
        "label":  "test",
        /*ro, opt, string, self-defined attendance name*/
        "statusValue":  1,
        /*ro, opt, int, status value*/
        "pictureURL":  "test",
        /*ro, opt, string, URL of the captured picture, range:[0,256]*/
        "visibleLightURL":  "test",
        /*ro, opt, string, visible light picture URL of the thermal imaging camera, range:[0,256]*/
        "thermalURL":  "test",
        /*ro, opt, string, URL of the thermal picture, range:[0,256]*/
        "faceBasemapURL":  "test",
        /*ro, opt, string, range:[0,256]*/
        "picturesNumber":  1,
        /*ro, opt, int, number of captured pictures*/
        "unlockType":  "password",
        /*ro, opt, enum, unlocking type, subType:string, desc:this node is returned when the minor type is MINOR_UNCLOCK_RECORD; "password" (unlock by
password), "hijcking" (unlock by duress), "card" (unlock by card), "householder" (unlock by householder), "centerplatform" (unlock by management center),
"bluetooth" (unlock by bluetooth), "qrcode" (unlocked via QR code), "face" (unlock by recognizing face), "fingerprint" (unlock by fingerprint)*/
        "classroomId":  "test",
        /*ro, opt, string, class ID*/
        "classroomName":  "test",
        /*ro, opt, string, class name*/
        "analysisModule":  "signageApp",
        /*ro, opt, enum, analysis module, subType:string, desc:this node is not returned, and the value is report via signage App; "signageApp" (signage
App), "faceSDK" (face picture SDK)*/
        "customInfo":  "test",
        /*ro, opt, string, custom information*/
        "helmet":  "unknown",
        /*ro, opt, enum, whether the person is wearing hard hat, subType:string, desc:"unknown", "yes", "no"*/
        "purePwdVerifyEnable":  true,
        /*ro, opt, bool, whether the device supports opening the door only by password,
        desc:opening the door only by password:
        the password in authentication method is person password; checking the repetition of person password is not supported by the device, it should be
performed by the upper-layer platform; adding, deleting, editing, and searching for person password locally is not supported by the device*/
        "appType":  "attendance",
        /*ro, opt, enum, application type (for FocSign products), subType:string, desc:"attendance" (Time & Attendance module), "signIn" (Check-In module)*/
        "HealthInfo": {
        /*ro, opt, object, health information*/
            "healthCode":  1,
            /*ro, opt, enum, health code status, subType:int, desc: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)*/
            "NADCode":  1,
            /*ro, opt, enum, nucleic acid test result, subType:int, desc:0 (no result), 1 (negative, which means normal), 2 (positive, which means
diagnosed), 3 (the result has expired)*/
            "NADMsg":  "test",
            /*ro, opt, string, range:[0,64]*/
            "NADTime":  1,
            /*ro, opt, enum, subType:int*/
            "travelCode":  1,
            /*ro, opt, enum, trip code, subType:int, desc:0 (no trip in the past 14 days), 1 (has left the current area left in the past 14 days), 2 (has
been to the high-risk area in the past 14 days), 3 (other)*/
            "travelInfo":  "test",
            /*ro, opt, string, trip information, desc:the empty string indicates that searching trip failed*/
            "vaccineStatus":  1,
            /*ro, opt, enum, whether the person is vaccinated, subType:int, desc:0 (not vaccinated), 1 (vaccinated)*/
            "vaccineNum":  1,
            /*ro, opt, int, step:1*/
            "vaccineMsg":  "test",
            /*ro, opt, string, range:[0,64]*/
            "ANTCode":  1,
            /*ro, opt, enum, subType:int*/
            "ANTMsg":  "test"
            /*ro, opt, string, range:[0,64]*/
        },
        "PhysicalInfo": {
        /*ro, opt, object, BMI information, desc:this node is obtained after authentication by BMI scales which is connected to MinMoe terminals*/
            "weight":  7000,
            /*ro, opt, int, weight, unit:kg*/
            "height":  18000
            /*ro, opt, int, height, unit:cm*/
        },
        "meetingID":  "test",
        /*ro, opt, string, meeting ID, range:[1,32]*/
        "PersonInfoExtends": [
        /*ro, opt, array, additional person information, subType:object, desc:this node displays additional person information on the device*/
            {
                "id":  1,
                /*ro, opt, int, extended ID of the additional person information, range:[1,32], desc:related URL: /ISAPI/AccessControl/personInfoExtendName?
format=json; this node is used for displaying the name of value; if ID does not exists, it starts from 1*/
                "value":  "test"
                /*ro, opt, string, extended content of the additional person information*/
            }
        ],
        "customPrompt":  "test",
        /*ro, opt, string, custom prompt message, range:[1,128], desc:this node is displayed when the authentication result is authenticated, authentication
failed, or stranger*/
        "FaceRect": {
        /*ro, opt, object, rectangle frame for human face, desc:the origin is the upper-left corner of the screen*/
            "height":  1.000,
```

```
            /*ro, req, float, height, range:[0.000,1.000]*/
            "width": 1.000,
            /*ro, req, float, width, range:[0.000,1.000]*/
            "x": 0.000,
            /*ro, req, float, X-coordinate of the upper-left corner of the frame, range:[0.000,1.000]*/
            "y": 0.000
            /*ro, req, float, Y-coordinate of the upper-left corner of the frame, range:[0.000,1.000]*/
        },
        "faceSimilarity": 90,
        /*ro, opt, int, Similarity, range:[0,100]*/
        "faceRecognitionDistance": 0.1,
        /*ro, opt, float, unit:m*/
        "eyesDistance": 20,
        /*ro, opt, int, range:[0,100], step:1*/
        "faceRecognitionFailedReason": "attackBlacklist",
        /*ro, opt, enum, subType:string*/
        "currentAuthenticationTimes": 1,
        /*ro, opt, int, range:[0,255], step:1*/
        "allowAuthenticationTimes": 1,
        /*ro, opt, int, range:[0,255], step:1*/
        "LocalAttendanceData": {
        /*ro, opt, object*/
            "attendanceResult": [
            /*ro, opt, array, subType:object*/
                {
                    "date": "1970-01-01",
                    /*ro, opt, date*/
                    "week": 1,
                    /*ro, opt, enum, subType:int, desc:1 (Monday), 2 (Tuesday), 3 (Wednesday), 4 (Thursday), 5 (Friday), 6 (Saturday), 7 (Sunday)*/
                    "personalAttendanceStatus": "normal"
                    /*ro, opt, enum, subType:string*/
                }
            ]
        },
        "hasRecord": true
        /*ro, opt, bool*/
    },
    "URLCertificationType": "digest"
    /*ro, opt, enum, picture URL authentication method, subType:string, desc:"no" (no authentication, it is used for the cloud protocol), "digest" (digest
authentication, it is used for local picture URL returned by NVR or DVR)*/
}
```

Parameter Name	Parameter Value	Parameter Type(Content-Type)	Content-ID	File Name	Description
AccessControllerEvent	[Message content]	application/json	--	--	--
Picture	[Binary picture data]	image/jpeg	pictureImage	Picture.jpg	--
VisibleLight	[Binary picture data]	image/jpeg	visibleLight_image	VisibleLight.jpg	--
Thermal	[Binary picture data]	image/jpeg	thermal_image	Thermal.jpg	--

Note: The protocol is transmitted in form format. See Chapter 4.5.1.4 for form framework description, as shown in the instance below.

```
--<frontier>
Content-Disposition: form-data; name=Parameter Name;filename=File Name
Content-Type: Parameter Type
Content-Length: ****
Content-ID: Content ID
Parameter Value
```

- Parameter Name: the name property of Content-Disposition in the header of form unit; it refers to the form unit name.
- Parameter Type (Content-Type): the Content-Type property in the header of form unit.
- File Name (filename): the filename property of Content-Disposition of form unit Headers. It exists only when the transmitted data of form unit is file, and it refers to the file name of form unit body.
- Parameter Value: the body content of form unit.

10.3.2 Access Control Schedule Management

10.3.2.1 Get the capability of clearing access control schedule configuration.

Request URL

GET /ISAPI/AccessControl/ClearPlansCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "ClearPlansCfg": {
    /*ro, req, object*/
        "ClearFlags": {
        /*ro, opt, object*/
            "doorStatusWeekPlan":  "true,false",
            /*ro, opt, string, whether to clear the week schedule of the door control*/
            "cardReaderWeekPlan":  "true,false",
            /*ro, opt, string, whether to clear the week schedule of the card reader authentication mode control*/
            "userRightWeekPlan":  "true,false",
            /*ro, opt, string, whether to clear the week schedule of the access permission control*/
            "doorStatusHolidayPlan":  "true,false",
            /*ro, opt, string, whether to clear the holiday schedule of the door control*/
            "cardReaderHolidayPlan":  "true,false",
            /*ro, opt, string, whether to clear the holiday schedule of the card reader authentication mode control*/
            "userRightHolidayPlan":  "true,false",
            /*ro, opt, string, whether to clear the holiday schedule of the access permission control*/
            "doorStatusHolidayGroup":  "true,false",
            /*ro, opt, string, whether to clear the holiday group of the door control*/
            "cardReaderHolidayGroup":  "true,false",
            /*ro, opt, string, whether to clear the holiday group of the card reader authentication mode control*/
            "userRightHolidayGroup":  "true,false",
            /*ro, opt, string, whether to clear the holiday group of the access permission control*/
            "doorStatusTemplate":  "true,false",
            /*ro, opt, string, whether to clear the schedule template of the door control*/
            "cardReaderTemplate":  "true,false",
            /*ro, opt, string, whether to clear the control schedule template of the card reader authentication mode*/
            "userRightTemplate":  "true,false"
            /*ro, opt, string, whether to clear the schedule template of the access permission control*/
        }
    }
}
```


10.3.2.2 Clear access control schedule configuration parameters

Request URL

PUT /ISAPI/AccessControl/ClearPlansCfg?format=json

Query Parameter

None

Request Message

```
{
    "ClearPlansCfg": {
    /*opt, object*/
        "ClearFlags": {
        /*opt, object*/
            "doorStatusWeekPlan":  true,
            /*opt, bool, whether to clear the week schedule of the door control*/
            "cardReaderWeekPlan":  true,
            /*opt, bool, whether to clear the week schedule of the card reader authentication mode control*/
            "userRightWeekPlan":  true,
            /*opt, bool, whether to clear the week schedule of the access permission control*/
            "doorStatusHolidayPlan":  true,
            /*opt, bool, whether to clear the holiday schedule of the door control*/
            "cardReaderHolidayPlan":  true,
            /*opt, bool, whether to clear the holiday schedule of the card reader authentication mode control*/
            "userRightHolidayPlan":  true,
            /*opt, bool, whether to clear the holiday schedule of the access permission control*/
            "doorStatusHolidayGroup":  true,
            /*opt, bool, whether to clear the holiday group of the door control*/
            "cardReaderHolidayGroup":  true,
            /*opt, bool, whether to clear the holiday group of the card reader authentication mode control*/
            "userRightHolidayGroup":  true,
            /*opt, bool, whether to clear the holiday group of the access permission control*/
            "doorStatusTemplate":  true,
            /*opt, bool, whether to clear the schedule template of the door control*/
            "cardReaderTemplate":  true,
            /*opt, bool, whether to clear the control schedule template of card reader authentication mode*/
            "userRightTemplate":  true
            /*opt, bool, whether to clear the schedule template of access permission control*/
        }
    }
}
```

Response Message

```
{
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded); it is required when an error occurred*/
    "statusString":  "ok",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "subStatusCode":  "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:when the value of statusCode is not 1, it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, opt, string, error information, desc:this node must be returned when the value of statusCode is not 1*/
}
```


10.3.2.3 Set holiday group parameters of door control schedule

Request URL

PUT /ISAPI/AccessControl/DoorStatusHolidayGroupCfg/<holidayGroupID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
holidayGroupID	string	--

Request Message

```
{
    "DoorStatusHolidayGroupCfg": {
    /*opt, object*/
        "enable":  true,
        /*req, bool, whether to enable, desc:whether to enable*/
        "groupName":  "test",
        /*req, string, holiday group name*/
        "holidayPlanNo":  "1,3,5"
        /*opt, string*/
    }
}
```

Response Message

```
{
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded); it is required when an error occurred*/
    "statusString":  "ok",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "subStatusCode":  "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:when the value of statusCode is not 1, it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, opt, string, error information, desc:this node must be returned when the value of statusCode is not 1*/
}
```

10.3.2.4 Get the holiday group configuration parameters of the door control schedule

Request URL

GET /ISAPI/AccessControl/DoorStatusHolidayGroupCfg/<holidayGroupID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
holidayGroupID	string	--

Request Message

None

Response Message

```
{
    "DoorStatusHolidayGroupCfg": {
    /*ro, opt, object*/
        "enable":  true,
        /*ro, req, bool, whether to enable: "true"-enable, "false"-disable*/
        "groupName":  "test",
        /*ro, req, string, holiday group name*/
        "holidayPlanNo":  "1,3,5"
        /*ro, req, string*/
    }
}
```

10.3.2.5 Get the configuration capability of door status parameters of holiday group

Request URL

GET /ISAPI/AccessControl/DoorStatusHolidayGroupCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "DoorStatusHolidayGroupCfg": {
    /*ro, opt, object*/
        "groupNo": {
        /*ro, opt, object, holiday group No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  16
            /*ro, opt, int, the maximum value*/
        },
        "enable":  "true,false",
        /*ro, opt, string, whether to enable, desc:true (enable)*/
        "groupName": {
        /*ro, opt, object, length of holiday group name*/
            "@min":  1,
            /*ro, opt, int, the minimum length*/
            "@max":  32
            /*ro, opt, int, the maximum length*/
        },
        "holidayPlanNo": {
        /*ro, opt, object, holiday group plan No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  16
            /*ro, opt, int, the maximum value*/
        }
    }
}
```

10.3.2.6 Get the configuration parameters of the door control holiday schedule

Request URL

GET /ISAPI/AccessControl/DoorStatusHolidayPlanCfg/<holidayPlanID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
holidayPlanID	string	--

Request Message

None

Response Message

```
{
    "DoorStatusHolidayPlanCfg": {
    /*ro, req, object*/
        "enable":  true,
        /*ro, req, bool, whether to enable, desc:true (enable), false (disable)*/
        "beginDate":  "2017-10-01",
        /*ro, req, date, start date of the holiday*/
        "endDate":  "2017-10-08",
        /*ro, req, date, end date of the holiday*/
        "HolidayPlanCfg": [
        /*ro, req, array, holiday schedule parameters, subType:object*/
            {
                "id":  1,
                /*ro, req, int, time period No., range:[1,8]*/
                "enable":  true,
                /*ro, req, bool, whether to enable, desc:true (enable), false (disable)*/
                "doorStatus":  "remainClosed",
                /*ro, req, enum, door status, subType:string, desc:“remainOpen”-remain open (access without authentication), “remainClosed”-remain closed
(access is not allowed), “normal”-access by authentication, "sleep", "invalid”, “induction”, “barrierFree”*/
                "TimeSegment": {
                /*ro, opt, object, time*/
                    "beginTime":  "00:00:00",
                    /*ro, req, time, start time of the time period, desc:device local time*/
                    "endTime":  "10:00:00"
                    /*ro, req, time, end time of the time period, desc:device local time*/
                }
            }
        ]
    }
}
```

10.3.2.7 Set parameters of door control holiday schedule

Request URL

PUT /ISAPI/AccessControl/DoorStatusHolidayPlanCfg/<holidayPlanID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
holidayPlanID	string	--

Request Message

```
{
    "DoorStatusHolidayPlanCfg": {
    /*ro, req, object*/
        "enable":  true,
        /*ro, req, bool, whether to enable, desc:"true"-enable, "false"-disable*/
        "beginDate":  "2017-10-01",
        /*ro, req, date, start date of the holiday*/
        "endDate":  "2017-10-08",
        /*ro, req, date, end data of the holiday*/
        "HolidayPlanCfg": [
        /*ro, req, array, holiday schedule parameters, subType:object*/
            {
                "id":  1,
                /*ro, req, int, time period No., range:[1,8]*/
                "enable":  true,
                /*ro, req, bool, whether to enable, desc:"true"-enable, "false"-disable*/
                "doorStatus":  "remainClosed",
                /*ro, req, enum, door status, subType:string, desc:"remainOpen"-remain open (access without authentication), "remainClosed"-remain closed
(access is not allowed), "normal"-access by authentication, "sleep", "invalid"*/
                "TimeSegment": {
                /*ro, opt, object, time*/
                    "beginTime":  "00:00:00",
                    /*ro, req, time, start time, desc:device local time*/
                    "endTime":  "10:00:00"
                    /*ro, req, time, end time, desc:device local time*/
                }
            }
        ]
    }
}
```

Response Message

```
{
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded); it is required when an error occurred*/
    "statusString":  "ok",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "subStatusCode":  "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:when the value of statusCode is not 1, it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, opt, string, error information, desc:this node must be returned when the value of statusCode is not 1*/
}
```


10.3.2.8 Get the configuration capability of the door control holiday schedule

Request URL

GET /ISAPI/AccessControl/DoorStatusHolidayPlanCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "DoorStatusHolidayPlanCfg": {
    /*ro, opt, object*/
        "planNo": {
        /*ro, opt, object, holiday schedule No.*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  16
            /*ro, opt, int*/
        },
        "enable":  "true,false",
        /*ro, opt, string, whether to enable: "true"-enable,"false"-disable, desc:"true" (enable), "false" (disable)*/
        "beginDate":  "1970-01-01",
        /*ro, opt, date, start date of the holiday*/
        "endDate":  "1971-01-01",
        /*ro, opt, date, end date of the holiday*/
        "HolidayPlanCfg": {
        /*ro, opt, object*/
            "maxSize":  8,
            /*ro, opt, int*/
            "id": {
            /*ro, opt, object, time period No.*/
                "@min":  1,
                /*ro, opt, int*/
                "@max":  8
                /*ro, opt, int*/
            },
            "enable":  "true,false",
            /*ro, opt, string, whether to enable: "true"-enable,"false"-disable, desc:"true" (enable), "false" (disable)*/
            "doorStatus": {
            /*ro, opt, object, door status*/
                "@opt":  "remainOpen,remainClosed,normal,sleep,invlid,induction,barrierFree"
                /*ro, opt, string, desc:"remainOpen" (remain open (access without authentication)), "remainClosed" (remain closed (access is not allowed)),
"normal" (access by authentication), "sleep", "invalid"*/
            },
            "TimeSegment": {
            /*ro, opt, object, time*/
                "beginTime":  "00:00:00",
                /*ro, opt, time, start time of the time period (device local time), desc:device local time*/
                "endTime":  "00:00:00",
                /*ro, opt, time, end time of the time period (device local time), desc:device local time*/
                "validUnit":  "minute"
                /*ro, opt, enum, time accuracy, subType:string, desc:"hour", "minute", "second"; if this node is not returned, the default time accuracy is
"minute"*/
            }
        }
    }
}
```

10.3.2.9 Get the configuration parameters of the door control schedule

Request URL

GET /ISAPI/AccessControl/DoorStatusPlan/<doorID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
doorID	string	--

Request Message

None

Response Message

```
{
    "DoorStatusPlan": {
    /*ro, req, object*/
        "templateNo":  1
        /*ro, req, int, schedule template No., desc:0-cancel linking the template with the schedule and restore to the default status (normal status)*/
    }
}
```

10.3.2.10 Set parameters of door control schedule

Request URL

PUT /ISAPI/AccessControl/DoorStatusPlan/<doorID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
doorID	string	--

Request Message

```
{
    "DoorStatusPlan": {
    /*req, object*/
        "templateNo":  1
        /*req, int, schedule template No., desc:0-cancel linking the template with the schedule and restore to the default status (normal status)*/
    }
}
```

Response Message

```
{
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded); it is required when an error occurred*/
    "statusString":  "ok",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "subStatusCode":  "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:when the value of statusCode is not 1, it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, opt, string, error information, desc:this node must be returned when the value of statusCode is not 1*/
}
```

10.3.2.11 Get the configuration capability of the door control schedule

Request URL

GET /ISAPI/AccessControl/DoorStatusPlan/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "DoorStatusPlan": {
    /*ro, opt, object*/
        "doorNo": {
        /*ro, opt, object, door No.*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  4
            /*ro, opt, int*/
        },
        "templateNo": {
        /*ro, opt, object, schedule template No.*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  16
            /*ro, opt, int*/
        }
    }
}
```

10.3.2.12 Set parameters of door control schedule template

Request URL

PUT /ISAPI/AccessControl/DoorStatusPlanTemplate/<planTemplateID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
planTemplateID	string	--

Request Message

```
{
    "DoorStatusPlanTemplate": {
    /*ro, opt, object, door control schedule template*/
        "enable": true,
        /*ro, req, bool, whether to enable, desc:"true"-enable, "false"-disable*/
        "templateName": "test",
        /*ro, req, string, template name*/
        "weekPlanNo": 1,
        /*ro, req, int, weekly schedule No.*/
        "holidayGroupNo": "1,3,5"
        /*ro, req, string, holiday group No., desc:holiday group No.*/
    }
}
```

Response Message

```
{
    "statusCode": 1,
    /*ro, opt, int, status code, desc:1 (succeeded); it is required when an error occurred*/
    "statusString": "ok",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "subStatusCode": "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "errorCode": 1,
    /*ro, opt, int, error code, desc:when the value of statusCode is not 1, it corresponds to subStatusCode*/
    "errorMsg": "ok"
    /*ro, opt, string, error information, desc:this node must be returned when the value of statusCode is not 1*/
}
```

10.3.2.13 Get parameters of door control schedule template

Request URL

GET /ISAPI/AccessControl/DoorStatusPlanTemplate/<planTemplateID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
planTemplateID	string	--

Request Message

None

Response Message

```
{
    "DoorStatusPlanTemplate": {
    /*ro, opt, object, door control schedule template*/
        "enable": true,
        /*ro, req, bool, whether to enable, desc:"true"-enable, "false"-disable*/
        "templateName": "test",
        /*ro, req, string, template name*/
        "weekPlanNo": 1,
        /*ro, req, int, weekly schedule No.*/
        "holidayGroupNo": "1,3,5"
        /*ro, req, string, holiday group No.*/
    }
}
```

10.3.2.14 Get the configuration capability of the door control schedule template

Request URL

GET /ISAPI/AccessControl/DoorStatusPlanTemplate/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "DoorStatusPlanTemplate": {
    /*ro, opt, object, schedule template*/
        "templateNo": {
        /*ro, opt, object, schedule template No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value of schedule template No.*/
            "@max":  16
            /*ro, opt, int, the maximum value of schedule template No.*/
        },
        "enable":  "true,false",
        /*ro, opt, string, whether to enable, desc:true (enable), false (disable)*/
        "templateName": {
        /*ro, opt, object, template name length*/
            "@min":  1,
            /*ro, opt, int, the minimum value of template name length*/
            "@max":  32
            /*ro, opt, int, the maximum value of template name length*/
        },
        "weekPlanNo": {
        /*ro, opt, object, weekly schedule No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value of weekly schedule No.*/
            "@max":  16
            /*ro, opt, int, the maximum value of weekly schedule No.*/
        },
        "holidayGroupNo": {
        /*ro, opt, object, holiday group No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value of holiday group No.*/
            "@max":  16
            /*ro, opt, int, the maximum value of holiday group No.*/
        }
    }
}
```

10.3.2.15 Set parameters of door control weekly schedule

Request URL

PUT /ISAPI/AccessControl/DoorStatusWeekPlanCfg/<weekPlanID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
weekPlanID	string	--

Request Message

```
{
    "DoorStatusWeekPlanCfg": {
    /*opt, object, weekly schedule of door control*/
        "enable":  true,
        /*req, bool, whether to enable, desc:"true"-enable, "false"-disable*/
        "WeekPlanCfg": [
        /*req, array, weekly schedule parameters, subType:object*/
            {
                "week":  "Monday",
                /*req, enum, days of the week, subType:string, desc:"Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"*/
                "id":  1,
                /*req, int, time period No., range:[1,8]*/
                "enable":  true,
                /*req, bool, whether to enable*/
                "doorStatus":  "remainClosed",
                /*req, enum, door control schedule, subType:string, desc:"remainOpen"-remain open (access without authentication), "remainClosed"-remain
closed (access is not allowed), "normal"-access by authentication, "sleep", "invalid"*/
                "TimeSegment": {
                /*req, object, time*/
                    "beginTime":  "00:00:00",
                    /*req, time, start time, desc:device local time*/
                    "endTime":  "10:00:00"
                    /*req, time, end time, desc:device local time*/
                }
            }
        ]
    }
}
```

Response Message

```
{
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded); it is required when an error occurred*/
    "statusString":  "ok",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "subStatusCode":  "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:when the value of statusCode is not 1, it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, opt, string, error information, desc:this node must be returned when the value of statusCode is not 1*/
}
```

10.3.2.16 Get the configuration parameters of the door control week schedule

Request URL

GET /ISAPI/AccessControl/DoorStatusWeekPlanCfg/<weekPlanID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
weekPlanID	string	--

Request Message

None

Response Message

```
{
    "DoorStatusWeekPlanCfg": {
    /*ro, opt, object, door control week schedule*/
        "enable":  true,
        /*ro, req, bool, whether to enable, desc:true (enable), false (disable)*/
        "WeekPlanCfg": [
        /*ro, req, array, week schedule parameters, subType:object*/
            {
                "week":  "Monday",
                /*ro, req, enum, days of the week, subType:string, desc:"Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"*/
                "id":  1,
                /*ro, req, int, time period No., range:[1,8]*/
                "enable":  true,
                /*ro, req, bool, whether to enable: "true"-enable, "false"-disable*/
                "doorStatus":  "remainClosed",
                /*ro, req, enum, door status, subType:string, desc:"remainOpen"-remain open (access without authentication), "remainClosed"-remain closed
(access is not allowed), "normal"-access by authentication, "sleep","invalid", "induction", "barrierFree"*/
                "TimeSegment": {
                /*ro, req, object, time*/
                    "beginTime":  "00:00:00",
                    /*ro, req, time, start time of the time period, desc:device local time*/
                    "endTime":  "10:00:00"
                    /*ro, req, time, end time of the time period, desc:device local time*/
                }
            }
        ]
    }
}
```

10.3.2.17 Get the configuration capability of the door control week schedule

Request URL

GET /ISAPI/AccessControl/DoorStatusWeekPlanCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "DoorStatusWeekPlanCfg": {
    /*ro, opt, object*/
        "planNo": {
        /*ro, opt, object, week schedule No.*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  16
            /*ro, opt, int*/
        },
        "enable":  "true,false",
        /*ro, opt, string, whether to enable: "true"-enable,"false"-disable, desc:"true" (enable), "false" (disable)*/
        "WeekPlanCfg": {
        /*ro, opt, object, week schedule parameters*/
            "maxSize":  56,
            /*ro, opt, int*/
            "week": {
            /*ro, opt, object*/
                "@opt":  "Monday,Tuesday,Wednesday,Thursday,Friday,Saturday,Sunday"
                /*ro, opt, string, desc:"Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"*/
            },
            "id": {
            /*ro, opt, object, weekly schedule No.*/
                "@min":  1,
                /*ro, opt, int*/
                "@max":  8
                /*ro, opt, int*/
            },
            "enable":  "true,false",
            /*ro, opt, string, whether to enable: "true"-enable,"false"-disable, desc:"true" (enable), "false" (disable)*/
            "doorStatus": {
            /*ro, opt, object, door status*/
                "@opt":  "remainOpen,remainClosed,normal,sleep,invalid,induction,barrierFree"
                /*ro, opt, string, desc:"remainOpen" (remain open (access without authentication)), "remainClosed" (remain closed (access is not allowed)),
"normal" (access by authentication), "sleep", "invalid"*/
            },
            "TimeSegment": {
            /*ro, opt, object, time*/
                "beginTime":  "00:00:00",
                /*ro, opt, time, start time of the time period (device local time), desc:device local time*/
                "endTime":  "10:00:00",
                /*ro, opt, time, end time of the time period (device local time), desc:device local time*/
                "validUnit":  "minute"
                /*ro, opt, enum, time accuracy, subType:string, desc:"hour", "minute", "second"; if this node is not returned, the default time accuracy is
"minute"*/
            }
        }
    }
}
```

10.3.3 Access Controller Management

10.3.3.1 Get the configuration capability of the access controller

Request URL

GET /ISAPI/AccessControl/AcsCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "AcsCfg": {
    /*ro, req, object*/
        "voicePrompt":  "true,false",
        /*ro, opt, string, whether to enable voice prompt, desc:"true" (yes), "false" (no)*/
    }
}
```

10.3.3.2 Set the parameters of the access controller

Request URL

PUT /ISAPI/AccessControl/AcsCfg?format=json

Query Parameter

None

Request Message

```
{
    "AcsCfg": {
    /*req, object, parameters of the access controller*/
        "voicePrompt": true,
        /*opt, bool, whether to enable voice prompt*/
    }
}
```

Response Message

```
{
    "statusCode": 1,
    /*ro, opt, int, status code, desc:status code*/
    "statusString": "ok",
    /*ro, opt, string, status description, range:[1,64], desc:status description*/
    "subStatusCode": "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:sub status code*/
    "errorCode": 1,
    /*ro, opt, int, error code, desc:error code*/
    "errorMsg": "ok"
    /*ro, opt, string, error description, desc:error description*/
}
```

