HTTP API Protocol User Guide

For IP Media Device

Version 1.9

2020-05

Document History

No	Release Notes	Date	Version	Author
1	Draft	2014-2-14	1.0	
2	Draft	2014-4-15	1.1	
3	Draft	2014-9-30	1.2	
4	Release	2014-11-6	1.2	Luyugang
5	Draft	2014-12-5	1.3	Chenshiguang
6	Draft	2015-12-22	1.4	Ouyangming
7	Draft	2016-5-25	1.5	Ouyangming
8	Release	2017-8-4	1.6	Liu Wenhan
9	Release	2017-11-22	1.7	ouyangss
10	Release	2019-10-22	1.8	Dengyongjun
11	Draft	2020-05-06	1.9	Dengyongjun

Contents

HTTP API PROTOCOL USER GUIDE I
FOR IP MEDIA DEVICE I
VERSION 1.9 I
2020-05 I
DOCUMENT HISTORY II
1 OVERVIEW 1
1.1 PREFACE 1
1.2 TRANSACTION 1
1.3 PROTOCOL DESCRIPTION 1
1.3.1 URL *2*
1.3.2 Connection Header Filed *2*
1.3.3 Authorization Header Field *3*
1.3.4 Entity Body Field *4*
1.3.5 Response Message *4*
1.3.6 Error Code *6*
1.4 PROTOCOL CONVENTIONS 6
1.4.1 XML Element Name *6*
1.4.2 XML Element Type *7*
1.4.3 The "types" Element *9*
1.4.4 Command catagory *9*
1.5 DEVICE DISCOVERY 10
2 SYSTEM COMMANDS 11
2.1 DEVICE INFORMATION 11
2.1.1 GetDeviceInfo *11*
2.1.2 GetDiskInfo *13*
2.1.3 GetChannelList *15*
2.1.4 GetAlarmInList *16*

2.1.5 GetAlarmOutList 17
2.1.6 GetDeviceDetail 18
2.2 DATE AND TIME 21
2.2.1 GetDateAndTime 21
2.2.2 SetDateAndTime 22
3 IMAGE COMMANDS 23
3.1 STREAM CAPABILITIES 23
3.1.1 GetStreamCaps 23
3.2 IMAGE CONFIGURATION 26
3.2.1 GetImageConfig 26
3.2.2 SetImageConfig 28
3.2.3 GetSnapshot 29
3.2.4 GetSnapshotByTime 29
3.3 STREAM CONFIGURATION 30
3.3.1 GetAudioStreamConfig 30
3.3.2 SetAudioStreamConfig 31
3.3.3 GetVideoStreamConfig 32
3.3.4 SetVideoStreamConfig 35
3.3.5 RequestKeyFrame 36
3.4 OSD 36
3.4.1 GetImageOsdConfig 36
3.4.2 SetImageOsdConfig 39
3.5 PRIVACY MASK 40
3.5.1 GetPrivacyMaskConfig 40
3.5.2 SetPrivacyMaskConfig 41
4 PTZ COMMANDS 42
4.1 PROTOCOL 42
4.1.1 PtzGetCaps 42
4.1.2 GetPtzConfig 43
..... 错误！未定义书签。
4.1.3 SetPtzConfig 44
4.2 PTZ CONTROL 45
4.2.1 PtzControl 45
4.2.2 PtzGotoPreset 46
4.2.3 PtzRunCruise 47
4.2.4 PtzStopCruise 48

4.3 PRESET....48
4.3.1 *PtzGetPresets*....48
4.3.2 *PtzAddPreset*....49
4.3.3 *PtzModifyPresetName*....50
4.3.4 *PtzDeletePreset*....50
4.3.5 *PtzModifyPresetPosition*....51
4.4 CRUISE....51
4.4.1 *PtzGetCruises*....51
4.4.2 *PtzGetCruise*....52
4.4.3 *PtzAddCruise*....53
4.4.4 *PtzModifyCruise*....54
4.4.5 *PtzDeleteCruise*....55
5 ALARM COMMANDS....56
5.1 MOTION DETECTION....56
5.1.1 *GetMotionConfig*....56
5.1.2 *SetMotionConfig*....58
5.2 ALARM....58
5.2.1 *GetAlarmInConfig*....58
5.2.2 *SetAlarmInConfig*....59
5.2.3 *ManualAlarmOut*....60
5.2.4 *GetAlarmOutConfig*....61
5.2.5 *SetAlarmOutConfig*....61
5.3 ALARMSTATUS....62
5.3.1 *GetAlarmStatus*....62
5.3.2 *GetAlarmServerConfig*....64
5.3.3 *SetAlarmServerConfig*....66
5.3.4 *SendAlarmStatus*....66
5.4 ALARMTRIGGER....67
5.4.1 *GetAlarmTriggerConfig*....67
5.4.2 *SetAlarmTriggerConfig*....69
5.5 SOUND-LIGHT ALARM....69
5.5.1 *GetAudioAlarmOutConfig*....69
5.5.2 *SetAudioAlarmOutConfig*....72
5.5.3 *AddCustomizeAudioAlarm*....73
5.5.4 *DeleteCustomizeAudioAlarm*....74
5.5.5 *AuditionCustomizeAudioAlarm*....76

5.5.6 *GetWhiteLightAlarmOutConfig* 77
5.5.7 *SetWhiteLightAlarmOutConfig* 78
5.6 ALARM PIR 79
5.6.1 *GetPirConfig* 79
5.6.2 *SetPirConfig* 80
6 PLAYBACK 81
6.1 RECORD SEARCH 81
6.1.1 *GetRecordType* 81
6.1.2 *SearchRecordDate* 82
6.1.3 *SearchByTime* 83
6.2 RECORDSTATUS 86
6.2.1 *GetRecordStatusInfo* 86
7 NETWORK COMMANDS 88
7.1 TCP/IPV4 88
7.1.1 *GetNetBasicConfig* 88
7.1.2 *SetNetBasicConfig* 89
7.2 PPPOE 90
7.2.1 *GetNetPppoeConfig* 90
7.2.2 *SetNetPppoeConfig* 91
7.3 PORT 91
7.3.1 *GetPortConfig* 91
7.3.2 *SetPortConfig* 92
7.4 DDNS 93
7.4.1 *GetDdnsConfig* 93
7.4.2 *SetDdnsConfig* 94
8 SECURITY COMMANDS 95
8.1 USER MANAGEMENT 95
8.1.1 *ModifyPassword* 95
9 MAINTAIN COMMANDS 97
9.1 REBOOT 97
9.1.1 *Reboot* 97
10 TALKBACK COMMANDS 98
10.1 TALKBACK 98

10.1.1 *Talkback* *98*

11 SMART COMMANDS 100

11.1 FACE DETECT & FACE COMPARISON 100

11.1.1 *GetSmartVfdConfig* *100*

11.1.2 *SetSmartVfdConfig* *104*

11.1.3 *AddTargetFace* *104*

11.1.4 *DeleteTargetFace* *108*

11.1.5 *EditTargetFace* *111*

11.1.6 *GetTargetFace* *113*

11.1.7 *SearchSnapFaceByTime* *117*

11.1.8 *SearchSnapFaceByKey* *118*

..... *错误！未定义书签。*

..... *错误！未定义书签。*

11.2 CROWD DENSITY DETECTION 121

11.2.1 *GetSmartCddConfig* *121*

11.2.2 *SetSmartCddConfig* *123*

11.3 PEOPLE COUNTING 123

11.3.1 *GetSmartCpcConfig* *123*

11.3.2 *SetSmartCpcConfig* *125*

11.4 PEOPLE INTRUSION 125

11.4.1 *GetSmartIpdConfig* *125*

11.4.2 *SetSmartIpdConfig* *126*

11.5 LINE CROSSING 127

11.5.1 *GetSmartPerimeterConfig* *127*

11.5.2 *SetSmartPerimeterConfig* *129*

11.6 INTRUSION 129

11.6.1 *GetSmartTripwireConfig* *129*

11.6.2 *SetSmartTripwireConfig* *131*

11.7 OBJECT REMOVAL 132

11.7.1 *GetSmartOscConfig* *132*

11.7.2 *SetSmartOscConfig* *133*

11.8 EXCEPTION 134

11.8.1 *GetSmartAvdConfig* *134*

11.8.2 *SetSmartAvdConfig* *135*

11.9 LICENSE PLATE RECOGNITION 135

11.9.1 *GetSmartVehicleConfig* *135*

11.9.2 *SetSmartVehicleConfig* *141*

11.9.3 AddVehiclePlate 141
11.9.4 DeleteVehiclePlate 143
11.9.5 EditVehiclePlate 144
11.9.6 GetVehiclePlate 145
11.9.7 GetVehiclePlateProgress 147
11.10 REGION ENTRANCE 147
11.10.1 GetSmartAoiEntryConfig 147
11.10.2 SetSmartAoiEntryConfig 150
11.11 REGION ENTRANCE 151
11.11.1 GetSmartAoiLeaveConfig 151
11.11.2 SetSmartAoiLeaveConfig 154
11.12 TARGET COUNTING 154
11.12.1 GetSmartPassLineCountConfig 154
11.12.2 SetSmartPassLineCountConfig 159
11.12.3 GetPassLineCountStatistics 159
11.13 THERMOGRAPHIC TEMPERATURE MEASUREMENT 160
11.13.1 GetMeasureTemperatureConfig 160
11.13.2 SetMeasureTemperatureConfig 162
11.13.3 GetTemperatureCalibrationConfig 163
11.13.4 SetTemperatureCalibrationConfig 164
11.13.5 GetMeasureTemperatureScheduleConfig 165
11.13.6 SetMeasureTemperatureScheduleConfig 166
11.13.7 GetDotTemperature 166
11.14 INFRARED TEMPERATURE CONTROL 168
11.14.1 GetAccessControlConfig 168
11.14.2 SetAccessControlConfig 169
11.14.3 SetPassUnlockConfig ..... 错误！未定义书签。
11.14.4 UnLockingByPassword 170
11.14.5 GetTakeTemperatureConfig 171
11.14.6 SetTakeTemperatureConfig 173
11.14.7 GetWearmaskDetectConfig 173
11.14.8 SetWearmaskDetectConfig 174
12 SCHEDULE COMMANDS 175
12.1 SCHEDULE 175
12.1.1 GetScheduleConfig 175
12.1.2 SetScheduleConfig 178

12.1.3 SetScheduleConfigEx *179*

ANNEX A **1**

A.1 CHANGE LOG 1

1 Overview

1.1 Preface

This document details the API of IP media devices. Programmers can access and configure IP media devices following the API.

1.2 Transaction

The HTTP API transaction starts from a request from a client application, usually a web browser. The web server on the IP media devices processes the request and sends the response back to the client application. The HTTP requests taken in POST form as described in the following paragraphs. If the request is successful, the IP media video device will return a HTTP header contains 200 OK. The HTTP Body will contain actual result or error message if an error occurs.

1.3 Protocol Description

The client application should use POST form to send requests to the IP media devices. Other forms are not supported in this specification.

1.3.1 URL

The URL scheme is used to specify a request to the device locate device resources via a specific protocol in the network.This section defines the syntax and semantics for HTTP URLs.

<protocol>:///<host>[:port]</cmd name>[/channelId][/action name]

protocol: URL scheme for the particular request. The HTTP protocol isallowedin this specification.

host: The host field refer to the hostname, IP address, or the FQDN(Fully Qualified Domain Name) of an IP device.

port: The port field refer to the port number of that host on which the identified resource is located at the IP device listening for TCP connections. If the port is empty or not given, the default port is assumed. For HTTP, the default port 80.

cmd name: The specific command to an IP device.

channelId: The channel identification for an IP device. For the IP camera, this field can be omitted, the default channelId is “1”.

action name: This field is optional. It acts as a sub operation for complex commands.

1.3.2 Connection Header Filed

Requests from the video management system or the client application are packed in HTTP messages. A request message composed of three parts: the connection header field, the authorization header field, and the entity body field.

HTTP/1.1 is implemented and utilized according to RFC 2616 in the IP devices. For a video management system or client application that uses persistent connection for multiple transactions, it is required to implement “Connection: Keep-Alive ”HTTP header field as follows.

POST http://192.168.6.37/PtzAddPreset

HTTP/1.1

…

Content-Length: 135

…

Connection: Keep-Alive

…

1.3.3 Authorization Header Field

When a video management system or client application sends any request to the IP device, it must be authenticated by means of Basic Access according to RFC 2617.

Authorization header field needs to be sent along with each request, and if a user is authenticated, the request will follow the normal execution flow. For the request with no authentication credentials, unauthorized HTTP response (401) will be returned with WWW-Authenticate header field.

For example:

1. An HTTP request from the client application should include the “Authorization” information as follows, the “YWRtaW46MTIzNDU2” is the encoded result of “admin:123456” by base64:

POST http://192.168.6.37/PtzAddPreset

HTTP/1.1

…

Authorization: Basic YWRtaW46MQ==

…

2. The device responses the following to a request with no authentication credentials:

401 Unauthorized

WWW-Authenticate: Basic realm=”XXXXXX”

Then the client application encodes the username and password with base64, and sends the following request:

Authorization: Basic VXZVXZ.

1.3.4 Entity Body Field

Some requests will include entity body field. The Content-Type entity-header field indicates the media type of the entity body. The Content-Type may be designated as “application/xml; charset=’UTF-8’”. For example:

```
POST http://192.168.6.37/PtzAddPreset
 HTTP/1.1
...
Content-Type: application/xml; charset=”UTF-8”
...
<?xml version="1.0" encoding="utf-8" ?>
<presetInfo>
<name>preset1</name>
</presetInfo>
```

1.3.5 Response Message

The response message from the IP device is a standard HTTP response, information can be included in the entity body field in XML format. This information includes the result to a request message, or the detailed parameters that required by a request message.

A successful response that includes the result is as follows:

```
HTTP/1.1 200 OK
…
Content-Type: application/xml; charset=”UTF-8”
Content-Length: 66
```

```
Connection: close
…
<?xml version="1.0" encoding="UTF-8"?>
<config status="success"/>
```

A successful response that includes the detailed parameters is as follows:

```
HTTP/1.1 200 OK
…
Content-Type: application/xml; charset=”UTF-8”
Content-Length: 66
Connection: close
…
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
…
<deviceInfo>
<supportTalk type="boolean">true</supportTalk>
        …
</deviceInfo>
</config>
```

When a request cannot be executed correctly, an application fail response that includes an error result in the entity body will be sent from the IP device. Meantime, the HTTP answer is 400 to indicate the client application. For example:

```
HTTP/1.1 400 Bad Request
```

...

Content-Type: application/xml

Content-Length: 66

Connection: close

<?xml version="1.0" encoding="utf-8" ?>

<config status="failed" errorCode="1"/>

The detailed “errorCode” will be described in the following section.

1.3.6 Error Code

Error Code	Description
1	“Invalid Request”: The request URL is not supported by the device. There is something wrong with “cmd name”, “channelId”, or “action name”.
2	“Invalid XML Format”: The entity’s XML format is not recognized by the system.
3	“Invalid XML Content”:An incomplete message or a message containing some out-of-range parameters.
4	Permission denied
5	Network port num error

1.4 Protocol Conventions

1.4.1 XML Element Name

There will be several words in one element name, in this case, the first letter of the first word should be in lower case, the first letter of other words should be in upper case, and all other letters should be in lower case.

1.4.2 XML Element Type

Each element has an attribute “type”, which defines the data type of the element. The basic data types are listed as follows:

Type	Description
boolean	The same as “bool” in C++, available value is “true” or “false”.
int8	8 bit integer, the same as “char” in C/C++.
uint8	Unsigned 8 bit integer, the same as “unsigned char” in C/C++.
int16	16 bit integer, the same as “short” in C/C++.
uint16	Unsigned 16 bit integer, the same as “unsigned short” in C/C++.
int32	32 bit integer, the same as “long” in C/C++.
uint32	Unsigned 32 bit integer, the same as “unsigned long” in C/C++.
int64	64 bit integer, the same as “long long” in C/C++.
uint64	Unsigned 64 bit integer, the same as “unsigned long long” in C/C++.
string	A string of characters, like the “string” in C++.

Type	Description
list	List of basic or advanced types.

For the element with type “int8/uint8/int16/uint16/int32/uint32/int64/uint64”, two more attributes “min” and “max” can be optional, which mean the minimum and maximum value of this element. For example:

```
<bright type="uint8" min="0" max="100" default="50">50</bright>
```

For the element with type “string” attribute, two more attributes “minLen” and “maxLen” are optional, which mean the minimum and maximum length of the character string. When the type “string”attribute is used, the string itself should be packed in the CDATA segment. For example:

```
<ntpServer type="string" minLen="0" maxLen="127" default="time.windows.com"><![CDATA[time.windows.com]]></ntpServer>
```

For the element with type “list” attribute, the attribute “maxCount” should be used for the variable list, which means the maximum item counts for this list, and the attribute “count” should be used for the list with constant items. There should be an “itemType” sub element after the element with type “list” attribute. Some “item” sub element should be included after the “itemType” sub element to indicate the value for the list. For example:

```
<content type="list" count="6">
<itemType type="string" minLen="0" maxLen="32" default="00000000000000000000000000000000"/>
<item><![CDATA[11111111111111111111]]></item>
<item><![CDATA[22222222222222222222]]></item>
<item><![CDATA[33333333333333333333]]></item>
<item><![CDATA[44444444444444444444]]></item>
<item><![CDATA[55555555555555555555]]></item>
<item><![CDATA[66666666666666666666]]></item>
```

```
</content>
```

1.4.3 The "types" Element

When the basic data types cannot meet the demands, the "types" element should be used to define advanced data types. We don't define any advanced data types in this document. Either, all advanced data types that will be used in a message should be defined in the message body. This means **" The messages themselves are documents"**.

In the "types" element, only the "enum" type can be defined. For example, an "enum" type is defined as follows:

```
<types>
    <userType>
        <enum>administrator</enum>
        <enum>advance</enum>
        <enum>normal</enum>
    </userType>
</types>
```

It is not allowed for the client application to define advanced data types with the "types" element in request messages. The client application should study advanced data types from the response messages. Advanced data types defined in the corresponding response message can be used directly in a request message by the client application. The Client application can also study advanced data types from other elements except for "types" in the message entity from the device.

1.4.4 Command catagory

We divide all commands into different categories that will be detailed in the following paragraphs.

System commands.

Image commands.

PTZ commands.

Alarm commands.

Playback commands

Network commands.

Security commands.

Maintain commands.

Talkback commands

Smart commands

Schedule commands

1.5 Device discovery

The IP media devices support UPnP protocol for device discovery.

The IP devices support Universal Plug and Play (UPnP) technology to discovery/locate themselves. An UPnP compatible device will automatically announce its network address supported devices and services types when connected to a network, therefore becoming “plug-and-play” by allowing clients recognize those information and begin using this device immediately.

The UPnP architecture supports zero-configuration networking, and the device can dynamically join a network, obtain IP address, announce its name, convey its capabilities upon request, and gets the on-line status and capabilities of other devices. DHCP and DNS servers are optional and are only used if they are available on the network. Devices can leave the network automatically without leaving any unwanted status information behind. UPnP was published as a 73-part International Standard, ISO/IEC 29341, in December, 2008 [6][7][8].

After a control point has discovered a device, the control point still needs more operations to request more information about the device or to interact with it.