10.3.3.3 Get the configuration parameters of the access controller

Request URL

GET /ISAPI/AccessControl/AcsCfg?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "AcsCfg": {
    /*ro, req, object*/
        "RS485Backup": true,
        /*ro, opt, bool, whether to enable downstream RS-485 communication redundancy*/
        "showCapPic": true,
        /*ro, opt, bool, whether to display the captured picture*/
        "showUserInfo": true,
        /*ro, opt, bool, whether to display user information*/
        "overlayUserInfo": true,
        /*ro, opt, bool, whether to overlay user information*/
        "voicePrompt": true,
        /*ro, opt, bool, whether to enable voice prompt*/
        "uploadCapPic": true,
        /*ro, opt, bool, whether to upload the picture from linked capture, desc:whether to upload the picture from linked capture*/
        "saveCapPic": true,
        /*ro, opt, bool, whether to save the captured picture, desc:whether to save the captured picture*/
        "inputCardNo": true,
        /*ro, opt, bool, whether to allow inputting card No. on keypad*/
        "enableWifiDetect": true,
        /*ro, opt, bool, whether to enable Wi-Fi probe*/
        "enable3G4G": true,
        /*ro, opt, bool, whether to enable 3G/4G*/
        "protocol": "Private",
        /*ro, opt, enum, communication protocol type of the card reader, subType:string, desc:"Private" (private protocol), "OSDP" (OSDP protocol)*/
        "enableCaptureCertificate": true,
        /*ro, opt, bool, whether to enable capturing the ID picture, desc:true (yes), false (no). If this node does not exist, it indicates that this
function is not supported*/
        "showPicture": true,
        /*ro, opt, bool, whether to display the authenticated picture, desc:whether to display the authenticated picture*/
        "showPresetPicture": true,
        /*ro, opt, bool*/
        "showEmployeeNo": true,
        /*ro, opt, bool, whether to display the authenticated employee ID*/
        "showName": true,
        /*ro, opt, bool, whether to display the authenticated name*/
        "showPhoneNo": true,
        /*ro, opt, bool*/
        "desensitiseEmployeeNo": true,
        /*ro, opt, bool, whether to enable employee No. de-identification for local UI display, dep:or,{$.AcsCfg.showEmployeeNo,eq,true}*/
        "desensitiseName": true,
```

```
        /*ro, opt, bool, whether to enable name de-identification for local UI display, dep:or,{$.AcsCfg.showName,eq,true}*/
        "desensitisePhoneNo": true,
        /*ro, opt, bool, dep:or,{$.AcsCfg.showPhoneNo,eq,true}*/
        "thermalEnabled": true,
        /*ro, opt, bool, whether to enable temperature measurement*/
        "thermalMode": true,
        /*ro, opt, bool, whether to enable temperature measurement only mode*/
        "thermalPictureEnabled": true,
        /*ro, opt, bool, whether to enable uploading visible light pictures in temperature measurement only mode: true-enable,false-disable (default). This field is used to control uploading captured pictures and visible light pictures*/
        "thermalIp": "192.168.1.1",
        /*ro, opt, string, IP address of the thermography device, desc:for access control devices, each device only requires one IP address; for metal detector doors, this field does not need to be configured*/
        "highestThermalThreshold": 37.3,
        /*ro, opt, float, upper limit of the temperature threshold*/
        "lowestThermalThreshold": 38.5,
        /*ro, opt, float, lower limit of the temperature threshold*/
        "thermalDoorEnabled": false,
        /*ro, opt, bool, whether to open the door according to the temperature threshold, desc:whether to open the door when the temperature is above the upper limit (highestThermalThreshold) or below the lower limit (lowestThermalThreshold) of the threshold: true (open the door), false (not open the door (default))*/
        "QRCodeEnabled": false,
        /*ro, opt, bool, whether to enable QR code function*/
        "remoteCheckDoorEnabled": false,
        /*ro, opt, bool, whether to enable controlling the door by remote verification, desc:whether to enable controlling the door by remote verification*/
        "checkChannelType": "Ezviz",
        /*ro, opt, enum, verification channel type, subType:string, dep:or,{$.AcsCfg.remoteCheckDoorEnabled,eq,true}, desc:verification channel type*/
        "channelIp": "test",
        /*ro, opt, string, IP address of the verification channel, dep:and,{$.AcsCfg.checkChannelType,eq,PrivateSDK}, desc:this field is valid when checkChannelType is "PrivateSDK"*/
        "needDeviceCheck": true,
        /*ro, opt, bool, dep:or,{$.AcsCfg.remoteCheckDoorEnabled,eq,true}*/
        "remoteCheckTimeout": 5,
        /*ro, opt, int, range:[1,10], unit:s, dep:or,{$.AcsCfg.remoteCheckDoorEnabled,eq,true}*/
        "remoteCheckVerifyMode": 1,
        /*ro, opt, enum, subType:int, dep:or,{$.AcsCfg.remoteCheckDoorEnabled,eq,true}*/
        "offlineDevCheckOpenDoorEnabled": false,
        /*ro, opt, bool, dep:or,{$.AcsCfg.remoteCheckDoorEnabled,eq,true}*/
        "remoteCheckWithISAPIListen": "async",
        /*ro, opt, enum, subType:string, dep:or,{$.AcsCfg.checkChannelType,eq,ISAPIListen}*/
        "uploadVerificationPic": true,
        /*ro, opt, bool, whether to upload the authenticated picture, desc:whether to upload the authenticated picture*/
        "uploadVerificationPicType": 0,
        /*ro, opt, enum, subType:int*/
        "saveVerificationPic": true,
        /*ro, opt, bool, whether to save the authenticated picture, desc:whether to save the authenticated picture*/
        "saveFacePic": true,
        /*ro, opt, bool, whether to save the registered face picture, desc:whether to save the registered face picture*/
        "thermalUnit": "celsius",
        /*ro, opt, enum, temperature unit, subType:string, desc:"celsius", "fahrenheit"*/
        "highestThermalThresholdF": 1.0,
        /*ro, opt, float, the maximum value of the temperature threshold, desc:the value is accurate to one decimal place, and the unit is Fahrenheit this node is used to check whether to open the door when the temperature is higher than the threshold*/
        "lowestThermalThresholdF": 1.0,
        /*ro, opt, float, the minimum value of the temperature threshold, desc:the value is accurate to one decimal place, and the unit is Fahrenheit this node is used to check whether to open the door when the temperature is higher than the threshold*/
        "enable5G": true,
        /*ro, opt, bool, whether to enable 5G*/
        "thermalCompensation": 1.0,
        /*ro, opt, float, temperature compensation, desc:float,temperature compensation,the value is accurate to one decimal place. The unit depends on the node thermalUnit. If the node thermalUnit does not exist,the default unit is Celsius*/
        "externalCardReaderEnabled": true,
        /*ro, opt, bool*/
        "combinationAuthenticationTimeout": 1,
        /*ro, opt, int, range:[1,20], step:1, unit:s*/
        "combinationAuthenticationLimitOrder": true,
        /*ro, opt, bool*/
        "passwordEnabled": true,
        /*ro, opt, bool*/
        "showGender": true,
        /*ro, opt, bool, whether to display gender*/
        "showSignInTime": true,
        /*ro, opt, bool*/
        "showsCustomInfo": true,
        /*ro, opt, bool*/
        "showMobileWebQRCode": true,
        /*ro, opt, bool*/
        "fireAlarmInputType": "alwaysOpen",
        /*ro, opt, enum, subType:string*/
        "buzzerEnabled": true,
        /*ro, opt, bool*/
        "saveVPAudioFile": false,
        /*ro, opt, bool*/
        "saveVPAudioFileByAuth": false,
        /*ro, opt, bool*/
        "faceDuplicateCheckEnabled": false,
        /*ro, opt, bool*/
        "localControllerBackupMode": 1,
        /*ro, opt, enum, subType:int*/
        "maxlocalControllerNum": 64,
        /*ro, opt, int*/
        "saveFpPicByCollectionMode": false,
        /*ro, opt, bool*/
```

```
        "faceBatchModelingMode":  "recognitionPriority",
        /*ro, opt, enum, subType:string*/
        "externalAuthResultDisplayEnabled":  false
        /*ro, opt, bool*/
    }
}
```

10.3.3.4 Get the capability of getting the working status of

the access controller **Request URL**

GET /ISAPI/AccessControl/AcsWorkStatus/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "AcsWorkStatus": {
    /*ro, req, object*/
        "doorLockStatus": {
        /*ro, opt, object, door lock status (relay status): 0-normally close, 1-normally open, 2-short-circuit alarm, 3-broken-circuit alarm, 4-exception
alarm*/
            "@opt":  "0,1,2,3,4"
            /*ro, opt, string*/
        },
        "doorStatus": {
        /*ro, opt, object, door (floor) status: 1-sleep, 2-remain unlocked (free), 3-remain locked (disabled), 4-normal status (controlled)*/
            "@opt":  "1,2,3,4"
            /*ro, opt, string*/
        },
        "magneticStatus": {
        /*ro, opt, object, magnetic contact status: 0-normally close,1-normally open, 2-short-circuit alarm, 3-broken-circuit alarm, 4-exception alarm*/
            "@opt":  "0,1,2,3,4"
            /*ro, opt, string, magnetic contact status*/
        },
        "powerSupplyStatus": {
        /*ro, opt, object, device power supply status: "ACPowerSupply"-alternative current, "BatteryPowerSupply"-storage battery power supply*/
            "@opt":  "ACPowerSupply,BatteryPowerSupply"
            /*ro, opt, string*/
        },
        "hostAntiDismantleStatus": {
        /*ro, opt, object, tampering status of the access control device: "close"-disabled, "open"-enabled*/
            "@opt":  "close,open"
            /*ro, opt, string*/
        },
        "cardReaderOnlineStatus": {
        /*ro, opt, object, online status of the authentication unit*/
            "@min":  1,
            /*ro, opt, int, the minimum value, range:[1,8]*/
            "@max":  1
            /*ro, opt, int, the maximum value, range:[1,8]*/
        },
        "cardReaderAntiDismantleStatus": {
        /*ro, opt, object, tampering status of the authentication unit*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  1
            /*ro, opt, int*/
        },
        "cardReaderVerifyMode": {
        /*ro, opt, object, current authentication mode of the authentication unit, desc:1-sleep,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*/
            "@opt":  "1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34"
            /*ro, opt, string*/
        },
        "alarmInStatus": {
        /*ro, opt, object, No. of input port with alarms*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  1
            /*ro, opt, int*/
        },
        "alarmOutStatus": {
        /*ro, opt, object, No. of output port with alarms*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  1
            /*ro, opt, int*/
        },
        "cardNum": {
        /*ro, opt, object, number of added cards*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  1
            /*ro, opt, int*/
        },
    }
}
```


10.3.3.5 Get the working status of the access controller

Request URL

GET /ISAPI/AccessControl/AcsWorkStatus?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "AcsWorkStatus": {
    /*ro, req, object*/
        "doorLockStatus":  [1, 2, 1, 2],
        /*ro, opt, enumarray, door lock status (relay status), subType:int, desc:door lock status (relay status): 0 (normally close), 1 (normally open), 2 (short-circuit alarm), 3 (broken-circuit alarm), 4 (exception alarm). For example, [1,2,1,2] indicates that door lock 1 is normally open, door lock 2 triggers short-circuit alarm, door lock 3 is normally open, and door lock 4 triggers short-circuit alarm*/
        "doorStatus":  [1, 2, 1, 2],
        /*ro, opt, enumarray, door (floor) status, subType:int, desc:door (floor) status: 1 (sleep), 2 (remain unlocked (free)), 3 (remain locked (disabled)), 4 (normal status (controlled)). For example, [1,2,1,2] indicates that door 1 is sleeping, door 2 remains unlocked, door 3 is sleeping, and door 4 remains unlocked*/
        "magneticStatus":  [1, 2, 1, 2],
        /*ro, opt, enumarray, magnetic contact status, subType:int, desc:magnetic contact status: 0 (normally close), 1 (normally open), 2 (short-circuit alarm), 3 (broken-circuit alarm), 4 (exception alarm). For example, [1,2,1,2] indicates that magnetic contact No.1 is normally open, magnetic contact No.2 triggers short-circuit alarm, magnetic contact No.3 is normally open, and magnetic contact No.4 triggers short-circuit alarm*/
        "caseStatus": [1, 3, 5],
        /*ro, opt, array, event trigger status, subType:int, desc:event trigger status, e.g., [1,3,5] indicates that event trigger No.1, No.3, and No.5 have input*/
        "batteryVoltage":  50,
        /*ro, opt, int, storage battery power voltage, desc:storage battery power voltage, the actual value will be 10 times of this value, unit: Volt*/
        "batteryLowVoltage":  false,
        /*ro, opt, bool, whether the storage battery is in low voltage status, desc:"true" (yes), "false" (no)*/
        "powerSupplyStatus":  "ACPowerSupply",
        /*ro, opt, enum, device power supply status, subType:string, desc:"ACPowerSupply" (alternative current), "BatteryPowerSupply" (storage battery power supply)*/
        "multiDoorInterlockStatus":  "close",
        /*ro, opt, enum, multi-door interlocking status, subType:string, desc:"close" (disabled), "open" (enabled)*/
        "antiSneakStatus":  "open",
        /*ro, opt, enum, anti-passback status, subType:string, desc:"close" (disabled), "open" (enabled)*/
        "hostAntiDismantleStatus":  "open",
        /*ro, opt, enum, tampering status of the access control device, subType:string, desc:"close" (disabled), "open" (enabled)*/
        "indicatorLightStatus":  "onLine",
        /*ro, opt, enum, indicator status, subType:string, desc:"offLine" (offline), "onLine" (online)*/
        "cardReaderOnlineStatus": [1, 3, 5],
        /*ro, opt, array, online status of the authentication unit, subType:int, desc:online status of the authentication unit, e.g., [1,3,5] indicates that authentication unit No.1, No.3, and No.5 are online*/
        "netReaderOnlineStatus": [1, 3, 5],
        /*ro, opt, array, subType:int*/
        "POEPortList": [
        /*ro, opt, array, subType:object, range:[1,8]*/
            {
                "port":  1,
                /*ro, req, int, range:[1,8]*/
                "readerID": [1, 2, 3]
                /*ro, opt, array, subType:int*/
            }
        ],
        "cardReaderAntiDismantleStatus": [1, 3, 5],
        /*ro, opt, array, tampering status of the authentication unit, subType:int, desc:tampering status of the authentication unit, e.g., [1,3,5] indicates that the tampering function of authentication unit No.1, No.3, and No.5 is enabled*/
        "cardReaderVerifyMode":  [3, 5, 3, 5],
        /*ro, opt, enumarray, current authentication mode of the authentication unit, subType:int, desc:1 (sleep), 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). For example, [3,5,3,5] indicates that the authentication mode of authentication unit 1 is "card", the authentication mode of authentication unit 2 is "fingerprint", the authentication mode of authentication unit 3 is "card", and the authentication mode of authentication unit 4 is "fingerprint"*/
        "setupAlarmStatus": [1, 3, 5],
        /*ro, opt, array, No. of armed input port, subType:int, desc:No. of armed input port, e.g., [1,3,5] indicates that input port No.1, No.3, and No.5 are armed*/
        "alarmInStatus": [1, 3, 5],
        /*ro, opt, array, No. of input port with alarms, subType:int, desc:No. of input port with alarms, e.g., [1,3,5] indicates that input port No.1, No.3, and No.5 trigger alarms*/
        "alarmOutStatus": [1, 3, 5],
        /*ro, opt, array, No. of output port with alarms, subType:int, desc:No. of output port with alarms, e.g., [1,3,5] indicates that output port No.1, No.3, and No.5 trigger alarms*/
        "cardNum":  3,
        /*ro, opt, int, number of added cards, desc:number of added cards*/
        "fireAlarmStatus":  "normal",
        /*ro, opt, enum, fire alarm status, subType:string, desc:fire alarm status: "normal", "shortCircuit" (short-circuit alarm), "brokenCircuit" (broken-circuit alarm)*/
        "batteryChargeStatus":  "charging",
        /*ro, opt, enum, battery charging status, subType:string, desc:battery charging status: "charging", "uncharged"*/
        "masterChannelControllerStatus":  "onLine",
        /*ro, opt, enum, online status of the main-lane controller, subType:string, desc:online status of the main lane controller: "offLine" (offline), "onLine" (online)*/
        "slaveChannelControllerStatus":  "onLine",
        /*ro, opt, enum, online status of the sub lane controller, subType:string, desc:online status of the sub lane controller: "offLine" (offline), "onLine" (online)*/
        "antiSneakServerStatus":  "normal",
        /*ro, opt, enum, anti-passback server status, subType:string, desc:anti-passback server status: "disable" (disabled), "normal", "disconnect" (disconnected)*/
        "netStatus":  "connect",
        /*ro, opt, enum, subType:string*/
        "InterfaceStatusList": [
        /*ro, opt, array, subType:object*/
```

```
            {
                "id": 1,
                /*ro, opt, int*/
                "netStatus": "connect"
                /*ro, opt, enum, subType:string*/
            }
        ],
        "signalStatus": "noSignal",
        /*ro, opt, enum, subType:string*/
        "sipStatus": "connect",
        /*ro, opt, enum, subType:string*/
        "ezvizStatus": "unregistered",
        /*ro, opt, enum, subType:string*/
        "voipStatus": "unregistered",
        /*ro, opt, enum, subType:string*/
        "wifiStatus": "connect",
        /*ro, opt, enum, subType:string*/
        "TFCardStatus": "mounted",
        /*ro, opt, enum, subType:string*/
        "acrossHostInterlockStatus": "disable",
        /*ro, opt, enum, subType:string*/
        "faceReceivingStatus": "receiving",
        /*ro, opt, enum, subType:string*/
        "QRCodeReaderStatusList": [
        /*ro, opt, array, subType:object*/
            {
                "id": 1,
                /*ro, opt, int*/
                "QRCodeReaderStatus": "onLine"
                /*ro, opt, enum, subType:string*/
            }
        ]
    }
}
```

10.3.4 Access Point Unit Management

10.3.4.1 Get the door configuration capability

Request URL

GET /ISAPI/AccessControl/Door/param/<doorID>/capabilities

Query Parameter

Parameter Name	Parameter Type	Description
doorID	string	--

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<DoorParam xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, attr:version{req, string, protocolVersion}-->
  <doorNo min="1" max="10">
    <!--ro, opt, int, door No., attr:min{req, int},max{req, int}-->1
  </doorNo>
  <doorName min="1" max="32">
    <!--ro, opt, string, door name, attr:min{req, int},max{req, int}-->test
  </doorName>
  <magneticType opt="alwaysClose,alwaysOpen">
    <!--ro, opt, enum, magnetic contact type, subType:string, attr:opt{req, string}, desc:"alwaysClose" (remain locked), "alwaysOpen" (remain unlocked)--
>alwaysClose
  </magneticType>
  <openButtonType opt="alwaysClose,alwaysOpen">
    <!--ro, opt, enum, door button type, subType:string, attr:opt{req, string}, desc:"alwaysClose" (remain locked), "alwaysOpen" (remain unlocked)--
>alwaysClose
  </openButtonType>
  <openDuration min="1" max="255">
    <!--ro, opt, int, door open duration (floor relay action time), attr:min{req, int},max{req, int}-->1
  </openDuration>
  <disabledOpenDuration min="1" max="255">
    <!--ro, opt, int, door open duration by disability card (delay duration of closing the door), attr:min{req, int},max{req, int}-->1
  </disabledOpenDuration>
  <magneticAlarmTimeout min="0" max="255">
    <!--ro, opt, int, alarm time of magnetic contact detection timeout, range:[0,255], unit:s, attr:min{req, int},max{req, int}, desc:alarm time of magnetic
contact detection timeout,which is between 0 and 255,0 refers to not triggering alarm,unit: second-->1
  </magneticAlarmTimeout>
  <enableDoorLock opt="true,false">
    <!--ro, opt, bool, whether to enable locking door when the door is closed, attr:opt{req, string}-->true
```

```
  </enableDoorLock>
  <enableLeaderCard opt="true,false">
    <!--ro, opt, bool, whether to enable remaining open with first card, attr:opt{req, string}-->true
  </enableLeaderCard>
  <leaderCardMode opt="disable,alwaysOpen,authorize">
    <!--ro, opt, enum, first card mode, subType:string, attr:opt{req, string}, desc:"disable", "alwaysOpen" (remain open with first card), "authorize"
(first card authentication)-->disable
  </leaderCardMode>
  <leaderCardOpenDuration min="1" max="1440">
    <!--ro, opt, int, duration of remaining open with first card, attr:min{req, int},max{req, int}-->1
  </leaderCardOpenDuration>
  <stressPassword min="1" max="8">
    <!--ro, opt, string, duress password, attr:min{req, int},max{req, int}, desc:the maximum length is 8 bytes, and the duress password should be encoded by
Base64 for transmission-->test
  </stressPassword>
  <superPassword min="1" max="8">
    <!--ro, opt, string, super password, attr:min{req, int},max{req, int}, desc:the maximum length is 8 bytes, and the duress password should be encoded by
Base64 for transmission-->test
  </superPassword>
  <unlockPassword min="1" max="8">
    <!--ro, opt, string, dismiss password,t, attr:min{req, int},max{req, int}, desc:the maximum length is 8 bytes, and the duress password should be encoded
by Base64 for transmission-->test
  </unlockPassword>
  <useLocalController opt="true,false">
    <!--ro, opt, bool, whether it is connected to the distributed controller, attr:opt{req, string}-->true
  </useLocalController>
  <localControllerID min="0" max="64">
    <!--ro, opt, int, distributed controller No., range:[0,64], attr:min{req, int},max{req, int}, desc:ro,distributed controller No.,which is between 1 and
64,0-unregistered-->1
  </localControllerID>
  <localControllerDoorNumber min="0" max="4">
    <!--ro, opt, int, distributed controller door No., range:[0,4], attr:min{req, int},max{req, int}, desc:ro,distributed controller door No.,which is
between 1 and 4,0-unregistered-->1
  </localControllerDoorNumber>
  <localControllerStatus opt="0,1,2,3,4,5,6,7,8,9">
    <!--ro, opt, enum, online status of the distributed controller, subType:int, attr:opt{req, string}, desc:0-offline, 1-network online, 2-RS-485 serial
port 1 on loop circuit 1, 3-RS-485 serial port 2 on loop circuit 1, 4-RS-485 serial port 1 on loop circuit 2, 5-RS-485 serial port 2 on loop circuit 2, 6-
RS-485 serial port 1 on loop circuit 3, 7-RS-485 serial port 2 on loop circuit 3,8-RS-485 serial port 1 on loop circuit 4,  9-RS-485 serial port 2 on loop
circuit 4-->1
  </localControllerStatus>
  <lockInputCheck opt="true,false">
    <!--ro, opt, bool, whether to enable door lock input detection, attr:opt{req, string}-->true
  </lockInputCheck>
  <lockInputType opt="alwaysClose,alwaysOpen">
    <!--ro, opt, enum, door lock input type, subType:string, attr:opt{req, string}, desc:"alwaysClose" (remain locked), "alwaysOpen" (remain unlocked)--
>alwaysClose
  </lockInputType>
  <doorTerminalMode opt="preventCutAndShort,preventCutAndShort,common">
    <!--ro, opt, enum, working mode of door terminal, subType:string, attr:opt{req, string}, desc:working mode of door terminal: "preventCutAndShort"-
prevent from broken-circuit and short-circuit (default), "common"-->preventCutAndShort
  </doorTerminalMode>
  <openButton opt="true,false">
    <!--ro, opt, bool, whether to enable door button, attr:opt{req, string}, desc:whether to enable door button: "true"-yes (default),"false"-no-->true
  </openButton>
  <ladderControlDelayTime min="1" max="255">
    <!--ro, opt, int, elevator control delay time (for visitor), attr:min{req, int},max{req, int}-->1
  </ladderControlDelayTime>
  <Leader>
    <!--ro, opt, object-->
    <continuousVerificationTimes min="1" max="10">
      <!--ro, opt, int, range:[1,10], attr:min{req, int},max{req, int}-->1
    </continuousVerificationTimes>
    <continuousVerificationDuration min="1" max="10">
      <!--ro, opt, int, range:[5,60], unit:s, attr:min{req, int},max{req, int}-->20
    </continuousVerificationDuration>
    <effectiveTimeEnabled opt="true,false">
      <!--ro, req, bool, attr:opt{req, string}-->true
    </effectiveTimeEnabled>
    <dayBeginTime>
      <!--ro, opt, time, dep:and,{$.DoorParam.Leader.effectiveTimeEnabled,eq,true}-->00:00:00
    </dayBeginTime>
    <beginEffectiveTime>
      <!--ro, opt, time, dep:and,{$.DoorParam.Leader.effectiveTimeEnabled,eq,true}-->00:00:00
    </beginEffectiveTime>
    <endEffectiveTime>
      <!--ro, opt, time, dep:and,{$.DoorParam.Leader.effectiveTimeEnabled,eq,true}-->00:00:00
    </endEffectiveTime>
  </Leader>
  <verificationPassOpenDoor opt="true,false">
    <!--ro, opt, bool, attr:opt{req, string}-->true
  </verificationPassOpenDoor>
  <relayReverseEnabled opt="true,false" def="false">
    <!--ro, opt, bool, attr:opt{req, string},def{req, bool}-->true
  </relayReverseEnabled>
</DoorParam>
```


10.3.4.2 Set the door (floor) parameters

Request URL

PUT /ISAPI/AccessControl/Door/param/<doorID>

Query Parameter

Parameter Name	Parameter Type	Description
doorID	string	--

Request Message

```
<?xml version="1.0" encoding="UTF-8"?>

<DoorParam xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--req, object, door parameter, attr:version{req, string, protocolVersion}-->
  <doorName>
    <!--opt, string, door name-->test
  </doorName>
  <magneticType>
    <!--opt, enum, door magnetic sensor type, subType:string, desc:"alwaysClose” (remain locked), "alwaysOpen” (remain unlocked)-->alwaysClose
  </magneticType>
  <openButtonType>
    <!--opt, enum, open door button type, subType:string, desc:"alwaysClose” (remain locked), "alwaysOpen” (remain unlocked)-->alwaysClose
  </openButtonType>
  <openDuration>
    <!--opt, int, door open duration, range:[1,255], unit:s-->1
  </openDuration>
  <disabledOpenDuration>
    <!--opt, int, door open duration by disability card (delay duration of closing the door), range:[1,255], unit:s-->1
  </disabledOpenDuration>
  <magneticAlarmTimeout>
    <!--opt, int, alarm time of magnetic contact detection timeout, range:[0,255], unit:s, desc:0 refers to not triggering alarm-->1
  </magneticAlarmTimeout>
  <enableDoorLock>
    <!--opt, bool, lock door when door closed-->true
  </enableDoorLock>
  <enableLeaderCard>
    <!--opt, bool, whether to enable remaining open with first card-->true
  </enableLeaderCard>
  <leaderCardMode>
    <!--opt, enum, first card mode, subType:string, desc:first card mode: "disable","alwaysOpen"-remain open with first card,"authorize"-first card
authentication. If this node is configured,the node <enableLeaderCard> is invalid-->disable
  </leaderCardMode>
  <leaderCardOpenDuration>
    <!--opt, int, duration of remaining open with first card, range:[0,1440], unit:min, dep:and,{$.DoorParam.leaderCardMode,eq,alwaysOpen}-->1
  </leaderCardOpenDuration>
  <stressPassword>
    <!--opt, string, duress password, desc:the maximum length is 8 bytes, and the duress password should be encoded by Base64 for transmission-->test
  </stressPassword>
  <superPassword>
    <!--opt, string, super password, desc:the maximum length is 8 bytes, and the duress password should be encoded by Base64 for transmission-->test
  </superPassword>
  <unlockPassword>
    <!--opt, string, unlock password, desc:the maximum length is 8 bytes, and the duress password should be encoded by Base64 for transmission-->test
  </unlockPassword>
  <useLocalController>
    <!--opt, bool, whether it is connected to the distributed controller-->true
  </useLocalController>
  <localControllerID>
    <!--opt, int, distributed controller No., range:[0,64], desc:ro,distributed controller No.,which is between 1 and 64,0-unregistered-->1
  </localControllerID>
  <localControllerDoorNumber>
    <!--opt, int, distributed controller door No., range:[0,4], desc:ro,distributed controller door No.,which is between 1 and 4,0-unregistered-->1
  </localControllerDoorNumber>
  <localControllerStatus>
    <!--opt, enum, online status of the distributed controller, subType:int, desc:ro,online status of the distributed controller: 0-offline,1-network
online,2-RS-485 serial port 1 on loop circuit 1,3-RS-485 serial port 2 on loop circuit 1,4-RS-485 serial port 1 on loop circuit 2,5-RS-485 serial port 2 on
loop circuit 2,6-RS-485 serial port 1 on loop circuit 3,7-RS-485 serial port 2 on loop circuit 3,8-RS-485 serial port 1 on loop circuit 4,9-RS-485 serial
port 2 on loop circuit 4-->1
  </localControllerStatus>
  <lockInputCheck>
    <!--opt, bool, whether to enable door lock input detection-->true
  </lockInputCheck>
  <lockInputType>
    <!--opt, enum, door lock input type, subType:string, desc:"alwaysClose” (remain locked), "alwaysOpen” (remain unlocked), lt remains locked by default--
>alwaysClose
  </lockInputType>
  <doorTerminalMode>
    <!--opt, enum, working mode of door terminal, subType:string, desc:"preventCutAndShort” (prevent from broken-circuit and short-circuit
(default)),"common"-->preventCutAndShort
  </doorTerminalMode>
  <openButton>
    <!--opt, bool, whether to enable door button-->true
  </openButton>
  <ladderControlDelayTime>
    <!--opt, int, elevator control delay time (for visitor), range:[1,255], unit:min-->1
  </ladderControlDelayTime>
  <Leader>
    <!--opt, object-->
    <continuousVerificationTimes>
      <!--opt, int, range:[1,10]-->1
```

```
      <!--opt, int, range:[1,10]-->1
    </continuousVerificationTimes>
    <continuousVerificationDuration>
      <!--opt, int, range:[5,60], unit:s-->20
    </continuousVerificationDuration>
    <effectiveTimeEnabled>
      <!--req, bool-->true
    </effectiveTimeEnabled>
    <dayBeginTime>
      <!--opt, time, dep:and,{$.DoorParam.Leader.effectiveTimeEnabled,eq,true}-->00:00:00
    </dayBeginTime>
    <beginEffectiveTime>
      <!--opt, time, dep:and,{$.DoorParam.Leader.effectiveTimeEnabled,eq,true}-->00:00:00
    </beginEffectiveTime>
    <endEffectiveTime>
      <!--opt, time, dep:and,{$.DoorParam.Leader.effectiveTimeEnabled,eq,true}-->00:00:00
    </endEffectiveTime>
  </Leader>
  <verificationPassOpenDoor>
    <!--opt, bool-->true
  </verificationPassOpenDoor>
  <relayReverseEnabled>
    <!--opt, bool-->true
  </relayReverseEnabled>
</DoorParam>
```

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response message, attr:version{ro, req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL-->null
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:int, desc:0 (OK), 1 (OK), 2 (Device Busy), 3 (Device Error), 4 (Invalid Operation), 5 (Invalid XML Format), 6
(Invalid XML Content), 7 (Reboot Required)-->0
  </statusCode>
  <statusString>
    <!--ro, req, enum, status description, subType:string, desc:“OK” (succeeded), “Device Busy”, “Device Error”, “Invalid Operation”, “Invalid XML Format”,
“Invalid XML Content”, “Reboot” (reboot device)-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, error code description, desc:error code description-->OK
  </subStatusCode>
</ResponseStatus>
```


10.3.4.3 Get the door (floor) configuration parameters

Request URL

GET /ISAPI/AccessControl/Door/param/<doorID>

Query Parameter

Parameter Name	Parameter Type	Description
doorID	string	--

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<DoorParam xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, attr:version{req, string, protocolVersion}-->
  <doorName>
    <!--ro, opt, string, door name-->test
  </doorName>
  <magneticType>
    <!--ro, opt, enum, magnetic contact type, subType:string, desc:"alwaysClose"-remain locked, "alwaysOpen"-remain unlocked-->alwaysClose
  </magneticType>
  <openButtonType>
    <!--ro, opt, enum, door button type, subType:string, desc:"alwaysClose"-remain locked, "alwaysOpen"-remain unlocked-->alwaysClose
  </openButtonType>
  <openDuration>
    <!--ro, opt, int, door open duration, range:[1,255], unit:s-->1
  </openDuration>
  <disabledOpenDuration>
    <!--ro, opt, int, door open duration by disability card (delay duration of closing the door), range:[1,255], unit:s-->1
```

```
  </disabledOpenDuration>
  <magneticAlarmTimeout>
    <!--ro, opt, int, alarm time of magnetic contact detection timeout, range:[0,255], unit:s, desc:0 refers to not triggering alarm-->1
  </magneticAlarmTimeout>
  <enableDoorLock>
    <!--ro, opt, bool, whether to enable locking door when the door is closed-->true
  </enableDoorLock>
  <enableLeaderCard>
    <!--ro, opt, bool, whether to enable remaining open with first card. This node is invalid when leaderCardMode is configured-->true
  </enableLeaderCard>
  <leaderCardMode>
    <!--ro, opt, enum, first card mode, subType:string, desc:"disable","alwaysOpen"-remain open with first card,"authorize"-first card authentication. If this node is configured,the node <enableLeaderCard> is invalid-->disable
  </leaderCardMode>
  <leaderCardOpenDuration>
    <!--ro, opt, int, duration of remaining open with first card, range:[0,1440], unit:min, dep:and,{$.DoorParam.LeaderCardMode,eq,alwaysOpen}-->1
  </leaderCardOpenDuration>
  <stressPassword>
    <!--ro, opt, string, duress password, desc:the maximum length is 8 bytes, and the duress password should be encoded by Base64 for transmission-->test
  </stressPassword>
  <superPassword>
    <!--ro, opt, string, super password, desc:wo,super password,the maximum length is 8 bytes,and the super password should be encoded by Base64 for transmission-->test
  </superPassword>
  <unlockPassword>
    <!--ro, opt, string, dismiss password, desc:the maximum length is 8 bytes, and the dismiss password should be encoded by Base64 for transmission-->test
  </unlockPassword>
  <useLocalController>
    <!--ro, opt, bool, whether it is connected to the distributed controller-->true
  </useLocalController>
  <localControllerID>
    <!--ro, opt, int, distributed controller No., which is between 1 and 64, range:[0,64], desc:0-unregistered-->1
  </localControllerID>
  <localControllerDoorNumber>
    <!--ro, opt, int, distributed controller door No., range:[0,4], desc:0-unregistered-->1
  </localControllerDoorNumber>
  <localControllerStatus>
    <!--ro, opt, enum, online status of the distributed controller, subType:int, desc:0-offline,1-network online,2-RS-485 serial port 1 on loop circuit 1,3-RS-485 serial port 2 on loop circuit 1,4-RS-485 serial port 1 on loop circuit 2,5-RS-485 serial port 2 on loop circuit 2,6-RS-485 serial port 1 on loop circuit 3,7-RS-485 serial port 2 on loop circuit 3,8-RS-485 serial port 1 on loop circuit 4,9-RS-485 serial port 2 on loop circuit 4-->1
  </localControllerStatus>
  <lockInputCheck>
    <!--ro, opt, bool, whether to enable door lock input detection-->true
  </lockInputCheck>
  <lockInputType>
    <!--ro, opt, enum, door lock input type, subType:string, desc:"alwaysClose"-remain locked (default), "alwaysOpen"-remain unlocked-->alwaysClose
  </lockInputType>
  <doorTerminalMode>
    <!--ro, opt, enum, working mode of door terminal, subType:string, desc:"preventCutAndShort"-prevent from broken-circuit and short-circuit (default),"common"-->preventCutAndShort
  </doorTerminalMode>
  <openButton>
    <!--ro, opt, bool, whether to enable door button: "true"-yes (default), "false"-no-->true
  </openButton>
  <ladderControlDelayTime>
    <!--ro, opt, int, elevator control delay time (for visitor), range:[1,255], unit:min-->1
  </ladderControlDelayTime>
  <Leader>
    <!--ro, opt, object-->
    <continuousVerificationTimes>
      <!--ro, opt, int, range:[1,10]-->1
    </continuousVerificationTimes>
    <continuousVerificationDuration>
      <!--ro, opt, int, range:[5,60], unit:s-->20
    </continuousVerificationDuration>
    <effectiveTimeEnabled>
      <!--ro, req, bool-->true
    </effectiveTimeEnabled>
    <dayBeginTime>
      <!--ro, opt, time, dep:and,{$.DoorParam.Leader.effectiveTimeEnabled,eq,true}-->00:00:00
    </dayBeginTime>
    <beginEffectiveTime>
      <!--ro, opt, time, dep:and,{$.DoorParam.Leader.effectiveTimeEnabled,eq,true}-->00:00:00
    </beginEffectiveTime>
    <endEffectiveTime>
      <!--ro, opt, time, dep:and,{$.DoorParam.Leader.effectiveTimeEnabled,eq,true}-->00:00:00
    </endEffectiveTime>
  </Leader>
  <verificationPassOpenDoor>
    <!--ro, opt, bool-->true
  </verificationPassOpenDoor>
  <relayReverseEnabled>
    <!--ro, opt, bool-->true
  </relayReverseEnabled>
</DoorParam>
```


10.3.4.4 Get the status of the secure door control unit

Request URL

GET /ISAPI/AccessControl/DoorSecurityModule/moduleStatus

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ModuleStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, opt, object, module status, attr:version{req, string, protocolVersion}-->
  <securityModuleNo min="1" max="256">
    <!--ro, req, string, secure door control unit No., attr:min{req, int},max{req, int}-->test
  </securityModuleNo>
  <onlineStatus opt="0,1">
    <!--ro, req, enum, online status, subType:int, attr:opt{req, string}, desc:0 (offline), 1(online)-->1
  </onlineStatus>
  <desmantelStatus opt="0,1">
    <!--ro, req, enum, tamper-proof status, subType:int, attr:opt{req, string}, desc:0 (the unit is not tampered), 1(the unit is tampered)-->1
  </desmantelStatus>
</ModuleStatus>
```

10.3.4.5 Get the capability of getting the status of the secure door control unit

Request URL

GET /ISAPI/AccessControl/DoorSecurityModule/moduleStatus/capabilities

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ModuleStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, opt, object, module status, attr:version{req, string, protocolVersion}-->
  <securityModuleNo min="1" max="256">
    <!--ro, req, string, secure door control unit No., attr:min{req, int},max{req, int}-->test
  </securityModuleNo>
  <onlineStatus opt="0,1">
    <!--ro, req, enum, online status, subType:int, attr:opt{req, string}, desc:0 (offline), 1 (online)-->1
  </onlineStatus>
  <desmantelStatus opt="0,1">
    <!--ro, req, enum, tampering status, subType:int, attr:opt{req, string}, desc:0 (the unit is not tampered), 1 (the unit is tampered)-->1
  </desmantelStatus>
</ModuleStatus>
```

10.3.5 Anti-Passback

10.3.5.1 Get the anti-passing back configuration capability

Request URL

GET /ISAPI/AccessControl/AntiSneakCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "AntiSneakCfg": {
    /*ro, req, object*/
        "enable":  "true,false",
        /*ro, req, string, whether to enable anti-passing back*/
        "startCardReaderNo": {
        /*ro, opt, object, first card reader No., desc:first card reader No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  4,
            /*ro, opt, int, the maximum value*/
            "@opt": [1, 4]
            /*ro, opt, array, subType:int*/
        }
    }
}
```

10.3.5.2 Get the parameters of anti-passback configuration

Request URL

GET /ISAPI/AccessControl/AntiSneakCfg?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "AntiSneakCfg": {
    /*ro, req, object*/
        "enable":  true,
        /*ro, req, bool, whether to enable anti-passback*/
        "startCardReaderNo":  1
        /*ro, opt, int, first card reader No., desc:first card reader No.,0-no first card reader*/
    }
}
```

10.3.5.3 Set the anti-passing back parameters

Request URL

PUT /ISAPI/AccessControl/AntiSneakCfg?format=json

Query Parameter

None

Request Message

```
{
    "AntiSneakCfg": {
    /*req, object*/
        "enable":  true,
        /*req, bool, whether to enable anti-passing back*/
        "startCardReaderNo":  1
        /*opt, int, first card reader No., desc:0-no first card reader*/
    }
}
```

Response Message

```
{
    "requestURL":  "test",
    /*ro, opt, string, URI*/
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded). It is required when an error occurred*/
    "statusString":  "ok",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded). It is required when an error occurred*/
    "subStatusCode":  "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded). It is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:when the value of statusCode is not 1, it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, opt, string, error information, desc:this node is required when the value of statusCode is not 1*/
}
```

10.3.5.4 Get the anti-passing back configuration parameters of a specified card reader

Request URL

GET /ISAPI/AccessControl/CardReaderAntiSneakCfg/<cardReaderID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
cardReaderID	string	--

Request Message

None

Response Message

```
{
    "CardReaderAntiSneakCfg": {
    /*ro, req, object*/
        "enable":  true,
        /*ro, req, bool, whether to enable the anti-passing back function of the card reader, desc:"true" (enable), "false" (disable)*/
        "followUpCardReader": [2, 3, 4]
        /*ro, opt, array, following card reader No. after the first card reader, subType:int, desc:[2,3,4] indicates that card reader No. 2, No. 3, or No. 4
can be swiped after the first card reader*/
    }
}
```

10.3.5.5 Set anti-passing back parameters of a card reader

Request URL

PUT /ISAPI/AccessControl/CardReaderAntiSneakCfg/<cardReaderID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
cardReaderID	string	--

Request Message

```
{
    "CardReaderAntiSneakCfg": {
    /*req, object, anti-passing back parameters of a card reader*/
        "enable":  true,
        /*req, bool, whether to enable the anti-passing back function of the card reader, desc:"true"-enable, "false"-disable*/
        "followUpCardReader": [2, 3, 4]
        /*opt, array, following card reader No. after the first card reader, subType:int, desc:e.g., [2,3,4] indicates that card reader No. 2, No. 3, and
No. 4 can be swiped after the first card reader*/
    }
}
```

Response Message

```
{
    "requestURL":  "test",
    /*ro, opt, string, URI*/
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded); it is required when an error occurred*/
    "statusString":  "ok",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "subStatusCode":  "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:when the value of statusCode is not 1, it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, opt, string, error information, desc:this node is required when the value of statusCode is not 1*/
}
```

10.3.5.6 Get the configuration capability of anti-passing back parameters of card readers

Request URL

GET /ISAPI/AccessControl/CardReaderAntiSneakCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "CardReaderAntiSneakCfg": {
    /*ro, req, object*/
        "cardReaderNo": {
        /*ro, opt, object, card reader No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  512,
            /*ro, opt, int, the maximum value*/
            "@opt": [1, 4]
            /*ro, opt, array, subType:int*/
        },
        "enable":  "true,false",
        /*ro, req, string, whether to enable the anti-passing back function of the card reader*/
        "followUpCardReader": {
        /*ro, opt, object, array,following card reader No. after the first card reader*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  512,
            /*ro, opt, int*/
            "@opt": [1, 4]
            /*ro, opt, array, subType:int*/
        }
    }
}
```

10.3.5.7 Get the capability of clearing anti-passback records

Request URL

GET /ISAPI/AccessControl/ClearAntiSneak/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "ClearAntiSneak": {
    /*ro, req, object*/
        "EmployeeNoList": {
        /*ro, opt, object*/
            "maxSize":  32,
            /*ro, opt, int*/
            "employeeNo": {
            /*ro, opt, object, employee No. (person ID)*/
                "@min":  1,
                /*ro, opt, int*/
                "@max":  32
                /*ro, opt, int*/
            }
        },
        "clearMode": {
        /*ro, opt, object*/
            "@opt": ["employeeNo", "all"]
            /*ro, opt, array, subType:string*/
        }
    }
}
```

10.3.5.8 Clear anti-passback records in the device

Request URL

PUT /ISAPI/AccessControl/ClearAntiSneak?format=json

Query Parameter

None

Request Message

```
{
    "ClearAntiSneak": {
    /*req, object, clear anti-passback records*/
        "EmployeeNoList": [
        /*opt, array, person ID list, subType:object, dep:and,{$.ClearAntiSneak.clearMode,eq,employeeNo}*/
            {
                "employeeNo":  "test"
                /*opt, string, employee No. (person ID)*/
            }
        ],
        "clearMode":  "employeeNo"
        /*opt, enum, clearing mode, subType:string*/
    }
}
```

Response Message

```
{
    "requestURL":  "test",
    /*ro, opt, string, URI*/
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded). It is required when an error occurred*/
    "statusString":  "ok",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded). It is required when an error occurred*/
    "subStatusCode":  "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded). It is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:it is required when the value of statusCode is not 1, and it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, opt, string, error information, desc:this node is required when the value of statusCode is not 1*/
}
```

10.3.5.9 Get the capability of clearing anti-passback parameters

Request URL

GET /ISAPI/AccessControl/ClearAntiSneakCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "ClearAntiSneakCfg": {
    /*ro, req, object, the capability of clearing anti-passback*/
        "ClearFlags": {
        /*ro, req, object*/
            "antiSneak":  "true,false"
            /*ro, req, string, whether to clear the anti-passback parameter*/
        }
    }
}
```

10.3.5.10 Clear anti-passing back parameters

Request URL

PUT /ISAPI/AccessControl/ClearAntiSneakCfg?format=json

Query Parameter

None

Request Message

```
{
    "ClearAntiSneakCfg": {
    /*req, object*/
        "ClearFlags": {
        /*req, object*/
            "antiSneak":  true
            /*req, bool, whether to clear the anti-passing back parameters*/
        }
    }
}
```

Response Message

```
{
    "requestURL":  "test",
    /*ro, opt, string, URI*/
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded); it is required when an error occurred*/
    "statusString":  "ok",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "subStatusCode":  "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:when the value of statusCode is not 1, it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, opt, string, error information, desc:this node is required when the value of statusCode is not 1*/
}
```

10.3.6 Authentication Schedule Management

10.3.6.1 Set control schedule parameters of card reader authentication mode

Request URL

PUT /ISAPI/AccessControl/CardReaderPlan/<cardReaderID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
cardReaderID	string	--

Request Message

```
{
    "CardReaderPlan": {
    /*req, object*/
        "templateNo":  1
        /*req, int, schedule template No., desc:0-cancel linking the template to the schedule and restore to the default status (normal status)*/
    }
}
```

Response Message

```
{
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded); it is required when an error occurred*/
    "statusString":  "ok",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "subStatusCode":  "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:when the value of statusCode is not 1, it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, opt, string, error information, desc:this node must be returned when the value of statusCode is not 1*/
}
```

10.3.6.2 Get the control schedule configuration parameters of the card reader authentication mode

Request URL

GET /ISAPI/AccessControl/CardReaderPlan/<cardReaderID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
cardReaderID	string	--

Request Message

None

Response Message

```
{
    "CardReaderPlan": {
    /*ro, req, object, schedule template structure*/
        "templateNo":  1
        /*ro, req, int, schedule template number, desc:0-cancel linking the template to the schedule and restore to the default status (normal status)*/
    }
}
```

10.3.6.3 Get the control schedule configuration capability of the card reader authentication mode

Request URL

GET /ISAPI/AccessControl/CardReaderPlan/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "CardReaderPlan": {
    /*ro, opt, object, card reader No. node*/
        "cardReaderNo": {
        /*ro, opt, object, card reader No.*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  4
            /*ro, opt, int*/
        },
        "templateNo": {
        /*ro, opt, object, schedule template No. node*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  16,
            /*ro, opt, int*/
            "@opt": [65535, 65534, 65533]
            /*ro, opt, array, subType:int*/
        },
        "verifyMode": {
        /*ro, opt, object, authentication mode, dep:and,{$.CardReaderPlan.templateNo.@opt[*],eq,65535}*/
            "@opt":
"cardAndPw,card,cardOrPw,fp,fpAndPw,fpOrCard,fpAndCard,fpAndCardAndPw,faceOrFpOrCardOrPw,faceAndFp,faceAndPw,faceAndCard,face,employeeNoAndPw,fpOrPw,employe
eNoAndFp,employeeNoAndFpAndPw,faceAndFpAndCard,faceAndPwAndFp,employeeNoAndFace,faceOrfaceAndCard,fpOrface,cardOrfaceOrPw,cardOrFace,cardOrFaceOrFp,cardOrFp
OrPw,faceOrPw,employeeNoAndFaceAndPw,cardOrFaceOrFaceAndCard,iris,faceOrFpOrCardOrPwOrIris,faceOrCardOrPwOrIris,sleep,invalid"
            /*ro, opt, string, desc:"cardAndPw" (card + password), "card" (card), "cardOrPw" (card or password), "fp" (fingerprint), "fpAndPw" (fingerprint
+ password), "fpOrCard" (fingerprint or card), "fpAndCard" (fingerprint + card), "fpAndCardAndPw" (fingerprint + card + password), "faceOrFpOrCardOrPw"
(face or fingerprint or card or password), "faceAndFp" (face + fingerprint), "faceAndPw" (face + password), "faceAndCard" (face + card), "face" (face),
"employeeNoAndPw" (employee No. + password), "fpOrPw" (fingerprint or password), "employeeNoAndFp" (employee No. + fingerprint), "employeeNoAndFpAndPw"
(employee No. + fingerprint + password), "faceAndFpAndCard" (face + fingerprint + card), "faceAndPwAndFp" (face + password + fingerprint),
"employeeNoAndFace" (employee No. + face), "faceOrfaceAndCard" (face or face + card), "fpOrface" (fingerprint or face), "cardOrfaceOrPw" (card or face or
password), "cardOrFace" (card or face), "cardOrFaceOrFp" (card or face or fingerprint), "cardOrFpOrPw" (card or fingerprint or password), "faceOrPw" (face
or password), "employeeNoAndFaceAndPw" (employee No. + face + card), "cardOrFaceOrFaceAndCard"(card or face or face + card), "iris",
"faceOrFpOrCardOrPwOrIris"(face or fingerprint or card or password or iris), "faceOrCardOrPwOrIris" (face or  card or password or iris), "sleep",
"invalid"*/
        }
    }
}
```

10.3.6.4 Get the holiday group configuration parameters of the control schedule of the card reader authentication mode

Request URL

GET /ISAPI/AccessControl/VerifyHolidayGroupCfg/<holidayGroupID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
holidayGroupID	string	--

Request Message

None

Response Message

```
{
    "VerifyHolidayGroupCfg": {
    /*ro, opt, object*/
        "enable":  true,
        /*ro, req, bool, whether to enable, desc:true (yes), false-no (default)*/
        "groupName":  "test",
        /*ro, req, string, holiday group name*/
        "holidayPlanNo":  "1,3,5"
        /*ro, opt, string, holiday group schedule No., desc:holiday group schedule No.*/
    }
}
```

10.3.6.5 Set holiday group parameters of control schedule of card reader authentication mode

Request URL

PUT /ISAPI/AccessControl/VerifyHolidayGroupCfg/<holidayGroupID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
holidayGroupID	string	--

Request Message

```
{
    "VerifyHolidayGroupCfg": {
    /*opt, object, holiday group parameters of control schedule of card reader authentication mode*/
        "enable":  true,
        /*req, bool, whether to enable, desc:"true"-enable, "false"-disable*/
        "groupName":  "test",
        /*req, string, holiday group name*/
        "holidayPlanNo":  "1,3,5"
        /*opt, string, holiday group schedule No., desc:holiday group schedule No.*/
    }
}
```

Response Message

```
{
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded); it is required when an error occurred*/
    "statusString":  "ok",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "subStatusCode":  "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:when the value of statusCode is not 1, it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, opt, string, error information, desc:this node must be returned when the value of statusCode is not 1*/
}
```

10.3.6.6 Get the holiday group configuration capability of the control schedule of the card reader authentication mode

Request URL

GET /ISAPI/AccessControl/VerifyHolidayGroupCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "VerifyHolidayGroupCfg": {
    /*ro, opt, object*/
        "groupNo": {
        /*ro, opt, object, holiday group No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  16
            /*ro, opt, int, the maximum value*/
        },
        "enable":  "true,false",
        /*ro, opt, string, whether to enable, desc:whether to enable: "true"-enable,"false"-disable*/
        "groupName": {
        /*ro, opt, object, length of holiday group name*/
            "@min":  1,
            /*ro, opt, int, the minimum length*/
            "@max":  32
            /*ro, opt, int, the maximum length*/
        },
        "holidayPlanNo": {
        /*ro, opt, object, holiday group schedule No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  16
            /*ro, opt, int, the maximum value*/
        }
    }
}
```

10.3.6.7 Get holiday schedule parameters of the card reader authentication mode

Request URL

GET /ISAPI/AccessControl/VerifyHolidayPlanCfg/<holidayPlanID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
holidayPlanID	string	--

Request Message

None

Response Message

```
{
    "VerifyHolidayPlanCfg": {
    /*ro, opt, object, holiday schedule parameters of the card reader authentication mode*/
        "enable":  true,
        /*ro, req, bool, whether to enable, desc:true (enable), false (disable)*/
        "beginDate":  "1970-01-01",
        /*ro, req, date, start date of the holiday, desc:device local time*/
        "endDate":  "1970-01-02",
        /*ro, req, date, end date of the holiday, desc:device local time*/
        "HolidayPlanCfg": [
        /*ro, req, array, holiday schedule parameters, subType:object*/
            {
                "id":  1,
                /*ro, req, int, time period No., range:[1,8]*/
                "enable":  true,
                /*ro, req, bool, whether to enable the holiday schedule, desc:true (enable), false (disable)*/
                "verifyMode":  "cardAndPw",
                /*ro, req, enum, authentication mode, subType:string, desc:"cardAndPw" (card+password), "card" (card), "cardOrPw" (card or password), "fp"
(fingerprint), "fpAndPw" (fingerprint+password), "fpOrCard" (fingerprint or card), "fpAndCard" (fingerprint+card), "fpAndCardAndPw"
(fingerprint+card+password), "faceOrFpOrCardOrPw" (face or fingerprint or card or password), "faceAndFp" (face+fingerprint), "faceAndPw" (face+password),
"faceAndCard" (face+card), "face" (face), "employeeNoAndPw" (employee No.+password), "fpOrPw" (fingerprint or password), "employeeNoAndFp" (employee
No.+fingerprint), "employeeNoAndFpAndPw" (employee No.+fingerprint+password), "faceAndFpAndCard" (face+fingerprint+card), "faceAndPwAndFp"
(face+password+fingerprint), "employeeNoAndFace" (employee No.+face), "faceOrfaceAndCard" (face or face+card), "fpOrface" (fingerprint or face),
"cardOrfaceOrPw" (card or face or password), "cardOrFace" (card or face), "cardOrFaceOrFp" (card or face or fingerprint), "cardOrFpOrPw" (card or
fingerprint or password), "iris" (iris), "faceOrFpOrCardOrPwOrIris" (face or fingerprint or card or password or iris), "faceOrCardOrPwOrIris" (face or card
or password or iris), "cardOrFace" (card or face), "cardOrFaceOrFp" (card or face or password), "faceOrPw" (face or password), "employeeNoAndFaceAndPw"
(employee No.+face+password), "cardOrFaceOrFaceAndCard" (card or face or face+card), "faceOrFpOrPw" (face or fingerprint or password),
"cardOrFpOrFaceOrIris" (card or fingerpriont or face or iris), "fpOrFaceOrIrisOrPw" (fingerprint or face or iris or password), "cardOrFpOrIrisOrPw" (card or
fingerprint or iris or password), "cardOrIrisOrPw" (card or iris or password), "cardAndIris" (card+ris), "fpAndIri" (fingerprint+iris), "faceAndIris"
(face+iris), "irisAndPw" (iris+password), "cardAndIrisAndPw" (card+iris+password), "faceAndIrisAndPw" (face+iris+password), "cardAndFaceAndIris"
(card+face+iris)*/
                "TimeSegment": {
                /*ro, opt, object, time*/
                    "beginTime":  "00:00:00",
                    /*ro, req, time, start time of the time period, desc:device local time*/
                    "endTime":  "10:00:00"
                    /*ro, req, time, end time of the time period, desc:device local time*/
                }
            }
        ]
    }
}
```


10.3.6.8 Set holiday schedule parameters of card reader authentication mode

Request URL

PUT /ISAPI/AccessControl/VerifyHolidayPlanCfg/<holidayPlanID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
holidayPlanID	string	--

Request Message

```
{
    "VerifyHolidayPlanCfg": {
    /*opt, object, holiday schedule parameters of the card reader authentication mode*/
        "enable": true,
        /*req, bool, whether to enable, desc:"true"-enable, "false"-disable*/
        "beginDate": "1970-01-01",
        /*req, date, start date of the holiday, desc:device local time*/
        "endDate": "1970-01-02",
        /*req, date, end date of the holiday, desc:device local time*/
        "HolidayPlanCfg": [
        /*req, array, holiday schedule parameters, subType:object*/
            {
                "id": 1,
                /*req, int, time period No., range:[1,8]*/
                "enable": true,
                /*req, bool, whether to enable, desc:"true"-enable, "false"-disable*/
                "verifyMode": "cardAndPw",
                /*req, enum, authentication mode, subType:string, desc:"cardAndPw" (card+password), "card" (card), "cardOrPw" (card or password), "fp"
(fingerprint), "fpAndPw" (fingerprint+password), "fpOrCard" (fingerprint or card), "fpAndCard" (fingerprint+card), "fpAndCardAndPw"
(fingerprint+card+password), "faceOrFpOrCardOrPw" (face or fingerprint or card or password), "faceAndFp" (face+fingerprint), "faceAndPw" (face+password),
"faceAndCard" (face+card), "face" (face), "employeeNoAndPw" (employee No.+password), "fpOrPw" (fingerprint or password), "employeeNoAndFp" (employee
No.+fingerprint), "employeeNoAndFpAndPw" (employee No.+fingerprint+password), "faceAndFpAndCard" (face+fingerprint+card), "faceAndPwAndFp"
(face+password+fingerprint), "employeeNoAndFace" (employee No.+face), "faceOrfaceAndCard" (face or face+card), "fpOrface" (fingerprint or face),
"cardOrfaceOrPw" (card or face or password), "cardOrFace" (card or face), "cardOrFaceOrFp" (card or face or fingerprint), "cardOrFpOrPw" (card or
fingerprint or password), "iris" (iris), "faceOrFpOrCardOrPwOrIris" (face or fingerprint or card or password or iris), "faceOrCardOrPwOrIris" (face or card
or password or iris), "cardOrFace" (card or face), "cardOrFaceOrFp" (card or face or password), "faceOrPw" (face or password), "employeeNoAndFaceAndPw"
(employee No.+face+password), "cardOrFaceOrFaceAndCard" (card or face or face+card), "faceOrFpOrPw" (face or fingerprint or password),
"cardOrFpOrFaceOrIris" (card or fingerpriont or face or iris), "fpOrFaceOrIrisOrPw" (fingerprint or face or iris or password), "cardOrFpOrIrisOrPw" (card or
fingerprint or iris or password), "cardOrIrisOrPw" (card or iris or password), "cardAndIris" (card+ris), "fpAndIri" (fingerprint+iris), "faceAndIris"
(face+iris), "irisAndPw" (iris+password), "cardAndIrisAndPw" (card+iris+password), "faceAndIrisAndPw" (face+iris+password), "cardAndFaceAndIris"
(card+face+iris)*/
                "TimeSegment": {
                /*opt, object, time*/
                    "beginTime": "00:00:00",
                    /*req, time, start time of the time period, desc:device local time*/
                    "endTime": "10:00:00"
                    /*req, time, end time of the time period, desc:device local time*/
                }
            }
        ]
    }
}
```

Response Message