2 System commands

2.1 Device Information

2.1.1 GetDeviceInfo

GetDeviceInfo	
Description	To get the IP media device's information.
Typical URL	POST or GET http://<host>[:port]/GetDeviceInfo
Channel ID	None
Action name	None
Entity Data	None
Successful Response	The device information will be included in the entity of the successful response. For example:

GetDeviceInfo

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
    <deviceInfo>
        <deviceName type="string"><![CDATA[212]]></deviceName>
        <model type="string"><![CDATA[TD-9421M]]></model>
        <brand type="string"><![CDATA[IPC]]></brand>
        <deviceDescription type="string"><![CDATA[IPCamera]]></deviceDescription>
        <audioInCount type="uint32">1</audioInCount>
        <audioOutCount type="uint32">1</audioOutCount>
        <integratedPtz type="boolean">true</integratedPtz>
        <supportRS485Ptz type="boolean">false</supportRS485Ptz>
        <supportSDCard type="boolean">true</supportSDCard>
        <alarmInCount type="uint32">1</alarmInCount>
        <alarmOutCount type="uint32">1</alarmOutCount>
        <softwareVersion type="string"><![CDATA[4.0.0 beta1]]></softwareVersion>
        <softwareBuildDate type="string"><![CDATA[2013-12-24]]></softwareBuildDate>
        <kernelVersion type="string"><![CDATA[20111010]]></kernelVersion>
        <hardwareVersion type="string"><![CDATA[1.3]]></hardwareVersion>
        <mac type="string"><![CDATA[00:18:ae:98:38:fd]]></mac>
        <sn type="string"><![CDATA[2E323D9463D5]]></sn>
        <chlMaxCount type="uint32">9</chlMaxCount>
    </deviceInfo>
</config>
```

[Tips]:

This command is designed for the client application to obtain the basic information from the specific media device.

- For the fixed-channel devices such as IPC or DVR, the items “audioInCount”, “audioOutCount”, “alarmInCount” and “alarmOutCount” will be included in the successful response.
- For the variable-channel devices such as NVR, these items are optional. The client application can use “GetChannelList”, “GetAlarmInList”, “GetAlarmOutList”, “GetStreamCpas” commands to obtain the information.

2.1.2 GetDiskInfo

GetDiskInfo	
Description	To get the IP media device's disk information.
Typical URL	POST or GET http://<host>[:port]/GetDiskInfo
Channel ID	None
Action name	None
Entity Data	None
Successful Response	The device information will be included in the entity of the successful response. For example:

GetDiskInfo

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
    <types>
        <diskStatus>
            <enum>read</enum>
            <enum>read/write</enum>
            <enum>unformat</enum>
            <enum>formatting</enum>
            <enum>exception</enum>
        </diskStatus>
    </types>
    <diskInfo type="list" count="1">
        <item>
            <id type="string"><![CDATA[{5B457B2A-D467-834E-B1E8-22F3450DA873}]]></id>
            <totalSpace type="uint32">953869</totalSpace>
            <freeSpace type="uint32">847872</freeSpace>
            <imageFreeSpace type="uint32">847872</imageFreeSpace>
            <diskStatus type="diskStatus">read/write</diskStatus>
        </item>
    </diskInfo>
</config>
```

[Tips]:

The “totalSpace” and “freeSpace” are in mb.

There is empty “diskInfo” node if there is no disk on device.

The enums, “read”, “read/write” and “unformat”, are supported by NVR and DVR.

The enums, “read/write”, “unformat”, “formatting” and “exception”, are supported by IPC.

The “imageFreeSpace” is supported by IPC only.

2.1.3 GetChannelList

GetChannelList	
Description	To get the IP media device's channel list.
Typical URL	POST or GET http://<host>[:port]/GetChannelList
Channel ID	None
Action name	None
Entity Data	None
Successful Response	The channel list will be included in the entity of the successful response. For example:

GetChannelList
<?xml version="1.0" encoding="UTF-8"?> <config version="1.0" xmlns="http://www.ipc.com/ver10"> <types> <channelStatus> <enum>online</enum> <enum>offline</enum> <enum>videoOn</enum> <enum>videoLoss</enum> </channelStatus> </types> <channelIDList type="list" count="4"/> <itemType type="string" maxLen="20"/> <item channelStatus="online">1</item> <item channelStatus="online">2</item> <item channelStatus="online">3</item> <item channelStatus="online">4</item> </config> --- [Tips]: This command is designed for multi-channel device and not mandatory for IP cameras. If the "deviceDescription" item is equal to "IPCamera" in the response message for "GetDeviceInfo" command, this command should not be sent to the device.

2.1.4 GetAlarmInList

GetAlarmInList	
Description	To get the IP media device's alarmin list.
Typical URL	POST or GET http://<host>[:port]/GetAlarmInList
Channel ID	None
Action name	None

GetAlarmInList	
Entity Data	None
Successful Response	The alarmin list will be included in the entity of the successful response. For example:

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

<config version="1.0" xmlns="http://www.ipc.com/ver10">

<alarmInIDList type="list" count="8"></alarmInIDList>

<itemType type="string" maxLen="20"/>

<item>1</item>

<item>2</item>

<item>3</item>

<item>4</item>

<item>5</item>

<item>6</item>

<item>7</item>

<item>8</item>

</config>

[Tips]:

This command is designed for multi-channel device and not mandatory for IP cameras. If the "deviceDescription" item is equal to "IPCamera" in the response message for "GetDeviceInfo" command, this command should not be sent to the device.

2.1.5 GetAlarmOutList

GetAlarmOutList	
Description	To get the IP media device's alarmout list.
Typical URL	POST or GET http://<host>[:port]/GetAlarmOutList
Channel ID	None
Action name	None

GetAlarmOutList	
Entity Data	None
Successful Response	The alarmout list will be included in the entity of the successful response. For example:

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

<config version="1.0" xmlns="http://www.ipc.com/ver10">

<alarmOutIDList type="list" count="4"></alarmOutIDList>

<itemTypetype="string" maxLen="20"/>

<item>1</item>

<item>2</item>

<item>3</item>

<item>4</item>

</config>

[Tips]:

This command is designed for multi-channel device and not mandatory for IP cameras. If the "deviceDescription" item is equal to "IPCamera" in the response message for "GetDeviceInfo" command, this command should not be sent to the device.

2.1.6 GetDeviceDetail

GetDeviceDetail	
Description	To get device's details.
Typical URL	POST or GET http://<host>[:port]/GetDeviceDetail
Channel ID	None
Action name	None
Entity Data	None
Successful Response	The device detail will be included in the entity of the successful response. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
    <detail>
        <property>
            <deviceName type="string"><![CDATA[IPC]]></deviceName>
            <deviceDescription type="string"><![CDATA[IPCamera]]></deviceDescription>
            <model type="string"><![CDATA[TD-9421M]]></model>
            <brand type="string"><![CDATA[IPC]]></brand>
            <sn type="string"><![CDATA[2E323D9463D5]]></sn>
            <mac type="string"><![CDATA[00:18:ae:98:38:fd]]></mac>
            <softwareVersion type="string"><![CDATA[4.0.0 beta1]]></softwareVersion>
            <softwareBuildDate type="string"><![CDATA[2013-12-24]]></softwareBuildDate>
            <kernelVersion type="string"><![CDATA[20111010]]></kernelVersion>
            <hardwareVersion type="string"><![CDATA[1.3]]></hardwareVersion>
            <apiVersion type="string"><![CDATA[1.7]]></apiVersion>
        </property>
        <smart>
            <supportTripwire type="boolean">false</supportTripwire>
            <supportPerimeter type="boolean">false</supportPerimeter>
            <supportOsc type="boolean">false</supportOsc>
            <supportAvd type="boolean">false</supportAvd>
            <supportVfd type="boolean">false</supportVfd>
            <supportCpc type="boolean">false</supportCpc>
            <supportCdd type="boolean">false</supportCdd>
            <supportIpd type="boolean">false</supportIpd>
            <supportVfdMatch type="boolean">false</supportVfdMatch>
            <supportvehicle type="boolean">false</supportvehicle>
            <supportAoiEntry type="boolean">false</supportAoiEntry>
            <supportAoiLeave type="boolean">false</supportAoiLeave>
            <supportPassLineCount type="boolean">false</supportPassLineCount>
            <supportThermal type="boolean">true</supportThermal>
        </smart>
```

```
        <image>
            <supportAZ type="boolean">true</supportAZ>
            <supportROI type="boolean">true</supportROI>
            <supportInfraredLamp type="boolean">false</supportInfraredLamp>
            <supportWatermark type="boolean">true</supportWatermark>
            <supportPrivateMask type="boolean">true</supportPrivateMask>
        </image>
        <alarm>
            <supportMultiMotionSensitivity type="boolean">false</supportMultiMotionSensitivity>
            <supportAlarmServer type="boolean">false</supportAlarmServer>
            <alarmInCount type="uint32">1</alarmInCount>
            <alarmOutCount type="uint32">1</alarmOutCount>
            <supportAudioAlarmOut type="boolean">false</supportAudioAlarmOut>
            <supportWhiteLightAlarmOut type="boolean">false</supportWhiteLightAlarmOut>
        </alarm>
        <system>
            <supportSnmp type="boolean">true</supportSnmp>
            <audioInCount type="uint32">1</audioInCount>
            <audioOutCount type="uint32">1</audioOutCount>
            <integratedPtz type="boolean">true</integratedPtz>
            <supportRS485Ptz type="boolean">false</supportRS485Ptz>
            <supportSDCard type="boolean">true</supportSDCard>
            <chlMaxCount type="uint32">9</chlMaxCount>
        </system>
    </detail>
</config>
```

[Tips]:

2.2 Date and Time

2.2.1 GetDateAndTime

GetDateAndTime	
Description	To get the IP media device's system date and time.
Typical URL	POST or GEThttp://<host>[:port]/GetDateAndTime
Channel ID	None
Action name	None
Entity Data	None
Successful Response	The device time and date will be included in the entity of the Successful response. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
    <types>
        <synchronizeType>
            <enum>manually</enum>
            <enum>NTP</enum>
        </synchronizeType>
    </types>
    <time>
        <timezoneInfo>
            <timeZone type="string"><![CDATA[GMT0BST,M3.5.0/1,M10.5.0]]></timeZone>
            <daylightSwitch type="boolean">false</daylightSwitch>
        </timezoneInfo>
        <synchronizeInfo>
            <type type="synchronizeType">manually</type>
            <ntpServer type="string" maxLen="127"><![CDATA[time.windows.com]]></ntpServer>
            <currentTime type="string"><![CDATA[2014-01-09 15:07:28]]></currentTime>
```

GetDateAndTime
</synchronizeInfo> </time> </config> [Tips]: The element “timeZone” announces the time zone information. “GMT0BST,M3.5.0/1,M10.5.0”, this time zone, standard time named GMT and daylight saving time named BST, has daylight saving time. The standard local time is GMT. Daylight saving time, 1 hour ahead of GMT, starts the last Sunday in March at 01:00 and ends the last Sunday in October at 02:00.

2.2.2 SetDateAndTime

SetDateAndTime	
Description	To set the IP media device’s system date and time.
Typical URL	POST http://<host>[:port]/SetDateAndTime
Channel ID	None
Action name	None
Entity Data	The device time and date will be included in the entity of request message. The whole “time” element in the “GetDataAndTime” should be included in entity of this message. Any attributes for the “time” element or sub elements should not be included.
Successful Response	The standard successful result response that described in 1.3.5.

3 Image commands

3.1 Stream Capabilities

3.1.1 GetStreamCaps

GetStreamCaps	
Description	To get the IP media device's streams' capabilities for specific channel.
Typical URL	POST or GEThttp://<host>[:port]/GetStreamCaps[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The stream capabilities will be included in the entity of the Successful response. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
<types>
<resolution>
    <enum>1920x1080</enum>
        <enum>1280x720</enum>
        <enum>704x576</enum>
        <enum>352x288</enum>
</resolution>
<encodeType>
```

GetStreamCaps

```
<enum>h264</enum>
<enum>mpeg4</enum>
<enum>mjpeg</enum>
<enum>h264plus</enum>
<enum>h265plus</enum>
<enum>h264smart</enum>
<enum>h265smart</enum>
                ...
</encodeType>
<encodeLevel>
<enum>baseLine</enum>
<enum>mainProfile</enum>
<enum>highProfile</enum>
</encodeLevel>
</types>
<rtspPort type="uint16">554</rtspPort>
<streamList type="list" count="4">
<item id="1">
<streamName type="string"><![CDATA[profile1]]></streamName>
<resolutionCaps type="list" count="1">
<itemType type="resolution"/>
<item maxFrameRate="25">1920x1080</item>
</resolutionCaps>
<encodeTypeCaps type="list" count="1">
<itemType type="encodeType"/>
<item>h264</item>
</encodeTypeCaps>
<encodeLevelCaps type="list" count="3">
<itemType type="encodeLevel"/>
<item>baseLine</item>
<item>mainProfile</item>
```

GetStreamCaps

<item>highProfile</item>

</encodeLevelCaps>

</item>

<item id="2">

<streamName type="string"><![CDATA[profile2]]></streamName>

<resolutionCaps type="list" count="3">

<itemType type="resolution"/>

<item maxFrameRate="10">1920x1080</item>

<item maxFrameRate="25">1280x720</item>

<item maxFrameRate="25">704x480</item>

</resolutionCaps>

<encodeTypeCaps type="list" count="1">

<itemType type="encodeType"/>

<item>h264</item>

</encodeTypeCaps>

<encodeLevelCaps type="list" count="3">

<itemType type="encodeLevel"/>

<item>baseLine</item>

<item>mainProfile</item>

<item>highProfile</item>

</encodeLevelCaps>

</item>

…

</streamList>

</config>

[Tips]:

The “count=4” means the channel supports 4 streams at the same time. Each stream’s capability is announced in the “item” sub element. The “streamName” announces the name of each stream. The client application, can obtain the specific stream of NVR/DVR by the following URL.

rtsp://<host><:port>?chID=<channelId>&streamType=<streamType>

“streamtype” can be main or sub

GetStreamCaps
The client application, can obtain the specific stream of IPC by the following URL. rtsp://<host><:port>/<streamName> The “resolutionCaps” announces optional combinations for frame rate and resolution. The “encodeTypeCaps” announces optional compression types. The “encodeLevelCaps” optional compression levels. For the reason that the capabilities for each stream are not the same, we omit the “itemType” element after the “streamList” element. The “id” attribute for each item starts from “1”.

3.2 Image Configuration

3.2.1 GetImageConfig

GetImageConfig	
Description	To get the IP media device’s image configuration for specific channel.
Typical URL	POST or GEThttp://<host>[:port]/GetImageConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The image configuration will be included in the entity of the Successful response. For example:
<?xml version="1.0" encoding="UTF-8"?> <config version="1.0" xmlns="http://www.ipc.com/ver10"> <types> <frequency> <enum>60HZ</enum> <enum>50HZ</enum> </frequency> <whitebalanceMode>	

GetImageConfig

```
<enum>auto</enum>
<enum>manual</enum>
<enum>outdoor</enum>
<enum>indoor</enum>
</whitebalanceMode>
<IRCutMode>
<enum>auto</enum>
<enum>day</enum>
<enum>night</enum>
</IRCutMode>
</types>
<image>
<frequency type="frequency" default="50HZ">50HZ</frequency>
<bright type="uint8" min="0" max="100" default="50">50</bright>
<contrast type="uint8" min="0" max="100" default="55">55</contrast>
<hue type="uint8" min="0" max="100" default="50">50</hue>
<saturation type="uint8" min="0" max="100" default="50">50</saturation>
<mirrorSwitch type="boolean" default="false">false</mirrorSwitch>
<flipSwitch type="boolean" default="false">false</flipSwitch>
<WDR>
    <switch type="boolean" default="false">false</switch>
    <value type="uint8" default="128">128</value>
</WDR>
<whiteBalance>
<mode type="whitebalanceMode" default="auto">auto</mode>
<red type="uint32" min="0" max="100" default="50">50</red>
<blue type="uint32" min="0" max="100" default="50">50</blue>
</whiteBalance>
<denoise>
<switch type="boolean" default="false">false</switch>
    <value type="uint8" default="24">24</value>
```

GetImageConfig
</denoise> <irisSwitch type="boolean" default="false">false</irisSwitch> <sharpen> <switch type="boolean" default="true">true</switch> <value type="uint8" default="80">80</value> </sharpen> <IRCutMode type="IRCutMode" default="auto">auto</IRCutMode> <backLightAdjust> <switch type="boolean" default="true">true</switch> <value type="uint8" min="150" max="255" default="200">200</value> </backLightAdjust> </image> </config>

3.2.2 SetImageConfig

SetImageConfig	
Description	To set the IP media device's image configuration for specific channel.
Typical URL	POST http://<host>[:port]/SetImageConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The image configuration for specific channel should be included in the entity of request message. The whole "image" element in the "GetImagConfig" or some parameters that need to be changed can be included in entity of this message. Any attributes for the "image" element or sub elements should not be included. The following example changes the "saturation" parameter.

SetImageConfig	
<?xml version="1.0"?> <config version="1.0" xmlns="http://www.ipc.com/ver10"> <image> <saturation>65</saturation> </image> </config>	
Successful Response	The standard successful result response that described in 1.3.5.

3.2.3 GetSnapshot

GetSnapshot	
Description	To get a picture encoded by jpg for specific channel.
Typical URL	POST or GEThttp://<host>[:port]/GetSnapshot[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	A picture encoded by jpg.

3.2.4 GetSnapshotByTime

GetSnapshotByTime	
Description	To get a key frame for specific channel on specific time.
Typical URL	POST or GEThttp://<host>[:port]/GetSnapshotByTime[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.

GetSnapshotByTime	
Action name	None
Entity Data	Optional.The time be included in the entity of the request message as search history picture. For example: <?xml version="1.0" encoding="UTF-8"?> <config version="1.0" xmlns="http://www.ipc.com/ver10"> <search> <time type="string"><![CDATA[2017-01-09 15:07:28]]></time> <length type = “uint16”>10</length> </search> </config>
Successful Response	The snapshot data from the specific time.
[Tips]: 1. It returns the data from “time” in “length” seconds. 2. The response maybe one key frame of H.264 or H.265, or a picture encoded by jpg. Get the type from the http head content-Type.	

3.3 Stream Configuration

3.3.1 GetAudioStreamConfig

GetAudioStreamConfig	
Description	To get the IP media device’s audio stream configuration for specific channel.
Typical URL	POST or GET http://<host>[:port]/GetAudioStreamConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None

GetAudioStreamConfig	
Entity Data	None
Successful Response	The audio stream configuration will be included in the entity of the Successful response. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
<types>
<audioEncode>
<enum>G711A</enum>
<enum>G711U</enum>
<enum>G726</enum>
</audioEncode>
<audioInput>
    <enum>MIC</enum>
    <enum>LIN</enum>
</audioInput>
<audioOutput>
    <enum>TALKBACK</enum>
    <enum>ALARM_AUDIO</enum>
</audioOutput>
</types>
<audioInSwitch type="boolean">true</audioInSwitch>
<audioEncode type="audioEncode">G711A</audioEncode>
<audioInput type="audioInput">MIC</audioInput>
<audioOutput type="audioOutput">TALKBACK</audioOutput>
<loudSpeaker type="audioOutput">ALARM_AUDIO</loudSpeaker>
</config>
```

3.3.2 SetAudioStreamConfig

SetAudioStreamConfig

SetAudioStreamConfig	
Description	To set the IP media device's audio stream configuration for specific channel.
Typical URL	POST http://<host>[:port]/SetAudioStreamConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The audio stream configuration for specific channel should be included in the entity of request message. The whole "audioEncode" element in the "GetAudioStreamConfig" can be included in entity of this message. Any attributes for the "audioEncode" element or sub elements should not be included.
Successful Response	The standard successful result response that described in 1.3.5.