```
{
    "statusCode": 1,
    /*ro, opt, int, status code, desc:1 (succeeded); it is required when an error occurred*/
    "statusString": "OK",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "subStatusCode": "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded); it is required when an error occurred*/
    "errorCode": 1,
    /*ro, opt, int, error code, desc:when the value of statusCode is not 1, it corresponds to subStatusCode*/
    "errorMsg": "ok"
    /*ro, opt, string, error information, desc:this node must be returned when the value of statusCode is not 1*/
}
```


10.3.6.9 Get the holiday schedule configuration capability of the card reader authentication mode

Request URL

GET /ISAPI/AccessControl/VerifyHolidayPlanCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "VerifyHolidayPlanCfg": {
    /*ro, opt, object, holiday schedule of card reader authentication mode*/
        "planNo": {
        /*ro, opt, object, holiday schedule template No.*/
            "@min":  1,
            /*ro, opt, int, maximum value*/
            "@max":  16
            /*ro, opt, int, minimum value*/
        },
        "enable":  "true,false",
        /*ro, opt, string, whether to enable, desc:true (enable), false (disable)*/
        "beginDate":  "1970-01-01",
        /*ro, opt, date, start date of the holiday (device local time), desc:(device local time)*/
        "endDate":  "1970-01-02",
        /*ro, opt, date, end date of the holiday (device local time), desc:(device local time)*/
        "HolidayPlanCfg": {
        /*ro, opt, object, holiday schedule parameters*/
            "maxSize":  8,
            /*ro, opt, int, maximum value*/
            "id": {
            /*ro, opt, object, time period No.*/
                "@min":  1,
                /*ro, opt, int, minimum value*/
                "@max":  8
                /*ro, opt, int, maximum value*/
            },
            "enable":  "true,false",
            /*ro, opt, string, whether to enable, desc:true (enable), false (disable)*/
            "verifyMode": {
            /*ro, opt, object, authentication mode*/
                "@opt":
"cardAndPw,card,cardOrPw,fp,fpAndPw,fpOrCard,fpAndCard,fpAndCardAndPw,faceOrFpOrCardOrPw,faceAndFp,faceAndPw,faceAndCard,face,employeeNoAndPw,fpOrPw,employe
eNoAndFp,employeeNoAndFpAndPw,faceAndFpAndCard,faceAndPwAndFp,employeeNoAndFace,faceOrfaceAndCard,fpOrface,cardOrfaceOrPw,cardOrFace,cardOrFaceOrFp,faceOrPw
,employeeNoAndFaceAndPw,cardOrFaceOrFaceAndCard,iris,faceOrFpOrCardOrPwOrIris,faceOrCardOrPwOrIris,sleep,invalid,cardOrFpOrFaceOrIris,fpOrFaceOrIrisOrPw,car
dOrFpOrIrisOrPw,cardOrIrisOrPw,cardAndIris,fpAndIris,faceAndIris,irisAndPw,cardAndIrisAndPw,faceAndIrisAndPw,cardAndFaceAndIris,Pw,cardOrFpOrPw,faceOrFpOrPw
,cardAndVp,faceAndVp,fpAndVp,pwAndVp"
                /*ro, opt, string, authentication mode, desc:"cardAndPw" (card+password), "card" (card), "cardOrPw" (card or password), "fp" (fingerprint),
"fpAndPw" (fingerprint+password), "fpOrCard" (fingerprint or card), "fpAndCard" (fingerprint+card), "fpAndCardAndPw" (fingerprint+card+password),
"faceOrFpOrCardOrPw" (face or fingerprint or card or password), "faceAndFp" (face+fingerprint), "faceAndPw" (face+password), "faceAndCard" (face+card),
"face" (face), "employeeNoAndPw" (employee No.+password), "fpOrPw" (fingerprint or password), "employeeNoAndFp" (employee No.+fingerprint),
"employeeNoAndFpAndPw" (employee No.+fingerprint+password), "faceAndFpAndCard" (face+fingerprint+card), "faceAndPwAndFp" (face+password+fingerprint),
"employeeNoAndFace" (employee No.+face), "faceOrfaceAndCard" (face or face+card), "fpOrface" (fingerprint or face), "cardOrfaceOrPw" (card or face or
password), "cardOrFace" (card or face), "cardOrFaceOrFp" (card or face or fingerprint), "cardOrFpOrPw" (card or fingerprint or password), "iris" (iris),
"faceOrFpOrCardOrPwOrIris" (face or fingerprint or card or password or iris), "faceOrCardOrPwOrIris" (face or card or password or iris), "cardOrFace" (card
or face), "cardOrFaceOrFp" (card or face or password), "faceOrPw" (face or password), "employeeNoAndFaceAndPw" (employee No.+face+password),
"cardOrFaceOrFaceAndCard" (card or face or face+card), "faceOrFpOrPw" (face or fingerprint or password), "cardOrFpOrFaceOrIris" (card or fingerpriont or
face or iris), "fpOrFaceOrIrisOrPw" (fingerprint or face or iris or password), "cardOrFpOrIrisOrPw" (card or fingerprint or iris or password),
"cardOrIrisOrPw" (card or iris or password), "cardAndIris" (card+ris), "fpAndIri" (fingerprint+iris), "faceAndIris" (face+iris), "irisAndPw"
(iris+password), "cardAndIrisAndPw" (card+iris+password), "faceAndIrisAndPw" (face+iris+password), "cardAndFaceAndIris" (card+face+iris)*/
            },
            "TimeSegment": {
            /*ro, opt, object, time*/
                "beginTime":  "00:00:00",
                /*ro, opt, time, start time, desc:(device local time)*/
                "endTime":  "10:00:00",
                /*ro, opt, time, end time, desc:(device local time)*/
                "validUnit":  "minute"
                /*ro, opt, enum, time accuracy, subType:string, desc:"hour", "minute", "second". If this node is not returned, the default time accuracy is
"minute"*/
            }
        },
        "purePwdVerifyEnable":  true
        /*ro, opt, bool*/
    }
}
```


10.3.6.10 Get the schedule template parameters of card reader authentication mode

Request URL

GET /ISAPI/AccessControl/VerifyPlanTemplate/<planTemplateID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
planTemplateID	string	--

Request Message

None

Response Message

```
{
    "VerifyPlanTemplate": {
    /*ro, opt, object, the schedule template parameters of card reader authentication mode*/
        "enable":  true,
        /*ro, req, bool, whether to enable, desc:"true"-enable, "false"-disable*/
        "templateName":  "test",
        /*ro, req, string, template name*/
        "weekPlanNo":  1,
        /*ro, req, int, week schedule No.*/
        "holidayGroupNo":  "1,3,5"
        /*ro, req, string, holiday group No., desc:holiday group No.*/
    }
}
```

10.3.6.11 Set the schedule template parameters of the card reader authentication mode

Request URL

PUT /ISAPI/AccessControl/VerifyPlanTemplate/<planTemplateID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
planTemplateID	string	--

Request Message

```
{
    "VerifyPlanTemplate": {
    /*opt, object*/
        "enable":  true,
        /*req, bool, whether to enable, desc:true (yes), false (no)*/
        "templateName":  "test",
        /*req, string, template name*/
        "weekPlanNo":  1,
        /*req, int, week schedule No.*/
        "holidayGroupNo":  "1,3,5"
        /*req, string, holiday group No., desc:holiday group No.*/
    }
}
```

Response Message

```
{
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded). It is required when an error occurred*/
    "statusString":  "ok",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded). It is required when an error occurred*/
    "subStatusCode":  "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded). It is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:it is required when the value of statusCode is not 1, and it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, opt, string, error information, desc:this field is required when the value of statusCode is not 1*/
}
```

10.3.6.12 Get the schedule template configuration capability of the card reader authentication mode

Request URL

GET /ISAPI/AccessControl/VerifyPlanTemplate/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "VerifyPlanTemplate": {
    /*ro, opt, object*/
        "templateNo": {
        /*ro, opt, object, schedule template No.*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  16
            /*ro, opt, int*/
        },
        "enable":  "true,false",
        /*ro, opt, string, whether to enable, desc:true (yes), false (no)*/
        "templateName": {
        /*ro, opt, object, template name length*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  32
            /*ro, opt, int*/
        },
        "weekPlanNo": {
        /*ro, opt, object, weekly schedule No.*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  16
            /*ro, opt, int*/
        },
        "holidayGroupNo": {
        /*ro, opt, object, holiday group No.*/
            "@min":  1,
            /*ro, opt, int*/
            "@max":  16
            /*ro, opt, int*/
        }
    }
}
```


10.3.6.13 Get the week schedule configuration parameters of the card reader authentication mode

Request URL

GET /ISAPI/AccessControl/VerifyWeekPlanCfg/<weekPlanID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
weekPlanID	string	--

Request Message

None

Response Message

```
{
    "VerifyWeekPlanCfg": {
    /*ro, opt, object, the week schedule configuration parameters of the card reader authentication mode*/
        "enable":  true,
        /*ro, req, bool, whether to enable the week schedule configuration parameters of the card reader authentication mode, desc:true (enable), false
(disable)*/
        "WeekPlanCfg": [
        /*ro, req, array, week schedule parameters, subType:object*/
            {
                "week":  "Monday",
                /*ro, req, enum, day of a week, subType:string, desc:"Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"*/
                "id":  1,
                /*ro, req, int, time period No., range:[1,8]*/
                "enable":  true,
                /*ro, req, bool, whether to enable week schedule*/
                "verifyMode":  "cardAndPw",
                /*ro, req, enum, authentication mode, subType:string, desc:"cardAndPw" (card+password), "card" (card), "cardOrPw" (card or password), "fp"
(fingerprint), "fpAndPw" (fingerprint+password), "fpOrCard" (fingerprint or card), "fpAndCard" (fingerprint+card), "fpAndCardAndPw"
(fingerprint+card+password), "faceOrFpOrCardOrPw" (face or fingerprint or card or password), "faceAndFp" (face+fingerprint), "faceAndPw" (face+password),
"faceAndCard" (face+card), "face" (face), "employeeNoAndPw" (employee No.+password), "fpOrPw" (fingerprint or password), "employeeNoAndFp" (employee
No.+fingerprint), "employeeNoAndFpAndPw" (employee No.+fingerprint+password), "faceAndFpAndCard" (face+fingerprint+card), "faceAndPwAndFp"
(face+password+fingerprint), "employeeNoAndFace" (employee No.+face), "faceOrfaceAndCard" (face or face+card), "fpOrface" (fingerprint or face),
"cardOrfaceOrPw" (card or face or password), "cardOrFace" (card or face), "cardOrFaceOrFp" (card or face or fingerprint), "cardOrFpOrPw" (card or
fingerprint or password), "iris" (iris), "faceOrFpOrCardOrPwOrIris" (face or fingerprint or card or password or iris), "faceOrCardOrPwOrIris" (face or card
or password or iris), "cardOrFace" (card or face), "cardOrFaceOrFp" (card or face or password), "faceOrPw" (face or password), "employeeNoAndFaceAndPw"
(employee No.+face+password), "cardOrFaceOrFaceAndCard" (card or face or face+card), "faceOrFpOrPw" (face or fingerprint or password),
"cardOrFpOrFaceOrIris" (card or fingerpriont or face or iris), "fpOrFaceOrIrisOrPw" (fingerprint or face or iris or password), "cardOrFpOrIrisOrPw" (card or
fingerprint or iris or password), "cardOrIrisOrPw" (card or iris or password), "cardAndIris" (card+ris), "fpAndIris" (fingerprint+iris), "faceAndIris"
(face+iris), "irisAndPw" (iris+password), "cardAndIrisAndPw" (card+iris+password), "faceAndIrisAndPw" (face+iris+password), "cardAndFaceAndIris"
(card+face+iris)*/
                "TimeSegment": {
                /*ro, req, object, time*/
                    "beginTime":  "00:00:00",
                    /*ro, req, time, start time, desc:device local time*/
                    "endTime":  "10:00:00"
                    /*ro, req, time, end time, desc:device local time*/
                }
            }
        ]
    }
}
```

10.3.6.14 Set the week schedule parameters of the card reader authentication mode

Request URL

PUT /ISAPI/AccessControl/VerifyWeekPlanCfg/<weekPlanID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
weekPlanID	string	--

Request Message

```
{
    "VerifyWeekPlanCfg": {
    /*opt, object, the week schedule configuration parameters of the card reader authentication mode*/
        "enable": true,
        /*req, bool, whether to enable, desc:"true" (enable), "false" (disable)*/
        "WeekPlanCfg": [
        /*req, array, week schedule parameters, subType:object*/
            {
                "week": "Monday",
                /*req, enum, days of the week, subType:string, desc:"Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"*/
                "id": 1,
                /*req, int, time period No., range:[1,8]*/
                "enable": true,
                /*req, bool, whether to enable week schedule*/
                "verifyMode": "cardAndPw",
                /*req, enum, authentication mode, subType:string, desc:"cardAndPw" (card+password), "card" (card), "cardOrPw" (card or password), "fp"
(fingerprint), "fpAndPw" (fingerprint+password), "fpOrCard" (fingerprint or card), "fpAndCard" (fingerprint+card), "fpAndCardAndPw"
(fingerprint+card+password), "faceOrFpOrCardOrPw" (face or fingerprint or card or password), "faceAndFp" (face+fingerprint), "faceAndPw" (face+password),
"faceAndCard" (face+card), "face" (face), "employeeNoAndPw" (employee No.+password), "fpOrPw" (fingerprint or password), "employeeNoAndFp" (employee
No.+fingerprint), "employeeNoAndFpAndPw" (employee No.+fingerprint+password), "faceAndFpAndCard" (face+fingerprint+card), "faceAndPwAndFp"
(face+password+fingerprint), "employeeNoAndFace" (employee No.+face), "faceOrfaceAndCard" (face or face+card), "fpOrface" (fingerprint or face),
"cardOrfaceOrPw" (card or face or password), "cardOrFace" (card or face), "cardOrFaceOrFp" (card or face or fingerprint), "cardOrFpOrPw" (card or
fingerprint or password), "iris" (iris), "faceOrFpOrCardOrPwOrIris" (face or fingerprint or card or password or iris), "faceOrCardOrPwOrIris" (face or card
or password or iris), "cardOrFace" (card or face), "cardOrFaceOrFp" (card or face or password), "faceOrPw" (face or password), "employeeNoAndFaceAndPw"
(employee No.+face+password), "cardOrFaceOrFaceAndCard" (card or face or face+card), "faceOrFpOrPw" (face or fingerprint or password),
"cardOrFpOrFaceOrIris" (card or fingerprint or face or iris), "fpOrFaceOrIrisOrPw" (fingerprint or face or iris or password), "cardOrFpOrIrisOrPw" (card or
fingerprint or iris or password), "cardOrIrisOrPw" (card or iris or password), "cardAndIris" (card+ris), "fpAndIri" (fingerprint+iris), "faceAndIris"
(face+iris), "irisAndPw" (iris+password), "cardAndIrisAndPw" (card+iris+password), "faceAndIrisAndPw" (face+iris+password), "cardAndFaceAndIris"
(card+face+iris)*/
                "TimeSegment": {
                /*req, object, time*/
                    "beginTime": "00:00:00",
                    /*req, time, start time of the time period, desc:device local time*/
                    "endTime": "10:00:00"
                    /*req, time, end time of the time period, desc:device local time*/
                }
            }
        ]
    }
}
```

Response Message

```
{
    "statusCode": 1,
    /*ro, opt, int, status code, desc:1 (succeeded). It is required when an error occurred*/
    "statusString": "OK",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded). It is required when an error occurred*/
    "subStatusCode": "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded). It is required when an error occurred*/
    "errorCode": 1,
    /*ro, opt, int, error code, desc:it is required when the value of statusCode is not 1, and it corresponds to subStatusCode*/
    "errorMsg": "ok"
    /*ro, opt, string, error information, desc:this field is required when the value of statusCode is not 1*/
}
```


10.3.6.15 Get the weekly schedule configuration capability of the card reader authentication mode

Request URL

GET /ISAPI/AccessControl/VerifyWeekPlanCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "VerifyWeekPlanCfg": {
    /*ro, opt, object, weekly schedule parameters of the card reader authentication mode*/
        "planNo": {
        /*ro, opt, object, weekly schedule No.*/
            "@min":  1,
            /*ro, opt, int, minimum value*/
            "@max":  16
            /*ro, opt, int, maximum value*/
        },
        "enable":  "true,false",
        /*ro, opt, string, whether to enable, desc:"true"-enable, "false"-disable*/
        "WeekPlanCfg": {
        /*ro, opt, object, week schedule parameters*/
            "maxSize":  56,
            /*ro, opt, int, maximum value*/
            "week": {
            /*ro, opt, object, day of a week*/
                "@opt":  "Monday,Tuesday,Wednesday,Thursday,Friday,Saturday,Sunday"
                /*ro, opt, string, day of a week, desc:"Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"*/
            },
            "id": {
            /*ro, opt, object, weekly schedule No.*/
                "@min":  1,
                /*ro, opt, int, minimum value*/
                "@max":  8
                /*ro, opt, int, maximum value*/
            },
            "enable":  "true,false",
            /*ro, opt, string, whether to enable, desc:true (enable), false (disable)*/
            "verifyMode": {
            /*ro, opt, object, authentication mode*/
                "@opt":
"cardAndPw,card,cardOrPw,fp,fpAndPw,fpOrCard,fpAndCard,fpAndCardAndPw,faceOrFpOrCardOrPw,faceAndFp,faceAndPw,faceAndCard,face,employeeNoAndPw,fpOrPw,employe
eNoAndFp,employeeNoAndFpAndPw,faceAndFpAndCard,faceAndPwAndFp,employeeNoAndFace,faceOrfaceAndCard,fpOrface,cardOrfaceOrPw,cardOrFace,cardOrFaceOrFp,faceOrPw
,employeeNoAndFaceAndPw,cardOrFaceOrFaceAndCard,iris,faceOrFpOrCardOrPwOrIris,faceOrCardOrPwOrIris,sleep,invalid,cardOrFpOrFaceOrIris,fpOrFaceOrIrisOrPw,car
dOrFpOrIrisOrPw,cardOrIrisOrPw,cardAndIris,fpAndIris,faceAndIris,irisAndPw,cardAndIrisAndPw,faceAndIrisAndPw,cardAndFaceAndIris,Pw,cardOrFpOrPw,faceOrFpOrPw
,cardAndVp,faceAndVp,fpAndVp,pwAndVp"
                /*ro, opt, string, authentication mode, desc:"cardAndPw" (card+password), "card" (card), "cardOrPw" (card or password), "fp" (fingerprint),
"fpAndPw" (fingerprint+password), "fpOrCard" (fingerprint or card), "fpAndCard" (fingerprint+card), "fpAndCardAndPw" (fingerprint+card+password),
"faceOrFpOrCardOrPw" (face or fingerprint or card or password), "faceAndFp" (face+fingerprint), "faceAndPw" (face+password), "faceAndCard" (face+card),
"face" (face), "employeeNoAndPw" (employee No.+password), "fpOrPw" (fingerprint or password), "employeeNoAndFp" (employee No.+fingerprint),
"employeeNoAndFpAndPw" (employee No.+fingerprint+password), "faceAndFpAndCard" (face+fingerprint+card), "faceAndPwAndFp" (face+password+fingerprint),
"employeeNoAndFace" (employee No.+face), "faceOrfaceAndCard" (face or face+card), "fpOrface" (fingerprint or face), "cardOrfaceOrPw" (card or face or
password), "cardOrFace" (card or face), "cardOrFaceOrFp" (card or face or fingerprint), "cardOrFpOrPw" (card or fingerprint or password), "iris" (iris),
"faceOrFpOrCardOrPwOrIris" (face or fingerprint or card or password or iris), "faceOrCardOrPwOrIris" (face or card or password or iris), "cardOrFace" (card
or face), "cardOrFaceOrFp" (card or face or password), "faceOrPw" (face or password), "employeeNoAndFaceAndPw" (employee No.+face+password),
"cardOrFaceOrFaceAndCard" (card or face or face+card), "faceOrFpOrPw" (face or fingerprint or password), "cardOrFpOrFaceOrIris" (card or fingerprint or
face or iris), "fpOrFaceOrIrisOrPw" (fingerprint or face or iris or password), "cardOrFpOrIrisOrPw" (card or fingerprint or iris or password),
"cardOrIrisOrPw" (card or iris or password), "cardAndIris" (card+ris), "fpAndIri" (fingerprint+iris), "faceAndIris" (face+iris), "irisAndPw"
(iris+password), "cardAndIrisAndPw" (card+iris+password), "faceAndIrisAndPw" (face+iris+password), "cardAndFaceAndIris" (card+face+iris), "sleep",
"invalid"*/
            },
            "TimeSegment": {
            /*ro, opt, object, time*/
                "beginTime":  "00:00:00",
                /*ro, opt, time, start time, desc:start time of the time period (device local time)*/
                "endTime":  "10:00:00",
                /*ro, opt, time, end time, desc:end time of the time period (device local time)*/
                "validUnit":  "minute"
                /*ro, opt, enum, time accuracy, subType:string, desc:If this node is not returned, it indicates that the time accuracy is "minute". "hour",
"minute", "second"*/
            }
        },
        "purePwdVerifyEnable":  true
        /*ro, opt, bool*/
    }
}
```

10.3.7 Credential Recognition Unit Management

10.3.7.1 Set the card reader parameters

Request URL

PUT /ISAPI/AccessControl/CardReaderCfg/<cardReaderID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
cardReaderID	string	--

Request Message

```
{
    "CardReaderCfg": {
    /*req, object, card reader information*/
        "enable": true,
        /*req, bool, whether to enable, desc:true-yes, false-no*/
        "okLedPolarity": "cathode",
        /*opt, enum, OK LED polarity, subType:string, desc:"cathode", "anode"”*/
        "errorLedPolarity": "cathode",
        /*opt, enum, Error LED polarity, subType:string, desc:"cathode", "anode"*/
        "buzzerPolarity": "cathode",
        /*opt, enum, buzzer polarity, subType:string, desc:"cathode", "anode"*/
        "swipeInterval": 1,
        /*opt, int, time interval of repeated authentication, unit:s, desc:it is valid for authentication modes such as fingerprint, card, face, etc.*/
        "pressTimeout": 1,
        /*opt, int, timeout to reset entry on keypad, unit:s*/
        "enableFailAlarm": true,
        /*opt, bool, whether to enable excessive failed authentication attempt alarm*/
        "maxReadCardFailNum": 1,
        /*opt, int, maximum number of failed authentication attempts*/
        "enableTamperCheck": true,
        /*opt, bool, whether to enable tampering detection*/
        "offlineCheckTime": 1,
        /*opt, int, time to detect after the card reader is offline, unit:s*/
        "fingerPrintCheckLevel": 1,
        /*opt, enum, fingerprint recognition level, subType:int, desc:1-1/10 false acceptance rate (FAR), 2-1/100 false acceptance rate (FAR), 3-1/1000 false acceptance rate (FAR), 4-1/10000 false acceptance rate (FAR), 5-1/100000 false acceptance rate (FAR), 6-1/1000000 false acceptance rate (FAR), 7-1/10000000 false acceptance rate (FAR), 8-1/100000000 false acceptance rate (FAR), 9-3/100 false acceptance rate (FAR), 10-3/1000 false acceptance rate (FAR), 11-3/10000 false acceptance rate (FAR), 12-3/100000 false acceptance rate (FAR), 13-3/1000000 false acceptance rate (FAR), 14-3/10000000 false acceptance rate (FAR), 15-3/100000000 false acceptance rate (FAR), 16-Automatic Normal, 17-Automatic Secure, 18-Automatic More Secure (currently not support)*/
        "useLocalController": true,
        /*opt, bool, whether it is connected to the distributed controller*/
        "localControllerID": 1,
        /*opt, int, distributed controller No., range:[0,64], dep:and,{$.CardReaderCfg.localControllerID,eq,true}, desc:0-unregistered. This field is valid only when useLocalController is "true"*/
        "localControllerReaderID": 1,
        /*opt, int, card reader ID of the distributed controller, dep:and,{$.CardReaderCfg.localControllerID,eq,true}, desc:0-unregistered. This field is valid only when useLocalController is "true"*/
        "cardReaderChannel": 1,
        /*opt, enum, communication channel No. of the card reader, subType:int, dep:and,{$.CardReaderCfg.localControllerID,eq,true}, desc:0-Wiegand or offline, 1-RS-485A, 2-RS-485B. This field is valid only when useLocalController is "true"*/
        "fingerPrintImageQuality": 1,
        /*opt, enum, fingerprint image quality, subType:int, desc:1-low quality (V1), 2-medium quality (V1), 3-high quality (V1), 4-highest quality (V1), 5-low quality (V2), 6-medium quality (V2), 7-high quality (V2), 8-highest quality (V2)*/
        "fingerPrintContrastTimeOut": 1,
        /*opt, enum, fingerprint comparison timeout, subType:int, desc:it is between 1 and 20, unit: second, 255-infinite*/
        "fingerPrintRecogizeInterval": 1,
        /*opt, enum, fingerprint scanning interval, subType:int, desc:it is between 1 and 10, unit: second, 255-no delay*/
        "fingerPrintMatchFastMode": 1,
        /*opt, enum, fingerprint matching quick mode, subType:int, desc:1-quick mode 1, 2-quick mode 2, 3-quick mode 3, 4-quick mode 4, 5-quick mode 5, 255-automatic*/
        "fingerPrintModuleSensitive": 1,
        /*opt, enum, fingerprint module sensitivity, subType:int, desc:fingerprint module sensitivity,which is between 1 and 8*/
        "fingerPrintModuleLightCondition": "outdoor",
        /*opt, enum, fingerprint module light condition, subType:string, desc:"outdoor", "indoor”*/
        "faceMatchThresholdN": 1,
        /*opt, int, threshold of face picture 1:N comparison, range:[0,100], desc:threshold of face picture 1:N comparison,which is between 0 and 100*/
        "faceQuality": 1,
        /*opt, int, face picture quality, range:[0,100]*/
        "faceRecogizeTimeOut": 1,
        /*opt, enum, face recognition timeout, subType:int, desc:it is between 1 and 20, unit: second, 255-infinite*/
        "faceRecogizeInterval": 1,
        /*opt, enum, face recognition interval, subType:int, desc:it is between 1 and 10, unit: second, 255-no delay*/
        "cardReaderFunction": ["fingerPrint", "face", "fingerVein", "iris", "card"],
        /*opt, enumarray, card reader type, subType:string, desc:"fingerPrint"-fingerprint, "face", "fingerVein"-finger vein, “iris”. For example, ["fingerPrint","face"] indicates that the card reader supports both fingerprint and face*/
        "cardReaderDescription": "Wiegand\u000485Offline",
        /*opt, string, card reader description, desc:if the card reader is the Wiegand card reader or if offline, this field will be set to "Wiegand" or "485Offline”*/
        "faceImageSensitometry": 1,
        /*opt, int, face picture exposure, range:[0,655535]*/
        "livingBodyDetect": true,
        /*opt, bool, whether to enable human detection*/
        "faceMatchThreshold1": 1,
        /*opt, int, threshold of face picture 1:1 comparison, range:[0,100], desc:threshold of face picture 1:1 comparison,which is between 0 and 100*/
        "buzzerTime": 1,
        /*opt, int, buzzing duration, range:[0,59999], unit:s, desc:0-long buzzing*/
        "faceMatch1SecurityLevel": 1,
        /*opt, enum, security level of face 1:1 recognition, subType:int, desc:1-normal, 2-high, 3-higher*/
        "faceMatchNSecurityLevel": 1,
        /*opt, enum, security level of face 1:N recognition, subType:int, desc:1-normal, 2-high, 3-higher*/
        "envirMode": "indoor",
        /*opt, enum, environment mode of face recognition, subType:string, desc:"indoor", "other”*/
        "liveDetLevelSet": "low",
        /*opt, enum, threshold level of liveness detection, subType:string, desc:"low", "middle"-medium, "high"*/
        "liveDetThreshold": 1,
        /*opt, int, range:[0,100]*/
        "liveDetAntiAttackCntLimit": 1,
        /*opt, int, number of anti-attacks of liveness detection,, range:[1,255], desc:this value should be configured as the same one on both client and device*/
        "enableLiveDetAntiAttack": true,
        /*opt, bool, whether to enable anti-attack for liveness detection*/
        "liveDetAntiAttackLockedTime": 1,
```

```
        /*opt, int, range:[0,300], unit:s*/
        "supportDelFPByID":  true,
        /*opt, bool, whether the card reader supports deleting fingerprint by fingerprint ID, desc:true-yes, false-no*/
        "fingerPrintCapacity":  1,
        /*opt, int, fingerprint capacity*/
        "fingerPrintNum":  1,
        /*opt, int, number of added fingerprints*/
        "defaultVerifyMode":  "cardAndPw",
        /*opt, enum, default authentication mode of the fingerprint and card reader (factory defaults):, subType:string, desc:factory defaults; "cardAndPw"
(card+password), "card" (card), "cardOrPw" (card or password), "fp" (fingerprint), "fpAndPw" (fingerprint+password), "fpOrCard" (fingerprint or card),
"fpAndCard" (fingerprint+card), "fpAndCardAndPw" (fingerprint+card+password), "faceOrFpOrCardOrPw" (face or fingerprint or card or password), "faceAndFp"
(face+fingerprint), "faceAndPw" (face+password), "faceAndCard" (face+card), "face" (face), "employeeNoAndPw" (employee No.+password), "fpOrPw" (fingerprint
or password), "employeeNoAndFp" (employee No.+fingerprint), "employeeNoAndFpAndPw" (employee No.+fingerprint+password), "faceAndFpAndCard"
(face+fingerprint+card), "faceAndPwAndFp" (face+password+fingerprint), "employeeNoAndFace" (employee No.+face), "faceOrfaceAndCard" (face or face+card),
"fpOrface" (fingerprint or face), "cardOrfaceOrPw" (card or face or password), "cardOrFace" (card or face), "cardOrFaceOrFp" (card or face or fingerprint),
"cardOrFpOrPw" (card or fingerprint or password), "iris" (iris), "faceOrFpOrCardOrPwOrIris" (face or fingerprint or card or password or iris),
"faceOrCardOrPwOrIris" (face or card or password or iris), "cardOrFace" (card or face), "cardOrFaceOrFp" (card or face or password), "faceOrPw" (face or
password), "employeeNoAndFaceAndPw" (employee No.+face+password), "cardOrFaceOrFaceAndCard" (card or face or face+card), "faceOrFpOrPw" (face or fingerprint
or password), "cardOrFpOrFaceOrIris" (card or fingerprint or face or iris), "fpOrFaceOrIrisOrPw" (fingerprint or face or iris or password),
"cardOrFpOrIrisOrPw" (card or fingerprint or iris or password), "cardOrIrisOrPw" (card or iris or password), "cardAndIris" (card+ris), "fpAndIri"
(fingerprint+iris), "faceAndIris" (face+iris), "irisAndPw" (iris+password), "cardAndIrisAndPw" (card+iris+password), "faceAndIrisAndPw"
(face+iris+password), "cardAndFaceAndIris" (card+face+iris)*/
        "faceRecogizeEnable":  1,
        /*opt, enum, whether to enable facial recognition, subType:int, desc:1-enable, 2-disable, 3-attendence checked in/out by recognition of multiple
faces*/
        "FPAlgorithmVersion":  "test",
        /*opt, string, fingerprint algorithm library version, range:[1,32]*/
        "cardReaderVersion":  "test",
        /*opt, string, card reader version, range:[1,32]*/
        "enableReverseCardNo":  true,
        /*opt, bool, whether to enable reversing the card No.*/
        "independSwipeIntervals":  0,
        /*opt, int, time interval of person authentication, unit: second. This time interval is calculated for each person separately and is different from
swipeInterval, desc:time interval of person authentication,unit: second. This time interval is calculated for each person separately and is different from
swipeInterval*/
        "maskFaceMatchThresholdN":  1,
        /*opt, int, 1:N face picture (face with mask and normal background) comparison threshold, range:[0,100], desc:1:N face picture (face with mask and
normal background) comparison threshold,value range: [0,100]*/
        "maskFaceMatchThreshold1":  1,
        /*opt, int, 1:1 face picture (face with mask and normal background) comparison threshold, range:[0,100], desc:1:1 face picture (face with mask and
normal background) comparison threshold,value range: [0,100]*/
        "faceMotionDetLevel":  "low",
        /*opt, enum, face motion detection level, subType:string, desc:"high", "medium", "low"*/
        "showMode":  "normal",
        /*opt, enum, display mode, subType:string, desc:this node is not valid; simple mode indicates that the device displays authentication results
exclude employee No., name, etc.; the device applies normal mode by default; advertisement mode indicates that the device displays both advertisement and
authentication results; meeting mode indicates that the device displays check-in page of the conference; custom mode indicates that the device displays
layout of the interface customized by users; "concise"-simple mode, "normal"-normal mode (default), "advertising"-advertisement mode, "meeting"-meeting
mode, "selfDefine"-custom mode*/
        "enableScreenOff":  true,
        /*opt, bool, whether to enable auto locking the screen*/
        "screenOffTimeout":  1
        /*opt, int, time, step:1, unit:s*/
    }
}
```