3.3.3 GetVideoStreamConfig

GetVideoStreamConfig	
Description	To get the IP media device's video stream configuration for specific channel.
Typical URL	POST or GEThttp://<host>[:port]/GetVideoStreamConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The video stream configuration will be included in the entity of the successful response. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
<types>
<bitRateType>
<enum>VBR</enum>
<enum>CBR</enum>
```

GetVideoStreamConfig

```
</bitRateType>
<quality>
<enum>lowest</enum>
<enum>lower</enum>
<enum>medium</enum>
<enum>higher</enum>
<enum>highest</enum>
</quality>
<encodeType>
    <enum>h264</enum>
    <enum>h265</enum>
    <enum>h264plus</enum>
    <enum>h265plus</enum>
    <enum>h264smart</enum>
    <enum>h265smart</enum>
    <enum>mjpeg</enum>
</encodeType>
</types>
<streams type="list" count="4">
<item id="1">
<name type="string" maxLen="32"><![CDATA[profile1]]></name>
<resolution>1920x1080</resolution>
<frameRate type="uint32">25</frameRate>
<bitRateType type="bitRateType">CBR</bitRateType>
<maxBitRate type="uint32" min="64" max="12288">4096</maxBitRate>
<bitRateLists>
<item>2048</item>
<item>3072</item>
<item>4096</item>
<item>6144</item>
<item>8192</item>
```

GetVideoStreamConfig

```
</bitRateLists>
<encodeTypeCaps type="list">
<itemType type="encodeType" />
<item>h264</item>
<item>h265</item>
<item>h264plus</item>
<item>h265plus</item>
<item>h264smart</item>
<item>h265smart</item>
<item>mjpeg</item>
</encodeTypeCaps>
<encodeType>h264</encodeType>
<encodeLevel>baseLine</encodeLevel>
<quality type="quality">highest</quality>
<GOP type="uint32" min="30" max="200">100</GOP>
</item>
<item id="2">
<name type="string" maxLen="32"><![CDATA[profile2]]></name>
<resolution>1280x720</resolution>
<frameRate type="uint32">25</frameRate>
<bitRateType type="bitRateType">CBR</bitRateType>
<maxBitRate type="uint32" min="64" max="10240">2048</maxBitRate>
<bitRateLists>
<item>256</item>
<item>512</item>
<item>768</item>
<item>1024</item>
<item>2048</item>
</bitRateLists>
<encodeTypeCaps type="list">
<itemType type="encodeType" />
```

GetVideoStreamConfig
<item>h264</item> <item>h265</item> <item>h264plus</item> <item>h265plus</item> <item>mjpeg</item> </encodeTypeCaps> <encodeType>h264</encodeType> <encodeLevel>baseLine</encodeLevel> <quality type="quality">highest</quality> <GOP type="uint32" min="30" max="200">100</GOP> </item> ... </streams> </config> [Tips]: The “count=4” means the channel supports 4 streams at the same time. Each stream’s current video configuration is announced in the “item” sub element. The value of each stream’s “resolution”, “framRate”, “encodeType”, and “encodeLevel” should be in the scope of the corresponding capability announced in the “GeteStreamCaps” successful respond message. The “maxBitRate” element means the bitrate in kbps. The “id” attribute for each item starts from “1”.

3.3.4 SetVideoStreamConfig

SetVideoStreamConfig	
Description	To set the IP media device’s video stream configuration for specific channel.
Typical URL	POST http://<host>[:port]/SetVideoStreamConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None

SetVideoStreamConfig	
Entity Data	The video stream configuration for specific channel should be included in the entity of request message. The whole “streams” element in the “GetVideoStreamConfig” can be included in entity of this message. Any attributes for the “streams” element or sub elements should not be included. The value of each stream’s “resolution”, “framRate”, “encodeType”, and “encodeLevel” should be in the scope of the corresponding capability announced in the “GeteStreamCaps” successful respond message.
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]: IPC does not support	

3.3.5 RequestKeyFrame

RequestKeyFrame	
Description	It is used to request the device to encode a key frame for specific channel.
Typical URL	POST or GEThttp://<host>[:port]/RequestKeyFrame [/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The standard successful result response that described in 1.3.5.

3.4 OSD

3.4.1 GetImageOsdConfig

GetImageOsdConfig	
Description	To get the IP media device’s image OSD configuration for specific channel.

GetImageOsdConfig	
Typical URL	POST or GET http://<host>[:port]/GetImageOsdConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The image OSD configuration will be included in the entity of the Successful response. For example:

GetImageOsdConfig

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
<types>
<dateFormat>
<enum>year-month-day</enum>
<enum>month-day-year</enum>
<enum>day-month-year</enum>
</dateFormat>
</types>
<imageOsd>
<time>
<switch type="boolean">true</switch>
<X type="uint32">0</X>
<Y type="uint32">0</Y>
<dateFormat type="dateFormat">year-month-day</dateFormat>
</time>
<channelName>
<switch type="boolean">false</switch>
<X type="uint32">0</X>
<Y type="uint32">0</Y>
<name type="string" maxLen="19"><![CDATA[name]]></name>
</channelName>
</imageOsd>
</config>
```

[Tips]:

The “X” and “Y” element announce the horizontal and vertical position based in the 10000*10000 resolution.

3.4.2 SetImageOsdConfig

SetImageOsdConfig	
Description	To set the IP media device's image OSD configuration for specific channel.
Typical URL	POST http://<host>[:port]/SetImageOsdConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The image OSD configuration for specific channel should be included in the entity of request message. The whole "imageOsd" element in the "GetImagOsdConfig" or some parameters that need to be changed can be included in entity of this message. Any attributes for the "imageOsd" element or sub elements should not be included. The following example changes the "channelName" element:
<?xml version="1.0" encoding="UTF-8"?> <config version="1.0" xmlns="http://www.ipc.com/ver10"> <imageOsd> <channelName> <switch>true</switch> <X>100</X> <Y>100</Y> <name><![CDATA[camera01]]></name> </channelName> </imageOsd> </config>	
Successful Response	The standard successful result response that described in 1.3.5.

3.5 Privacy Mask

3.5.1 GetPrivacyMaskConfig

GetPrivacyMaskConfig	
Description	To get the IP media device's privacy mask configuration for specific channel.
Typical URL	POST or GET http://<host>[:port]/GetPrivacyMaskConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The privacy mask configuration will be included in the entity of the Successful response. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
<types>
<color>
<enum>black</enum>
<enum>white</enum>
<enum>gray</enum>
</color>
</types>
<privacyMask type="list" count="4">
<itemType>
<switch type="boolean"/>
<rectangle>
<X type="uint32"/>
<Y type="uint32"/>
<width type="uint32"/>
<height type="uint32"/>
```

GetPrivacyMaskConfig
</rectangle> <color type="color"/> </itemType> <item> <switch>false</switch> <rectangle> <X>0</X> <Y>0</Y> <width>0</width> <height>0</height> </rectangle> <color>black</color> </item> ... </privacyMask> </config> [Tips]: The “X” and “Y” element announce the horizontal and vertical position based in the 640*480 resolution.

3.5.2 SetPrivacyMaskConfig

SetPrivacyMaskConfig	
Description	To set the IP media device’s privacy mask configuration for specific channel.
Typical URL	POST http://<host>[:port]/SetPrivacyMaskConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None

SetPrivacyMaskConfig	
Entity Data	The privacy mask configuration for specific channel should be included in the entity of request message. The whole “privacyMask” element in the “GetPrivacyMaskConfig” should be included in entity of this message. Any attributes for the “privacyMask” element or sub elements should not be included.
Successful Response	The standard successful result response that described in 1.3.5.

4 PTZ commands

4.1 Protocol

4.1.1 PtzGetCaps

PtzGetCaps	
Description	To get the IP media device’s PTZ capabilities mask information for specific channel.
Typical URL	POST or GET http://<host>[:port]/PtzGetCaps[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None

PtzGetCaps	
Successful Response	The PTZ capabilities will be included in the entity of the Successful response. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
    <caps>
        <controlMinSpeed type="uint32">1</controlMinSpeed>
        <controlMaxSpeed type="uint32">8</controlMaxSpeed>
        <presetMaxCount type="uint32">255</presetMaxCount>
        <cruiseMaxCount type="uint32">8</cruiseMaxCount>
        <cruisePresetMinSpeed type="uint32">1</cruisePresetMinSpeed>
        <cruisePresetMaxSpeed type="uint32">8</cruisePresetMaxSpeed>
        <cruisePresetMaxHoldTime type="uint32">240</cruisePresetMaxHoldTime>
        <cruisePresetMaxCount type="uint32">16</cruisePresetMaxCount>
    </caps>
</config>
```

[Tips]:

The sub elements in the “caps” element announce the scope of each parameter. For example, the “ptzControlMinSpeed” announce the minimum speed for the PTZ control command, the “ptzControlMaxSpeed” announce the maximum speed for the PTZ control command.

4.1.2 GetPtzConfig

GetPtzConfig	
Description	To get the IP media device’s PTZ protocol configuration for specific channel.
Typical URL	POST or GET http://<host>[:port]/GetPtzConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None

GetPtzConfig	
Successful Response	The PTZ protocol configuration will be included in the entity of the Successful response. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.7" xmlns="http://www.ipc.com/ver10">
<types>
<language>
<enum>en</enum>
<enum>cn</enum>
</language>
<autoExitTime>
<enum>off</enum>
<enum>15sec</enum>
<enum>30sec</enum>
<enum>60sec</enum>
<enum>90sec</enum>
<enum>120sec</enum>
</autoExitTime>
<protocol>
<enum>PELCOP</enum>
<enum>PELCOD</enum>
</protocol>
<baudRate>
<enum>1200</enum>
<enum>2400</enum>
<enum>4800</enum>
<enum>9600</enum>
</baudRate>
</types>
<ptzSettings>
<autoPtzFlip type="boolean">true</autoPtzFlip>
<language type="string">en</language>
<autoExitTime type="string">off</autoExitTime>
<rs485>
<idType>SW</idType>
<demoId min="0" max="255">1</demoId>
<protocol type="string"><![CDATA[PELCOD]]></protocol>
<baudRate type= "baudRate">2400</baudRate>
</rs485>
</ptzSettings>
</config>
```

4.1.3 SetPtzConfig

SetPtzConfig

SetPtzConfig	
Description	To set the IP media device's PTZ protocol configuration for specific channel.
Typical URL	POST http://<host>[:port]/SetPtzConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The PTZ protocol configuration for specific channel should be included in the entity of request message. The whole "ptzSettings" element in the "**GetPtzConfig**" should be included in entity of this message. Any attributes for the "ptzSettings" element or sub elements should not be included.
Successful Response	The standard successful result response that described in 1.3.5.

4.2 PTZ Control

4.2.1 PtzControl

PtzControl	
Description	To start control PTZ for a specific channel of the IP media device.
Typical URL	POST http://<host>[:port]/PtzControl[/channelId]</action_name>
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.

PtzControl	
Action name	Up: to move up Down: to move down Left: to move left Right: to move right LeftUp: to move left and up LeftDown: to move left and down RightUp: to move right and up RightDown: to move right and down Near: to focus near Far: to focus far ZoomIn: to zoom in ZoomOut: to zoom out IrisOpen: to open the iris IrisClose: to close the iris Stop: to stop current action
Entity Data	The PTZ's action information that needs to be executed will be included in the entity of the request message. For example:
<?xml version="1.0" encoding="utf-8" ?> <actionInfo version="1.0" xmlns="http://www.ipc.com/ver10"> <speed>4</speed> </actionInfo> [Tips]: The value of "speed" should be in the scope of the corresponding capability announced in the "PtzGetCaps" successful respond message.	
Successful Response	The standard successful result response that described in 1.3.5.

4.2.2 PtzGotoPreset

PtzGotoPreset

PtzGotoPreset	
Description	To run the PTZ to one preset for a specific channel of the IP media device.
Typical URL	POST http://<host>[:port]/PtzGotoPreset[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The destination PTZ preset's ID will be included in the entity of the request message. For example:
<?xml version="1.0" encoding="utf-8" ?> <presetInfo version="1.0" xmlns="http://www.ipc.com/ver10"> <id>2</id> </presetInfo>	
Successful Response	The standard successful result response that described in 1.3.5.

4.2.3 PtzRunCruise

PtzRunCruise	
Description	To run one PTZ's cruise for a specific channel of the IP media device.
Typical URL	POST http://<host>[:port]/PtzRunCruise[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The PTZ cruise's ID that needs to be run will be included in the entity of the request message. For example:
<?xml version="1.0" encoding="utf-8" ?> <cruiseInfo version="1.0" xmlns="http://www.ipc.com/ver10"> <id>1</id> </cruiseInfo>	
Successful Response	The standard successful result response that described in 1.3.5.

4.2.4 PtzStopCruise

PtzStopCruise	
Description	To stop the PTZ cruise for a specific channel of the IP media device.
Typical URL	POST http://<host>[:port]/PtzStopCruise[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The standard successful result response that described in 1.3.5.

4.3 Preset

4.3.1 PtzGetPresets

PtzGetPresets	
Description	To get the IP media device's PTZ presets list for specific channel.
Typical URL	POST or GET http://<host>[:port]/PtzGetPresets[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The PTZ presets list will be included in the entity of the Successful response. For example:

PtzGetPresets
<?xml version="1.0" encoding="UTF-8"?> <config version="1.0" xmlns="http://www.ipc.com/ver10"> <presetInfo type="list" maxCount="360"> <itemType type="string" maxLen="10"></itemType> <item id="1"><![CDATA[DDD]]></item> </presetInfo> </config> --- [Tips]: The “id” attribute for each item starts from “1”.

4.3.2 PtzAddPreset

PtzAddPreset	
Description	To add one preset for a specific channel of the IP media device.
Typical URL	POST http://<host>[:port]/PtzAddPreset[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The PTZ preset name for current position will be included in the entity of the request message. The “name” should accord with the “name” type in the “itemType” that announced in “PtzGetPresets” message. For example:
<?xml version="1.0" encoding="utf-8" ?> <presetInfo version="1.0" xmlns="http://www.ipc.com/ver10"> <name><![CDATA[dd]]></name> </presetInfo>	
Successful Response	The standard successful result response that described in 1.3.5.

4.3.3 PtzModifyPresetName

PtzModifyPresetName	
Description	To modify one preset's name for a specific channel of the IP media device.
Typical URL	POST http://<host>[:port]/PtzModifyPresetName[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The PTZ preset's ID and new name will be included in the entity of the request message. For example:
<?xml version="1.0" encoding="utf-8" ?> <presetInfo version="1.0" xmlns="http://www.ipc.com/ver10"> <id>1</id> <name><![CDATA[aa1]]></name> </presetInfo>	
Successful Response	The standard successful result response that described in 1.3.5.

4.3.4 PtzDeletePreset

PtzDeletePreset	
Description	To delete one preset for a specific channel of the IP media device.
Typical URL	POST http://<host>[:port]/PtzDeletePreset[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The PTZ preset's ID that needs to be deleted will be included in the entity of the request message. For example:

PtzDeletePreset	
<?xml version="1.0" encoding="utf-8" ?> <presetInfo version="1.0" xmlns="http://www.ipc.com/ver10"> <id>1</id> </presetInfo>	
Successful Response	The standard successful result response that described in 1.3.5.

4.3.5 PtzModifyPresetPosition

PtzModifyPresePosition	
Description	To modify one preset's position to current position for a specific channel of the IP media device.
Typical URL	POST http://<host>[:port]/PtzModifyPresetPosition[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The PTZ preset's ID that needs to be modified to current position will be included in the entity of the request message. For example:
<?xml version="1.0" encoding="utf-8" ?> <presetInfo version="1.0" xmlns="http://www.ipc.com/ver10"> <id>3</id> </presetInfo>	
Successful Response	The standard successful result response that described in 1.3.5.

4.4 Cruise

4.4.1 PtzGetCruises

PtzGetCruises

PtzGetCruises	
Description	To get the IP media device's PTZ cruises list for specific channel.
Typical URL	POST or GET http://<host>[:port]/PtzGetCruises[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The PTZ cruises list will be included in the entity of the Successful response. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
<cruiseInfo type="list" maxCount="8">
<itemType type="string" maxLen="31"></itemType>
<item id="1"><![CDATA[SSS]]></item>
</cruiseInfo>
</config>
```

[Tips]:

The "id" attribute for each item starts from "1".

4.4.2 PtzGetCruise

PtzGetCruise	
Description	To get one cruise configuration of the IP media device's specific channel.
Typical URL	POST http://<host>[:port]/GetPtzCruise[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The PTZ cruise's ID that needs to be queried will be included in the entity of the request message. For example:

PtzGetCruise	
<?xml version="1.0" encoding="utf-8" ?> <cruiseInfo version="1.0" xmlns="http://www.ipc.com/ver10"> <id>1</id> </cruiseInfo>	
Successful Response	The PTZ cruise's information will be included in the entity of the Successful response. For example:
<?xml version="1.0" encoding="UTF-8"?> <cruiseInfo> <id type="uint32">1</id> <name type="string" maxLen="31"><![CDATA[SSS]]></name> <presetInfo type="list" maxCount="16"> <itemType> <name type="string" maxLen="11"/> <speed type="uint32" min="1" max="8"/> <holdTime type="uint32" min="5" max="240"/> </itemType> <item id="1"> <name><![CDATA[DDD]]></name> <speed>5</speed> <holdTime>5</holdTime> </item> </presetInfo> </cruiseInfo> --- [Tips]: The "id" attribute for each item starts from "1".	

4.4.3 PtzAddCruise

PtzAddCruise

PtzAddCruise	
Description	To add one cruise for a specific channel of the IP media device.
Typical URL	POST http://<host>[:port]/PtzAddCruise[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The PTZ cruise configuration for specific channel should be included in the entity of request message. The whole “cruiseInfo” element in the “GetPtzCruise” should be included in entity of this message. Any attributes for the “cruiseInfo” element or sub elements should not be included. For example:

```
<?xml version="1.0"?>
<cruiseInfo version="1.0" xmlns="http://www.ipc.com/ver10">
<name type="string"><![CDATA[c2]]></name>
<presetInfo>
<item id="2">
<speed>5</speed>
<holdTime>5</holdTime>
</item>
        ...
</presetInfo>
</cruiseInfo>
```

[Tips]:

The “id” attribute for each item starts from “1”.

Successful Response	The standard successful result response that described in 1.3.5.

4.4.4 PtzModifyCruise

PtzModifyCruise	
Description	To modify one cruise information of the IP media device’s specific channel.

PtzModifyCruise	
Typical URL	POST http://<host>[:port]/PtzModifyCruise[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The PTZ cruise configuration for specific channel should be included in the entity of request message. The whole “cruiseInfo” element in the “GetPtzCruise” should be included in entity of this message. Any attributes for the “cruiseInfo” element or sub elements should not be included.
Successful Response	The standard successful result response that described in 1.3.5.

4.4.5 PtzDeleteCruise

PtzDeleteCruise	
Description	To delete one cruise of the IP media device’s specific channel.
Typical URL	POST http://<host>[:port]/PtzDeleteCruise[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The PTZ cruise’s ID that needs to be deleted will be included in the entity of the request message. For example:
<?xml version="1.0" encoding="utf-8" ?> <cruiseInfo version="1.0" xmlns="http://www.ipc.com/ver10"> <id>2</id> </cruiseInfo>	
Successful Response	The standard successful result response that described in 1.3.5.

5 Alarm commands

5.1 Motion Detection

5.1.1 GetMotionConfig

GetMotionConfig	
Description	To get the IP media device's motion configuration for specific channel.
Typical URL	POST or GET http://<host>[:port]/GetMotionConfig [/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The motion configuration information will be included in the entity of the Successful response. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
<motion>
<switch type="boolean">false</switch>
<sensitivity type="int32" min="0" max="8">4</sensitivity>
<alarmHoldTime type="uint32">20</alarmHoldTime>
<area type="list" count="18">
<itemType type="string" minLen="22" maxLen="22"></itemType>
<item><![CDATA[1111111111111111111111]]></item>
<item><![CDATA[1111111111111111111111]]></item>
```

GetMotionConfig

<item><![CDATA[1111111111111111111111]]></item>

<item><![CDATA[1111111111111111111111]]></item>

<item><![CDATA[1111111111111111111111]]></item>

<item><![CDATA[1111111111111111111111]]></item>

<item><![CDATA[1111111111111111111111]]></item>

<item><![CDATA[1111111111111111111111]]></item>

<item><![CDATA[1111111111111111111111]]></item>

<item><![CDATA[1111111111111111111111]]></item>

<item><![CDATA[1111111111111111111111]]></item>

<item><![CDATA[1111111111111111111111]]></item>

<item><![CDATA[1111111111111111111111]]></item>

<item><![CDATA[1111111111111111111111]]></item>

<item><![CDATA[1111111111111111111111]]></item>

<item><![CDATA[1111111111111111111111]]></item>

<item><![CDATA[1111111111111111111111]]></item>

<item><![CDATA[1111111111111111111111]]></item>

</area>

<triggerAlarmOut type="list" count="1">

<itemType type="boolean"></itemType>

<item id="1">false</item>

</triggerAlarmOut>

</motion>

</config>

[Tips]:

There are 18 sub items in the “area” element, each item is a string with fixed length 22. This means a 22x18 motion detection areas, if corresponding character is “1”, the switch for this detection area is on.

The “id” attribute for each item starts from “1”.

5.1.2 SetMotionConfig

SetMotionConfig	
Description	To set the IP media device's motion configuration for specific channel.
Typical URL	POST http://<host>[:port]/SetMotionConfig [/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The motion detection configuration for specific channel should be included in the entity of request message. The whole "motion" element in the "GetMotionConfig" should be included in entity of this message. Any attributes for the "motion" element or sub elements should not be included.
Successful Response	The standard successful result response that described in 1.3.5.

5.2 Alarm

5.2.1 GetAlarmInConfig

GetAlarmInConfig	
Description	To get the IP media device's alarm input configuration for specific alarm input channel.
Typical URL	POST or GET http://<host>[:port]/GetAlarmInConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default alarm input channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The alarm inputs configuration will be included in the entity of the Successful response. For example:
<?xml version="1.0" encoding="UTF-8"?>	

GetAlarmInConfig
<config version="1.0" xmlns="http://www.ipc.com/ver10"> <types> <alarmInVoltage> <enum>NO</enum> <enum>NC</enum> </alarmInVoltage> </types> <sensor> <id type="uint32">1</id> <sensorName type="string" maxLen="11"><![CDATA[Sensor1]]></sensorName> <switch type="boolean">true</switch> <voltage type="alarmInVoltage">NO</voltage> <alarmHoldTime type="uint32">10</alarmHoldTime> <triggerAlarmOut type="list" count="1"> <itemTypetype="boolean"></itemType> <item id="1">true</item> </triggerAlarmOut> </sensor> </config> [Tips]: The “id” attribute for each item starts from “1”.

5.2.2 SetAlarmInConfig

SetAlarmInConfig	
Description	To set the IP media device’s alarm inputs configuration for specific alarm input channel.
Typical URL	POST http://<host>[:port]/SetAlarmInConfig[/channelId]

SetAlarmInConfig	
Channel ID	Optional. If none channel ID included in the URL, the default alarm input channel ID is 1.
Action name	None
Entity Data	The alarm input configuration for specific channel should be included in the entity of request message. The whole “sensor” element in the “GetAlarmInConfig” should be included in entity of this message. Any attributes for the “sensor” element or sub elements should not be included.
Successful Response	The standard successful result response that described in 1.3.5.

5.2.3 ManualAlarmOut

ManualAlarmOut	
Description	To manually set the IP media device’s alarm output status for specific alarm output channel.
Typical URL	POST http://<host>[:port]/ManualAlarmOut[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default alarm output channel ID is 1.
Action name	None
Entity Data	The new status for the specific alarm output will be included in the entity of request message. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
<action>
<status>true</status>
</action>
</config>
```

[Tips]:

The “status” element is Boolean type.