Response Message

```
{
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded). It is required when an error occurred*/
    "statusString":  "OK",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded). It is required when an error occurred*/
    "subStatusCode":  "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded). It is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:when the value of statusCode is not 1, it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, opt, string, error description, desc:this node is required when the value of statusCode is not 1*/
}
```

10.3.7.2 Get the card reader configuration parameters

Request URL

GET /ISAPI/AccessControl/CardReaderCfg/<cardReaderID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
cardReaderID	string	--

Request Message

None

Response Message

```
{
    "CardReaderCfg": {
    /*ro, req, object, card reader information*/
        "enable": true,
        /*ro, req, bool, whether to enable, desc:true (enable), false (disable)*/
        "okLedPolarity": "cathode",
        /*ro, opt, enum, OK LED polarity, subType:string, desc:"cathode", "anode"*/
        "errorLedPolarity": "cathode",
        /*ro, opt, enum, Error LED polarity, subType:string, desc:"cathode", "anode"*/
        "buzzerPolarity": "cathode",
        /*ro, opt, enum, buzzer polarity, subType:string, desc:“cathode", "anode"*/
        "swipeInterval": 1,
        /*ro, opt, int, time interval of repeated authentication, unit:s, desc:which is valid for authentication modes such as fingerprint, card, face,
etc.*/
        "pressTimeout": 1,
        /*ro, opt, int, timeout to reset entry on keypad, unit:s*/
        "enableFailAlarm": true,
        /*ro, opt, bool, whether to enable excessive failed authentication attempts alarm*/
        "maxReadCardFailNum": 1,
        /*ro, opt, int, maximum number of failed authentication attempts*/
        "enableTamperCheck": true,
        /*ro, opt, bool, whether to enable tampering detection*/
        "offlineCheckTime": 1,
        /*ro, opt, int, time to detect after the card reader is offline, unit:s*/
        "fingerPrintCheckLevel": 1,
        /*ro, opt, enum, fingerprint recognition level, subType:int, desc:1-1/10 false acceptance rate (FAR), 2-1/100 false acceptance rate (FAR), 3-1/1000
false acceptance rate (FAR), 4-1/10000 false acceptance rate (FAR), 5-1/100000 false acceptance rate (FAR), 6-1/1000000 false acceptance rate (FAR), 7-
1/10000000 false acceptance rate (FAR), 8-1/100000000 false acceptance rate (FAR), 9-3/100 false acceptance rate (FAR), 10-3/1000 false acceptance rate
(FAR), 11-3/10000 false acceptance rate (FAR), 12-3/100000 false acceptance rate (FAR), 13-3/1000000 false acceptance rate (FAR), 14-3/10000000 false
acceptance rate (FAR), 15-3/100000000 false acceptance rate (FAR), 16-Automatic Normal, 17-Automatic Secure, 18-Automatic More Secure (currently not
support)*/
        "useLocalController": true,
        /*ro, opt, bool, whether it is connected to the distributed controller*/
        "localControllerID": 1,
        /*ro, opt, int, distributed controller No., range:[0,64], dep:and,{$.CardReaderCfg.localControllerID,eq,true}, desc:which is between 1 and 64, 0-
unregistered. This field is valid only when useLocalController is "true”*/
        "localControllerReaderID": 1,
        /*ro, opt, int, card reader ID of the distributed controller, 0-unregistered, dep:and,{$.CardReaderCfg.localControllerID,eq,true}, desc:this field
is valid only when useLocalController is "true”*/
        "cardReaderChannel": 1,
        /*ro, opt, enum, communication channel No. of the card reader, subType:int, dep:and,{$.CardReaderCfg.localControllerID,eq,true}, desc:0-Wiegand or
offline, 1-RS-485A, 2-RS-485B. This field is valid only when useLocalController is "true”*/
        "fingerPrintImageQuality": 1,
        /*ro, opt, enum, fingerprint image quality, subType:int, desc:1-low quality (V1), 2-medium quality (V1), 3-high quality (V1), 4-highest quality
(V1), 5-low quality (V2), 6-medium quality (V2), 7-high quality (V2), 8-highest quality (V2)*/
        "fingerPrintContrastTimeOut": 1,
        /*ro, opt, enum, fingerprint comparison timeout, subType:int, desc:fingerprint comparison timeout,which is between 1 and 20,unit: second,255-
infinite*/
        "fingerPrintRecogizeInterval": 1,
        /*ro, opt, enum, fingerprint scanning interval, subType:int, desc:fingerprint scanning interval,which is between 1 and 10,unit: second,255-no
delay*/
        "fingerPrintMatchFastMode": 1,
        /*ro, opt, enum, fingerprint matching quick mode, subType:int, desc:1-quick mode 1, 2-quick mode 2, 3-quick mode 3, 4-quick mode 4, 5-quick mode 5,
255-automatic*/
        "fingerPrintModuleSensitive": 1,
        /*ro, opt, enum, fingerprint module sensitivity, subType:int, desc:fingerprint module sensitivity,which is between 1 and 8*/
        "fingerPrintModuleLightCondition": "outdoor",
        /*ro, opt, enum, fingerprint module light condition, subType:string, desc:"outdoor", "indoor”*/
        "faceMatchThresholdN": 1,
        /*ro, opt, int, threshold of face picture 1:N comparison,which is between 0 and 100, range:[0,100]*/
        "faceQuality": 1,
        /*ro, opt, int, face picture quality, range:[0,100]*/
        "faceRecogizeTimeOut": 1,
        /*ro, opt, enum, face recognition timeout, subType:int, desc:face recognition timeout,which is between 1 and 20,unit: second,255-infinite*/
        "faceRecogizeInterval": 1,
        /*ro, opt, enum, face recognition interval, subType:int, desc:face recognition interval,which is between 1 and 10,unit: second,255-no delay*/
        "cardReaderFunction": ["fingerPrint", "face", "card"],
        /*ro, opt, enumarray, card reader type, subType:string, desc:"fingerPrint”, "face", "fingerVein". For example, ["fingerPrint", "face"] indicates
that the card reader supports both fingerprint and face*/
        "cardReaderDescription": "Wiegand\u000485Offline",
        /*ro, opt, string, card reader description, desc:if the card reader is the Wiegand card reader or if offline, this field will be set to "Wiegand" or
"485Offline”*/
        "faceImageSensitometry": 1,
        /*ro, opt, int, face picture exposure, range:[0,655535]*/
        "livingBodyDetect": true,
        /*ro, opt, bool, whether to enable human detection*/
        "faceMatchThreshold1": 1,
        /*ro, opt, int, threshold of face picture 1:1 comparison, range:[0,100]*/
        "buzzerTime": 1,
        /*ro, opt, int, buzzing duration, range:[0,59999], unit:s, desc:buzzing duration,which is between 0 and 5999,unit: second,0-long buzzing*/
        "faceMatch1SecurityLevel": 1,
        /*ro, opt, enum, security level of face 1:1 recognition, subType:int, desc:1 (normal), 2 (high), 3 (higher)*/
        "faceMatchNSecurityLevel": 1,
        /*ro, opt, enum, security level of face 1:N recognition: 1-normal,2-high,3-higher, subType:int, desc:1 (normal), 2 (high), 3 (higher)*/
        "envirMode": "other",
        /*ro, opt, enum, environment mode of face recognition, subType:string, desc:"indoor", "other”*/
        "liveDetLevelSet": "low",
        /*ro, opt, enum, threshold level of liveness detection, subType:string, desc:"low", "middle", "high”*/
```

```
        "liveDetThreshold":  1,
        /*ro, opt, int, range:[0,100]*/
        "liveDetAntiAttackCntLimit":  1,
        /*ro, opt, int, number of anti-attacks of liveness detection, range:[1,255], desc:this value should be configured as the same one on both client and
device*/
        "enableLiveDetAntiAttack":  true,
        /*ro, opt, bool, whether to enable anti-attack for liveness detection*/
        "liveDetAntiAttackLockedTime":  1,
        /*ro, opt, int, range:[0,300], unit:s*/
        "supportDelFPByID":  true,
        /*ro, opt, bool, whether the card reader supports deleting fingerprint by fingerprint ID, desc:"true"-yes, "false"-no*/
        "fingerPrintCapacity":  1,
        /*ro, opt, int, fingerprint capacity*/
        "fingerPrintNum":  1,
        /*ro, opt, int, number of added fingerprints*/
        "defaultVerifyMode":  "cardAndPw",
        /*ro, opt, enum, default authentication mode of the fingerprint and card reader (factory defaults), subType:string, desc:factory defaults;
"cardAndPw" (card+password), "card" (card), "cardOrPw" (card or password), "fp" (fingerprint), "fpAndPw" (fingerprint+password), "fpOrCard" (fingerprint or
card), "fpAndCard" (fingerprint+card), "fpAndCardAndPw" (fingerprint+card+password), "faceOrFpOrCardOrPw" (face or fingerprint or card or password),
"faceAndFp" (face+fingerprint), "faceAndPw" (face+password), "faceAndCard" (face+card), "face" (face), "employeeNoAndPw" (employee No.+password), "fpOrPw"
(fingerprint or password), "employeeNoAndFp" (employee No.+fingerprint), "employeeNoAndFpAndPw" (employee No.+fingerprint+password), "faceAndFpAndCard"
(face+fingerprint+card), "faceAndPwAndFp" (face+password+fingerprint), "employeeNoAndFace" (employee No.+face), "faceOrfaceAndCard" (face or face+card),
"fpOrface" (fingerprint or face), "cardOrfaceOrPw" (card or face or password), "cardOrFace" (card or face), "cardOrFaceOrFp" (card or face or fingerprint),
"cardOrFpOrPw" (card or fingerprint or password), "iris" (iris), "faceOrFpOrCardOrPwOrIris" (face or fingerprint or card or password or iris),
"faceOrCardOrPwOrIris" (face or card or password or iris), "cardOrFace" (card or face), "cardOrFaceOrFp" (card or face or password), "faceOrPw" (face or
password), "employeeNoAndFaceAndPw" (employee No.+face+password), "cardOrFaceOrFaceAndCard" (card or face or face+card), "faceOrFpOrPw" (face or fingerprint
or password), "cardOrFpOrFaceOrIris" (card or fingerpriont or face or iris), "fpOrFaceOrIrisOrPw" (fingerprint or face or iris or password),
"cardOrFpOrIrisOrPw" (card or fingerprint or iris or password), "cardOrIrisOrPw" (card or iris or password), "cardAndIris" (card+ris), "fpAndIri"
(fingerprint+iris), "faceAndIris" (face+iris), "irisAndPw" (iris+password), "cardAndIrisAndPw" (card+iris+password), "faceAndIrisAndPw"
(face+iris+password), "cardAndFaceAndIris" (card+face+iris)*/
        "faceRecogizeEnable":  1,
        /*ro, opt, enum, whether to enable facial recognition, subType:int, desc:1 (enable), 2 (disable), 3 (attendence checked in/out by recognition of
multiple faces)*/
        "FPAlgorithmVersion":  "test",
        /*ro, opt, string, fingerprint algorithm library version, range:[1,32]*/
        "cardReaderVersion":  "test",
        /*ro, opt, string, card reader version, range:[1,32]*/
        "enableReverseCardNo":  true,
        /*ro, opt, bool, whether to enable reversing the card No.*/
        "independSwipeIntervals":  0,
        /*ro, opt, int, time interval of person authentication, desc:unit: second. This time interval is calculated for each person separately and is
different from swipeInterval*/
        "maskFaceMatchThresholdN":  1,
        /*ro, opt, int, 1:N face picture (face with mask and normal background) comparison threshold, range:[0,100]*/
        "maskFaceMatchThreshold1":  1,
        /*ro, opt, int, 1:1 face picture (face with mask and normal background) comparison threshold, range:[0,100]*/
        "faceMotionDetLevel":  "low",
        /*ro, opt, enum, face motion detection level, subType:string, desc:"high", "medium", "low"*/
        "showMode":  "normal",
        /*ro, opt, enum, display mode, subType:string, desc:this node is not valid; simple mode indicates that the device displays authentication results
exclude employee No., name, etc.; the device applies normal mode by default; advertisement mode indicates that the device displays both advertisement and
authentication results; meeting mode indicates that the device displays check-in page of the conference; custom mode indicates that the device displays
layout of the interface customized by users; "concise"-simple mode, "normal"-normal mode (default), "advertising"-advertisement mode, "meeting"-meeting
mode, "selfDefine"-custom mode*/
        "enableScreenOff":  true,
        /*ro, opt, bool, whether to enable auto locking the screen*/
        "screenOffTimeout":  1
        /*ro, opt, int, time, step:1, unit:s*/
    }
}
```


10.3.7.3 Get the configuration capability of the card reader

Request URL

GET /ISAPI/AccessControl/CardReaderCfg/capabilities?format=json&cardReaderID=<cardReaderID>

Query Parameter

Parameter Name	Parameter Type	Description
cardReaderID	string	--

Request Message

None

Response Message

```
{
    "CardReaderCfg": {
    /*ro, req, object, Set Card Reader Parameters*/
        "cardReaderNo": {
        /*ro, opt, object, card reader No., desc:card reader No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  512,
            /*ro, opt, int, the maximum value*/
        },
        "enable":  "true,false",
        /*ro, opt, string, whether to enable, desc:"true"-yes,"false"-no*/
        "swipeInterval": {
        /*ro, opt, object, time interval of repeated authentication, desc:it is valid for authentication modes such as fingerprint, card, face, etc., unit:
second*/
            "@min":  1,
            /*ro, req, int, the minimum value*/
            "@max":  255
            /*ro, req, int, the maximum value*/
        },
        "enableFailAlarm":  "true,false",
        /*ro, opt, string, whether to enable excessive failed authentication attempts alarm*/
        "maxReadCardFailNum": {
        /*ro, opt, object, maximum number of failed authentication attempts*/
            "@min":  1,
            /*ro, req, int, the minimum value*/
            "@max":  255
            /*ro, req, int, the maximum value*/
        },
        "enableTamperCheck":  "true,false",
        /*ro, opt, string, whether to enable tampering detection*/
        "offlineCheckTime": {
        /*ro, opt, object, time to detect after the card reader is offline, desc:unit: second*/
            "@min":  1,
            /*ro, req, int, the minimum value*/
            "@max":  255
            /*ro, req, int, the maximum value*/
        },
        "cardReaderDescription": {
        /*ro, opt, object, card reader description, desc:if the card reader is the Wiegand card reader or if offline, this field will be set to "Wiegand" or
"485Offline"*/
            "@min":  1,
            /*ro, req, int, the minimum value*/
            "@max":  16
            /*ro, req, int, the maximum value*/
        },
        "buzzerTime": {
        /*ro, opt, object, buzzing duration, desc:it is between 0 and 5999, unit: second, 0 (long buzzing)*/
            "@min":  0,
            /*ro, req, int, the minimum value*/
            "@max":  5999
            /*ro, req, int, the maximum value*/
        },
        "fingerPrintCapacity": {
        /*ro, opt, object, maximum number of fingerprints that can be added*/
            "@min":  1,
            /*ro, req, int, the minimum value*/
            "@max":  100
            /*ro, req, int, the maximum value*/
        },
        "fingerPrintNum": {
        /*ro, opt, object, number of added fingerprints*/
            "@min":  1,
            /*ro, req, int, the minimum value*/
            "@max":  100
            /*ro, req, int, the maximum value*/
        },
        "FPAlgorithmVersion": {
        /*ro, opt, object, fingerprint algorithm library version*/
            "@min":  1,
            /*ro, req, int, the minimum value*/
            "@max":  1
            /*ro, req, int, the maximum value*/
        },
        "cardReaderVersion": {
        /*ro, opt, object, card reader version*/
            "@min":  1,
            /*ro, req, int, the minimum value*/
            "@max":  1
            /*ro, req, int, the maximum value*/
        },
    }
}
```


10.3.7.4 Get the configuration capability of enabling NFC (Near-Field Communication) function

Request URL

GET /ISAPI/AccessControl/Configuration/NFCCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "NFCCfgCap": {
    /*ro, opt, object, configuration capability of enabling NFC (Near-Field Communication) function*/
        "enable":  "true,false"
        /*ro, req, string, whether to enable NFC function, desc:true-yes, false-no (default)*/
    }
}
```

10.3.7.5 Get the parameters of enabling NFC (Near-Field Communication) function

Request URL

GET /ISAPI/AccessControl/Configuration/NFCCfg?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "NFCCfg": {
    /*ro, req, object*/
        "enable":  true
        /*ro, req, bool, whether to enable NFC function, desc:true (yes), false (no). The value of this node is "false" by default*/
    }
}
```

10.3.7.6 Set the parameters of enabling NFC (Near-Field Communication) function

Request URL

PUT /ISAPI/AccessControl/Configuration/NFCCfg?format=json

Query Parameter

None

Request Message

```
{
    "NFCCfg": {
    /*req, object, configuration capability of enabling NFC (Near-Field Communication) function*/
        "enable":  true
        /*req, bool, whether to enable NFC function, desc:true-yes, false-no (default)*/
    }
}
```

Response Message

```
{
    "requestURL":  "test",
    /*ro, opt, string, URI*/
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded). It is required when an error occurred*/
    "statusString":  "test",
    /*ro, opt, string, status description, desc:"ok" (succeeded). It is required when an error occurred*/
    "subStatusCode":  "test",
    /*ro, opt, string, sub status code, desc:"ok" (succeeded). It is required when an error occurred*/
    "errorCode":  1,
    /*ro, req, int, error code, desc:it is required when the value of statusCode is not 1, and it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, req, string, error information, desc:this node is required when the value of statusCode is not 1*/
}
```

10.3.7.7 Get the configuration capability of enabling RF (Radio Frequency) card recognition

Request URL

GET /ISAPI/AccessControl/Configuration/RFCardCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "RFCardCfgCap": {
    /*ro, req, object*/
        "cardType": {
        /*ro, opt, object, card type, desc:"EMCard"-EM card, "M1Card"-M1 card, "CPUCard"-CPU card, "IDCard"-ID card, "DesfireCard"-DESFire card,
"FelicaCard"-FeliCa card*/
            "@opt": ["EMCard", "M1Card", "CPUCard", "IDCard", "FelicaCard"]
            /*ro, req, array, options, subType:string*/
        },
        "enabled": {
        /*ro, opt, object, whether to enable RF card recognition*/
            "@opt": [true, false]
            /*ro, req, array, options, subType:bool*/
        }
    }
}
```

10.3.7.8 Get the parameters of enabling RF (Radio Frequency) card recognition

Request URL

GET /ISAPI/AccessControl/Configuration/RFCardCfg?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "RFCardCfg": [
    /*ro, req, array, subType:object*/
        {
            "cardType":  "EMCard",
            /*ro, req, enum, card type, subType:string, desc:"EMCard" (EM card), "M1Card" (M1 card), "CPUCard" (CPU card), "IDCard" (ID card), "DesfireCard"
(DESFire card), "FelicaCard" (FeliCa card)*/
            "enabled":  true
            /*ro, req, bool, whether to enable RF card recognition, desc:true-yes, false-no*/
        }
    ]
}
```

10.3.7.9 Set the parameters of enabling RF (Radio Frequency) card recognition

Request URL

PUT /ISAPI/AccessControl/Configuration/RFCardCfg?format=json

Query Parameter

None

Request Message

```
{
    "RFCardCfg": [
    /*req, array, the parameters of enabling RF (Radio Frequency) card recognition, subType:object*/
        {
            "cardType":  "EMCard",
            /*req, enum, card type, subType:string, desc:"EMCard"(EM card), "M1Card"(M1 card), "CPUCard"(CPU card), "IDCard"(ID card), "DesfireCard"(DESFire
card), "FelicaCard"(FeliCa card)*/
            "enabled":  true
            /*req, bool, whether to enable RF card recognition, desc:"true"(yes), "false"(no)*/
        }
    ]
}
```

Response Message

```
{
    "requestURL":  "test",
    /*ro, opt, string, URI*/
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded). It is required when an error occurred*/
    "statusString":  "test",
    /*ro, opt, string, status description, desc:"ok" (succeeded). It is required when an error occurred*/
    "subStatusCode":  "test",
    /*ro, opt, string, sub status code, desc:"ok" (succeeded). It is required when an error occurred*/
    "errorCode":  1,
    /*ro, req, int, error code, desc:it is required when the value of statusCode is not 1, and it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, req, string, error information, desc:this node is required when the value of statusCode is not 1*/
}
```


10.3.7.10 Get the parameters of face recognition terminal

Request URL

GET /ISAPI/AccessControl/IdentityTerminal

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>
<IdentityTerminal xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, parameters of face recognition terminal, attr:version{req, string, protocolVersion}-->
  <terminalMode>
    <!--ro, opt, enum, terminal mode, subType:string, desc:"authMode" (authentication mode), "registerMode" (registration mode)-->authMode
  </terminalMode>
  <idCardReader>
    <!--ro, opt, enum, ID card reader model, subType:string, desc:"iDR210", "DS-K1F110-I", "DS-K1F1110-B", "DS-K1F1110-AB", "none", "DS-K1F1001-I(USB)",
"DS-K1F1002-I(USB)"-->iDR210
  </idCardReader>
  <camera>
    <!--ro, opt, enum, camera, subType:string, desc:camera model: C270,DS-2CS5432B-S-->C270
  </camera>
  <fingerPrintModule>
    <!--ro, opt, enum, fingerprint module type, subType:string, desc:"ALIWARD", ""-->ALIWARD
  </fingerPrintModule>
  <videoStorageTime>
    <!--ro, opt, int, time for saving video, range:[0,10], desc:unit: second-->1
  </videoStorageTime>
  <faceContrastThreshold>
    <!--ro, opt, int, face picture comparison threshold, range:[0,100]-->1
  </faceContrastThreshold>
  <twoDimensionCode>
    <!--ro, opt, enum, whether to enable QR code recognition, subType:string, desc:"enable", "disable"-->enable
  </twoDimensionCode>
  <blackListCheck>
    <!--ro, opt, enum, whether to enable blocklist verification, subType:string, desc:"enable", "disable"-->enable
  </blackListCheck>
```

```
  <idCardCheckCenter>
    <!--ro, opt, enum, ID card comparison mode, subType:string, desc:“local” (compare with ID card of local storage), “server” (compare with ID card of remote server storage)-->local
  </idCardCheckCenter>
  <faceAlgorithm>
    <!--ro, opt, enum, face picture algorithm, subType:string, desc:"DeepLearn" (deep learning algorithm), "Tradition" (third-party algorithm)-->DeepLearn
  </faceAlgorithm>
  <comNo>
    <!--ro, opt, int, COM No., range:[1,9]-->1
  </comNo>
  <memoryLearning>
    <!--ro, opt, enum, whether to enable learning and memory function, subType:string, desc:“enable”, “disable”-->enable
  </memoryLearning>
  <saveCertifiedImage>
    <!--ro, opt, enum, whether to enable saving authenticated picture, subType:string, desc:“enable”, “disable”-->enable
  </saveCertifiedImage>
  <MCUVersion>
    <!--ro, opt, string, MCU version information-->test
  </MCUVersion>
  <usbOutput>
    <!--ro, opt, enum, whether to enable USB output of ID card reader, subType:string, desc:“enable”, “disable”-->enable
  </usbOutput>
  <serialOutput>
    <!--ro, opt, enum, whether to enable serial port output of ID card reader, subType:string, desc:“enable”, “disable”-->enable
  </serialOutput>
  <readInfoOfCard>
    <!--ro, opt, enum, set content to be read from CPU card, subType:string, desc:“serialNo” (read serial No.), “file” (read file)-->serialNo
  </readInfoOfCard>
  <workMode>
    <!--ro, opt, enum, authentication mode, subType:string, desc:“passMode”, “accessControlMode”-->passMode
  </workMode>
  <ecoMode>
    <!--ro, opt, object, ECO mode-->
    <eco>
      <!--ro, opt, enum, whether to enable ECO mode, subType:string, desc:“enable”, “disable”-->enable
    </eco>
    <faceMatchThreshold1>
      <!--ro, opt, int, 1V1 face picture comparison threshold of ECO mode, range:[0,100]-->1
    </faceMatchThreshold1>
    <faceMatchThresholdN>
      <!--ro, opt, int, 1:N face picture comparison threshold of ECO mode, range:[0,100]-->1
    </faceMatchThresholdN>
    <changeThreshold>
      <!--ro, opt, int, switching threshold of ECO mode, range:[0,8], desc:switching threshold of ECO mode,which is between 0 and 8-->0
    </changeThreshold>
    <maskFaceMatchThresholdN>
      <!--ro, opt, int, 1:N face picture (face with mask and normal background picture) comparison threshold of ECO mode, range:[0,100]-->1
    </maskFaceMatchThresholdN>
    <maskFaceMatchThreshold1>
      <!--ro, opt, int, 1:1 face picture (face with mask and normal background picture) comparison threshold of ECO mode, range:[0,100]-->1
    </maskFaceMatchThreshold1>
    <alwaysInfrared>
      <!--ro, opt, bool, whether to enable infrared recognition, desc:if this node exists and changeThreshold is 8, it indicates that the device will not always enable the infrared recognition-->true
    </alwaysInfrared>
    <ageFaceMatchList>
      <!--ro, opt, array, list of different age groups in ECO mode, subType:object, range:[0,5], desc:list of different age groups in ECO mode-->
      <ageFaceMatch>
        <!--ro, opt, object, age group matching in ECO mode-->
        <ageLevel>
          <!--ro, opt, enum, age groups, subType:int, desc:0 (child who are 0~6 years old), 1 (teenager who are 7~17years old), 2 (youth and prime who are 18~40 years old), 3 (middle age who are 41~65 years old), 4 (elderly who are more than 66 years old)-->1
        </ageLevel>
        <ageFaceMatchThreshold1>
          <!--ro, opt, int, matching threshold when authenticating via age groups in ECO mode (1:1), range:[0,100]-->1
        </ageFaceMatchThreshold1>
        <ageFaceMatchThresholdN>
          <!--ro, opt, int, matching threshold when authenticating via age groups in ECO mode (1:N), range:[0,100]-->1
        </ageFaceMatchThresholdN>
      </ageFaceMatch>
    </ageFaceMatchList>
  </ecoMode>
  <readCardRule>
    <!--ro, opt, enum, card No. setting rule, subType:string, desc:"wiegand26", "wiegand34”-->wiegand26
  </readCardRule>
  <enableScreenOff>
    <!--ro, opt, bool, whether the device enters the sleep mode when there is no operation after the configured sleep time-->true
  </enableScreenOff>
  <screenOffTimeout>
    <!--ro, opt, int, sleep time, range:[0,3600], unit:s, dep:or,{$.IdentityTerminal.enableScreenOff,eq,true}, desc:unit: second-->1
  </screenOffTimeout>
  <enableScreensaver>
    <!--ro, opt, bool, whether to enable the screen saver function-->true
  </enableScreensaver>
  <faceModuleVersion>
    <!--ro, opt, string, face recognition module version, range:[1,32]-->test
  </faceModuleVersion>
  <showMode>
    <!--ro, opt, enum, display mode, subType:string, desc:"concise" (simple mode,only the authentication result will be displayed), "normal" (normal mode). The default mode is normal mode. If this node does not exist, the default mode is normal mode-->concise
  </showMode>
  <popUpPreviewWindow>
    <!--ro, opt, bool, whether to pop up live view window, dep:or,{$.IdentityTerminal.showMode,eq,advertising}-->true
```