ManualAlarmOut	
Successful Response	The standard successful result response that described in 1.3.5.

5.2.4 GetAlarmOutConfig

GetAlarmOutConfig	
Description	To get the IP media device's alarm output configuration for specific alarm output channel.
Typical URL	POST or GET http://<host>[:port]/GetAlarmOutConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default alarm output channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The specific alarm output configuration will be included in the entity of the Successful response. For example:
<?xml version="1.0" encoding="UTF-8"?> <config version="1.0" xmlns="http://www.ipc.com/ver10"> <alarmOut> <id type="uint32">1</id> <alarmOutName type="string" maxLen="11"><![CDATA[alarmOut1]]></alarmOutName> <alarmHoldTime type="uint32">20</alarmHoldTime> </alarmOut> </config>	

5.2.5 SetAlarmOutConfig

SetAlarmOutConfig	
Description	To set the IP media device's alarm output configuration for specific alarm output channel.

SetAlarmOutConfig	
Typical URL	POST http://<host>[:port]/SetAlarmOutConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default alarm output channel ID is 1.
Action name	None
Entity Data	The alarm output configuration for specific channel should be included in the entity of request message. The whole “alarmOut” element in the “GetAlarmOutConfig” should be included in entity of this message. Any attributes for the “alarmOut” element or sub elements should not be included.
Successful Response	The standard successful result response that described in 1.3.5.

5.3 AlarmStatus

5.3.1 GetAlarmStatus

GetAlarmStatus	
Description	To get the IP media device’s alarm trigger status.
Typical URL	POST or GET http://<host>[:port]/GetAlarmStatus
Channel ID	None
Action name	None
Entity Data	None
Successful Response	The alarm trigger status information will be included in the entity of the Successful response. For example:

GetAlarmStatus

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
    <alarmStatusInfo>
        <motionAlarm type="boolean" id="1">false</motionAlarm>
        <motionAlarm type="boolean" id="2">true</motionAlarm>
        <motionAlarm type="boolean" id="3">false</motionAlarm>
        <motionAlarm type="boolean" id="4">false</motionAlarm>
        <sensorAlarmIn type="list" count="4">
            <itemType type="boolean"/>
            <item id="1">false</item>
            <item id="2">false</item>
            <item id="3">false</item>
            <item id="4">false</item>
        </sensorAlarmIn>
        <perimeterAlarm type="boolean">false</perimeterAlarm>
        <tripwireAlarm type="boolean">false</tripwireAlarm>
        <cpcAlarm type="boolean">false</cpcAlarm>
        <oscAlarm type="boolean">false</oscAlarm>
        <cddAlarm type="boolean">false</cddAlarm>
        <ipdAlarm type="boolean">false</ipdAlarm>
        <vfdAlarm type="boolean">false</vfdAlarm>
        <avdAlarm type="boolean">false</avdAlarm>
    </alarmStatusInfo>
</config>
```

[Tips]:

The “id” attribute for each item starts from “1”.

The “id” attribute for sensorAlarm’s child item is the NO. of the sensors. And the sensor on the IPC who is the first channel will be 5 if there are 4 sensors on the NVR.

5.3.2 GetAlarmServerConfig

GetAlarmServerConfig	
Description	To get the alarm server configuration
Typical URL	POST or GET http://<host>[:port]/GetAlarmServerConfig
Channel ID	None
Action name	None
Entity Data	None
Successful Response	The alarm server configuration will be included in the entity of the Successful response. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
<alarmServer>
<serverAddr type="string"></serverAddress>
<serverPort type=" uint16">80</serverPort>
<enableHeartbeat type="boolean">true</enableHeartbeat>
<heartbeatInterval type="uint16">10</heartbeatInterval>
</alarmServer>
</config>
```

GetAlarmServerConfig

[Tips]:

1. The “heartbeatInterval” is in second.
2. The data sent to the server when the alarm is issued is as follows:

```
<config version="1.0" xmlns="http://www.ipc.com/ver10">
    <alarmStatusInfo>
        <motionAlarm type="boolean" id="1">true</motionAlarm>
    </alarmStatusInfo>
    <dataTime><![CDATA[2017-06-20 10:30:21]]></dataTime>
    <deviceInfo>
        <deviceName><![CDATA[Device Name]]></deviceName>
        <deviceNo.><![CDATA[1]]></deviceNo.>
        <sn><![CDATA[N563F0159MNK]]></sn>
        <ipAddress><![CDATA[192.168.3.100]]></ipAddress>
        <macAddress><![CDATA[78-24-AF-44-89-01]]></macAddress>
    </deviceInfo>
</config>
```

3. The heartbeat data send to server is as follows:

```
<config version="1.0" xmlns="http://www.ipc.com/ver10">
    <deviceInfo>
        <deviceName><![CDATA[Device Name]]></deviceName>
        <deviceNo.><![CDATA[1]]></deviceNo.>
        <sn><![CDATA[N563F0159MNK]]></sn>
        <ipAddress><![CDATA[192.168.3.100]]></ipAddress>
        <macAddress><![CDATA[78-24-AF-44-89-01]]></macAddress>
    </deviceInfo>
</config>
```

5.3.3 SetAlarmServerConfig

SetAlarmServerConfig	
Description	To set the alarm server configuration.
Typical URL	POST http://<host>[:port]/SetAlarmServerConfig
Channel ID	None
Action name	None
Entity Data	The alarm server configuration should be included in the entity of request message. The whole “alarmServer” element in the “GetAlarmServerConfig” should be included in entity of this message. Any attributes for the “alarmServer” element or sub elements should not be included.
Successful Response	The standard successful result response that described in 1.3.5.

5.3.4 SendAlarmStatus

SendAlarmStatus	
Description	To send the alarm status to the alarm server when an alarm happens. This command will be used by the device. The alarm server should provide HTTP service to receive this command.
Typical URL	POST http://<alarm server>[:port]/SendAlarmStatus
Channel ID	None
Action name	None
Entity Data	The alarm status should be included in the entity of request message. The whole “alarmStatusInfo” element in the response for “GetAlarmStatus” should be included in entity of this message.
Successful Response	None

5.4 AlarmTrigger

5.4.1 GetAlarmTriggerConfig

GetAlarmTriggerConfig	
Description	To get the IP media device's alarm trigger configuration.
Typical URL	POST or GET http://<host>[:port]/GetAlarmTriggerConfig[/channelId]</action_name>
Channel ID	The channel ID starts from 1.
Action name	The action names are defined as follows: alarmIn: schedule of alarmIn. In this scenario, the channelId is used as alarmIn ID. motion: schedule of motion avd: schedule of Abnormal Video Diagnosis cdd: schedule of Crowd Density Detection cpc: schedule of Cross-line People Counting ipd: schedule of Intruding People Detection tripwire: schedule of Tripwire Detection osc: schedule of Object Status Change perimeter: schedule of Perimeter Environment Assurance vfd: schedule of Video Face Detection vehicle:schedule of Video vehilce Detection aoientry: schedule of Aoi Entry Detection aoileave: schedule of Aoi Leave Detection passlinecount: schedule of Passline Count Detection
Entity Data	None
Successful Response	The alarm trigger configuration will be included in the entity of the successful response. For example:
<?xml version="1.0" encoding="utf-8"?> <config xmlns="http://www.ipc.com/ver10" version="1.0">	

```
    <triggerConfig>
        <snap type="list" maxCount="1" count="1">
            <item>
                <channelId type="uint32">1</channelId>
                <switch type="boolean">true</switch>
            </item>
        </snap>
        <record type="list" maxCount="1" count="1">
            <item>
                <channelId type="uint32">1</channelId>
                <switch type="boolean">true</switch>
            </item>
        </record>
        <triggerAlarmOut>
            <alarmOutList type="list" maxCount="1" count="1">
                <item>
                    <alarmOutId type="uint32">1</alarmOutId>
                </item>
            </alarmOutList>
        </triggerAlarmOut>
        <audiotype="list" maxCount="1" count="1">
            <item>
                <switch type="boolean">true</switch>
            </item>
        </audio>
        <whiteLighttype="list" maxCount="1" count="1">
            <item>
                <switch type="boolean">true</switch>
            </item>
        </whiteLight>
    </triggerConfig>
</config>
```

[Tips]:

5.4.2 SetAlarmTriggerConfig

SetAlarmTriggerConfig	
Description	To set the IP media device's alarm trigger configuration.
Typical URL	POST http://<host>[:port]/SetAlarmTriggerConfig[/channelId]</action_name>
Channel ID	The channel ID starts from 1.
Action name	The same as "GetAlarmTriggerConfig".
Entity Data	The whole "triggerConfig" elements in the "GetAlarmTriggerConfig" should be included in entity of this message.
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]:	

5.5 Sound-Light Alarm

5.5.1 GetAudioAlarmOutConfig

GetAudioAlarmOutConfig	
Description	
Typical URL	POST or GET http://<host>[:port]/GetAudioAlarmOutConfig[/channelId]
Channel ID	The channel ID starts from 1.
Action name	The action names are defined as follows:

Entity Data	None
Successful Response	For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.7" xmlns="http://www.ipc.com/ver10">
<types>
    <audioAlarmType>
    <enum value="Warning area, leave as soon as possible">1</enum>
    <enum value="Dangerous area, please do not approach">2</enum>
    <enum    value="No parking in this area">3</enum>
    <enum value="You have entered the real-time monitoring area">4</enum>
    <enum value="Hello, welcome">5</enum>
    <enum value="Do not touch valuables">6</enum>
    <enum value="Private area, no entry">7</enum>
    <enum value="Danger of water depth, pay attention to safety">8</enum>
    <enum value="High altitude, don't climb">9</enum>
    <enum value="Howling alarm sound">10</enum>
    <enum value="Abnormal temperature alarm">23</enum>
    <enum value="Hello world!">100</enum>
    </audioAlarmType>
    <audioLanguageType>
    <enum>zh-cn</enum>
    <enum>en-us</enum>
    <enum>customize</enum>
    </audioLanguageType>
</types>
<audioParamLimit>
    <audioFormat type="read-only">WAV</audioFormat>
    <sampleRate type="read-only">8000HZ</sampleRate>
    <audioChannel type="read-only">Monophonic</audioChannel>
    <audioDepth type="read-only">16bit</audioDepth>
    <audioFileSize type="read-only">less than 200K</audioFileSize>
```

```
</audioParamLimit>
<audioAlarmOut>
    <audioType type="audioAlarmType">10</audioType>
    <alarmTimes type="uint32" min="1" max="50" default="5">5</alarmTimes>
    <audioVolume type="uint32" min="0" max="100" default="100">100</audioVolume>
    <languageType type="audioLanguageType">en-us</languageType>
    <customize type="list" maxCount="10" count="10">
    <item>
    <id type="uint32">100</id>
    <audioName type="string" maxLen="128"><![CDATA[Hello world!]]></audioName>
    </item>
    <item>
    <id type="uint32">101</id>
    <audioName type="string" maxLen="128"><![CDATA[]]></audioName>
    </item>
    <item>
    <id type="uint32">102</id>
    <audioName type="string" maxLen="128"><![CDATA[]]></audioName>
    </item>
    <item>
    <id type="uint32">103</id>
    <audioName type="string" maxLen="128"><![CDATA[]]></audioName>
    </item>
    <item>
    <id type="uint32">104</id>
    <audioName type="string" maxLen="128"><![CDATA[]]></audioName>
    </item>
    <item>
    <id type="uint32">105</id>
    <audioName type="string" maxLen="128"><![CDATA[]]></audioName>
    </item>
    <item>
```

<id type="uint32">106</id>
<audioName type="string" maxLen="128"><![CDATA[]]></audioName>
</item>
<item>
<id type="uint32">107</id>
<audioName type="string" maxLen="128"><![CDATA[]]></audioName>
</item>
<item>
<id type="uint32">108</id>
<audioName type="string" maxLen="128"><![CDATA[]]></audioName>
</item>
<item>
<id type="uint32">109</id>
<audioName type="string" maxLen="128"><![CDATA[]]></audioName>
</item>
</customize>
</audioAlarmOut>
</config>

[Tips]:

修改说明：

1. audioAlarmType 枚举中增加自定义的添加音频文件显示。
2. audioLanguageType 枚举中增加自定义选项。
3. 消息体增加 customize 节点，目前最大支持 10 个自定义文件。

5.5.2 SetAudioAlarmOutConfig

SetAudioAlarmOutConfig	
Description	
Typical URL	POST http://<host>[:port]/SetAudioAlarmOutConfig[/channelId]

Channel ID	The channel ID starts from 1.
Action name	The same as "GetAudioAlarmOutConfig".
Entity Data	
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]:	

5.5.3 AddCustomizeAudioAlarm

AddCustomizeAudioAlarm	
Description	
Typical URL	POST or GET http://<host>[:port]/AddCustomizeAudioAlarm[/channelId]
Channel ID	The channel ID starts from 1.
Action name	The action names are defined as follows:
Entity Data	None
Successful Response	For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.7" xmlns="http://www.ipc.com/ver10">
<addAudioAlarm>
    <audioName type="string" maxLen="128"><![CDATA[welcome]]></audioName>
    <audioFileSize type="uint32" maxLen="102400" >123</audioFileSize>
    <audioFileData type="string"
maxLen="102400"><![CDATA[....base64encodeData...]]></audioFileData>
</addAudioAlarm>
</config>
```

Success Response:

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

<config xmlns="http://www.ipc.com/ver10" version="1.7">

<addAudioAlarm>

<id type="uint32">101</id>

</addAudioAlarm>

</config>

Failure Response:

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

<config version="1.0" xmlns="http://www.ipc.com/ver10" status="failed" errorCode="-107"/>

[Tips]:

1.errorCode 描述：

-101:未正常工作.

-102:参数错误.

-103:wav 音频内容不是 PCM 格式

-104:音频文件不是 WAV.

-105:采样率不是 8000HZ.

-106:保存音频文件失败.

-107:超出了最大自定义 10 个文件数量.

-108:音频文件大小超限制.

1. audioFileData 为 base64 加密后的数据
2. audioFileSize 为加密后的文件大小

5.5.4 DeleteCustomizeAudioAlarm

DeleteCustomizeAudioAlarm	
Description	

Typical URL	POST or GET http://<host>[:port]/DeleteCustomizeAudioAlarm[/channelId]
Channel ID	The channel ID starts from 1.
Action name	The action names are defined as follows:
Entity Data	None
Successful Response	For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.7" xmlns="http://www.ipc.com/ver10">
<deleteAudioAlarm>
    <id type="uint32">101</id>
    </deleteAudioAlarm>
</config>
```

Success Response:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10" status="success" errorCode="200" IssameOldPwd="false"/>
```

Failure Response:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10" status="failed" errorCode="-109"/>
```

[Tips]:

1.errorCode 描述：

-109：删除音频文件不存在。

2.id 为添加返回或者是 GetAudioAlarmOutConfig 返回自定义的节点

5.5.5 AuditionCustomizeAudioAlarm

AuditionCustomizeAudioAlarm	
Description	
Typical URL	POST or GET http://<host>[:port]/AuditionCustomizeAudioAlarm[/channelId]
Channel ID	The channel ID starts from 1.
Action name	The action names are defined as follows:
Entity Data	None
Successful Response	For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.7" xmlns="http://www.ipc.com/ver10">
<types>
    <audioAlarmType>
    <enum value="Warning area, leave as soon as possible">1</enum>
    <enum value="Dangerous area, please do not approach">2</enum>
    <enum    value="No parking in this area">3</enum>
    <enum value="You have entered the real-time monitoring area">4</enum>
    <enum value="Hello, welcome">5</enum>
    <enum value="Do not touch valuables">6</enum>
    <enum value="Private area, no entry">7</enum>
    <enum value="Danger of water depth, pay attention to safety">8</enum>
    <enum value="High altitude, don't climb">9</enum>
    <enum value="Howling alarm sound">10</enum>
    </audioAlarmType>
</types>
<auditionAudioAlarm>
    <audioType type="audioAlarmType">10</audioType>
</auditionAudioAlarm>
```

</config>

Success Response:

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

<config version="1.0" xmlns="http://www.ipc.com/ver10" status="success" errorCode="200" IssameOldPwd="false"/>

Failure Response:

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

<config version="1.0" xmlns="http://www.ipc.com/ver10" status="failed" errorCode="-110"/>

[Tips]:

1.errorCode 描述：

-110：正在告警中，不能试听。

5.5.6 GetWhiteLightAlarmOutConfig

GetWhiteLightAlarmOutConfig	
Description	
Typical URL	POST or GET http://<host>[:port]/GetWhiteLightAlarmOutConfig[/channelId]
Channel ID	The channel ID starts from 1.
Action name	The action names are defined as follows:
Entity Data	None
Successful Response	

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.7" xmlns="http://www.ipc.com/ver10">
    <types>
        <lightFrequency>
            <enum>low</enum>
            <enum>medium</enum>
            <enum>high</enum>
        </lightFrequency>
    </types>
    <whiteLightAlarmOut>
    <switch type="boolean">false</switch>
    <durationTime type="uint32" min="1" max="60" default="20">20</durationTime>
    <frequency type="lightFrequency">low</frequency>
    </whiteLightAlarmOut>
</config>
```

[Tips]:

5.5.7 SetWhiteLightAlarmOutConfig

SetWhiteLightAlarmOutConfig	
Description	
Typical URL	POST http://<host>[:port]/SetWhiteLightAlarmOutConfig[/channelId]
Channel ID	The channel ID starts from 1.
Action name	The same as "GetWhiteLightAlarmOutConfig".
Entity Data	
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]:	

5.6 Alarm PIR

5.6.1 GetPirConfig

GetPirConfig	
Description	To get the IP media device's PIR configuration for specific channel.
Typical URL	POST or GET http://<host>[:port]/GetPirConfig [/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The PIR configuration information will be included in the entity of the Successful response. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.7" xmlns="http://www.ipc.com/ver10">
<pir>
<switch type="boolean">true</switch>
<alarmHoldTime type="uint32">20</alarmHoldTime>
<triggerAlarmOut type="list" count="1">
<itemType type="boolean"/>
<item id="1">false</item>
</triggerAlarmOut>
<mail type="list" count="0">
<switch type="boolean">false</switch>
<subject type="string" maxLen="63"><![CDATA[]]></subject>
<content type="string" maxLen="255"><![CDATA[]]></content>
</mail>
<ftp type="list" count="0">
```

GetPirConfig
<switch type="boolean">false</switch> </ftp> <savePicSwitch type="boolean">false</savePicSwitch> <sdRecSwitch type="boolean">false</sdRecSwitch> <sendPush> <pushSwitch type="boolean">false</pushSwitch> <recordSwitch type="boolean">false</recordSwitch> <recordStreamIndex type="uint8">0</recordStreamIndex> <sendPicSwitch type="boolean">false</sendPicSwitch> <recordTime type="uint32">0</recordTime> <pushContent type="string" maxLen="127"><![CDATA[]]></pushContent> </sendPush> </pir> </config>
[Tips]:

5.6.2 SetPirConfig

SetPirConfig	
Description	To set the IP media device's PIR configuration for specific channel.
Typical URL	POST http://<host>[:port]/SetPirConfig [/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The motion detection configuration for specific channel should be included in the entity of request message. The whole "pir" element in the "GetPirConfig" should be included in entity of this message. Any attributes for the "pir" element or sub elements should not be included.
Successful Response	The standard successful result response that described in 1.3.5.