```
  </popUpPreviewWindow>
  <needDeviceCheck>
    <!--ro, opt, bool, whether it need device check in permission free mode, dep:or,{$.IdentityTerminal.workMode,eq,passMode}-->true
  </needDeviceCheck>
  <previewShowTime>
    <!--ro, opt, int, display duration in live view, range:[1,99], unit:s, dep:or,{$.IdentityTerminal.popUpPreviewWindow,eq,true}-->1
  </previewShowTime>
  <screensaverTimeout>
    <!--ro, opt, int, range:[0,3600], unit:s, dep:or,{$.IdentityTerminal.enableScreensaver,eq,true}-->1
  </screensaverTimeout>
  <screensaverDuration>
    <!--ro, opt, int, range:[0,3600], unit:s, dep:or,{$.IdentityTerminal.enableScreensaver,eq,true}-->1
  </screensaverDuration>
  <standbyTimeout>
    <!--ro, opt, int, range:[30,1800], unit:s-->30
  </standbyTimeout>
  <advertisingDisplayType>
    <!--ro, opt, enum, subType:string, dep:or,{$.IdentityTerminal.showMode,eq,advertising}-->full
  </advertisingDisplayType>
</IdentityTerminal>
```

10.3.7.11 Set the parameters of face recognition terminal

Request URL

PUT /ISAPI/AccessControl/IdentityTerminal

Query Parameter

None

Request Message

```
<?xml version="1.0" encoding="UTF-8"?>
<IdentityTerminal xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--req, object, the parameters of face recognition terminal, attr:version{req, string, protocolVersion}-->
  <terminalMode>
    <!--opt, enum, terminal mode, subType:string, desc:"authMode" (authentication mode), "registerMode" (registration mode)-->authMode
  </terminalMode>
  <idCardReader>
    <!--opt, enum, ID card reader, subType:string, desc:“iDR210”, “DS-K1F110-I”, “DS-K1F1110-B”, “DS-K1F1110-AB”, “none”, “DS-K1F1001-I (USB)”, “DS-K1F1002-
I (USB)”-->iDR210
  </idCardReader>
  <camera>
    <!--opt, enum, camera, subType:string, desc:"C270", "DS-2CS5432B-S"-->C270
  </camera>
  <fingerPrintModule>
    <!--opt, enum, fingerprint module, subType:string, desc:fingerprint module type: ALIWARD,-->ALIWARD
  </fingerPrintModule>
  <videoStorageTime>
    <!--opt, int, time for saving video, range:[0,10], desc:unit: second-->1
  </videoStorageTime>
  <faceContrastThreshold>
    <!--opt, int, face picture comparison threshold, range:[0,100]-->1
  </faceContrastThreshold>
  <twoDimensionCode>
    <!--opt, enum, whether to enable QR code recognition, subType:string, desc:“enable”, “disable”-->enable
  </twoDimensionCode>
  <blackListCheck>
    <!--opt, enum, whether to enable blocklist verification, subType:string, desc:“enable”, “disable”-->enable
  </blackListCheck>
  <idCardCheckCenter>
    <!--opt, enum, ID card comparison mode, subType:string, desc:“local” (compare with ID card of local storage), “server” (compare with ID card of remote
server storage)-->local
  </idCardCheckCenter>
  <faceAlgorithm>
    <!--opt, enum, face picture algorithm, subType:string, desc:"DeepLearn" (deep learning algorithm), "Tradition" (third-party algorithm)-->DeepLearn
  </faceAlgorithm>
  <comNo>
    <!--opt, int, COM No., range:[1,9]-->1
  </comNo>
  <memoryLearning>
    <!--opt, enum, whether to enable learning and memory function, subType:string, desc:“enable”, “disable”-->enable
  </memoryLearning>
  <saveCertifiedImage>
    <!--opt, enum, whether to enable saving authenticated picture, subType:string, desc:“enable”, “disable”-->enable
  </saveCertifiedImage>
  <MCUVersion>
    <!--opt, string, MCU Version-->test
  </MCUVersion>
  <usbOutput>
    <!--opt, enum, whether to enable USB output of ID card reader, subType:string, desc:“enable”, “disable”-->enable
  </usbOutput>
  <serialOutput>
    <!--opt, enum, whether to enable serial port output of ID card reade, subType:string, desc:“enable”, “disable”-->enable
  </serialOutput>
  <readInfoOfCard>
    <!--opt, enum, set content to be read from CPU card, subType:string, desc:“serialNo” (read serial No.), “file” (read file)-->serialNo
  </readInfoOfCard>
```

```
<workMode>
  <!--opt, enum, authentication mode, subType:string, desc:"passMode", "accessControlMode"-->passMode
</workMode>
<ecoMode>
  <!--opt, object, ECO mode-->
  <eco>
    <!--opt, enum, whether to enable ECO mode, subType:string, desc:"enable", "disable"-->enable
  </eco>
  <faceMatchThreshold1>
    <!--opt, int, 1V1 face picture comparison threshold of ECO mode, range:[0,100]-->1
  </faceMatchThreshold1>
  <faceMatchThresholdN>
    <!--opt, int, 1VN face picture comparison threshold of ECO mode, range:[0,100]-->1
  </faceMatchThresholdN>
  <changeThreshold>
    <!--opt, int, ECO mode threshold, range:[0,8], desc:switching threshold of ECO mode,which is between 0 and 8-->0
  </changeThreshold>
  <maskFaceMatchThresholdN>
    <!--opt, int, 1:N face picture (face with mask and normal background picture) comparison threshold of ECO mode, range:[0,100]-->1
  </maskFaceMatchThresholdN>
  <maskFaceMatchThreshold1>
    <!--opt, int, 1:1 face picture (face with mask and normal background picture) comparison threshold of ECO mode, range:[0,100]-->1
  </maskFaceMatchThreshold1>
  <alwaysInfrared>
    <!--opt, bool, whether to enable infrared recognition, desc:if this node exists and changeThreshold is 8, it indicates that the device will not always enable the infrared recognition-->true
  </alwaysInfrared>
  <ageFaceMatchList>
    <!--opt, array, list of different age groups in ECO mode, subType:object, range:[0,5], desc:before set this node, age group matching should be enabled, see URL: PUT /ISAPI/AccessControl/FaceCompareCond, and ageFaceMatchEnabled should be true-->
    <ageFaceMatch>
      <!--opt, object, age group matching in ECO mode-->
      <ageLevel>
        <!--opt, enum, age groups, subType:int, desc:0 (child who are 0~6 years old), 1 (teenager who are 7~17years old), 2 (youth and prime who are 18~40 years old), 3 (middle age who are 41~65 years old), 4 (elderly who are more than 66 years old)-->1
      </ageLevel>
      <ageFaceMatchThreshold1>
        <!--opt, int, matching threshold when authenticating via age groups in ECO mode (1:1), range:[0,100]-->1
      </ageFaceMatchThreshold1>
      <ageFaceMatchThresholdN>
        <!--opt, int, matching threshold when authenticating via age groups in ECO mode (1:N), range:[0,100]-->1
      </ageFaceMatchThresholdN>
    </ageFaceMatch>
  </ageFaceMatchList>
</ecoMode>
<readCardRule>
  <!--opt, enum, card No. setting rule, subType:string, desc:"wiegand26", "wiegand34"-->wiegand26
</readCardRule>
<enableScreenOff>
  <!--opt, bool, whether the device enters the sleep mode when there is no operation after the configured sleep time-->true
</enableScreenOff>
<screenOffTimeout>
  <!--opt, int, sleep time, range:[0,3600], unit:s, dep:or,{$.IdentityTerminal.enableScreenOff,eq,true}, desc:unit: second-->1
</screenOffTimeout>
<enableScreensaver>
  <!--opt, bool, whether to enable the screen saver function-->true
</enableScreensaver>
<faceModuleVersion>
  <!--opt, string, face recognition module version, range:[1,32]-->test
</faceModuleVersion>
<showMode>
  <!--opt, enum, display mode, subType:string, desc:simple mode indicates that the device displays authentication results exclude employee No., name, etc.; the device applies normal mode by default; advertisement mode indicates that the device displays both advertisement and authentication results; meeting mode indicates that the device displays check-in page of the conference; custom mode indicates that the device displays layout of the interface customized by users; "concise"-simple mode, "normal"-normal mode (default), "advertising"-advertisement mode, "meeting"-meeting mode, "selfDefine"-custom mode-->concise
</showMode>
<popUpPreviewWindow>
  <!--opt, bool, whether to pop up live view window, dep:or,{$.IdentityTerminal.showMode,eq,advertising}-->true
</popUpPreviewWindow>
<needDeviceCheck>
  <!--opt, bool, whether it need device check in permission free mode, dep:or,{$.IdentityTerminal.workMode,eq,passMode}-->true
</needDeviceCheck>
<previewShowTime>
  <!--opt, int, display duration in live view, range:[1,99], unit:s, dep:or,{$.IdentityTerminal.popUpPreviewWindow,eq,true}-->1
</previewShowTime>
<screensaverTimeout>
  <!--opt, int, range:[0,3600], unit:s, dep:or,{$.IdentityTerminal.enableScreensaver,eq,true}-->1
</screensaverTimeout>
<screensaverDuration>
  <!--opt, int, range:[0,3600], unit:s, dep:or,{$.IdentityTerminal.enableScreensaver,eq,true}-->1
</screensaverDuration>
<standbyTimeout>
  <!--opt, int, range:[30,1800], unit:s-->30
</standbyTimeout>
<advertisingDisplayType>
  <!--opt, enum, subType:string, dep:or,{$.IdentityTerminal.showMode,eq,advertising}-->full
</advertisingDisplayType>
</IdentityTerminal>
```


Response Message

```
<?xml version="1.0" encoding="UTF-8"?>
<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response status, attr:version{req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL, desc:request URL-->test
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:int, desc:read-only,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-->1
  </statusCode>
  <statusString>
    <!--ro, req, enum, read-only,status description, subType:string, desc:“OK” (succeeded), “Device Busy”, “Device Error”, “Invalid Operation”, “Invalid XML Format”, “Invalid XML Content”, “Reboot” (reboot device)-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, sub status code, desc:read-only,describe the error reason in detail-->test
  </subStatusCode>
</ResponseStatus>
```

10.3.7.12 Get configuration capability of face recognition terminal

Request URL

GET /ISAPI/AccessControl/IdentityTerminal/capabilities

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>
<IdentityTerminal xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, parameters of face recognition terminal, attr:version{req, string, protocolVersion}-->
  <terminalMode opt="authMode,registerMode">
    <!--ro, opt, enum, terminal mode, subType:string, attr:opt{req, string}, desc:"authMode” (authentication mode), "registerMode” (registration mode)-->authMode
  </terminalMode>
  <idCardReader opt="iDR210,DS-K1F110-I,DS-K1F1110-B,DS-K1F1110-AB,none ">
    <!--ro, opt, enum, ID card reader model, subType:string, attr:opt{req, string}, desc:“iDR210”, “DS-K1F110-I”, “DS-K1F1110-B”, “DS-K1F1110-AB”, “none”, “DS-K1F1001-I (USB)”, “DS-K1F1002-I (USB)”-->iDR210
  </idCardReader>
  <camera opt="C270,DS-2CS5432B-S">
    <!--ro, opt, enum, camera, subType:string, attr:opt{req, string}, desc:"C270", "DS-2CS5432B-S"-->C270
  </camera>
  <fingerPrintModule opt="ALIWARD,">
    <!--ro, opt, enum, fingerprint module, subType:string, attr:opt{req, string}, desc:fingerprint module-->ALIWARD
  </fingerPrintModule>
  <videoStorageTime min="0" max="10">
    <!--ro, opt, int, time for saving video (unit: second), range:[0,10], attr:min{req, int},max{req, int}, desc:unit: second-->1
  </videoStorageTime>
  <faceContrastThreshold min="0" max="100">
    <!--ro, opt, int, face picture comparison threshold, range:[0,100], attr:min{req, int},max{req, int}-->1
  </faceContrastThreshold>
  <twoDimensionCode opt="enable,disable">
    <!--ro, opt, enum, whether to enable QR code recognition, subType:string, attr:opt{req, string}, desc:“enable”, “disable”-->enable
  </twoDimensionCode>
  <blackListCheck opt="enable,disable">
    <!--ro, opt, enum, whether to enable blocklist verification, subType:string, attr:opt{req, string}, desc:“enable”, “disable”-->enable
  </blackListCheck>
  <idCardCheckCenter opt="local,server">
    <!--ro, opt, enum, ID card comparison mode, subType:string, attr:opt{req, string}, desc:“local” (compare with ID card of local storage), “server” (compare with ID card of remote server storage)-->local
  </idCardCheckCenter>
  <faceAlgorithm opt="DeepLearn,Tradition">
    <!--ro, opt, enum, face picture algorithm, subType:string, attr:opt{req, string}, desc:"DeepLearn" (deep learning algorithm), "Tradition" (third-party algorithm)-->DeepLearn
  </faceAlgorithm>
  <comNo min="1" max="9">
    <!--ro, opt, int, COM No., range:[1,9], attr:min{req, int},max{req, int}-->1
  </comNo>
  <memoryLearning opt="enable,disable">
    <!--ro, opt, enum, whether to enable learning and memory function, subType:object, attr:opt{req, string}, desc:“enable”, “disable”-->enable
  </memoryLearning>
  <saveCertifiedImage opt="enable,disable">
    <!--ro, opt, enum, whether to enable saving authenticated picture, subType:string, attr:opt{req, string}, desc:“enable”, “disable”-->enable
  </saveCertifiedImage>
  <MCUVersion min="1" max="10">
    <!--ro, opt, string, MCU version information, attr:min{req, int},max{req, int}-->test
  </MCUVersion>
  <usbOutput opt="enable,disable">
    <!--ro, opt, enum, whether to enable USB output of ID card reader, subType:string, attr:opt{req, string}, desc:“enable”, “disable”-->enable
  </usbOutput>
```

```
<serialOutput opt="enable,disable">
  <!--ro, opt, enum, whether to enable serial port output of ID card reader, subType:string, attr:opt{req, string}, desc:“enable”, “disable”-->enable
</serialOutput>
<readInfoOfCard opt="serialNo,file">
  <!--ro, opt, enum, set content to be read from CPU card, subType:string, attr:opt{req, string}, desc:"serialNo" (read the serial No.), "file" (read the file)-->serialNo
</readInfoOfCard>
<workMode opt="passMode,accessControlMode">
  <!--ro, opt, enum, authentication mode, subType:string, attr:opt{req, string}, desc:authentication mode-->passMode
</workMode>
<ecoMode>
  <!--ro, opt, object, ECO mode-->
  <eco opt="enable,disable">
    <!--ro, opt, enum, whether to enable ECO mode, subType:string, attr:opt{req, string}, desc:“enable”, “disable”-->enable
  </eco>
  <faceMatchThreshold1 min="0" max="100">
    <!--ro, opt, int, 1V1 face picture comparison threshold of ECO mod, range:[0,100], attr:min{req, int},max{req, int}-->1
  </faceMatchThreshold1>
  <faceMatchThresholdN min="0" max="100">
    <!--ro, opt, int, 1:N face picture comparison threshold of ECO mode, range:[0,100], attr:min{req, int},max{req, int}-->1
  </faceMatchThresholdN>
  <changeThreshold min="0" max="8">
    <!--ro, opt, int, switching threshold of ECO mode, range:[0,8], attr:min{req, int},max{req, int}, desc:switching threshold of ECO mode,which is between 0 and 8-->0
  </changeThreshold>
  <maskFaceMatchThresholdN min="0" max="100">
    <!--ro, opt, int, 1:N face picture (face with mask and normal background picture) comparison threshold of ECO mode, range:[0,100], attr:min{req, int},max{req, int}-->1
  </maskFaceMatchThresholdN>
  <maskFaceMatchThreshold1 min="0" max="100">
    <!--ro, opt, int, 1:1 face picture (face with mask and normal background picture) comparison threshold of ECO mode, range:[0,100], attr:min{req, int},max{req, int}-->1
  </maskFaceMatchThreshold1>
  <alwaysInfrared opt="true,false">
    <!--ro, opt, bool, whether to enable infrared recognition, attr:opt{req, string}, desc:if this node exists and changeThreshold is 8, it indicates that the device will not always enable the infrared recognition-->true
  </alwaysInfrared>
  <ageFaceMatchList size="5">
    <!--ro, opt, array, list of different age groups in ECO mode, subType:object, range:[0,5], attr:size{req, int}-->
    <ageFaceMatch>
      <!--ro, opt, object, age group matching in ECO mode-->
      <ageLevel opt="0,1,2,3,4">
        <!--ro, opt, enum, age groups, subType:int, attr:opt{req, string}, desc:0 (child who are 0~6 years old), 1 (teenager who are 7~17years old), 2 (youth and prime who are 18~40 years old), 3 (middle age who are 41~65 years old), 4 (elderly who are more than 66 years old)-->1
      </ageLevel>
      <ageFaceMatchThreshold1 min="0" max="100">
        <!--ro, opt, int, matching threshold when authenticating via age groups in ECO mode (1:1), range:[0,100], attr:min{req, int},max{req, int}-->1
      </ageFaceMatchThreshold1>
      <ageFaceMatchThresholdN min="0" max="100">
        <!--ro, opt, int, matching threshold when authenticating via age groups in ECO mode (1:N), range:[0,100], attr:min{req, int},max{req, int}-->1
      </ageFaceMatchThresholdN>
    </ageFaceMatch>
  </ageFaceMatchList>
</ecoMode>
<readCardRule opt="wiegand26,wiegand34">
  <!--ro, opt, enum, card No. setting rule, subType:string, attr:opt{req, string}, desc:card No. setting rule: "wiegand26","wiegand34"-->wiegand26
</readCardRule>
<enableScreenOff opt="true,false">
  <!--ro, opt, bool, whether the device enters the sleep mode when there is no operation after the configured sleep time, attr:opt{req, string}-->true
</enableScreenOff>
<screenOffTimeout min="0" max="3600">
  <!--ro, opt, int, sleep time, range:[0,3600], unit:s, attr:min{req, int},max{req, int}-->1
</screenOffTimeout>
<enableScreensaver opt="true,false">
  <!--ro, opt, bool, whether to enable the screen saver function, attr:opt{req, string}-->true
</enableScreensaver>
<faceModuleVersion min="0" max="10">
  <!--ro, opt, string, face recognition module version, attr:min{req, int},max{req, int}-->test
</faceModuleVersion>
<showMode opt="concise,normal,advertising,meeting,selfDefine,boxStatus,clock">
  <!--ro, opt, enum, display mode, subType:string, attr:opt{req, string}, desc:"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-->concise
</showMode>
<popUpPreviewWindow opt="true,false">
  <!--ro, opt, bool, whether to pop up live view window, dep:or,{$.IdentityTerminal.showMode,eq,advertising}, attr:opt{req, string}-->true
</popUpPreviewWindow>
<needDeviceCheck opt="true,false">
  <!--ro, opt, bool, whether it need device check in permission free mode, dep:or,{$.IdentityTerminal.workMode,eq,passMode}, attr:opt{req, string}-->true
</needDeviceCheck>
<previewShowTime min="1" max="99">
  <!--ro, opt, int, display duration in live view, range:[1,99], unit:s, attr:min{req, int},max{req, int}-->1
</previewShowTime>
<screensaverTimeout min="0" max="3600">
  <!--ro, opt, int, range:[0,3600], unit:s, attr:min{req, int},max{req, int}-->1
</screensaverTimeout>
<screensaverDuration min="0" max="3600">
  <!--ro, opt, int, range:[0,3600], unit:s, attr:min{req, int},max{req, int}-->1
</screensaverDuration>
<standbyTimeout min="30" max="1800">
  <!--ro, opt, int, range:[30,1800], unit:s, attr:min{req, int},max{req, int}-->30
</standbyTimeout>
<advertisingDisplayType opt="full,split">
  <!--ro, opt, enum, subType:string, attr:opt{req, string}-->full
</advertisingDisplayType>
```

```
</IdentityTerminal>
```

10.3.7.13 Set the parameters of M1 card encryption verification

Request URL

PUT /ISAPI/AccessControl/M1CardEncryptCfg

Query Parameter

None

Request Message

```
<?xml version="1.0" encoding="UTF-8"?>

<M1CardEncryptCfg xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--req, object, configuration capability of the M1 card encryption verification, attr:version{req, string, protocolVersion}-->
  <enable>
    <!--req, bool, whether to enable the function-->true
  </enable>
  <sectionID>
    <!--req, int, sector ID, desc:only one sector can be configured at a time-->1
  </sectionID>
</M1CardEncryptCfg>
```

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response status, attr:version{req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL, desc:request URL-->test
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:string, desc:0 (OK), 1 (OK), 2 (Device Busy), 3 (Device Error), 4 (Invalid Operation), 5 (Invalid XML Format), 6
(Invalid XML Content), 7 (Reboot Required)-->1
  </statusCode>
  <statusString>
    <!--ro, req, enum, status description, subType:string, desc:"OK,Device Busy,Device Error,Invalid Operation,Invalid XML Format,Invalid XML
Content,Reboot"-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, sub status code, desc:sub status code description-->test
  </subStatusCode>
</ResponseStatus>
```


10.3.7.14 Get the configuration parameters of M1 card encryption verification

Request URL

GET /ISAPI/AccessControl/M1CardEncryptCfg

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<M1CardEncryptCfg xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, configuration capability of the M1 card encryption verification, attr:version{req, string, protocolVersion}-->
  <enable>
    <!--ro, req, bool, whether to enable the function-->true
  </enable>
  <sectionID>
    <!--ro, req, int, sector ID, desc:sector ID,only one sector can be configured at a time-->1
  </sectionID>
</M1CardEncryptCfg>
```

10.3.7.15 Get the configuration capability of the M1 card encryption verification

Request URL

GET /ISAPI/AccessControl/M1CardEncryptCfg/capabilities

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<M1CardEncryptCfg xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, configuration capability of the M1 card encryption verification, attr:version{req, string, protocolVersion}-->
  <enable opt="true,false">
    <!--ro, req, bool, whether to enable, attr:opt{req, string}-->true
  </enable>
  <sectionID min="0" max="100">
    <!--ro, req, int, sector ID, range:[0,100], attr:min{req, int},max{req, int}-->1
  </sectionID>
</M1CardEncryptCfg>
```

10.3.7.16 Get the capability of Wiegand parameters

Request URL

GET /ISAPI/AccessControl/WiegandCfg/capabilities

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<WiegandCfg xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, configuration of Wiegand parameters, attr:version{req, string, protocolVersion}-->
  <wiegandNo min="1" max="4" opt="1,4">
    <!--ro, req, int, Wiegand interface No., range:[0,4], step:1, attr:min{opt, int},max{opt, int},opt{opt, string}, desc:Wiegand interface No.-->1
  </wiegandNo>
  <communicateDirection opt="receive,send">
    <!--ro, req, enum, communication direction, subType:string, attr:opt{req, string}, desc:"receive", "send"-->receive
  </communicateDirection>
  <wiegandMode
opt="wiegand26,wiegand34,wiegand27,wiegand35,Corporate1000_35,Corporate1000_48,H10302_37,H10304_37,wiegand_26CSN,H103130_32CSN,wiegand_56CSN,wiegand_58">
    <!--ro, opt, enum, Wiegand mode, subType:string, attr:opt{req, string}, desc:Wiegand mode-->wiegand26
  </wiegandMode>
  <inputWiegandMode>
    <!--ro, opt, enum, subType:string, dep:or,{$.WiegandCfg.communicateDirection,eq,receive}-->wiegand26
  </inputWiegandMode>
  <signalInterval min="1" max="20">
    <!--ro, opt, int, it is between 1 and 20,unit: ms, range:[1,20], attr:min{req, int},max{req, int}, desc:it is between 1 and 20,unit: ms-->1
  </signalInterval>
  <enable opt="true,false">
    <!--ro, opt, bool, whether to enable Wiegand parameters, attr:opt{req, string}-->true
  </enable>
  <pulseDuration min="1" max="10">
    <!--ro, opt, int, pulse duration, range:[1,10], attr:min{req, int},max{req, int}, desc:pulse duration-->1
  </pulseDuration>
  <facilityCodeEnabled opt="true,false">
    <!--ro, opt, bool, whether to enable facilityCode, attr:opt{req, string}-->true
  </facilityCodeEnabled>
  <facilityCode min="0" max="65535">
    <!--ro, opt, int, range:[0,65535], dep:and,{$.WiegandCfg.facilityCodeEnabled,eq,true}, attr:min{req, int, range:[0,65535]},max{req, int, range:
[0,65535]}-->1
  </facilityCode>
  <dataType opt="employeeNo,cardNo">
    <!--ro, opt, enum, data type, subType:string, dep:or,{$.WiegandCfg.wiegandMode,eq,send}, attr:opt{req, string}, desc:data type-->employeeNo
  </dataType>
</WiegandCfg>
```

10.3.7.17 Set Wiegand parameters

Request URL

PUT /ISAPI/AccessControl/WiegandCfg/wiegandNo/<wiegandID>

Query Parameter

Parameter Name	Parameter Type	Description
wiegandID	string	--

Request Message

```
<?xml version="1.0" encoding="UTF-8"?>

<WiegandCfg xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--wo, req, object, Wiegand parameters, attr:version{req, string, protocolVersion}-->
  <communicateDirection>
    <!--wo, req, enum, communication direction, subType:string, desc:"receive", "send"-->receive
  </communicateDirection>
  <wiegandMode>
    <!--wo, opt, enum, Wiegand mode, subType:string, dep:or,{$.WiegandCfg.communicateDirection,eq,send}, desc:"wiegand26", "wiegand34", "wiegand27",
"wiegand35", "Corporate1000_35", "Corporate1000_48", "H10302_37", "H10304_37", "wiegand_26CSN", "H103130_32CSN", "wiegand_56CSN", "wiegand_58"-->wiegand26
  </wiegandMode>
  <inputWiegandMode>
    <!--wo, opt, enum, subType:string, dep:or,{$.WiegandCfg.communicateDirection,eq,receive}-->wiegand26
  </inputWiegandMode>
  <signalInterval>
    <!--wo, opt, int, it is between 1 and 20,unit: ms, range:[1,20], desc:unit: ms-->1
  </signalInterval>
  <enable>
    <!--wo, opt, bool, whether to enable the function or not-->true
  </enable>
  <pulseDuration>
    <!--wo, opt, int, range:[1,10]-->1
  </pulseDuration>
  <facilityCodeEnabled>
    <!--opt, bool-->true
  </facilityCodeEnabled>
  <facilityCode>
    <!--opt, int, range:[0,65535], dep:and,{$.WiegandCfg.facilityCodeEnabled,eq,true}-->1
  </facilityCode>
  <dataType>
    <!--opt, enum, data type, subType:string, dep:or,{$.WiegandCfg.wiegandMode,eq,send}, desc:data type-->employeeNo
  </dataType>
</WiegandCfg>
```