6 Playback

6.1 Record Search

6.1.1 GetRecordType

GetRecordType	
Description	To get record types.
Typical URL	POST or GET http://<host>[:port]/GetRecordType
Channel ID	None.
Action name	None
Entity Data	None
Successful Response	The record types will be included in the entity of the Successful response. For example:

GetRecordType
<?xml version="1.0" encoding="UTF-8"?> <config version="1.0" xmlns="http://www.ipc.com/ver10"> <recTypeCaps type="list" count="6"> <itemType type="string" maxLen="20"/> <item>manual</item> <item>schedule</item> <item>motion</item> <item>sensor</item> <item>intel detection</item> <item>nic broken</item> </recTypeCaps> </config>
[Tips]: It returns the capability of recording for current device. The type "nic broken" is for IPC only.

6.1.2 SearchRecordDate

SearchByDate	
Description	To search the date list with record data for specific channel.
Typical URL	POST or GET http://<host>[:port]/SearchRecordDate[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The date list with record data will be included in the entity of the successful response. For example:

SearchByDate
<?xml version="1.0" encoding="UTF-8"?> <config version="1.0" xmlns="http://www.ipc.com/ver10"> <dateList type="list" count="6"> <itemType type="string"/> <item>2014-01-09</item> <item>2014-02-09</item> <item>2014-03-08</item> <item>2014-04-02</item> <item>2014-04-03</item> <item>2014-04-04</item> </datelist> </config>

6.1.3 SearchByTime

SearchByTime	
Description	To search record data segments for the specific channel by time.
Typical URL	POST or GET http://<host>[:port]/SearchByTime[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The start time and end time should be included in the entity of the request message as search condition. For example:

SearchByTime

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
    <search>
        <recTypes type="list">
            <itemType type="recType"></itemType>
            <item>manual</item>
            <item>schedule</item>
            <item>motion</item>
            <item>sensor</item>
            <item>intel detection</item>
            <item>nic broken</item>
        </recTypes>
        <starttime type="string"><![CDATA[2017-06-3000:00:00]]></starttime>
        <endtime type="string"><![CDATA[2017-06-3023:59:59]]></endtime>
    </search>
</config>
```

Successful Response	The searched record data segments will be included in the entity of the successful response. For example:

SearchByTime

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
    <timesectionList type="list" count="2">
    <item>
        <starttime type="string" seconds="827" recType="schedule"><![CDATA[2017-06-30 07:39:36]]></starttime>
    </item>
    <item>
        <starttime type="string" seconds="533" recType="schedule"><![CDATA[2017-06-30 07:54:03]]></starttime>
    </item>
    </timesectionList>
</config>
```

[Tips]:

The list count of “timesectionList” node is limit to 1000. Use shorter time to query when the list be limited.

The event type “nic broken” is for IPC only.

The client application can playback one specific record data segment through RTSP protocol. For example:

rtsp://<host><:rtspPort>/chID=0&date=2014-01-09&time=15:07:28&timelen=200[streamType=main][&action=backup]

When this URL is invoked by the client application, the first record data segment searched by the device will be playback through RTSP.

“streamType” can be “main” or “sub”

The “action” can be “playback” or “backup”. And the “backup” parameterwill make the data transmission as soon as possible.

If none “action”parameter include in the url, default is “playback.”

6.2 RecordStatus

6.2.1 GetRecordStatusInfo

GetRecordStatusInfo	
Description	To get the record status of the specific channel.
Typical URL	POST or GEThttp://<host>[:port]/GetRecordStatusInfo
Channel ID	None
Action name	None
Entity Data	None
Successful Response	The record status information will be included in the entity of the Successful response. For example:

GetRecordStatusInfo

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
    <types>
        <recordStatusType>
            <enum>no recording</enum>
            <enum>recording</enum>
            <enum>exception</enum>
        </recordStatusType>
    </types>
<recordStatusList type="list" count="4">
<itemType type="recordStatusType"/>
<item id="1">exception</item>
<item id="2">recording</item>
<item id="3">recording</item>
<item id="4">no recording</item>
</recordStatusList>
</config>
```

[Tips]:

The “id” attribute is the channel id.

7 Network commands

7.1 TCP/Ipv4

7.1.1 GetNetBasicConfig

GetNetBasicConfig	
Description	To get the IP media device's basic network configuration.
Typical URL	POST or GET http://<host>[:port]/GetNetBasicConfig
Channel ID	None
Action name	None
Entity Data	None
Successful Response	The basic network configuration will be included in the entity of the Successful response. For example:

GetNetBasicConfig
<?xml version="1.0" encoding="UTF-8"?> <config version="1.0" xmlns="http://www.ipc.com/ver10"> <types> <ipSettingMode> <enum>staticIp</enum> <enum>dhcp</enum> </ipSettingMode> </types> <tcpIp> <ipSettingMode type="ipSettingMode">staticIp</ipSettingMode> <staticIp type="string" minLen="7" maxLen="15"><![CDATA[192.168.6.36]]></staticIp> <staticIpRoute type="string" minLen="7" maxLen="15"><![CDATA[192.168.6.1]]></staticIpRoute> <staticIpMask type="string" minLen="7" maxLen="15"><![CDATA[255.255.255.0]]></staticIpMask> <dnsFromDhcpSwitch type="boolean">false</dnsFromDhcpSwitch> <dnsServer1 type="string" minLen="7" maxLen="15"><![CDATA[192.168.226.1]]></dnsServer1> <dnsServer2 type="string" minLen="7" maxLen="15"><![CDATA[8.8.8.8]]></dnsServer2> </tcpIp> </config>

7.1.2 SetNetBasicConfig

SetNetBasicConfig	
Description	To set the IP media device's basic network configuration.
Typical URL	POST http://<host>[:port]/SetNetBasicConfig
Channel ID	None
Action name	None
Entity Data	The basic network configuration should be included in the entity of request message. The whole "tcpIp" element in the "GetNetBasicConfig" should be included in entity of this message. Any attributes for the "tcpIp" element or sub elements should not be included.

SetNetBasicConfig	
Successful Response	The standard successful result response that described in 1.3.5.

7.2 PPPoE

7.2.1 GetNetPppoeConfig

GetNetPppoeConfig	
Description	To get the IP media device's network PPPOE configuration.
Typical URL	POST or GET http://<host>[:port]/GetNetPppoeConfig
Channel ID	None
Action name	None
Entity Data	None
Successful Response	The network PPPOE configuration will be included in the entity of the Successful response. For example:

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

<config version="1.0" xmlns="http://www.ipc.com/ver10">

<pppoe>

<switch type="boolean">false</switch>

<userName type="string"maxLen="63"><![CDATA[aaa]]></userName>

<password type="string"maxLen="63"><![CDATA[bbb]]></password>

</pppoe>

</config>

[Tips]:

The value of the "password" element will be none, for the reason that the "password" element is write-only.

7.2.2 SetNetPppoeConfig

SetNetPppoeConfig	
Description	To set the IP media device's network PPPOE configuration.
Typical URL	POST http://<host>[:port]/SetNetPppoeConfig
Channel ID	None
Action name	None
Entity Data	The network PPPOE configuration should be included in the entity of request message. The whole "pppoe" element in the "GetNetPppoeConfig" should be included in entity of this message. Any attributes for the "pppoe" element or sub elements should not be included. If the user doesn't need to change password, please omit the "password" element.
Successful Response	The standard successful result response that described in 1.3.5.

7.3 Port

7.3.1 GetPortConfig

GetPortConfig	
Description	To get the IP media device's network service ports configuration.
Typical URL	POST or GET http://<host>[:port]/GetPortConfig
Channel ID	None
Action name	None
Entity Data	None
Successful Response	The network service ports configuration will be included in the entity of the Successful response. For example:

GetPortConfig
<?xml version="1.0" encoding="UTF-8"?> <config version="1.0" xmlns="http://www.ipc.com/ver10"> <port> <httpPort type="uint16">80</httpPort> <netPort type="uint16">9008</netPort> <rtspPort type="uint16">554</rtspPort> </port> </config> [Tips]: The “httpPort” element announces the port for HTTP service. The “netPort” element announces the port for protocol service. The “rtspPort” element announces the port for RTSP service.

7.3.2 SetPortConfig

SetPortConfig	
Description	To set the IP media device’s network service ports configuration.
Typical URL	POST http://<host>[:port]/SetPortConfig
Channel ID	None
Action name	None
Entity Data	The network service ports configuration should be included in the entity of request message. The whole “port” element in the “GetPortConfig” should be included in entity of this message. Any attributes for the “port” element or sub elements should not be included.
Successful Response	The standard successful result response that described in 1.3.5.

7.4 DDNS

7.4.1 GetDdnsConfig

GetDdnsConfig	
Description	To get the IP media device's network DDNS configuration.
Typical URL	POST or GET http://<host>[:port]/GetDdnsConfig
Channel ID	None
Action name	None
Entity Data	None
Successful Response	The network DDNS configuration will be included in the entity of the Successful response. For example:

GetDdnsConfig

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
<types>
    <ddnsServerType>
        <enumrequireParameters="userName,password">www.88ip.net</enum>
        <enumrequireParameters="userName,password">www.dns2p.net</enum>
        <enumrequireParameters="userName,password">www.meibu.com</enum>
        <enum requireParameters="userName,password,domainName">www.dyndns.com</enum>
        <enum requireParameters="userName,password,domainName">www.no-ip.com</enum>
        <enum
requireParameters="userName,password,domainName,serverName">mintdns</enum>
        <enum requireParameters="userName,password,domainName">www.3322.org</enum>
    </ddnsServerType>
</types>
<ddns>
<switch type="boolean">false</switch>
<servertype type="ddnsServerType">www.88ip.com</servertype>
<userName type="string" maxLen="63"><![CDATA[aaa]]></userName>
<password type="string" maxLen="63"><![CDATA[]]></password>
<domainName type="string"maxLen="63"><![CDATA[ipc.88ip.com]]></domainName>
<serverName type="string" maxLen="63"><![CDATA[111]]></serverName>
</ddns>
</config>
```

[Tips]:

The value of the “password” element will be none, for the reason that the “password” element is write-only.

7.4.2 SetDdnsConfig

SetDdnsConfig

SetDdnsConfig	
Description	To set the IP media device's network DDNS configuration.
Typical URL	POST http://<host>[:port]/SetDdnsConfig
Channel ID	None
Action name	None
Entity Data	The network DDNS configuration should be included in the entity of request message. The whole "ddns" element in the "GetDdnsConfig" should be included in entity of this message. Any attributes for the "ddns" element or sub elements should not be included.If the user doesn't need to change password, please omit the "password" element.
Successful Response	The standard successful result response that described in 1.3.5.

8 Security commands

8.1 User Management

8.1.1 ModifyPassword

ModifyPassword	
Description	To modify the current login user's password for the IP media device.
Typical URL	POST http://<host>[:port]/ModifyPassword

<table>
<tr><th colspan="2">ModifyPassword</th></tr>
<tr><td>Channel ID</td><td>None</td></tr>
<tr><td>Action name</td><td>None</td></tr>
<tr><td>Entity Data</td><td>The new password will be included in the entity of request message. Any attributes for the “userPassword” element or sub elements should not be included. For example:</td></tr>
<tr><td colspan="2"><?xml version="1.0"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
<userPassword>
<oldPassword><![CDATA[YWFh]]></oldPassword>
<password><![CDATA[YmJi]]></password>
</userPassword>
</config>

[Tips]:
The “oldPassword” and “password” elements are all “string” type with maxLen“19”. They should be encoded by base64, the “YWFh” and “YmJi” are the encoded result for “aaa” and “bbb”.</td></tr>
<tr><td>Successful Response</td><td>The standard successful result response that described in 1.3.5.</td></tr>
</table>

9 Maintain commands

9.1 Reboot

9.1.1 Reboot

Reboot	
Description	To reboot the IP media device.
Typical URL	POST or GET http://<host>[:port]/Reboot
Channel ID	None
Action name	None
Entity Data	None
Successful Response	The standard successful result response that described in 1.3.5.

10 Talkback commands

10.1 Talkback

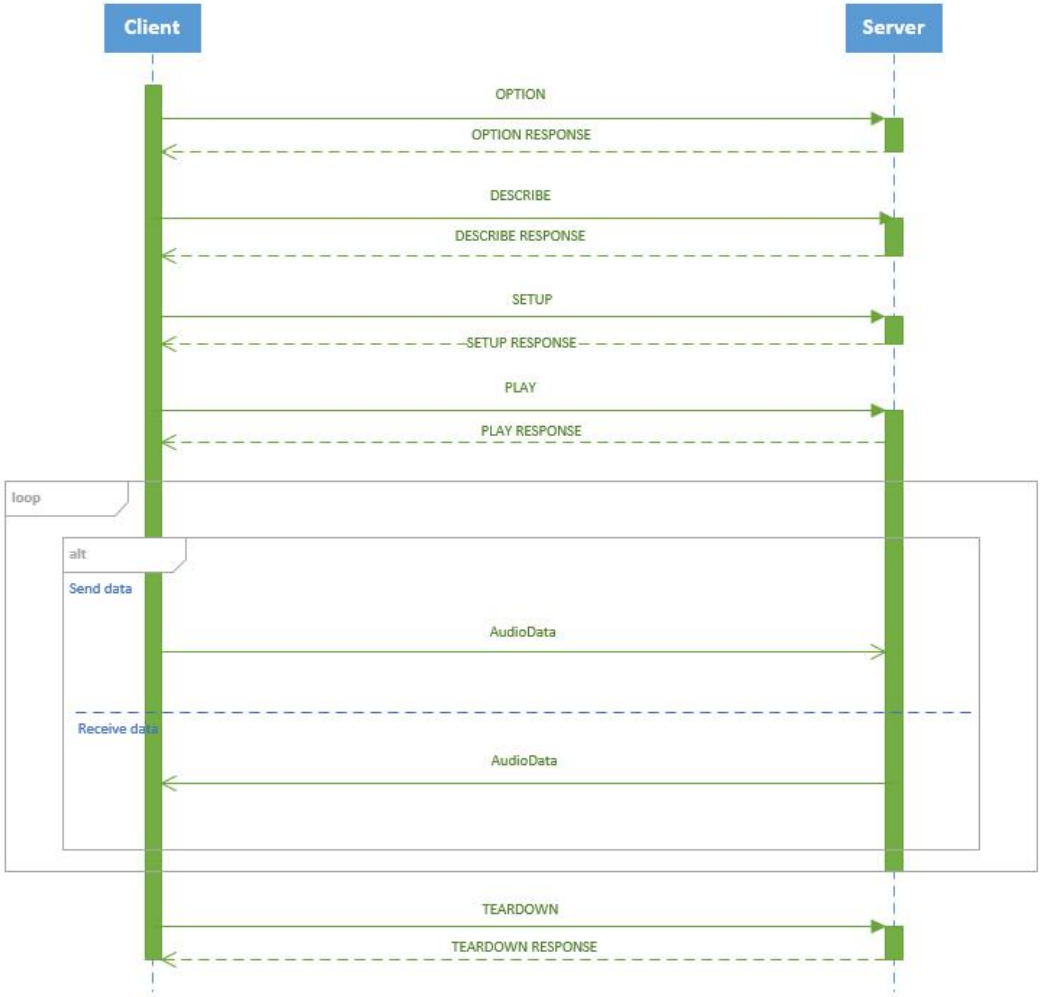
10.1.1 Talkback

profile_talk	
Description	Get the url that can used to send and receive the two-way audio data afterthe intercom opened.
Typical URL	GEThttp://<host>[:port]/profile_talk
Channel ID	None
Action name	None
Entity Data	None
Successful Response	The url for two-way audio data sending and receiving will be included in the entity of the successful response. For example:
<?xml version="1.0" encoding="UTF-8"?> <config version="1.0" xmlns="http://www.ipc.com/ver10"> <URL type="string">rtsp://192.168.0.9:554/intercom</URL> </config>	

profile_talk

[Tips]:

1. When the URL invoked by the client application, the two-way audio data stream can be passed through the RTSP protocol as below:



2. The RTSP error code is defined as below:

600	The device is busy
601	Audio open failed
602	No permission

3. Get the format of the tow-way data from rtp payload. And send the same format to device. It supports only single channel. The sampling rate is 8000HZ.The RTP size is a multiple of 320 bytes. Maximum of 320 * 5.

11 Smart commands

11.1 Face Detect & Face Comparison

11.1.1 GetSmartVfdConfig

GetSmartVfdConfig	
Description	To get the IP media device's Video Face Detection configuration.
Typical URL	POST or GET http://<host>[:port]/GetSmartVfdConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The VFD configuration will be included in the entity of the successful response. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.7" xmlns="http://www.ipc.com/ver10">
    <types>
        <mutexObjectType>
            <enum>cdd</enum>
            <enum>cpc</enum>
            <enum>ipd</enum>
            <enum>tripwire</enum>
            <enum>osc</enum>
            <enum>perimeter</enum>
```

```
            <enum>vfd</enum>
            <enum>avd</enum>
        </mutexObjectType>
        <detectModeType>
            <enum>auto</enum>
            <enum>fixedInterval</enum>
        </detectModeType>
        <alarmListType>
            <enum>blackList</enum>
            <enum>whiteList</enum>
            <enum>strangerList</enum>
        </alarmListType>
        <alarmModeType>
            <enum>faceAndIdentity</enum>
            <enum>faceOnly</enum>
        </alarmModeType>
        <senceModeType>
            <enum>accessControl</enum>
            <enum>securityMonitor</enum>
            <enum>customize</enum>
        </senceModeType>
    </types>
    <vfd>
        <mutexList type="list" count="2">
            <item>
                <object type="mutexObjectType">perimeter</object>
                <status type="boolean">false</status>
            </item>
            <item>
                <object type="mutexObjectType">tripwire</object>
                <status type="boolean">true</status>
            </item>
```

```
            </mutexList>
            <functionStatus type="int16">0</functionStatus>
            <switch type="boolean">false</switch>
            <detectMode>
                <mode type="detectModeType">fixedInterval</mode>
        <intervalTime type="uint16" Min="300"max="600000"default="5000">5000</intervalTime>
            <captureCycle type="uint16" min="1" max="65535" default="3">3</captureCycle>
            </detectMode>
            <alarmHoldTime type="uint32">3</alarmHoldTime>
            <saveFacePicture type="boolean">false</saveFacePicture>
            <saveSourcePicture type="boolean">false</saveSourcePicture>
            <regionInfo type="list" maxCount="1" count="1">
                <item type="rectangle">
                    <X1 type="uint32">262</X1>
                    <Y1 type="uint32">126</Y1>
                    <X2 type="uint32">9761</X2>
                    <Y2 type="uint32">9841</Y2>
                </item>
            </regionInfo>
            <maxFaceFrame type="uint16">5000</maxFaceFrame>
            <minFaceFrame type="uint16">1599</minFaceFrame>
            <faceMatch>
            <pushMode>
                <mode type="detectModeType">fixedInterval</mode>
                <intervalTime type="uint16" min="3" max="10" default="4">4</intervalTime>
            </pushMode>
        <similarityThreshold type="uint8" min="1" max="100" default="80">75</similarityThreshold>
            <alarmMode type="alarmModeType">faceOnly</alarmMode>
            <alarmList type="alarmListType">whiteList</alarmList>
            <triggerAlarmOut>
            <Io type="list" maxCount="8" count="2">
                <item>
```

```
                <alarmId type="uint32">0</alarmId>
                <switch type="boolean">false</switch>
            </item>
            <item>
                <alarmId type="uint32">1</alarmId>
                <switch type="boolean">false</switch>
            </item>
        </Io>
        </triggerAlarmOut>
        </faceMatch>
        <faceExp>
            <switch type="boolean">false</switch>
        <faceExpStrength type="uint32" min="0" max="100" default="50">50</faceExpStrength>
        </faceExp>
        <senceMode>
            <mode type="senceModeType">securityMonitor</mode>
            <spareTimeMatch type="boolean">true</spareTimeMatch>
            <nearPriority type="boolean">false</nearPriority>
        </senceMode>
        <senceModeInfo>
            <accessControlMode>
              <intervalTime type="uint16">500</intervalTime>
                <captureCycle type="uint16">65535</captureCycle>
                <spareTimeMatch type="boolean">false</spareTimeMatch>
                <nearPriority type="boolean">true</nearPriority>
            </accessControlMode>
            <securityMonitorMode>
                <intervalTime type="uint16">5000</intervalTime>
                <captureCycle type="uint16">3</captureCycle>
                <spareTimeMatch type="boolean">true</spareTimeMatch>
                <nearPriority type="boolean">false</nearPriority>
            </securityMonitorMode>
```

</senceModeInfo> </vfd> </config>
[Tips]: 1.The two coordinate points of "regionInfo.item" represent the two points of the rectangular diagonal.

11.1.2 SetSmartVfdConfig

SetSmartVfdConfig	
Description	To set the IP media device's Video Face Detection configuration.
Typical URL	POST http://<host>[:port]/SetSmartVfdConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The whole "vfd" element in the"GetSmartVfdConfig" should be included in entity of this message.
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]:	

11.1.3 AddTargetFace

AddTargetFace	
Description	Add the face info to the target lib.
Typical URL	POSThttp://<host>[:port]/AddTargetFace[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None

Entity Data	The "personInfo" and "faceImgs" elements will be included in the entity of request message. For example:

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
    <types>
        <listType>
            <enum>strangerList</enum>
            <enum>whiteList</enum>
            <enum>blackList</enum>
        </listType>
        <sexType>
            <enum>male</enum>
            <enum>female</enum>
        </sexType>
        <formatType>
            <enum>jpg</enum>
        </formatType>
    </types>
    <personInfo>
        <listType type="listType">whiteList</listType>
        <name type="string" maxLen="127"><![CDATA[user]]></name>
        <sex type="sexType">male</sex>
        <age type="uint32">34</age>
        <identifyNumber type="string" maxLen="127"><![CDATA[A123]]></identifyNumber>
        <telephone type="string" maxLen="63"><![CDATA[18888888888]]></telephone>
        <comment type="string" maxLen="63"><![CDATA[]]></comment>
    </personInfo>
    <faceImgs type="list" maxCount="5" count="2">
        <item>
            <pictureData type="string" maxLen="95576">
                <![CDATA[Base64 Picture Data]]>
            </pictureData>
            <pictureNum type="uint32">1</pictureNum>
            <width type="uint32">100</width>
```

```
            <height type="uint32">80</height>
            <format type="formatType">jpg</format>
            <size type="uint32">50000</size>
        </item>
        <item>
            <pictureData type="string" maxLen="95576">
                <![CDATA[Base64 Picture Data]]>
            </pictureData>
            <pictureNum type="uint32">2</pictureNum>
            <width type="uint32">200</width>
            <height type="uint32">180</height>
            <format type="formatType">jpg</format>
            <size type="uint32">60000</size>
        </item>
    </faceImgs>
</config>
```

Successful	The standard successful result response that described in 1.3.5.