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response message, attr:version{ro, req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL-->null
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:int, desc:0 (OK), 1 (OK), 2 (Device Busy), 3 (Device Error), 4 (Invalid Operation), 5 (Invalid XML Format), 6
(Invalid XML Content), 7 (Reboot Required)-->0
  </statusCode>
  <statusString>
    <!--ro, req, enum, status information, subType:string, desc:"OK" (succeeded), "Device Busy", "Device Error", "Invalid Operation", "Invalid XML Format",
"Invalid XML Content", "Reboot" (reboot device)-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, sub status code, which describes the error in details, desc:sub status code, which describes the error in details-->OK
  </subStatusCode>
</ResponseStatus>
```


10.3.7.18 Get Wiegand parameters

Request URL

GET /ISAPI/AccessControl/WiegandCfg/wiegandNo/<wiegandID>

Query Parameter

Parameter Name	Parameter Type	Description
wiegandID	string	--

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<WiegandCfg xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, Wiegand parameters, attr:version{req, string, protocolVersion}-->
  <communicateDirection>
    <!--ro, req, enum, communication direction, subType:string, desc:"receive", "send"-->receive
  </communicateDirection>
  <wiegandMode>
    <!--ro, opt, enum, Wiegand mode, subType:string, dep:or,{$.WiegandCfg.communicateDirection,eq,send}, desc:"wiegand26", "wiegand34", "wiegand27",
"wiegand35", "Corporate1000_35", "Corporate1000_48", "H10302_37", "H10304_37", "wiegand_26CSN", "H103130_32CSN", "wiegand_56CSN", "wiegand_58"-->wiegand26
  </wiegandMode>
  <inputWiegandMode>
    <!--ro, opt, enum, subType:string, dep:or,{$.WiegandCfg.communicateDirection,eq,receive}-->wiegand26
  </inputWiegandMode>
  <signalInterval>
    <!--ro, opt, int, Wiegand signal sending interval, range:[1,20], desc:unit: ms-->1
  </signalInterval>
  <enable>
    <!--ro, opt, bool, whether to enable the function or not-->true
  </enable>
  <pulseDuration>
    <!--ro, opt, int, range:[1,10]-->1
  </pulseDuration>
  <facilityCodeEnabled>
    <!--ro, opt, bool-->true
  </facilityCodeEnabled>
  <facilityCode>
    <!--ro, opt, int, range:[0,65535], dep:and,{$.WiegandCfg.facilityCodeEnabled,eq,true}-->1
  </facilityCode>
  <dataType>
    <!--ro, opt, enum, data type, subType:string, dep:or,{$.WiegandCfg.wiegandMode,eq,send}, desc:data type-->employeeNo
  </dataType>
</WiegandCfg>
```

10.3.8 Multi-Factor Authentication Management

10.3.8.1 Get the capability of clearing group parameters

Request URL

GET /ISAPI/AccessControl/ClearGroupCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "ClearGroupCfg": {
    /*ro, opt, object*/
        "ClearFlags": {
        /*ro, opt, object*/
            "groupCfg":  "true,false"
            /*ro, req, string, group parameters*/
        }
    }
}
```

10.3.8.2 Clear group parameters

Request URL

PUT /ISAPI/AccessControl/ClearGroupCfg?format=json

Query Parameter

None

Request Message

```
{
    "ClearGroupCfg": {
    /*opt, object, group parameters*/
        "ClearFlags": {
        /*opt, object*/
            "groupCfg":  true
            /*req, bool, whether to clear group parameters*/
        }
    }
}
```

Response Message

```
{
    "requestURL":  "test",
    /*ro, opt, string, URI*/
    "statusCode":  "test",
    /*ro, opt, string, status code*/
    "statusString":  "test",
    /*ro, opt, string, status description*/
    "subStatusCode":  "test",
    /*ro, opt, string, sub status code*/
    "errorCode":  1,
    /*ro, req, int, error code*/
    "errorMsg":  "ok"
    /*ro, req, string, error description*/
}
```

10.3.8.3 Get the group configuration parameters

Request URL

GET /ISAPI/AccessControl/GroupCfg/<groupID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
groupID	string	--

Request Message

None

Response Message

```
{
    "GroupCfg": {
    /*ro, opt, object, group parameters*/
        "enable":  true,
        /*ro, req, bool, whether to enable the function*/
        "ValidPeriodCfg": {
        /*ro, opt, object, validity period of the group*/
            "enable":  true,
            /*ro, req, bool, whether to enable validity period*/
            "beginTime":  "1970-01-01T00:00:00+08:00",
            /*ro, req, datetime, start time of the validity period (UTC time)*/
            "endTime":  "1970-01-01T00:00:00+08:00"
            /*ro, req, datetime, end time of the validity period (UTC time)*/
        },
        "groupName":  "test"
        /*ro, opt, string*/
    }
}
```

10.3.8.4 Set the group parameters

Request URL

PUT /ISAPI/AccessControl/GroupCfg/<groupID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
groupID	string	--

Request Message

```
{
    "GroupCfg": {
    /*opt, object, group parameters*/
        "enable":  true,
        /*req, bool, whether to enable the group*/
        "ValidPeriodCfg": {
        /*opt, object, validity period of the group*/
            "enable":  true,
            /*req, bool, whether to enable validity period*/
            "beginTime":  "1970-01-01T00:00:00+08:00",
            /*req, datetime, start time of the validity period (UTC time)*/
            "endTime":  "1970-01-01T00:00:00+08:00"
            /*req, datetime, end time of the validity period (UTC time)*/
        },
        "groupName":  "test"
        /*opt, string, group name*/
    }
}
```

Response Message

```
{
    "requestURL":  "test",
    /*ro, opt, string, request URL*/
    "statusCode":  "test",
    /*ro, opt, string, status code*/
    "statusString":  "test",
    /*ro, opt, string, status description*/
    "subStatusCode":  "test",
    /*ro, opt, string, sub status code*/
    "errorCode":  1,
    /*ro, req, int, error code*/
    "errorMsg":  "ok"
    /*ro, req, string, error information*/
}
```


10.3.8.5 Get the group configuration capability

Request URL

GET /ISAPI/AccessControl/GroupCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "GroupCfg": {
    /*ro, opt, object, group parameters*/
        "groupNo": {
        /*ro, opt, object, group No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  1
            /*ro, opt, int, the maximum value*/
        },
        "enable":  "true,false",
        /*ro, req, string, whether to enable the group*/
        "ValidPeriodCfg": {
        /*ro, req, object, whether to enable validity period parameters of the group*/
            "enable":  "true,false",
            /*ro, req, string, whether to enable validity period*/
            "beginTime": {
            /*ro, req, object, start time of the validity period (UTC time)*/
                "@min":  1,
                /*ro, opt, int, the minimum value*/
                "@max":  32
                /*ro, opt, int, the maximum value*/
            },
            "endTime": {
            /*ro, req, object, end time of the validity period (UTC time)*/
                "@min":  1,
                /*ro, opt, int, the minimum value*/
                "@max":  32
                /*ro, opt, int, the maximum value*/
            }
        },
        "groupName": {
        /*ro, opt, object, group name*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  32
            /*ro, opt, int, the maximum value*/
        }
    }
}
```

10.3.8.6 Get the configuration parameters of multi-factor authentication mode

Request URL

GET /ISAPI/AccessControl/MultiCardCfg/<doorID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
doorID	string	--

Request Message

None

Response Message

```
{
    "MultiCardCfg": {
    /*ro, opt, object*/
        "enable":  true,
        /*ro, req, bool, whether to enable multi-factor authentication*/
        "swipeIntervalTimeout":  10,
        /*ro, opt, int, timeout of swiping (authentication) interval*/
        "GroupCfg": [
        /*ro, opt, array, multi-factor authentication parameters, subType:object*/
            {
                "id":  1,
                /*ro, opt, int, multi-factor authentication No.*/
                "enable":  true,
                /*ro, opt, bool, whether to enable multi-factor authentication*/
                "enableOfflineVerifyMode":  true,
                /*ro, opt, bool, whether to enable verification mode when the access control device is offline (the super password will replace opening door
remotely)*/
                "templateNo":  1,
                /*ro, opt, int, schedule template No. to enable the multi-factor authentication*/
                "GroupCombination": [
                /*ro, opt, array, multi-factor authentication parameters, subType:object*/
                    {
                        "enable":  true,
                        /*ro, opt, bool, whether to enable multi-factor authentication*/
                        "memberNum":  3,
                        /*ro, opt, int, number of members swiping cards*/
                        "sequenceNo":  1,
                        /*ro, opt, int, serial No. of swiping cards of the multi-factor authentication group*/
                        "groupNo":  1
                        /*ro, opt, int, group No.*/
                    }
                ]
            }
        ]
    }
}
```

10.3.8.7 Set parameters of multi-factor authentication mode

Request URL

PUT /ISAPI/AccessControl/MultiCardCfg/<doorID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
doorID	string	--

Request Message

```
{
    "MultiCardCfg": {
    /*opt, object, parameters of multi-factor authentication mode*/
        "enable":  true,
        /*req, bool, whether to enable multi-factor authentication*/
        "swipeIntervalTimeout":  10,
        /*opt, int, timeout of swiping (authentication) interval*/
        "GroupCfg": [
        /*opt, array, multi-factor authentication parameters, subType:object*/
            {
                "id":  1,
                /*opt, int, multi-factor authentication No.*/
                "enable":  true,
                /*opt, bool, whether to enable multi-factor authentication*/
                "enableOfflineVerifyMode":  true,
                /*opt, bool, whether to enable verification mode when the access control device is offline (the super password will replace opening door
remotely)*/
                "templateNo":  1,
                /*opt, int, schedule template No. to enable the multi-factor authentication*/
                "GroupCombination": [
                /*opt, array, parameters of the multi-factor authentication group, subType:object*/
                    {
                        "enable":  true,
                        /*opt, bool, whether to enable multi-factor authentication*/
                        "memberNum":  3,
                        /*opt, int, number of members swiping cards*/
                        "sequenceNo":  1,
                        /*opt, int, serial No. of swiping cards of the multi-factor authentication group*/
                        "groupNo":  1
                        /*opt, int, group No.*/
                    }
                ]
            }
        ]
    }
}
```

Response Message

```
{
    "requestURL":  "test",
    /*ro, opt, string, URI*/
    "statusCode":  "test",
    /*ro, opt, string, status code*/
    "statusString":  "test",
    /*ro, opt, string, status description*/
    "subStatusCode":  "test",
    /*ro, opt, string, sub status code*/
    "errorCode":  1,
    /*ro, req, int, error code*/
    "errorMsg":  "ok"
    /*ro, req, string, error information*/
}
```


10.3.8.8 Get the configuration capability of multi-factor authentication mode

Request URL

GET /ISAPI/AccessControl/MultiCardCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "MultiCardCfg": {
    /*ro, opt, object*/
        "doorNo": {
        /*ro, opt, object, door No.*/
            "@min":  1,
            /*ro, opt, int, minimum value*/
            "@max":  256
            /*ro, opt, int, maximum value*/
        },
        "enable":  "true,false",
        /*ro, req, string, whether to enable multi-factor authentication*/
        "swipeIntervalTimeout": {
        /*ro, opt, object, timeout of swiping (authentication) interval*/
            "@min":  1,
            /*ro, opt, int, minimum value*/
            "@max":  255
            /*ro, opt, int, maximum value*/
        },
        "GroupCfg": {
        /*ro, opt, object, multi-factor authentication parameters*/
            "maxSize":  20,
            /*ro, opt, int, maximum value*/
            "id": {
            /*ro, opt, object, multi-factor authentication No.*/
                "@min":  1,
                /*ro, opt, int, minimum value*/
                "@max":  20
                /*ro, opt, int, maximum value*/
            },
            "enable":  "true,false",
            /*ro, opt, string, whether to enable multi-factor authentication*/
            "enableOfflineVerifyMode":  "true,false",
            /*ro, opt, string, whether to enable verification mode when the access control device is offline (the super password will replace opening door
remotely)*/
            "templateNo": {
            /*ro, opt, object, schedule template No. to enable the multi-factor authentication*/
                "@min":  1,
                /*ro, opt, int, minimum value*/
                "@max":  20
                /*ro, opt, int, maximum value*/
            },
            "GroupCombination": {
            /*ro, opt, object, parameters of the multi-factor authentication group*/
                "maxSize":  8,
                /*ro, opt, int, maximum value*/
                "enable":  "true,false",
                /*ro, opt, string, whether to enable multi-factor authentication*/
                "memberNum": {
                /*ro, opt, object, number of members swiping cards*/
                    "@min":  1,
                    /*ro, opt, int, minimum value*/
                    "@max":  20
                    /*ro, opt, int, maximum value*/
                },
                "sequenceNo": {
                /*ro, opt, object, serial No. of swiping cards of the multi-factor authentication group*/
                    "@min":  1,
                    /*ro, opt, int, minimum value*/
                    "@max":  8
                    /*ro, opt, int, maximum value*/
                },
                "groupNo": {
                /*ro, opt, object, group No.*/
                    "@min":  1,
                    /*ro, opt, int, minimum value*/
                    "@max":  20
                    /*ro, opt, int, the maximum value*/
                }
            }
        }
    }
}
```


10.3.9 Permission Schedules for Persons and Access Points

10.3.9.1 Set the holiday group parameters of the access permission control schedule

Request URL

PUT /ISAPI/AccessControl/UserRightHolidayGroupCfg/<holidayGroupID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
holidayGroupID	string	--

Request Message

```
{
    "UserRightHolidayGroupCfg": {
    /*req, object, the holiday group parameters of the access permission control schedule*/
        "enable": true,
        /*req, bool, whether to enable, desc:true (yes), false (no)*/
        "groupName": "test",
        /*req, string, holiday group name*/
        "holidayPlanNo": "1,3,5",
        /*req, string, holiday group schedule No., desc:holiday group schedule No.*/
        "operateType": "byTerminal",
        /*opt, enum, operation type, subType:string, desc:"byTerminal" (by terminal), "byOrg" (by organization), "byTerminalOrg" (by terminal
organization)*/
        "terminalNoList": [1, 2, 3, 4],
        /*opt, array, terminal ID list, subType:int, desc:this node is required when operation type is "byTerminal" or "byTerminalOrg"*/
        "orgNoList": [1, 2, 3, 4]
        /*opt, array, organization ID list, subType:int, desc:this node is required when operation type is "byOrg" or "byTerminalOrg"*/
    }
}
```

Response Message

```
{
    "requestURL": "test",
    /*ro, opt, string, URI*/
    "statusCode": "test",
    /*ro, opt, string, status code, desc:1 (succeeded). It is required when an error occurred*/
    "statusString": "test",
    /*ro, opt, string, status description, desc:"ok" (succeeded). It is required when an error occurred*/
    "subStatusCode": "test",
    /*ro, opt, string, sub status code, desc:"ok" (succeeded). It is required when an error occurred*/
    "errorCode": 1,
    /*ro, req, int, error code, desc:it is required when the value of statusCode is not 1, and it corresponds to subStatusCode*/
    "errorMsg": "ok"
    /*ro, req, string, error information, desc:this node is required when the value of statusCode is not 1*/
}
```


10.3.9.2 Get the holiday group configuration parameters of the access permission control schedule

Request URL

GET /ISAPI/AccessControl/UserRightHolidayGroupCfg/<holidayGroupID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
holidayGroupID	string	--

Request Message

None

Response Message

```
{
    "UserRightHolidayGroupCfg": {
    /*ro, req, object*/
        "enable": true,
        /*ro, req, bool, whether to enable, desc:true (yes), false (no)*/
        "groupName": "test",
        /*ro, req, string, holiday group name*/
        "holidayPlanNo": "1,3,5"
        /*ro, req, string, holiday group schedule No., desc:holiday group schedule No.*/
    }
}
```

10.3.9.3 Get the holiday group configuration capability of the access permission control

Request URL

GET /ISAPI/AccessControl/UserRightHolidayGroupCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "UserRightHolidayGroupCfg": {
    /*ro, req, object*/
        "groupNo": {
        /*ro, opt, object, holiday group No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  16
            /*ro, opt, int, the maximum value*/
        },
        "enable":  "true,false",
        /*ro, opt, string, whether it is enabled, desc:"true" (enabled), "false" (disabled)*/
        "groupName": {
        /*ro, opt, object, holiday group name*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  32
            /*ro, opt, int, the maximum value*/
        },
        "holidayPlanNo": {
        /*ro, opt, object, holiday group schedule No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  16
            /*ro, opt, int, the maximum value*/
        }
    }
}
```

10.3.9.4 Set the holiday schedule parameters of the access permission control

Request URL

PUT /ISAPI/AccessControl/UserRightHolidayPlanCfg/<holidayPlanID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
holidayPlanID	string	--

Request Message

```
{
    "UserRightHolidayPlanCfg": {
    /*req, object, the holiday schedule parameters of the access permission control*/
        "enable":  true,
        /*req, bool, whether to enable, desc:"true" (enable), "false" (disable)*/
        "beginDate":  "1970-01-01",
        /*req, date, start date of the holiday, desc:device local time*/
        "endDate":  "1970-01-01",
        /*req, date, end date of the holiday, desc:device local time*/
        "HolidayPlanCfg": [
        /*req, array, subType:object*/
            {
                "id":  1,
                /*req, int, time period No., range:[1,8], desc:it is between 1 and 8*/
                "enable":  true,
                /*req, bool, whether to enable, desc:"true" (enable), "false" (disable)*/
                "TimeSegment": {
                /*opt, object, time period*/
                    "beginTime":  "00:00:00",
                    /*req, time, start time of the time period, desc:device local time*/
                    "endTime":  "00:00:00"
                    /*req, time, end time of the time period, desc:device local time*/
                },
                "authenticationTimesEnabled":  true,
                /*opt, bool*/
                "authenticationTimes":  10
                /*opt, int, range:[1,255], step:1*/
            }
        ]
    }
}
```

Response Message

```
{
    "requestURL":  "test",
    /*ro, opt, string, URI*/
    "statusCode":  "test",
    /*ro, opt, string, status code, desc:1 (succeeded). It is required when an error occurred*/
    "statusString":  "test",
    /*ro, opt, string, status description, desc:"ok" (succeeded). It is required when an error occurred*/
    "subStatusCode":  "test",
    /*ro, opt, string, sub status code, desc:"ok" (succeeded). It is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:it is required when the value of statusCode is not 1, and it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, opt, string, error information, desc:this node is required when the value of statusCode is not 1*/
}
```


10.3.9.5 Get holiday schedule configuration parameters

Request URL

GET /ISAPI/AccessControl/UserRightHolidayPlanCfg/<holidayPlanID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
holidayPlanID	string	--

Request Message

None

Response Message

```
{
    "UserRightHolidayPlanCfg": {
    /*ro, req, object, holiday schedule configuration parameters*/
        "enable":  true,
        /*ro, req, bool, whether to enable, desc:true (enable), false (disable)*/
        "beginDate":  "1970-01-01",
        /*ro, req, date, start date of the holiday, desc:device local time*/
        "endDate":  "1970-01-01",
        /*ro, req, date, end date of the holiday, desc:device local time*/
        "HolidayPlanCfg": [
        /*ro, req, array, holiday schedule parameters, subType:object*/
            {
                "id":  1,
                /*ro, req, int, time period No., range:[1,8], desc:it is between 1 and 8*/
                "enable":  true,
                /*ro, req, bool, whether to enable, desc:true (enable), false (disable)*/
                "TimeSegment": {
                /*ro, req, object, time period*/
                    "beginTime":  "00:00:00",
                    /*ro, req, time, start time of the time period, desc:device local time*/
                    "endTime":  "00:00:00"
                    /*ro, req, time, end time of the time period, desc:device local time*/
                },
                "authenticationTimesEnabled":  true,
                /*ro, opt, bool*/
                "authenticationTimes":  10
                /*ro, opt, int, range:[1,255], step:1*/
            }
        ]
    }
}
```

10.3.9.6 Get the holiday schedule configuration capability of the access permission control

Request URL

GET /ISAPI/AccessControl/UserRightHolidayPlanCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "UserRightHolidayPlanCfg": {
    /*ro, req, object*/
        "planNo": {
        /*ro, opt, object, holiday schedule No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  16
            /*ro, opt, int, the maximum value*/
        },
        "enable":  "true,false",
        /*ro, opt, string, whether it is enabled, desc:"true" (enabled), "false" (disabled)*/
        "beginDate":  "1970-01-01",
        /*ro, opt, date, start date of the holiday, desc:(device local time)*/
        "endDate":  "1970-01-01",
        /*ro, opt, date, end date of the holiday, desc:(device local time)*/
        "HolidayPlanCfg": {
        /*ro, opt, object, holiday schedule parameter*/
            "maxSize":  8,
            /*ro, opt, int, the maximum value*/
            "id": {
            /*ro, opt, object, time period No.*/
                "@min":  1,
                /*ro, opt, int, the minimum value*/
                "@max":  8
                /*ro, opt, int, the maximum value*/
            },
            "enable":  "true,false",
            /*ro, opt, string, whether it is enabled, desc:"true" (enabled), "false" (disabled)*/
            "TimeSegment": {
            /*ro, opt, object, time period*/
                "beginTime":  "00:00:00",
                /*ro, opt, time, start time, desc:(device local time)*/
                "endTime":  "00:00:00",
                /*ro, opt, time, end time, desc:(device local time)*/
                "validUnit":  "minute"
                /*ro, opt, enum, time accuracy, subType:string, desc:"hour", "minute", "second". If this node is not returned, the default time accuracy is
"minute"*/
            },
            "authenticationTimesEnabled": {
            /*ro, opt, object*/
                "@opt": [true, false]
                /*ro, req, array, subType:bool*/
            },
            "authenticationTimes": {
            /*ro, opt, object*/
                "@min":  1,
                /*ro, req, int, range:[1,255], step:1*/
                "@max":  255
                /*ro, req, int, range:[1,255], step:1*/
            }
        }
    }
}
```


10.3.9.7 Get the schedule template configuration parameters of the access permission control

Request URL

GET /ISAPI/AccessControl/UserRightPlanTemplate/<planTemplateID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
planTemplateID	string	--

Request Message

None

Response Message

```
{
    "UserRightPlanTemplate": {
    /*ro, req, object*/
        "enable":  true,
        /*ro, req, bool, whether to enable, desc:true (yes), false (no)*/
        "templateName":  "test",
        /*ro, req, string, template name*/
        "weekPlanNo":  1,
        /*ro, req, int, week schedule No.*/
        "holidayGroupNo":  "1,3,5"
        /*ro, req, string, holiday group No., desc:holiday group No.*/
    }
}
```

10.3.9.8 Set the schedule template parameters of the access permission control

Request URL

PUT /ISAPI/AccessControl/UserRightPlanTemplate/<planTemplateID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
planTemplateID	string	--

Request Message

```
{
    "UserRightPlanTemplate": {
    /*req, object, the schedule template of the access permission control*/
        "enable":  true,
        /*req, bool, whether to enable, desc:true (yes), false (no)*/
        "templateName":  "test",
        /*req, string, template name*/
        "weekPlanNo":  1,
        /*req, int, week schedule No.*/
        "holidayGroupNo":  "1,3,5",
        /*req, string, holiday group No., desc:holiday group No.*/
        "operateType":  "byTerminal",
        /*opt, enum, operation type, subType:string, desc:"byTerminal" (by terminal), "byOrg" (by organization), "byTerminalOrg" (by terminal
organization)*/
        "terminalNoList": [1, 2, 3, 4],
        /*opt, array, terminal ID list, subType:int, desc:this node is required when operation type is "byTerminal" or "byTerminalOrg"*/
        "orgNoList": [1, 2, 3, 4]
        /*opt, array, organization ID list, subType:int, desc:this node is required when operation type is "byOrg" or "byTerminalOrg"*/
    }
}
```

Response Message

```
{
    "requestURL":  "test",
    /*ro, opt, string, URI*/
    "statusCode":  "test",
    /*ro, opt, string, status code, desc:1 (succeeded). It is required when an error occurred*/
    "statusString":  "test",
    /*ro, opt, string, status description, desc:"ok" (succeeded). It is required when an error occurred*/
    "subStatusCode":  "test",
    /*ro, opt, string, sub status code, desc:"ok" (succeeded). It is required when an error occurred*/
    "errorCode":  1,
    /*ro, req, int, error code, desc:it is required when the value of statusCode is not 1, and it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, req, string, error information, desc:this node is required when the value of statusCode is not 1*/
}
```

10.3.9.9 Get the schedule template configuration capability of the access permission control

Request URL

GET /ISAPI/AccessControl/UserRightPlanTemplate/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "UserRightPlanTemplate": {
    /*ro, opt, object*/
        "templateNo": {
        /*ro, opt, object, schedule template No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  16
            /*ro, opt, int, the maximum value*/
        },
        "enable":  "true,false",
        /*ro, opt, string, whether it is enabled, desc:"true" (enabled), "false" (disabled)*/
        "templateName": {
        /*ro, opt, object, template name*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  32
            /*ro, opt, int, the maximum value*/
        },
        "weekPlanNo": {
        /*ro, opt, object, weekly schedule No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  16
            /*ro, opt, int, the maximum value*/
        },
        "holidayGroupNo": {
        /*ro, opt, object, holiday group No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  16
            /*ro, opt, int, the maximum value*/
        }
    }
}
```

10.3.9.10 Get weekly schedule configuration parameters

Request URL

GET /ISAPI/AccessControl/UserRightWeekPlanCfg/<weekPlanID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
weekPlanID	string	--

Request Message

None

Response Message

```
{
    "UserRightWeekPlanCfg": {
    /*ro, opt, object, weekly schedule configuration parameters*/
        "enable":  true,
        /*ro, req, bool, whether to enable, desc:true (enable), false (disable)*/
        "WeekPlanCfg": [
        /*ro, req, array, weekly schedule parameters, subType:object*/
            {
                "week":  "Monday",
                /*ro, req, enum, day of the week, subType:string, desc:"Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"*/
                "id":  1,
                /*ro, req, int, time period No., range:[1,8], desc:it is between 1 and 8*/
                "enable":  true,
                /*ro, req, bool, whether to enable, desc:true (enable), false (disable)*/
                "TimeSegment": {
                /*ro, req, object, time period*/
                    "beginTime":  "10:10:00",
                    /*ro, req, string, start time of the time period, desc:device local time*/
                    "endTime":  "12:10:00"
                    /*ro, req, string, end time of the time period, desc:device local time*/
                },
                "authenticationTimesEnabled":  true,
                /*ro, opt, bool*/
                "authenticationTimes":  10
                /*ro, opt, int, range:[1,255], step:1*/
            }
        ]
    }
}
```

10.3.9.11 Set the week schedule parameters of the access permission control

Request URL

PUT /ISAPI/AccessControl/UserRightWeekPlanCfg/<weekPlanID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
weekPlanID	string	--

Request Message

```
{
    "UserRightWeekPlanCfg": {
    /*opt, object, the week schedule parameters of the access permission control*/
        "enable":  true,
        /*req, bool, whether to enable, desc:"true" (enable), "false" (disable)*/
        "WeekPlanCfg": [
        /*req, array, week schedule parameters, subType:object*/
            {
                "week":  "Monday",
                /*req, enum, days of the week, subType:string, desc:"Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"*/
                "id":  1,
                /*req, int, time period No., range:[1,8], desc:it is between 1 and 8*/
                "enable":  true,
                /*req, bool, whether to enable, desc:"true" (enable), "false" (disable)*/
                "TimeSegment": {
                /*req, object, time period*/
                    "beginTime":  "10:10:00",
                    /*req, string, start time of the time period, desc:(device local time)*/
                    "endTime":  "12:10:00"
                    /*req, string, end time of the time period, desc:(device local time)*/
                },
                "authenticationTimesEnabled":  true,
                /*opt, bool*/
                "authenticationTimes":  10
                /*opt, int, range:[1,255], step:1*/
            }
        ]
    }
}
```

Response Message

```
{
    "requestURL":  "test",
    /*ro, opt, string, URI*/
    "statusCode":  "test",
    /*ro, opt, string, status code, desc:1 (succeeded). It is required when an error occurred*/
    "statusString":  "test",
    /*ro, opt, string, status description, desc:"ok" (succeeded). It is required when an error occurred*/
    "subStatusCode":  "test",
    /*ro, opt, string, sub status code, desc:"ok" (succeeded). It is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:it is required when the value of statusCode is not 1, and it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, opt, string, error information, desc:this node is required when the value of statusCode is not 1*/
}
```

10.3.9.12 Get the weekly schedule configuration capability of the access permission control

Request URL

GET /ISAPI/AccessControl/UserRightWeekPlanCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "UserRightWeekPlanCfg": {
    /*ro, opt, object*/
        "planNo": {
        /*ro, opt, object, weekly schedule No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  16
            /*ro, opt, int, the maximum value*/
        },
        "enable":  "true,false",
        /*ro, opt, string, whether it is enabled, desc:"true" (enabled), "false" (disabled)*/
        "WeekPlanCfg": {
        /*ro, opt, object, weekly schedule parameters*/
            "maxSize":  56,
            /*ro, opt, int, the maximum value*/
            "week": {
            /*ro, opt, object, week*/
                "@opt":  "Monday"
                /*ro, opt, enum, days of the week, subType:string, desc:"Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"*/
            },
            "id": {
            /*ro, opt, object*/
                "@min":  1,
                /*ro, opt, int, the minimum value*/
                "@max":  8
                /*ro, opt, int, the maximum value*/
            },
            "enable":  "true,false",
            /*ro, opt, string, whether it is enabled, desc:"true" (enabled), "false" (disabled)*/
            "TimeSegment": {
            /*ro, opt, object, time period*/
                "beginTime":  "test",
                /*ro, opt, string, start time, desc:(device local time)*/
                "endTime":  "test",
                /*ro, opt, string, end time, desc:(device local time)*/
                "validUnit":  "minute"
                /*ro, opt, enum, time accuracy, subType:string, desc:"hour", "minute", "second". If this node is not returned, the default time accuracy is
"minute"*/
            },
            "authenticationTimesEnabled": {
            /*ro, opt, object*/
                "@opt": [true, false]
                /*ro, req, array, subType:bool*/
            },
            "authenticationTimes": {
            /*ro, opt, object*/
                "@min":  1,
                /*ro, req, int, range:[1,255], step:1*/
                "@max":  255
                /*ro, req, int, range:[1,255], step:1*/
            }
        }
    }
}
```


10.3.10 Person and Credential Management

10.3.10.1 Get the capability of deleting person information (including linked cards, fingerprints, and faces) and permissions

Request URL

GET /ISAPI/AccessControl/UserInfoDetail/Delete/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "UserInfoDetail": {
    /*ro, opt, object, user Information*/
        "mode": {
        /*ro, req, object*/
            "@opt":  "all,byEmployeeNo"
            /*ro, opt, string, deleting mode, desc:all (delete all), byEmployeeNo (delete by employee No. (person ID))*/
        },
        "EmployeeNoList": {
        /*ro, opt, object, person ID list, desc:person ID list*/
            "maxSize":  50,
            /*ro, opt, int*/
            "employeeNo": {
            /*ro, opt, object, employee No. (person ID)*/
                "@min":  1,
                /*ro, opt, int, the maximum value*/
                "@max":  32
                /*ro, opt, int, the minimum value*/
            }
        }
    }
}
```

10.3.10.2 Start deleting all person information (including linked cards, fingerprints, and faces) and permissions by employee No.