Response	
[Tips]: 1.目前只支持 jpg(jpeg)格式、大小限制 70k 以内图片上传 2.faceImgs 目前 A2 只支持 1 张人脸图	

11.1.4 DeleteTargetFace

DeleteTargetFace	
Description	Delete the face info from the target lib.
Typical URL	POST http://<host>[:port]/DeleteTargetFace[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The "deleteAction"element will be included in the entity of request message. For example:

```
<!-- example1: 按 ID 删除人员信息 -->
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
    <types>
        <deleteType>
            <enum>byPersonID</enum>
            <enum>byListType</enum>
            <enum>byName</enum>
            <enum>byIdentifyNumber</enum>
        </deleteType>
        <listType>
            <enum>strangerList</enum>
            <enum>whiteList</enum>
```

```
                <enum>blackList</enum>
            </listType>
        </types>
        <deleteAction>
            <deleteType type="deleteType">byPersonID</deleteType>
            <personID type="uint32">1543018104</personID>
        </deleteAction>
    </config>

    <!-- example2: 按名单类型删除人员信息 -->
    <?xml version="1.0" encoding="utf-8"?>
    <config xmlns="http://www.ipc.com/ver10" version="1.0">
        <types>
            <deleteType>
                <enum>byPersonID</enum>
                <enum>byListType</enum>
                <enum>byName</enum>
                <enum>byIdentifyNumber</enum>
            </deleteType>
            <listType>
                <enum>strangerList</enum>
                <enum>whiteList</enum>
                <enum>blackList</enum>
            </listType>
        </types>
        <deleteAction>
            <deleteType type="deleteType">byListType</deleteType>
            <listType type="listType">whiteList</listType>
        </deleteAction>
    </config>

    <!--example3: 按姓名删除人员信息-->
```

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
    <types>
        <deleteType>
            <enum>byPersonID</enum>
            <enum>byListType</enum>
            <enum>byName</enum>
            <enum>byIdentifyNumber</enum>
        </deleteType>
        <listType>
            <enum>strangerList</enum>
            <enum>whiteList</enum>
            <enum>blackList</enum>
        </listType>
    </types>
    <deleteAction>
        <deleteType type="deleteType">byName</deleteType>
        <name type="string" maxLen="127">
            <![CDATA[user]]>
        </name>
    </deleteAction>
</config>

<!--Example4: 按证件号码删除人员信息-->
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
    <types>
        <deleteType>
            <enum>byPersonID</enum>
            <enum>byListType</enum>
            <enum>byName</enum>
            <enum>byIdentifyNumber</enum>
```

<table>
<tr><td colspan="2">
</deleteType>
<listType>
<enum>strangerList</enum>
<enum>whiteList</enum>
<enum>blackList</enum>
</listType>
</types>
<deleteAction>
<deleteType type="deleteType">byIdentifyNumber</deleteType>
<identifyNumber type="string" maxLen="127">
<![CDATA[A123]]>
</identifyNumber>
</deleteAction>
</config>
</td></tr>
<tr><td>Successful Response</td><td>The standard successful result response that described in 1.3.5.</td></tr>
<tr><td colspan="2">[Tips]:</td></tr>
</table>

11.1.5 EditTargetFace

EditTargetFace	
Description	Edit the face info of the target lib.
Typical URL	POSThttp://<host>[:port]/EditTargetFace[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The "personInfo" and "faceImgs" elements will be included in the entity of request message. For example:
<?xml version="1.0" encoding="utf-8"?>	

```
<config xmlns="http://www.ipc.com/ver10" version="1.0">
    <types>
        <listType>
            <enum>strangerList</enum>
            <enum>whiteList</enum>
            <enum>blackList</enum>
        </listType>
        <sexType>
            <enum>male</enum>
            <enum>female</enum>
        </sexType>
        <formatType>
            <enum>jpg</enum>
        </formatType>
    </types>
    <personID type="uint32">1543018104</personID>
    <personInfo>
        <listType type="listType">whiteList</listType>
        <name type="string" maxLen="127"><![CDATA[user]]></name>
        <sex type="sexType">male</sex>
        <age type="uint32">34</age>
        <identifyNumber type="string" maxLen="127"><![CDATA[A123]]></identifyNumber>
        <telephone type="string" maxLen="63"><![CDATA[18888888888]]></telephone>
        <comment type="string" maxLen="63"><![CDATA[]]></comment>
    </personInfo>
    <faceImgs type="list" maxCount="5" count="2">
        <item>
            <pictureData type="string" maxLen="95576">
                <![CDATA[Base64 Picture Data]]>
            </pictureData>
            <pictureNum type="uint32">1</pictureNum>
            <width type="uint32">100</width>
```

<height type="uint32">80</height> <format type="formatType">jpg</format> <size type="uint32">50000</size> </item> <item> <pictureData type="string" maxLen="95576"> <![CDATA[Base64 Picture Data]]> </pictureData> <pictureNum type="uint32">2</pictureNum> <width type="uint32">200</width> <height type="uint32">180</height> <format type="formatType">jpg</format> <size type="uint32">60000</size> </item> </faceImgs> </config>

Successful Response	The standard successful result response that described in 1.3.5.
[Tips]:	

11.1.6 GetTargetFace

GetTargetFace	
Description	Get the face info from the target lib.
Typical URL	POSThttp://<host>[:port]/GetTargetFace[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None

Entity Data	The "queryAction" element will be included in the entity of request message. For example:

```
<!--example: 1  按名单类型查询人员 ID-->
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
    <types>
        <queryType>
            <enum>byPersonID</enum>
            <enum>byListType</enum>
            <enum>byName</enum>
            <enum>byIdentifyNumber</enum>
            <enum>byPersonID</enum>
        </queryType>
        <listType>
            <enum>strangerList</enum>
            <enum>whiteList</enum>
            <enum>blackList</enum>
        </listType>
    </types>
    <queryAction>
        <queryType type="queryType">byListType</queryType>
        <listType type="listType">whiteList</listType>
    </queryAction>
</config>

<!--example: 2  按姓名查询人员 ID-->
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
    <types>
        <queryType>
            <enum>byPersonID</enum>
            <enum>byListType</enum>
```

```
                <enum>byName</enum>
                <enum>byIdentifyNumber</enum>
            </queryType>
            <listType>
                <enum>strangerList</enum>
                <enum>whiteList</enum>
                <enum>blackList</enum>
            </listType>
        </types>
        <queryAction>
            <queryType type="queryType">byName</queryType>
            <name type="string" maxLen="127"><![CDATA[user]]></name>
        </queryAction>
</config>

<!--example: 3  按证件号码查询人员 ID-->
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
        <types>
            <queryType>
                <enum>byPersonID</enum>
                <enum>byListType</enum>
                <enum>byName</enum>
                <enum>byIdentifyNumber</enum>
            </queryType>
            <listType>
                <enum>strangerList</enum>
                <enum>whiteList</enum>
                <enum>blackList</enum>
            </listType>
        </types>
        <queryAction>
```

<queryType type="queryType">byIdentifyNumber</queryType>
<identifyNumber type="string" maxLen="127"><![CDATA[A123]]></identifyNumber>
</queryAction>
</config>

<!--example: 4 按查询人员编码-->
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
<types>
< queryType>
<enum>byPersonID</enum>
<enum>byListType</enum>
<enum>byName</enum>
<enum>byIdentifyNumber</enum>
</queryType>
<listType>
<enum>strangerList</enum>
<enum>whiteList</enum>
<enum>blackList</enum>
</listType>
</types>
<queryAction>
<queryType type="queryType">byPersonID</queryType>
<personID type="uint32">1543018104</personID>
</queryAction>

Successful Response	The "face" element will be included in the entity of the successful response. For example:

<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
<face>

```
        <personID type="list" maxCount="20000" count="4">
            <itemType type="uint32"/>
            <item>1543018099</item>
            <item>1543018100</item>
            <item>1543018101</item>
            <item>1543018104</item>
        </personID>
    </face>
</config>
```

[Tips]:

1. A2 IPC 目标库上限为 20000
2. 没有过滤条件则按顺序返回相册库 ID

11.1.7 SearchSnapFaceByTime

SearchSnapFaceByTime	
Description	Get the face info from the target lib.
Typical URL	POSThttp://<host>[:port]/SearchSnapFaceByTime[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The "queryAction" element will be included in the entity of request message. For example:

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
    <search>
        <starttime type="string"><![CDATA[2017-06-30 00:00:00]]></starttime>
        <endtime type="string"><![CDATA[2017-06-30 23:59:59]]></endtime>
```

</search>

</config>

Successful Response	The "face" element will be included in the entity of the successful response. For example:

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
    <captureFaceList type="list" count="3">
        <item>
            <snapTime type="uint64">6234564566</snapTime>
            <faceID type="uint32">66</faceID>
        </item>
        <item>
            <snapTime type="uint64">6234780985</snapTime>
            <faceID type="uint32">195</faceID>
        </item>
        <item>
            <snapTime type="uint64">7645456908</snapTime>
            <faceID type="uint32">10320</faceID>
        </item>
    </captureFaceList>
</config>
```

[Tips]:

最大返回 1000 条有效结果信息

11.1.8 SearchSnapFaceByKey

SearchSnapFaceByKey	
Description	Get the face info from the target lib.
Typical URL	POSThttp://<host>[:port]/SearchSnapFaceByKey[/channelId]

Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The "queryAction" element will be included in the entity of request message. For example:

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
    <search>
        <snapTime type="uint64">6234564566</snapTime>
        <faceID type="uint32">66</faceID>
        <requestPanoramicPic type="boolean">true</requestPanoramicPic>
        <requestPersonPic type="boolean">true</requestPersonPic>
    </search>
</config>
```

Successful Response	The "face" element will be included in the entity of the successful response. For example:

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
    <types>
        <listType>
            <enum>strangerList</enum>
            <enum>whiteList</enum>
            <enum>blackList</enum>
        </listType>
        <sexType>
            <enum>male</enum>
            <enum>female</enum>
        </sexType>
        <formatType>
            <enum>jpg</enum>
        </formatType>
```

```
</types>
<snapFace>
    <snapInfo>
        <time type="string"><![CDATA[2017-06-30 00:00:00]]></time>
        <pictureData type="string" maxLen="95576">
            <![CDATA[Base64 Picture Data]]>
        </pictureData>
        <width type="uint32">100</width>
        <height type="uint32">80</height>
        <format type="formatType">jpg</format>
        <size type="uint32">50000</size>
    </snapInfo>
    <panoramicInfo>
        <pictureData type="string" maxLen="95576">
            <![CDATA[Base64 Picture Data]]>
        </pictureData>
        <width type="uint32">100</width>
        <height type="uint32">80</height>
        <format type="formatType">jpg</format>
        <size type="uint32">50000</size>
    </panoramicInfo>
    <matchInfo>
        <similarity type="uint8">83</similarity>
        <threshold type="uint8">80</threshold>
        <temperature type="uint32">3650</temperature>
        <personInfo>
            <listType type="listType">whiteList</listType>
            <name type="string" maxLen="127"><![CDATA[user]]></name>
            <sex type="sexType">male</sex>
            <age type="uint32">34</age>
            <identifyNumber type="string" maxLen="127">
                <![CDATA[A123]]>
```

```
                    </identifyNumber>
                    <telephone type="string" maxLen="63"><![CDATA[18888888888]]></telephone>
                    <comment type="string" maxLen="63"><![CDATA[]]></comment>
                    <picInfo>
                        <pictureData type="string" maxLen="95576">
                            <![CDATA[Base64 Picture Data]]>
                        </pictureData>
                        <width type="uint32">100</width>
                        <height type="uint32">80</height>
                        <format type="formatType">jpg</format>
                        <size type="uint32">50000</size>
                    </picInfo>
                </personInfo>
            </matchInfo>
        </snapFace>
    </config>
```

[Tips]:

若没有比对则无 matchInfo 节点，若比对失败则无 personInfo 节点，requestPanoramicPic 为 false 则无 panoramicInfo 节点，requestPersonPic 为 false 则无 picInfo 节点

11.2 Crowd Density Detection

11.2.1 GetSmartCddConfig

GetSmartCddConfig	
Description	To get the IP media device's Crowd Density Detection configuration.
Typical URL	POST or GET http://<host>[:port]/GetSmartCddConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None

Entity Data	None
Successful Response	The CDD configuration will be included in the entity of the successful response. For example:

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
    <types>
        <refreshFrequency>
            <enum>500</enum>
            <enum>1000</enum>
            <enum>1500</enum>
            <enum>2000</enum>
        </refreshFrequency>
    </types>
    <cdd>
        <switch type="boolean">false</switch>
        <alarmHoldTime type="uint32">20</alarmHoldTime>
        <regionInfo type="list" maxCount="1" count="1">
            <item type="rectangle">
                <X1 type="uint32">2000</X1>
                <Y1 type="uint32">2000</Y1>
                <X2 type="uint32">8000</X2>
                <Y2 type="uint32">8000</Y2>
            </item>
        </regionInfo>
        <detectFrequency type="refreshFrequency">1000</detectFrequency>
        <triggerAlarmLevel type="uint32" min="1" max="100">1</triggerAlarmLevel>
    </cdd>
</config>
```

[Tips]:

1.The two coordinate points of "regionInfo.item" represent the two points of the rectangular diagonal.

2.The unit of "detectFrequency" is milliseconds.

11.2.2 SetSmartCddConfig

SetSmartCddConfig	
Description	To set the IP media device's Crowd Density Detection configuration.
Typical URL	POST http://<host>[:port]/SetSmartCddConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The whole "cdd" element in the "GetSmartCddConfig" should be included in entity of this message.
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]:	

11.3 People Counting

11.3.1 GetSmartCpcConfig

GetSmartCpcConfig	
Description	To get the IP media device's Cross-line People Counting configuration.
Typical URL	POST or GET http://<host>[:port]/GetSmartCpcConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The CPC configuration will be included in the entity of the successful response. For example:
<?xml version="1.0" encoding="utf-8"?>	

```
<config xmlns="http://www.ipc.com/ver10" version="1.0">
    <types>
        <statisticalPeriod>
            <enum>all</enum>
            <enum>daily</enum>
            <enum>weekly</enum>
            <enum>monthly</enum>
        </statisticalPeriod>
    </types>
    <cpc>
        <switch type="boolean">true</switch>
        <alarmHoldTime type="uint32">20</alarmHoldTime>
        <regionInfo type="list" maxCount="1" count="1">
            <item type="rectangle">
                <X1 type="uint32">2000</X1>
                <Y1 type="uint32">2000</Y1>
                <X2 type="uint32">8000</X2>
                <Y2 type="uint32">8000</Y2>
            </item>
        </regionInfo>
        <directionInfo type="list" maxCount="1" count="1">
            <item>
                <startX type="uint32">2000</startX >
                <startY type="uint32">5000</startY >
                <endX type="uint32">8000</endX >
                <endY type="uint32">5000</endY >
            </item>
        </directionInfo>
        <detectSensitivity type="uint32" min="1" max="3">2</detectSensitivity>
        <crossInThreshold type="uint32" min="1" max="655350">1000</crossInThreshold>
        <crossOutThreshold type="uint32" min="1" max="655350">1000</crossOutThreshold>
        <twoWayDiffThreshold type="uint32" min="1" max="655350">500</twoWayDiffThreshold>
```

<forceReset type="boolean">false</forceReset> <statisticalPeriod type="statisticalPeriod">daily</statisticalPeriod> </cpc> </config>
[Tips]: 1.The two coordinate points of "regionInfo.item" represent the two points of the rectangular diagonal.

11.3.2 SetSmartCpcConfig

SetSmartCpcConfig	
Description	To set the IP media device's Cross-line People Counting configuration.
Typical URL	POST http://<host>[:port]/SetSmartCpcConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The whole "cpc" element in the "GetSmartCpcConfig" should be included in entity of this message.
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]:	

11.4 People Intrusion

11.4.1 GetSmartIpdConfig

GetSmartIpdConfig	
Description	To get the IP media device's Intruding People Detection configuration.
Typical URL	POST or GET http://<host>[:port]/GetSmartIpdConfig[/channelId]

Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The IPD configuration will be included in the entity of the successful response. For example:

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
    <ipd>
        <switch type="boolean">true</switch>
        <alarmHoldTime type="uint32">20</alarmHoldTime>
        <detectSensitivity type="uint32" min="1" max="3">2</detectSensitivity>
    </ipd>
</config>
```

[Tips]:

11.4.2 SetSmartIpdConfig

SetSmartIpdConfig	
Description	To set the IP media device's Intruding People Detection configuration.
Typical URL	POST http://<host>[:port]/SetSmartIpdConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The whole "ipd" element in the "GetSmartIpdConfig" should be included in entity of this message.
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]:	

11.5 Line Crossing

11.5.1 GetSmartPerimeterConfig

GetSmartPerimeterConfig	
Description	To get the IP media device's Perimeter configuration.
Typical URL	POST or GET http://<host>[:port]/GetSmartPerimeterConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The Perimeter configuration will be included in the entity of the successful response. For example:

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
    <perimeter>
        <switch type="boolean">true</switch>
<alarmHoldTime type="uint32">20</alarmHoldTime>
<objectFilter>
<car>
        <switch type="boolean">true</switch>
        <sensitivity type="uint32" max="100" min="1" default="50">50</sensitivity>
    </car>
    <person>
        <switch type="boolean">true</switch>
        <sensitivity type="uint32" max="100" min="1" default="50">50</sensitivity>
    </person>
    <motor>
        <switch type="boolean">true</switch>
        <sensitivity type="uint32" max="100" min="1" default="50">50</sensitivity>
    </motor>
```

```
</objectFilter>
<--! maxTargetFrame minTargetFrame    researved -->
<maxTargetFrame type="uint16">0</maxTargetFrame>
<minTargetFrame type="uint16">0</minTargetFrame>
<saveTargetPicture type="boolean">false</saveTargetPicture>
<saveSourcePicture type="boolean">false</saveSourcePicture>
            <regionInfo type="list" maxCount="4" count="1">
                  <item>
                        <pointGroup type="list" maxCount="8" count="4">
                              <item>
                                    <X type="uint32">4075</X>
                                    <Y type="uint32">2466</Y>
                              </item>
                              <item>
                                    <X type="uint32">8025</X>
                                    <Y type="uint32">2833</Y>
                              </item>
                              <item>
                                    <X type="uint32">8150</X>
                                    <Y type="uint32">6366</Y>
                              </item>
                              <item>
                                    <X type="uint32">4475</X>
                                    <Y type="uint32">7233</Y>
                              </item>
                        </pointGroup>
                  </item>
            </regionInfo>
      </perimeter>
</config>
```

[Tips]:

11.5.2 SetSmartPerimeterConfig

SetSmartPerimeterConfig	
Description	To set the IP media device's Perimeter configuration.
Typical URL	POST http://<host>[:port]/SetSmartPerimeterConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The whole "perimeter" element in the "GetSmartPerimeterConfig" should be included in entity of this message.
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]:	