Request URL

PUT /ISAPI/AccessControl/UserInfoDetail/Delete?format=json

Query Parameter

None

Request Message

```
{
    "UserInfoDetail": {
    /*opt, object, user Information*/
        "mode":  "all",
        /*req, enum, deleting mode, subType:string, desc:deleting mode*/
        "EmployeeNoList": [
        /*opt, array, person ID list, subType:object*/
            {
                "employeeNo":  "test"
                /*opt, string, employee No.*/
            }
        ],
        "operateType":  "byTerminal",
        /*opt, enum, operation mode, subType:string, desc:"byTerminal" (by terminal), "byOrg" (by organization), "byTerminalOrg" (by terminal
organization)*/
        "terminalNoList": [1, 2, 3, 4],
        /*opt, array, terminal list, subType:int, dep:and,{$.UserInfoDetail.operateType,eq,byTerminal}*/
        "orgNoList": [1, 2, 3, 4]
        /*opt, array, organization list, subType:int, dep:or,{$.UserInfoDetail.operateType,eq,byOrg},{$.UserInfoDetail.operateType,eq,byTerminalOrg}*/
    }
}
```

Response Message

```
{
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded). It is required when an error occurred*/
    "statusString":  "ok",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded). It is required when an error occurred*/
    "subStatusCode":  "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded). It is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:when the value of statusCode is not 1, it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, opt, string, error information, desc:this field is required when the value of statusCode is not 1*/
}
```

10.3.10.3 Get the status of deleting all person information (including linked cards, fingerprints, and faces) and permissions by employee No

Request URL

GET /ISAPI/AccessControl/UserInfoDetail/DeleteProcess?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "UserInfoDetailDeleteProcess": {
    /*ro, req, object*/
        "status":  "processing",
        /*ro, req, enum, status, subType:string, desc:status*/
        "percent":  100
        /*ro, opt, int, range:[0,100], dep:or,{$.UserInfoDetailDeleteProcess.status,eq,processing}*/
    }
}
```

10.3.11 Person and Credential Management

10.3.11.1 Get the capability of deleting person information (including linked cards, fingerprints, and faces) and permissions

Request URL

GET /ISAPI/AccessControl/UserInfoDetail/Delete/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "UserInfoDetail": {
    /*ro, opt, object, user Information*/
        "mode": {
        /*ro, req, object*/
            "@opt":  "all,byEmployeeNo"
            /*ro, opt, string, deleting mode, desc:all (delete all), byEmployeeNo (delete by employee No. (person ID))*/
        },
        "EmployeeNoList": {
        /*ro, opt, object, person ID list, desc:person ID list*/
            "maxSize":  50,
            /*ro, opt, int*/
            "employeeNo": {
            /*ro, opt, object, employee No. (person ID)*/
                "@min":  1,
                /*ro, opt, int, the maximum value*/
                "@max":  32
                /*ro, opt, int, the minimum value*/
            }
        }
    }
}
```

10.3.11.2 Start deleting all person information (including linked cards, fingerprints, and faces) and permissions by employee No.

Request URL

PUT /ISAPI/AccessControl/UserInfoDetail/Delete?format=json

Query Parameter

None

Request Message

```
{
    "UserInfoDetail": {
    /*opt, object, user Information*/
        "mode":  "all",
        /*req, enum, deleting mode, subType:string, desc:deleting mode*/
        "EmployeeNoList": [
        /*opt, array, person ID list, subType:object*/
            {
                "employeeNo":  "test"
                /*opt, string, employee No.*/
            }
        ],
        "operateType":  "byTerminal",
        /*opt, enum, operation mode, subType:string, desc:"byTerminal" (by terminal), "byOrg" (by organization), "byTerminalOrg" (by terminal
organization)*/
        "terminalNoList": [1, 2, 3, 4],
        /*opt, array, terminal list, subType:int, dep:and,{$.UserInfoDetail.operateType,eq,byTerminal}*/
        "orgNoList": [1, 2, 3, 4]
        /*opt, array, organization list, subType:int, dep:or,{$.UserInfoDetail.operateType,eq,byOrg},{$.UserInfoDetail.operateType,eq,byTerminalOrg}*/
    }
}
```

Response Message

```
{
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded). It is required when an error occurred*/
    "statusString":  "ok",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded). It is required when an error occurred*/
    "subStatusCode":  "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded). It is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:when the value of statusCode is not 1, it corresponds to subStatusCode*/
    "errorMsg":  "ok"
    /*ro, opt, string, error information, desc:this field is required when the value of statusCode is not 1*/
}
```

10.3.11.3 Get the status of deleting all person information (including linked cards, fingerprints, and faces) and permissions by employee No

Request URL

GET /ISAPI/AccessControl/UserInfoDetail/DeleteProcess?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "UserInfoDetailDeleteProcess": {
    /*ro, req, object*/
        "status":  "processing",
        /*ro, req, enum, status, subType:string, desc:status*/
        "percent":  100
        /*ro, opt, int, range:[0,100], dep:or,{$.UserInfoDetailDeleteProcess.status,eq,processing}*/
    }
}
```

10.3.12 Remote Door Control

10.3.12.1 Remotely control the door or elevator

Request URL

PUT /ISAPI/AccessControl/RemoteControl/door/<doorID>

Query Parameter

Parameter Name	Parameter Type	Description
doorID	string	--

Request Message

```
<?xml version="1.0" encoding="UTF-8"?>

<RemoteControlDoor xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--req, object, attr:version{req, string, protocolVersion}-->
  <cmd>
    <!--req, enum, command, subType:string, desc:"open" (open the door), "close" (close the door (controlled)), "alwaysOpen" (remain unlocked (free)),
"alwaysClose" (remain open (disabled)), "visitorCallLadder" (call elevator (visitor)), "householdCallLadder" (call elevator (resident))-->open
  </cmd>
  <password>
    <!--opt, string, range:[8,16]-->test
  </password>
  <employeeNo>
    <!--wo, opt, string, employee No., range:[0,32], desc:employee No.-->test
  </employeeNo>
  <channelNo>
    <!--opt, int, range:[0,256], step:1-->1
  </channelNo>
  <controlType>
    <!--opt, enum, subType:string-->monitor
  </controlType>
</RemoteControlDoor>
```

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<ResponseStatus xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, response message, attr:version{ro, req, string, protocolVersion}-->
  <requestURL>
    <!--ro, req, string, request URL-->null
  </requestURL>
  <statusCode>
    <!--ro, req, enum, status code, subType:int, desc:0 (OK), 1 (OK), 2 (Device Busy), 3 (Device Error), 4 (Invalid Operation), 5 (Invalid XML Format), 6
(Invalid XML Content), 7 (Reboot Required)-->0
  </statusCode>
  <statusString>
    <!--ro, req, enum, status information, subType:string, desc:"OK" (succeeded), "Device Busy", "Device Error", "Invalid Operation", "Invalid XML Format",
"Invalid XML Content", "Reboot" (reboot device)-->OK
  </statusString>
  <subStatusCode>
    <!--ro, req, string, sub status code, desc:sub status code, which describes the error in details-->OK
  </subStatusCode>
</ResponseStatus>
```


10.3.12.2 Get the capability set of remote door status control

Request URL

GET /ISAPI/AccessControl/RemoteControl/door/capabilities

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<RemoteControlDoor xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, remote door status control, attr:version{req, float, protocolVersion}-->
  <doorNo min="1" max="10">
    <!--ro, opt, int, range of the door No., attr:min{req, int},max{req, int}-->1
  </doorNo>
  <cmd opt="open,close,alwaysOpen,alwaysClose,visitorCallLadder,householdCallLadder,resume">
    <!--ro, req, enum, command, subType:string, attr:opt{req, string}, desc:"open" (open the door), "close" (close the door (controlled)), "alwaysOpen"
(remain unlocked (free)), "alwaysClose" (remain locked (disabled)), "visitorCallLadder" (call elevator (visitor)), "householdCallLadder" (call elevator
(resident))-->open
  </cmd>
  <password min="8" max="16">
    <!--ro, opt, string, door opening password, range:[8,16], attr:min{req, int, range:[8,16]},max{req, int, range:[8,16]}-->test
  </password>
  <employeeNo min="0" max="32">
    <!--ro, opt, string, employee No., range:[0,32], attr:min{req, int, range:[0,32]},max{req, int, range:[0,32]}-->test
  </employeeNo>
  <channelNo min="0" max="10">
    <!--ro, opt, int, attr:min{req, int},max{req, int}-->0
  </channelNo>
  <controlType opt="monitor,calling">
    <!--ro, opt, enum, subType:string, attr:opt{req, string}-->monitor
  </controlType>
</RemoteControlDoor>
```

10.3.12.3 Get the capability of configuring password for remote door control

Request URL

GET /ISAPI/AccessControl/remoteControlPWCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "RemoteControlPWCfg": {
    /*ro, opt, object*/
        "password": {
        /*ro, opt, object, password for remote door control*/
            "@min":  "6",
            /*ro, opt, string*/
            "@max":  "6"
            /*ro, opt, string*/
        }
    }
}
```

10.3.12.4 Configure the password for remote door control

Request URL

PUT /ISAPI/AccessControl/remoteControlPWCfg/door/<doorID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
doorID	string	--

Request Message

```
{
    "RemoteControlPWCfg": {
    /*opt, object, password configuration for remote door control*/
        "password":  "test"
        /*opt, string, password for remote door control*/
    }
}
```

Response Message

```
{
    "requestURL":  "test",
    /*ro, opt, string, URI*/
    "statusCode":  "test",
    /*ro, opt, string, status code*/
    "statusString":  "test",
    /*ro, opt, string, status description*/
    "subStatusCode":  "test",
    /*ro, opt, string, sub status code*/
    "errorCode":  1,
    /*ro, req, int, error code*/
    "errorMsg":  "ok"
    /*ro, req, string, error information*/
}
```

10.3.12.5 Get the password for remotely controlling the door

Request URL

GET /ISAPI/AccessControl/remoteControlPWCfg/door/<doorID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
doorID	string	--

Request Message

None

Response Message

```
{
    "RemoteControlPWCfg": {
    /*ro, opt, object, password settings for remotely controlling the door*/
        "password":  "test"
        /*ro, opt, string, password for remotely controlling the door*/
    }
}
```

10.3.12.6 Get the capability of verifying the password for remote door control

Request URL

GET /ISAPI/AccessControl/remoteControlPWCheck/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "RemoteControlPWCfg": {
    /*ro, opt, object*/
        "password": {
        /*ro, opt, object, password for remote door control, desc:password for remote door control (or EZ verification code). The password must contain 6
digits and it ranges from 000000 to 999999*/
            "@min":  6,
            /*ro, opt, int, the minimum value*/
            "@max":  6
            /*ro, opt, int, the maximum value*/
        }
    }
}
```

10.3.12.7 Verify the password for remote door control

Request URL

PUT /ISAPI/AccessControl/remoteControlPWCheck/door/<doorID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
doorID	string	--

Request Message

```
{
    "RemoteControlPWCheck": {
    /*opt, object, password verification for remote door control*/
        "password":  "test"
        /*opt, string, password for remote door control (or EZVIZ verification code)*/
    }
}
```

Response Message

```
{
    "statusCode":  1,
    /*ro, opt, int, status code, desc:1 (succeeded). It is required when an error occurred*/
    "statusString":  "ok",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded). It is required when an error occurred*/
    "subStatusCode":  "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded). It is required when an error occurred*/
    "errorCode":  1,
    /*ro, opt, int, error code, desc:it corresponds to subStatusCode when statusCode is not 1*/
    "errorMsg":  "ok"
    /*ro, opt, string, error information, desc:this field is required when the value of statusCode is not 1*/
}
```

10.4 Credentials Collection

10.4.1 Card Online Adding

10.4.1.1 Get the capability of collecting card information

Request URL

GET /ISAPI/AccessControl/CaptureCardInfo/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "CardInfoCap": {
    /*ro, req, object*/
        "cardNo": {
        /*ro, opt, object, card No.*/
            "@min":  1,
            /*ro, req, int, the minimum length, range:[1,32]*/
            "@max":  32
            /*ro, req, int, the maximum length, range:[1,32]*/
        },
        "cardType": ["TypeA_M1", "TypeA_CPU", "TypeB", "ID_125K", "FelicaCard", "DesfireCard"],
        /*ro, opt, array, card type, subType:string, range:[1,6]*/
        "readerID": {
        /*ro, opt, object*/
            "@min":  1,
            /*ro, req, int*/
            "@max":  8
            /*ro, req, int*/
        }
    }
}
```

10.4.1.2 Collect card information by the card reading module of the device

Request URL

GET /ISAPI/AccessControl/CaptureCardInfo?format=json&readerID=<readerID>

Query Parameter

Parameter Name	Parameter Type	Description
readerID	string	--

Request Message

None

Response Message

```
{
    "CardInfo": {
    /*ro, req, object, card information*/
        "cardNo":  "abcd1234",
        /*ro, req, string, card No.*/
        "cardType":  "TypeA_M1",
        /*ro, opt, enum, card type, subType:string, desc:"TypeA_M1", "TypeA_CPU", "TypeB", "ID_125K", "FelicaCard" (FeliCa card), "DesfireCard" (DESFire
card)*/
        "readerID":  1
        /*ro, opt, int, range:[1,8]*/
    }
}
```

10.4.2 Fingerprint Online Adding

10.4.2.1 Collect fingerprint information

Request URL

POST /ISAPI/AccessControl/CaptureFingerPrint

Query Parameter

None

Request Message

```
<?xml version="1.0" encoding="UTF-8"?>

<CaptureFingerPrintCond xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--opt, object, Collect fingerprint information conditions, attr:version{req, string, protocolVersion}-->
  <fingerNo>
    <!--req, int, finger No., range:[1,10]-->1
  </fingerNo>
</CaptureFingerPrintCond>
```

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<CaptureFingerPrint xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, attr:version{req, string, protocolVersion}-->
  <fingerData>
    <!--ro, opt, string, fingerprint data, range:[1,768], desc:which, should be encoded by Base64-->test
  </fingerData>
  <fingerNo>
    <!--ro, req, int, finger No., range:[1,10]-->1
  </fingerNo>
  <fingerPrintQuality>
    <!--ro, req, int, fingerprint quality, range:[1,100]-->1
  </fingerPrintQuality>
</CaptureFingerPrint>
```

Parameter Name	Parameter Value	Parameter Type(Content-Type)	Content-ID	File Name	Description
CaptureFingerPrint	[Message content]	application/xml	--	--	--
fingerPrintPic	[Binary picture data]	image/jpeg		fingerPrintPic.jpg	--

Note： The protocol is transmitted in form format. See Chapter 4.5.1.4 for form framework description, as shown in the instance below.

```
--<frontier>
Content-Disposition: form-data; name=Parameter Name;filename=File Name
Content-Type: Parameter Type
Content-Length: ****
Content-ID: Content ID
Parameter Value
```

- Parameter Name: the name property of Content-Disposition in the header of form unit; it refers to the form unit name.
- Parameter Type (Content-Type): the Content-Type property in the header of form unit.
- File Name (filename): the filename property of Content-Disposition of form unit Headers. It exists only when the transmitted data of form unit is file, and it refers to the file name of form unit body.
- Parameter Value: the body content of form unit.

10.4.2.2 Get the fingerprint collection capability

Request URL

GET /ISAPI/AccessControl/CaptureFingerPrint/capabilities

Query Parameter

None

Request Message

None

Response Message

```
<?xml version="1.0" encoding="UTF-8"?>

<CaptureFingerPrint xmlns="http://www.isapi.org/ver20/XMLSchema" version="2.0">
  <!--ro, req, object, collect fingerprint information, attr:version{req, string, protocolVersion}-->
  <CaptureFingerPrintCond>
    <!--ro, req, object, finger No.-->
    <fingerNo min="1" max="10">
      <!--ro, opt, int, fingerprint No., range:[1,10], attr:min{req, int},max{req, int}-->1
    </fingerNo>
  </CaptureFingerPrintCond>
  <fingerData min="1" max="768">
    <!--ro, opt, string, fingerprint data, range:[1,768], attr:min{req, int},max{req, int}-->test
  </fingerData>
  <fingerNo min="1" max="10">
    <!--ro, opt, int, fingerprint No., range:[1,10], attr:min{req, int},max{req, int}-->1
  </fingerNo>
  <fingerPrintQuality min="1" max="100">
    <!--ro, opt, int, fingerprint quality, range:[1,100], attr:min{req, int},max{req, int}-->1
  </fingerPrintQuality>
</CaptureFingerPrint>
```


10.5 Security Control Device (General)

10.5.1 Event Message Push Management

10.5.1.1 Get the parameters of the report uploading method

Request URL

GET /ISAPI/SecurityCP/ReportCenterCfg/<centerID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
centerID	string	--

Request Message

None

Response Message

```
{
    "ReportCenterCfg": {
    /*ro, req, object*/
        "enable": true,
        /*ro, opt, bool, whether to enable the function*/
        "ChanAlarmMode": [
        /*ro, opt, array, alarm channel of the center group, subType:object*/
            {
                "id": 1,
                /*ro, opt, enum, channel ID, subType:int, desc:channel ID: 1-main channel,2-backup channel 1,3-backup channel 2,4-backup channel 3*/
                "chanAlarmMode": "T1"
                /*ro, opt, enum, alarm channel mode, subType:string, desc:"T1" (T1 channel), "T2" (T2 channel), "N1" (N1 channel), "N2" (N2 channel), "G1"
(G1 channel), "G2" (G2 channel), "N3" ()N3 channel, ”N4" (N4 channel)*/
            }
        ]
    }
}
```

10.5.1.2 Set the parameters of the report uploading method

Request URL

PUT /ISAPI/SecurityCP/ReportCenterCfg/<centerID>?format=json

Query Parameter

Parameter Name	Parameter Type	Description
centerID	string	--

Request Message

```
{
    "ReportCenterCfg": {
    /*req, object*/
        "enable": true,
        /*opt, bool, whether to enable uploading report*/
        "ChanAlarmMode": [
        /*opt, array, alarm channel of the center group, subType:object*/
            {
                "id": 1,
                /*opt, enum, channel ID, subType:int, desc:1 (main channel), 2 (backup channel 1), 3 (backup channel 2), 4 (backup channel 3)*/
                "chanAlarmMode": "T1"
                /*opt, enum, alarm channel mode, subType:string, desc:"T1" (T1 channel), "T2" (T2 channel), "N1" (N1 channel), "N2" (N2 channel), "G1" (G1
channel), "G2" (G2 channel), "N3" ()N3 channel, ”N4" (N4 channel)*/
            }
        ]
    }
}
```

Response Message

```
{
    "statusCode": 1,
    /*ro, opt, int, status code, desc:1 (succeeded). It is required when an error occurred*/
    "statusString": "ok",
    /*ro, opt, string, status description, range:[1,64], desc:"ok" (succeeded). It is required when an error occurred*/
    "subStatusCode": "ok",
    /*ro, opt, string, sub status code, range:[1,64], desc:"ok" (succeeded). It is required when an error occurred*/
    "errorCode": 1,
    /*ro, opt, int, error code, desc:it is required when the value of statusCode is not 1, it corresponds to subStatusCode*/
    "errorMsg": "ok"
    /*ro, opt, string, error description, desc:this field is required when the value of statusCode is not 1*/
}
```

10.5.1.3 Get the configuration capability of the report uploading method

Request URL

GET /ISAPI/SecurityCP/ReportCenterCfg/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "ReportCenterCfg": {
    /*ro, req, object*/
        "CenterID": {
        /*ro, opt, object, center group No.*/
            "@min":  1,
            /*ro, opt, int, the minimum value*/
            "@max":  1
            /*ro, opt, int, the maximum value*/
        },
        "enable":  "true,false",
        /*ro, opt, string, whether to enable uploading report*/
        "ChanAlarmMode": {
        /*ro, opt, object, alarm channel of the center group*/
            "maxSize":  1,
            /*ro, opt, int, the maximum number of channels*/
            "id": {
            /*ro, opt, object, channel ID, desc:1 (main channel), 2 (backup channel 1), 3 (backup channel 2), 4 (backup channel 3)*/
                "@min":  1,
                /*ro, opt, int, the minimum value*/
                "@max":  2
                /*ro, opt, int, the maximum value*/
            },
            "chanAlarmMode": {
            /*ro, opt, object, alarm channel mode, desc:"T1" (T1 channel), "T2" (T2 channel), "N1" (N1 channel), "N2" (N2 channel), "G1" (G1 channel), "G2"
(G2 channel), "N3" (N3 channel), "N4" (N4 channel)*/
                "@opt":  "T1,T2,N1,N2,G1,G2,N3,N4"
                /*ro, opt, string*/
            }
        }
    }
}
```

10.6 Zone Alarm

10.6.1 Security Control Panel Management

10.6.1.1 Get the capability of security control panel

Request URL

GET /ISAPI/SecurityCP/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "SecurityCPCap": {
    /*ro, req, object, capability node*/
        "partitionNum":  64,
        /*ro, opt, int, number of partitions that can be set, desc:1 by default*/
        "localZoneNum":  16,
        /*ro, req, int, number of local zones*/
        "extendZoneNum":  64,
        /*ro, opt, int, number of extended zones*/
        "wirelessZoneNum":  64,
        /*ro, opt, int, number of wireless zones*/
        "localRelayNum":  1,
        /*ro, req, int, number of local triggers*/
        "extendRelayNum":  1,
        /*ro, opt, int, number of extended triggers*/
        "wirelessRelayNum":  1,
        /*ro, opt, int, number of wireless triggers*/
        "repeater":  4,
        /*ro, opt, int, number of repeaters*/
        "sirenNum":  8,
        /*ro, opt, int, number of sounders*/
        "userNum":  1,
        /*ro, req, int, number of users*/
        "adminNum":  2,
        /*ro, opt, int, number of administrators*/
        "installerNum":  21,
        /*ro, opt, int, number of installers*/
```

```
        "operatorNum":  41,
        /*ro, opt, int, number of operators*/
        "ARCNum":  4,
        /*ro, opt, int, number of alarm receiving centers*/
        "phoneNum":  3,
        /*ro, opt, int, number of phone numbers*/
        "outputModNum":  8,
        /*ro, opt, int, number of output modules*/
        "cardNum":  64,
        /*ro, opt, int, number of cards*/
        "keypadNum":  8,
        /*ro, opt, int, number of keypads*/
        "cardReaderNum":  8,
        /*ro, opt, int, number of card readers*/
        "protocolOptimizationVersion":  "v1.0",
        /*ro, opt, string, supported protocol optimizing version*/
        "remoteCtrlNum":  64,
        /*ro, opt, int, number of remote controls*/
        "alarmLampNum":  8,
        /*ro, opt, int, number of strobe lights*/
        "electricLockNum":  64,
        /*ro, opt, int, number of electric locks*/
        "detectorModel": {
        /*ro, opt, object, supported detector models*/
            "@opt": ["0x00001", "0x00002"]
            /*ro, opt, array, options, subType:string*/
        },
        "sirenModel": {
        /*ro, opt, object, supported sounder models*/
            "@opt": ["0x7A001", "0x7A011"]
            /*ro, opt, array, options, subType:string*/
        },
        "keypadModel": {
        /*ro, opt, object, supported keypad models*/
            "@opt": ["0x92000", "0x92010"]
            /*ro, opt, array, options, subType:string*/
        },
        "cardReaderModel": {
        /*ro, opt, object, supported card reader models*/
            "@opt": ["0x90000", "0x90010"]
            /*ro, opt, array, options, subType:string*/
        },
        "remoteCtrlModel": {
        /*ro, opt, object, supported remote control models*/
            "@opt": ["0x81000", "0x81010"]
            /*ro, opt, array, options, subType:string*/
        },
        "repeaterModel": {
        /*ro, opt, object, supported repeater models*/
            "@opt": ["0x80000", "0x80010"]
            /*ro, opt, array, options, subType:string*/
        },
        "outputModuleModel": {
        /*ro, opt, object, supported output module models*/
            "@opt": ["0x71001", "0x71011"]
            /*ro, opt, array, options, subType:string*/
        },
        "isSptLogSearch":  true,
        /*ro, opt, bool, whether it supports log search*/
        "isSptLocalUser":  true,
        /*ro, opt, bool, whether it supports local users*/
        "isSptConfiguration":  true,
        /*ro, opt, bool, whether it supports configuring security control panel, desc:/ISAPI/SecurityCP/Configuration*/
        "isSptControl":  true,
        /*ro, opt, bool, whether it supports security control panel control, desc:/ISAPI/SecurityCP/control*/
        "isSptStatus":  true,
        /*ro, opt, bool, whether it supports monitoring security control panel status, desc:/ISAPI/SecurityCP/status*/
        "isSptZoneInfo":  true,
        /*ro, opt, bool, whether it supports getting the information of a specific zone, desc:/ISAPI/SecurityCP/Info/zones/<ID>?format=json*/
        "isSptSirenInfo":  true,
        /*ro, opt, bool, whether it supports getting the information of a specific sounder, desc:/ISAPI/SecurityCP/Info/siren/<ID>?format=json*/
        "isSptPaceTest":  true,
        /*ro, opt, bool, whether it supports pacing, desc:/ISAPI/SecurityCP/paceTest*/
        "isSptStandardCfg":  true,
        /*ro, opt, bool, whether it supports standard configuration for security control panel, desc:/ISAPI/SecurityCP/standardCfg*/
        "isSptPircamCapture":  true,
        /*ro, opt, bool, whether it supports pircam (detector equipped with camera) capture, desc:/ISAPI/SecurityCP/pircam/channels/<ID>/picture?
format=json*/
        "isSptManualControlCapture":  true,
        /*ro, opt, bool*/
        "isSptGetPictureByURL":  true,
        /*ro, opt, bool*/
        "isSptOneKeyAlarm":  true,
        /*ro, opt, bool, whether it supports one-push alarm, desc:/ISAPI/SecurityCP/control/oneKeyAlarm. For compatibility, this node will also be returned
in the capability message JSON_HostControlCap after calling the URI /ISAPI/SecurityCP/control/capabilities?format=json by GET method*/
        "isSptOneKeyAlarmSoundCtrl":  true,
        /*ro, opt, bool*/
        "transmitterNum":  8,
        /*ro, opt, int, the number of transmitters*/
        "transmitterModel": {
        /*ro, opt, object, supported model of transmitters*/
            "@opt": ["0x71001"]
            /*ro, opt, array, options, subType:string*/
        },
```

```
        "localAccessModuleType": {
        /*ro, opt, object, onboard access module type*/
            "@opt": ["localTransmitter", "localZone", "localRelay", "localSiren"]
            /*ro, opt, array, options, subType:string, desc:"localTransmitter" (onboard transmitter), "localZone" (onboard zone module), "localRelay"
(onboard relay module), "localSiren" (onboard sounder module)*/
        },
        "networkZoneModuleNum":  8
        /*ro, opt, int, number of network zone modules*/
    }
}
```

10.6.1.2 Get the configuration capability of security control panel

Request URL

GET /ISAPI/SecurityCP/Configuration/capabilities?format=json

Query Parameter

None

Request Message

None

Response Message

```
{
    "HostConfigCap": {
    /*ro, req, object, capability node*/
        "ExDevice": {
        /*ro, opt, object, peripheral node*/
            "isSptOutput":  true,
            /*ro, opt, bool, whether it supports relay management, desc:/ISAPI/SecurityCP/Configuration/outputs*/
        },
        "isSptReportCenterCfg":  true,
        /*ro, opt, bool, whether it supports configuring the report uploading method, desc:/ISAPI/SecurityCP/ReportCenterCfg/<ID>?format=json*/
        "isSptAlarmOutCfg":  true,
        /*ro, opt, bool, whether it supports configuring alarm output parameters, desc:/ISAPI/SecurityCP/AlarmOutCfg/<ID>?format=json*/
        "isSptSetAlarmHostOut":  true,
        /*ro, opt, bool, whether it supports setting alarm output, desc:/ISAPI/SecurityCP/SetAlarmHostOut?format=json*/
    }
}
```


11 How-To Video Guidance

If you need access to corresponding video guidance for device integration, please register on https://tpp.hikvision.com and visit our Training Center: https://tpp.hikvision.com/tpp/Training. The Training Center is specifically designed to provide technical training and guidance resources for our partners. On this platform, you can find integration video tutorials for various devices, enabling better understanding and learning of the integration process. To offer more personalized service, our Training Center also supports filtering by integration protocols, devices, and applications.