11.6 Intrusion

11.6.1 GetSmartTripwireConfig

GetSmartTripwireConfig	
Description	To get the IP media device's Tripwire configuration.
Typical URL	POST or GET http://<host>[:port]/GetSmartTripwireConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The Tripwire configuration will be included in the entity of the successful response. For example:
<?xml version="1.0" encoding="utf-8"?>	

```
<config xmlns="http://www.ipc.com/ver10" version="1.0">
    <types>
        <tripwireDirection>
            <enum>none</enum>
            <enum>rightortop</enum>
            <enum>leftorbotton</enum>
        </tripwireDirection>
    </types>
    <tripwire>
        <switch type="boolean">false</switch>
        <alarmHoldTime type="uint32">20</alarmHoldTime>
        <objectFilter>
            <car>
                <switch type="boolean">true</switch>
                <sensitivity type="uint32" max="100" min="1" default="50">50</sensitivity>
            </car>
            <person>
                <switch type="boolean">true</switch>
                <sensitivity type="uint32" max="100" min="1" default="50">50</sensitivity>
            </person>
            <motor>
                <switch type="boolean">true</switch>
                <sensitivity type="uint32" max="100" min="1" default="50">50</sensitivity>
            </motor>
        </objectFilter>
        <--! maxTargetFrame minTargetFrame    researved -->
        <maxTargetFrame type="uint16">0</maxTargetFrame>
        <minTargetFrame type="uint16">0</minTargetFrame>
        <saveTargetPicture type="boolean">false</saveTargetPicture>
        <saveSourcePicture type="boolean">false</saveSourcePicture>
        <lineInfo type="list" maxCount="4" count="1">
            <item>
```

```
                    <direction type="tripwireDirection">rightortop</direction>
                    <startPoint>
                        <X type="uint32">10</X>
                        <Y type="uint32">10</Y>
                    </startPoint>
                    <endPoint>
                        <X type="uint32">1000</X>
                        <Y type="uint32">1000</Y>
                    </endPoint>
                </item>
            </lineInfo>
        </tripwire>
</config>
```

[Tips]:

11.6.2 SetSmartTripwireConfig

SetSmartTripwireConfig	
Description	To set the IP media device's Tripwire configuration.
Typical URL	POST http://<host>[:port]/SetSmartTripwireConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The whole "tripwire" element in the "GetSmartTripwireConfig" should be included in entity of this message.
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]:	

11.7 Object Removal

11.7.1 GetSmartOscConfig

GetSmartOscConfig	
Description	To get the IP media device's Object Status Change configuration.
Typical URL	POST or GET http://<host>[:port]/GetSmartOscConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The OSC configuration will be included in the entity of the successful response. For example:

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
    <types>
        <oscObject>
            <enum>abandum</enum>
            <enum>objstolen</enum>
        </oscObject>
    </types>
    <osc>
        <switch type="boolean">true</switch>
        <oscObject type="oscObject">abandum</oscObject>
        <alarmHoldTime type="uint32">20</alarmHoldTime>
        <regionInfo type="list" maxCount="4" count="1">
            <item>
                <regionName type="string" maxLen="15"><![CDATA[object]]></regionName>
                <pointGroup type="list" maxCount="8" count="4">
                    <item>
```

```
                        <X type="uint32">4075</X>
                        <Y type="uint32">2466</Y>
                    </item>
                    <item>
                        <X type="uint32">8025</X>
                        <Y type="uint32">2833</Y>
                    </item>
                    <item>
                        <X type="uint32">8150</X>
                        <Y type="uint32">6366</Y>
                    </item>
                    <item>
                        <X type="uint32">4475</X>
                        <Y type="uint32">7233</Y>
                    </item>
                </pointGroup>
            </item>
        </regionInfo>
    </osc>
</config>
```

[Tips]:

11.7.2 SetSmartOscConfig

SetSmartOscConfig	
Description	To set the IP media device's Object Status Change configuration.
Typical URL	POST http://<host>[:port]/SetSmartOscConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None

Entity Data	The whole "osc" element in the "GetSmartOscConfig" should be included in entity of this message.
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]:	

11.8 Exception

11.8.1 GetSmartAvdConfig

GetSmartAvdConfig	
Description	To get the IP media device's Abnormal Video Diagnosis configuration.
Typical URL	POST or GET http://<host>[:port]/GetSmartAvdConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The AVD configuration will be included in the entity of the successful response. For example:

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
    <avd>
        <alarmHoldTime type="uint32">20</alarmHoldTime>
        <sceneChangeSwitch type="boolean">true</sceneChangeSwitch>
        <clarityAbnormalSwitch type="boolean">true</clarityAbnormalSwitch>
        <colorAbnormalSwitch type="boolean">true</colorAbnormalSwitch>
        <sensitivity type="uint32" min="1" max="100">100</sensitivity>
    </avd>
</config>
```

[Tips]:

11.8.2 SetSmartAvdConfig

SetSmartAvdConfig	
Description	To set the IP media device's Abnormal Video Diagnosis configuration.
Typical URL	POST http://<host>[:port]/SetSmartAvdConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The whole "avd" element in the "GetSmartAvdConfig" should be included in entity of this message.
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]:	

11.9 License Plate Recognition

11.9.1 GetSmartVehicleConfig

GetVehicleConfig	
Description	To get vehicle's details.
Typical URL	POST or GET http://<host>[:port]/GetSmartVehicleConfig
Channel ID	None
Action name	None
Entity Data	None
Successful	The device detail will be included in the entity of the successful

Response	response. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.7" xmlns="http://www.ipc.com/ver10">
    <types>
<detectModeType>
<enum>auto</enum>
<enum>fixedInterval</enum>
</detectModeType>
<mutexObjectType>
<enum>cdd</enum>
<enum>cpc</enum>
<enum>ipd</enum>
<enum>tripwire</enum>
<enum>osc</enum>
<enum>perimeter</enum>
<enum>vfd</enum>
<enum>avd</enum>
<enum>aoientry</enum>
<enum>aoileave</enum>
<enum>h264s</enum>
<enum>h265s</enum>
</mutexObjectType>
<plateAreaType>
<enum>CN</enum>
<enum>HK</enum>
<enum>TW</enum>
<enum>USA</enum>
<enum>ISRAEL</enum>
</plateAreaType>
<alarmListType>
<enum>blackList</enum>
<enum>whiteList</enum>
```

```
<enum>strangerList</enum>
</alarmListType>
</types>
    <vehicle>
        <mutexList type="list" count="0">
        <item>
        <object type="mutexObjectType">h264s</object>
        <status type="boolean">false</status>
        <item>
        <object type="mutexObjectType">h265s</object>
        <status type="boolean">false</status>
        </item>
        </mutexList>
        <switch type="boolean">false</switch>
        <plateSencitivity type="uint8">49</plateSencitivity>
        <plateSupportArea type="plateAreaType">CN</plateSupportArea>
        <faultTolerance type="uint8">0</faultTolerance>
        <regionInfo type="list" maxCount="1" count="1">
            <item>
            <X1 type="uint32">0</X1>
            <Y1 type="uint32">0</Y1>
            <X2 type="uint32">10000</X2>
            <Y2 type="uint32">10000</Y2>
            </item>
        </regionInfo>
        <plateSize type="list" maxCount="1" count="1">
            <item>
            <MinWidth type="int32">982</MinWidth>
            <MinHeight type="int32">982</MinHeight>
            <MaxWidth type="int32">2982</MaxWidth>
            <MaxHeight type="int32">2982</MaxHeight>
            </item>
```

```
            </plateSize>
            <maskArea type="list" count="4">
                <item>
                <point type="list" maxCount="8" count="0">
                </point>
                </item>
                <item>
                <point type="list" maxCount="8" count="0">
                </point>
                </item>
                <item>
                <point type="list" maxCount="8" count="0">
                </point>
                </item>
                <item>
                <point type="list" maxCount="8" count="0">
                </point>
                </item>
            </maskArea>
            <plateMatch>
                <item>
                    <alarmList type="alarmListType">whiteList</alarmList>
                        <triggerAlarmOut>
                            <Io type="list" maxCount="8" count="2">
                                <item>
                                    <alarmId type="uint32">0</alarmId>
                                    <switch type="boolean">false</switch>
                                </item>
                                <item>
                                    <alarmId type="uint32">1</alarmId>
                                    <switch type="boolean">false</switch>
                                </item>
```

```
                    </Io>
                </triggerAlarmOut>
            </item>
            <item>
                <alarmList type="alarmListType">whiteList</alarmList>
                <triggerAlarmOut>
                    <Io type="list" maxCount="8" count="2">
                        <item>
                            <alarmId type="uint32">0</alarmId>
                            <switch type="boolean">false</switch>
                        </item>
                        <item>
                            <alarmId type="uint32">1</alarmId>
                            <switch type="boolean">false</switch>
                        </item>
                    </Io>
                </triggerAlarmOut>
            </item>
            <item>
                <alarmList type="alarmListType">whiteList</alarmList>
                <triggerAlarmOut>
                    <Io type="list" maxCount="8" count="2">
                        <item>
                            <alarmId type="uint32">0</alarmId>
                            <switch type="boolean">false</switch>
                        </item>
                        <item>
                            <alarmId type="uint32">1</alarmId>
                            <switch type="boolean">false</switch>
                        </item>
                    </Io>
                </triggerAlarmOut>
```

```
                </item>
            </plateMatch>
            <triggerConfig>
                <alarmHoldTime type="uint32">20</alarmHoldTime>
                <sdSnapSwitch type="boolean">true</sdSnapSwitch>
                <sdRecSwitch type="boolean">true</sdRecSwitch>
                <triggerAlarmOut>
                    <alarmOutList type="list" maxCount="2" count="2">
                        <item>
                            <alarmOutId type="uint32">0</alarmOutId>
                            <alarmSwitch type="boolean">false</alarmSwitch>
                        </item>
                        <item>
                            <alarmOutId type="uint32">1</alarmOutId>
                            <alarmSwitch type="boolean">false</alarmSwitch>
                        </item>
                    </alarmOutList>
                </triggerAlarmOut>
                    <triggerMail>
                        <switch type="boolean">false</switch>
                        <subject type="string" maxLen="63"><![CDATA[]]></subject>
                        <content type="string" maxLen="255"><![CDATA[]]></content>
                        <recvList type="list" maxCount="5" count="0">
                        </recvList>
                    </triggerMail>
                    <triggerFtp>
                        <switch type="boolean">false</switch>
                        <ftpServerList type="list" maxCount="1" count="0">
                    </ftpServerList>
                    </triggerFtp>
            </triggerConfig>
        </vehicle>
```

</config>
[Tips]:

11.9.2 SetSmartVehicleConfig

SetVehicleConfig	
Description	To set the IP media device's Video Vehicle Detection configuration.
Typical URL	POST http://<host>[:port]/SetSmartVehicleConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The whole "vehilce" element in the"GetVehilceConfig" should be included in entity of this message.
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]:	

11.9.3 AddVehiclePlate

AddVehiclePlate	
Description	To set the schedulein batches.
Typical URL	POST or GET http://<host>[:port]/AddVehiclePlate
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
request:	

<?xml version="1.0" encoding="utf-8" ?>
<config>
<vehiclePlates type="list" count="1">
<item>
<carPlateNumber type ="string"><![CDATA[粤 B 123456]]></carPlateNumber>
<beginTime type ="string"><![CDATA[2019/08/22 00:00:00]]></beginTime>
<endTime type ="string"><![CDATA[2019/08/22 23:59:59]]></endTime>
<carPlateColor type ="string"><![CDATA[]]></carPlateColor>
<carPlateType type ="string"><![CDATA[小型车]]></carPlateType>
<carType type ="unit32"><![CDATA[undefined]]></carType>
<carOwner type ="string"><![CDATA[TEST]]></carOwner>
<carColor type ="string"><![CDATA[undefined]]></carColor>
<plateItemType type ="string">strangerList</plateItemType>
</item>
</vehiclePlates>
</config>

respone:

<?xml version="1.0" encoding="UTF-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.7">
<vehiclePlatesReply type="list" count="0">
</vehiclePlatesReply>
</config>

Successful Response	The standard successful result response that described in 1.3.5.

1.返回失败列表，如果 count 为 0 表示全部成功

11.9.4 DeleteVehiclePlate

DeleteVehiclePlate	
Description	To set the schedulein batches.
Typical URL	POST or GET http://<host>[:port]/DeleteVehiclePlate
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None

request:

```
<?xml version="1.0" encoding="utf-8" ?>
<config>
<vehiclePlates>
<keyList type="list" count="1">
<item>
<keyId type="unit32">1566443406</keyId>
</item>
</keyList>
<listType>
</listType>
<carPlateNum><![CDATA[]]></carPlateNum>
</vehiclePlates>
</config>
```

respone:

```
<?xml version="1.0" encoding="UTF-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.7">
<vehiclePlatesReply type="list" count="0">
</vehiclePlatesReply>
</config>
```

Successful Response	The standard successful result response that described in 1.3.5.
删除车牌支持如下三选一进行删除： 1. 支持 keyid list 批量删除。 2. 支持黑白名单删除 3. 支持车牌模糊删除	

11.9.5 EditVehiclePlate

EditVehiclePlate	
Description	To set the schedulein batches.
Typical URL	POST or GET http://<host>[:port]/EditVehiclePlate
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None

request:

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.7">
   <vehiclePlates>
        <keyId type="unit32">1300</keyId>
        <carPlateNumber type="string">粤 BCY113</carPlateNumber>
        <beginTime type="string">2019-1-1 12:23:00</beginTime>
        <endTime type="string">2019-1-1 12:23:00</endTime>
        <carPlateColor type="string">red</carPlateColor>
        <carPlateType type="string">1566</carPlateType>
        <carType type="unit32">1566</carType>
        <carOwner type="string">dengyuhui</carOwner>
        <carColor type="string">red</carColor>
```

<table>
<tr><td colspan="2">
<plateItemType type="string">1566</plateItemType>
</vehiclePlates>
</config>

respone:
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.7">
<vehiclePlatesReply>
<keyId type="unit32">1300</keyId>
<status type="unit32">0</status>
</vehiclePlatesReply>
</config>
</td></tr>
<tr><td>Successful Response</td><td>The standard successful result response that described in 1.3.5.</td></tr>
<tr><td colspan="2">1.respone 中的 status 等于 0 标识成功，非 0 标识修改失败</td></tr>
</table>

11.9.6 GetVehiclePlate

<table>
<tr><th colspan="2">GetVehiclePlate</th></tr>
<tr><td>Description</td><td>To set the schedulein batches.</td></tr>
<tr><td>Typical URL</td><td>POST or GET http://<host>[:port]/GetVehiclePlate</td></tr>
<tr><td>Channel ID</td><td>Optional. If none channel ID included in the URL, the default channel ID is 1.</td></tr>
<tr><td>Action name</td><td>None</td></tr>
<tr><td>Entity Data</td><td>None</td></tr>
<tr><td colspan="2">
<config xmlns="http://www.ipc.com/ver10" version="1.7">
<vehiclePlates type="list" maxCount="10000" count="1">
<searchFilter>
<pageIndex type="unit32">1300</pageIndex>
</td></tr>
</table>

<pageSize type="unit32">1</pageSize>
<listType type="unit32">0</listType>
<carPlateNum type="string">0</carPlateNum>
</item>
</searchFilter>
</config>

respone:

<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.7">
<vehiclePlates type="list" maxCount="10000" count="1">
<item>
<keyId type="unit32">1300</keyId>
<carPlateNumber type="string">粤 BCY113</carPlateNumber>
<beginTime type="string">2019-1-1 12:23:00</beginTime>
<endTime type="string">2019-1-1 12:23:00</endTime>
<carPlateColor type="string">red</carPlateColor>
<carPlateType type="string">1566</carPlateType>
<carType type="unit32">1566</carType>
<carOwner type="string">dengyuhui</carOwner>
<carColor type="string">red</carColor>
<plateItemType type="string">1566</plateItemType>
</item>
</vehiclePlates>
</config>

Successful Response	The standard successful result response that described in 1.3.5.

[Tips]:

1、 The "GetDeviceDetail" includes how many sensors the device supported.

2、The "types" is defined by this document to constrain how the "schedule.object" is filled out, it can

not be included in this message.

11.9.7 GetVehiclePlateProgress

GetVehiclePlateProgress	
Description	批量导入车牌库的进度
Typical URL	POST or GET http://<host>[:port]/GetVehiclePlateProgress
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
<?xml version="1.0" encoding="UTF-8"?> <config xmlns="http://www.ipc.com/ver10" version="1.7"> <vehiclePlatesReply>10000</vehiclePlatesReply> </config>	
Successful Response	The standard successful result response that described in 1.3.5.
结果按照换算为%比时除以 100，表示比例	

11.10 Region Entrance

11.10.1 GetSmartAoiEntryConfig

GetSmartAoiEntryConfig	
Description	
Typical URL	POST or GEThttp://<host>[:port]/GetSmartAoiEntryConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.

GetSmartAoiEntryConfig	
Action name	None
Entity Data	None
Successful Response	The configuration will be included in the entity of the successful response. For example:

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.7">
<types>
<mutexObjectType>
<enum>cdd</enum>
<enum>cpc</enum>
<enum>ipd</enum>
<enum>tripwire</enum>
<enum>osc</enum>
<enum>perimeter</enum>
<enum>vfd</enum>
<enum>avd</enum>
<enum>vehicle</enum>
</mutexObjectType>
</types>
<aoientry>
<mutexList type="list" count="2">
<item>
<object type="mutexObjectType">tripwire</object>
<status type="boolean">false</status>
</item>
<item>
<object type="mutexObjectType">vfd</object>
<status type="boolean">false</status>
</item>
</mutexList>
```

GetSmartAoiEntryConfig

```
<switch type="boolean">false</switch>
<objectFilter>
    <car>
<switch type="boolean">true</switch>
    <sensitivity type="uint32" max="100" min="1" default="50">85</sensitivity>
    </car>
    <person>
    <switch type="boolean">true</switch>
    <sensitivity type="uint32" max="100" min="1" default="50">60</sensitivity>
    </person>
    <motor>
    <switch type="boolean">true</switch>
    <sensitivity type="uint32" max="100" min="1" default="50">85</sensitivity>
    </motor>
</objectFilter>
  <saveTargetPicture type="boolean">false</saveTargetPicture>
  <saveSourcePicture type="boolean">false</saveSourcePicture>
<boundary type="list" count="4">
    <item>
        <point type="list" maxCount="6" count="5">
            <item><X type="unit32">1300</X><Y type="unit32">1566</Y></item>
            <item><X type="unit32">1925</X><Y type="unit32">8433</Y></item>
            <item><X type="unit32">8650</X><Y type="unit32">9033</Y></item>
            <item><X type="unit32">8725</X><Y type="unit32">833</Y></item>
            <item><X type="unit32">1200</X><Y type="unit32">1133</Y></item>
        </point>
    </item>
    <item>
        <point type="list" maxCount="6" count="0">
        </point>
    </item>
```

GetSmartAoiEntryConfig
<item> <point type="list" maxCount="6" count="0"> </point> </item> <item> <point type="list" maxCount="6" count="0"> </point> </item> </boundary> </aoientry> </config> [Tips]:

11.10.2 SetSmartAoiEntryConfig

SetSmartAoiEntryConfig	
Description	To set the IP media device's video stream configuration for specific channel.
Typical URL	POST http://<host>[:port]/SetSmartAoiEntryConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	
Successful Response	The standard successful result response that described in 1.3.5.

SetSmartAoiEntryConfig
[Tips]: IPC does not support

11.11 Region Entrance

11.11.1 GetSmartAoiLeaveConfig

GetSmartAoiLeaveConfig	
Description	
Typical URL	POST or GEThttp://<host>[:port]/GetSmartAoiLeaveConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The configuration will be included in the entity of the successful response. For example:

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.7">
<types>
<mutexObjectType>
<enum>cdd</enum>
<enum>cpc</enum>
<enum>ipd</enum>
<enum>tripwire</enum>
<enum>osc</enum>
<enum>perimeter</enum>
<enum>vfd</enum>
```

GetSmartAoiLeaveConfig

```
            <enum>avd</enum>
            <enum>vehicle</enum>
            </mutexObjectType>
            </types>
          <aoileave>
          <mutexList type="list" count="2">
            <item>
            <object type="mutexObjectType">tripwire</object>
            <status type="boolean">false</status>
            </item>
            <item>
            <object type="mutexObjectType">vfd</object>
            <status type="boolean">false</status>
            </item>
        </mutexList>
        <switch type="boolean">false</switch>
        <objectFilter>
        <car>
              <switch type="boolean">true</switch>
              <sensitivity type="uint32" max="100" min="1" default="50">85</sensitivity>
        </car>
        <person>
              <switch type="boolean">true</switch>
              <sensitivity type="uint32" max="100" min="1" default="50">60</sensitivity>
        </person>
        <motor>
              <switch type="boolean">true</switch>
              <sensitivity type="uint32" max="100" min="1" default="50">85</sensitivity>
        </motor>
  </objectFilter>
          <saveTargetPicture type="boolean">false</saveTargetPicture>
```

GetSmartAoiLeaveConfig

```
        <saveSourcePicture type="boolean">false</saveSourcePicture>
    <boundary type="list" count="4">
        <item>
            <point type="list" maxCount="6" count="5">
                <item><X type="unit32">1300</X><Y type="unit32">1566</Y></item>
                <item><X type="unit32">1925</X><Y type="unit32">8433</Y></item>
                <item><X type="unit32">8650</X><Y type="unit32">9033</Y></item>
                <item><X type="unit32">8725</X><Y type="unit32">833</Y></item>
                <item><X type="unit32">1200</X><Y type="unit32">1133</Y></item>
            </point>
        </item>
        <item>
            <point type="list" maxCount="6" count="0">
            </point>
        </item>
        <item>
            <point type="list" maxCount="6" count="0">
            </point>
        </item>
        <item>
            <point type="list" maxCount="6" count="0">
            </point>
        </item>
    </boundary>
</aoileave>
</config>
```

[Tips]:

11.11.2 SetSmartAoiLeaveConfig

SetSmartAoiEntryConfig	
Description	To set the IP media device's video stream configuration for specific channel.
Typical URL	POST http://<host>[:port]/SetSmartAoiEntryConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]: IPC does not support	

11.12 Target Counting

11.12.1 GetSmartPassLineCountConfig

GetSmartPassLineCountConfig	
Description	
Typical URL	POST or GEThttp://<host>[:port]/GetPassLineCountConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful	The configuration will be included in the entity of the successful

GetSmartPassLineCountConfig	
Response	response. For example:

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.7">
<types>
<direction>
<enum>none</enum>
<enum>rightortop</enum>
<enum>leftorbotton</enum>
</direction>
<mutexObjectType>
<enum>cdd</enum>
<enum>cpc</enum>
<enum>ipd</enum>
<enum>tripwire</enum>
<enum>osc</enum>
<enum>perimeter</enum>
<enum>vfd</enum>
<enum>avd</enum>
<enum>vehicle</enum>
</mutexObjectType>
<countCycleType>
<enum>day</enum>
<enum>week</enum>
<enum>month</enum>
<enum>off</enum>
</countCycleType>
</types>
<passlinecount>
<mutexList type="list" count="2">
<item>
<object type="mutexObjectType">tripwire</object>
```

GetSmartPassLineCountConfig

```
<status type="boolean">false</status>
</item>
<item>
<object type="mutexObjectType">vfd</object>
<status type="boolean">false</status>
</item>
</mutexList>
<switch type="boolean">false</switch>
 <objectFilter>
      <car>
          <switch type="boolean">true</switch>
          <sensitivity type="uint32" max="100" min="1" default="50">85</sensitivity>
      </car>
      <person>
          <switch type="boolean">true</switch>
          <sensitivity type="uint32" max="100" min="1" default="50">60</sensitivity>
      </person>
      <motor>
          <switch type="boolean">true</switch>
          <sensitivity type="uint32" max="100" min="1" default="50">85</sensitivity>
      </motor>
 </objectFilter>
<saveTargetPicture type="boolean">false</saveTargetPicture>
<saveSourcePicture type="boolean">false</saveSourcePicture>
 <countPeriod>
    <countTimeType type="countCycleType">off</countTimeType>
    <daily>
       <dateSpan type="uint32">0</dateSpan>
       <dateTimeSpan type="string">00:00:00</dateTimeSpan>
    </daily>
    <weekly>
```

GetSmartPassLineCountConfig

```
        <dateSpan type="uint32">0</dateSpan>
        <dateTimeSpan type="string">00:00:00</dateTimeSpan>
      </weekly>
      <monthly>
        <dateSpan type="uint32">0</dateSpan>
        <dateTimeSpan type="string">00:00:00</dateTimeSpan>
      </monthly>
    </countPeriod>
    <countOSD>
      <switch type="boolean">true</switch>
      <X type="uint32">6600</X>
      <Y type="uint32">100</Y>
      <osdFormat type="string"><![CDATA[Entrance: human-# car-# bike-# \nExit    : human-# car-# bike-#]]></osdFormat>
    </countOSD>
    <line type="list" count="4">
    <item>
    <direction type="direction">rightortop</direction>
    <startPoint>
    <X type="uint32">0</X>
    <Y type="uint32">0</Y>
    </startPoint>
    <endPoint>
    <X type="uint32">0</X>
    <Y type="uint32">0</Y>
    </endPoint></item>
    <item>
    <direction type="direction">rightortop</direction>
    <startPoint>
    <X type="uint32">0</X>
    <Y type="uint32">0</Y>
```

GetSmartPassLineCountConfig

</startPoint>

<endPoint>

<X type="uint32">0</X>

<Y type="uint32">0</Y>

</endPoint></item>

<item>

<direction type="direction">rightortop</direction>

<startPoint>

<X type="uint32">0</X>

<Y type="uint32">0</Y>

</startPoint>

<endPoint>

<X type="uint32">0</X>

<Y type="uint32">0</Y>

</endPoint></item>

<item>

<direction type="direction">rightortop</direction>

<startPoint>

<X type="uint32">0</X>

<Y type="uint32">0</Y>

</startPoint>

<endPoint>

<X type="uint32">0</X>

<Y type="uint32">0</Y>

</endPoint></item></line>

</passlinecount>

</config>

[Tips]:~~<?xml version="1.0" encoding="utf-8"?>~~

~~<config xmlns="http://www.ipc.com/ver10" version="1.7">~~

~~<types>~~

~~<direction>~~

GetSmartPassLineCountConfig

<enum>none</enum>

<enum>rightortop</enum>

<enum>leftorbotton</enum>

</direction>

<mutexObjectType>

<enum>cdd</enum>

<enum>cpc</enum>

<enum>ipd</enum>

<enum>tripwire</enum>

<enum>osc</enum>

<enum>perimeter</enum>

<enum>vfd</enum>

<enum>avd</enum>

<enum>vehicle</enum>

</mutexObjectType>

</types>

<passlinecount>

<mutexList type="list" count="2">

<item>

<object type="mutexObjectType">tripwire</object>

<status type="boolean">false</status>

</item>

<item>

<object type="mutexObjectType">vfd</object>

<status type="boolean">false</status>

</item>

</mutexList>

<switch type="boolean">false</switch>

<objectFilter>

<car>

<switch type="boolean">true</switch>

GetSmartPassLineCountConfig


```
        <sensitivity type="uint32" max="100" min="1" default="50">85</sensitivity>
        </car>
        <person>
        <switch type="boolean">true</switch>
        <sensitivity type="uint32" max="100" min="1" default="50">60</sensitivity>
        </person>
        <motor>
        <switch type="boolean">true</switch>
        <sensitivity type="uint32" max="100" min="1" default="50">85</sensitivity>
        </motor>
 </objectFilter>
<saveTargetPicture type="boolean">false</saveTargetPicture>
<saveSourcePicture type="boolean">false</saveSourcePicture>
<countTimeSpan>7<countTimeSpan>
<line type="list" count="4">
<item>
<direction type="direction">rightortop</direction>
<startPoint>
<X type="uint32">0</X>
<Y type="uint32">0</Y>
</startPoint>
<endPoint>
<X type="uint32">0</X>
<Y type="uint32">0</Y>
</endPoint></item>
<item>
<direction type="direction">rightortop</direction>
<startPoint>
<X type="uint32">0</X>
<Y type="uint32">0</Y>
</startPoint>
```

GetSmartPassLineCountConfig

~~<endPoint>~~

~~<X type="uint32">0</X>~~

~~<Y type="uint32">0</Y>~~

~~</endPoint></item>~~

~~<item>~~

~~<direction type="direction">rightortop</direction>~~

~~<startPoint>~~

~~<X type="uint32">0</X>~~

~~<Y type="uint32">0</Y>~~

~~</startPoint>~~

~~<endPoint>~~

~~<X type="uint32">0</X>~~

~~<Y type="uint32">0</Y>~~

~~</endPoint></item>~~

~~<item>~~

~~<direction type="direction">rightortop</direction>~~

~~<startPoint>~~

~~<X type="uint32">0</X>~~

~~<Y type="uint32">0</Y>~~

~~</startPoint>~~

~~<endPoint>~~

~~<X type="uint32">0</X>~~

~~<Y type="uint32">0</Y>~~

~~</endPoint></item></line>~~

~~</passlinecount>~~

~~</config>~~

[Tips]:

11.12.2 SetSmartPassLineCountConfig

GetPassLineCountConfig	
Description	To set the IP media device's video stream configuration for specific channel.
Typical URL	POST http://<host>[:port]/SetPassLineCountConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]:Manual Reset <?xml version="1.0" encoding="utf-8"?> <config xmlns="http://www.ipc.com/ver10" version="1.0"> <passlinecount> <forceReset type="boolean">true</forceReset> </passlinecount> </config>	
[Tips]: IPC does not support	

11.12.3 GetPassLineCountStatistics

GetPassLineCountStatistics	
Description	Get current statistics
Typical URL	POST or GET http://<host>[:port]/GetPassLineCountStatistics[/channelId]

GetPassLineCountStatistics	
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The configuration will be included in the entity of the successful response. For example:

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.7">
  <entranceCount>
    <person type="uint32">0</person>
    <car type="uint32">0</car>
    <bike type="uint32">0</bike>
  </entranceCount>
  <exitCount>
    <person type="uint32">0</person>
    <car type="uint32">0</car>
    <bike type="uint32">0</bike>
  </exitCount>
</config>
```

11.13 Thermographic Temperature Measurement

1.1. GetMeasureTemperatureConfig

GetMeasureTemperatureConfig

Description	To get thermal imaging temperature measurement detail information.
Typical URL	POST or GET http://<host>[:port]/ GetMeasureTemperatureConfig
Channel ID	None
Action name	None
Entity Data	None
Successful Response	The device detail will be included in the entity of the successful response. For example:

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

<config xmlns="http://www.ipc.com/ver10" version="1.7">
  <ThermalAlarmConfig>
    <alarmSwitch default="false" type="boolean">true</alarmSwitch>
    <highThreshold>
      <highTemperatureSwitch default="false" type="boolean">false</highTemperatureSwitch>
      <highTemperatureValue type="uint32" max="4500"
min="3000">4000</highTemperatureValue>
    </highThreshold>
    <lowThreshold>
      <lowTemperatureSwitch default="false" type="boolean">true</lowTemperatureSwitch>
      <lowTemperatureValue type="uint32" max="4500"
min="3000">4000</lowTemperatureValue>
    </lowThreshold>
    <triggerConfig>
      <alarmHoldTime type="uint32">3</alarmHoldTime>
      <sdSnapSwitch type="boolean">false</sdSnapSwitch>
      <sdRecSwitch type="boolean">false</sdRecSwitch>
      <triggerAlarmOut>
        <alarmOutList type="list" maxCount="1" count="1">
          <item>
            <alarmOutId type="uint32">0</alarmOutId>
```

```
                    <alarmSwitch type="boolean">false</alarmSwitch>
                </item>
            </alarmOutList>
        </triggerAlarmOut>
        <triggerMail>
            <switch type="boolean">false</switch>
            <subject type="string" maxLen="63"><![CDATA[]]></subject>
            <content type="string" maxLen="255"><![CDATA[]]></content>
            <recvList type="list" maxCount="5" count="0"></recvList>
        </triggerMail>
        <triggerFtp>
            <switch type="boolean">false</switch>
            <ftpServerList type="list" maxCount="1" count="0"></ftpServerList>
        </triggerFtp>
        <triggerAudio>
            <switch type="boolean">false</switch>
        </triggerAudio>
        <triggerWhiteLight>
            <switch type="boolean">false</switch>
        </triggerWhiteLight>
    </triggerConfig>
</ThermalAlarmConfig>
</config>
```

[Tips]:

1.2. SetMeasureTemperatureConfig

SetMeasureTemperatureConfig	
Description	To set the IP media device's thermal imaging temperature measurement detail information.
Typical URL	POST http://<host>[:port]/ SetMeasureTemperatureConfig [/channelId]

SetMeasureTemperatureConfig	
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	
Successful Response	The standard successful result response that described in 1.1.
[Tips]:	

1.3. GetTemperatureCalibrationConfig

GetTemperatureCalibrationConfig	
Description	To get thermal imaging temperature correction detail information.
Typical URL	POST or GET http://<host>[:port]/ GetTemperatureCalibrationConfig
Channel ID	None
Action name	None
Entity Data	None
Successful Response	The device detail will be included in the entity of the successful response. For example:

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

<config xmlns="http://www.ipc.com/ver10" version="1.0">
  <ScenseMode>
    <enum>correction</enum>
    <enum>monitoring</enum>
  </ScenseMode>
```

```
    <calibrationData>
      <mode type="ScenseMode">monitoring</mode>
      <envTemperature type="uint32" min="0" max="50">25</envTemperature>
      <envHumidity type="uint32" min="0" max="100">40</envHumidity>
      <objDistance type="uint32" min="1" max="10">2</objDistance>
      <Radiate type="uint32" min="1" max="100">80</Radiate>
      <blackBody>
        <switch type="boolean">false</switch>
        <blackPositionX type="uint32" min="0" max="10000">9400</blackPositionX>
        <blackPositionY type="uint32" min="0" max="10000">6387</blackPositionY>
        <blackTemperature type="uint32" min="0" max="100">20</blackTemperature>
      </blackBody>
      <correctionTemperature type="int32" min="-30" max="30">-3</correctionTemperature>
    </calibrationData>
</config>
```

[Tips]:

1.4. SetTemperatureCalibrationConfig

SetTemperatureCalibrationConfig	
Description	To set the IP media device's thermal imaging temperature calibration detail information.
Typical URL	POST http://<host>[:port]/ SetTemperatureCalibrationConfig [/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	

SetTemperatureCalibrationConfig	
Successful Response	The standard successful result response that described in 1.3.
[Tips]:	

1.5. GetMeasureTemperatureScheduleConfig

GetMeasureTemperatureScheduleConfig	
Description	To get thermal imaging temperature schedule detail information.
Typical URL	POST or GET http://<host>[:port]/ GetMeasureTemperatureScheduleConfig
Channel ID	None
Action name	None
Entity Data	None
Successful Response	The device detail will be included in the entity of the successful response. For example:

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

<config xmlns="http://www.ipc.com/ver10" version="1.7">
  <schedule maxTimeSpan="100" maxYearlyDay="31">
    <period mode="weekly" start="00:00" end="23:59" day="sunday"/>
    <period mode="weekly" start="00:00" end="23:59" day="monday"/>
    <period mode="weekly" start="00:00" end="23:59" day="tuesday"/>
    <period mode="weekly" start="00:00" end="23:59" day="wednesday"/>
    <period mode="weekly" start="00:00" end="23:59" day="thursday"/>
    <period mode="weekly" start="00:00" end="23:59" day="friday"/>
    <period mode="weekly" start="00:00" end="23:59" day="saturday"/>
```

<period mode="yearly" start="00:00" end="23:59" date="04-20"/> </schedule> </config>
[Tips]:

1.6. SetMeasureTemperatureScheduleConfig

SetMeasureTemperatureScheduleConfig	
Description	To set the IP media device's thermal imaging temperature schedule detail information.
Typical URL	POST http://<host>[:port]/ SetMeasureTemperatureScheduleConfig [/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	
Successful Response	The standard successful result response that described in 1.5.
[Tips]:	

1.7. GetDotTemperature

GetDotTemperature	
Description	Gets the temperature at the position of the input coordinate.
Typical URL	POST or GET http://<host>[:port]/ GetDotTemperature

Channel ID	None
Action name	None
Entity Data	<?xml version="1.0" encoding="utf-8"?> <config xmlns="http://www.ipc.com/ver10" version="1.0"> <dotTemperature> <hotX type="uint32" min="0" max="10000">0</hotX> <hotY type="uint32" min="0" max="10000">0</hotY> </dotTemperature> </config>
Successful Response	The device detail will be included in the entity of the successful response. For example:

<?xml version="1.0" encoding="utf-8"?>
<config xmlns="http://www.ipc.com/ver10" version="1.0">
<types>
<tempUnitsType>
<enum>centigrade</enum>
<enum>Fahrenheit</enum>
</tempUnitsType>
</types>
<dotTemperature>
<hotX type="uint32" min="0" max="10000">0</hotX>
<hotY type="uint32" min="0" max="10000">0</hotY>
<temperature type="int">3650</temperature>
<tempUnits type="tempUnitsType">centigrade</tempUnits>
</dotTemperature>
</config>

[Tips]:

11.14 Infrared Temperature Control

1. GetAccessControlConfig

GetAccessControlConfig	
Description	To get the IP media device's AccessControl configuration.
Typical URL	POST or GET http://<host>[:port]/ GetAccessControlConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The AccessControl configuration will be included in the entity of the successful response. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.7" xmlns="http://www.ipc.com/ver10">
<AccessControl>
<matchOpenMode type="boolean">true</matchOpenMode>
<temperatureOpen type="boolean">false</temperatureOpen>
<wearmaskOpen type="boolean">false</wearmaskOpen>
<passOpenMode>
<switch type="boolean">false</switch>
<password type="string" maxLen="15"><![CDATA[]]></password>
</passOpenMode>
<OpenDelayTime type="uint8" min="0" max="10" default="3">3</OpenDelayTime>
<OpenHoldTime type="uint8" min="1" max="10" default="5">5</OpenHoldTime>
<tamperProtection type="boolean">false</tamperProtection>
<alarmHoldTime type="uint32">20</alarmHoldTime>
<triggerAlarmOut type="list" count="2"><itemType type="boolean"/>
```

<item id="0">false</item>
<item id="1">false</item>
</triggerAlarmOut>
<mail type="list" count="0">
<switch type="boolean">false</switch>
<subject type="string" maxLen="63"><![CDATA[]]></subject>
<content type="string" maxLen="255"><![CDATA[]]></content>
</mail>
<ftp type="list" count="0">
<switch type="boolean">false</switch>
</ftp>
<savePicSwitch type="boolean">false</savePicSwitch>
<sdRecSwitch type="boolean">false</sdRecSwitch>
<audioSwitch type="boolean">false</audioSwitch>
</AccessControl>
</config>

[Tips]:

passOpenMode: Is password unlocking supported

matchOpenMode: Whether face recognition unlocking is supported (on by default)

OpenDelayTime: Unlocking delay time

OpenHoldTime: Unlocking duration (from time to automatic closing)

tamperProtection: Anti disassembly alarm linkage

2. SetAccessControlConfig

SetAccessControlConfig	
Description	To set the IP media device's AccessControl configuration.
Typical URL	POST http://<host>[:port]/ SetAccessControlConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID

	is 1.
Action name	None
Entity Data	The same as " GetAccessControlConfig ".
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]:	

3. UnLockingByPassword

UnLockingByPassword	
Description	Enter password to unlock..
Typical URL	POST http://<host>[:port]/UnLockingByPassword
Channel ID	None
Action name	None
Entity Data	The password will be included in the entity of request message. For example:
<?xml version="1.0"?> <config version="1.0" xmlns="http://www.ipc.com/ver10"> <unlocking> <password type="string" maxLen="15"><![CDATA[MTIzNDU2]]></password> </unlocking> </config>	
Successful Response	The standard successful result response that described in 1.3.5.

[Tips]:

4. GetTakeTemperatureConfig

GetTakeTemperatureConfig	
Description	To get the IP media device's temperature configuration.
Typical URL	POST or GET http://<host>[:port]/GetTakeTemperatureConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The temperature configuration will be included in the entity of the successful response. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.7" xmlns="http://www.ipc.com/ver10">
<types>
<tempUnitsType>
<enum>centigrade</enum>
<enum>Fahrenheit</enum>
</tempUnitsType>
</types>
<TakeTemperature>
<takeEanble type="boolean" default="false">false</takeEanble>
<tempUnits type="tempUnitsType">centigrade</tempUnits>
<highThreshold>
<switch type="boolean" default="false">false</switch>
<value type="uint32" min="0" max="10000">3720</value>
</highThreshold>
```

```
<lowThreshold>
<switch type="boolean" default="false">false</switch>
<value type="uint32"    min="0" max="10000">3600</value>
</lowThreshold>
<FhighThreshold>
<switch type="boolean" default="false">false</switch>
<value type="uint32" min="3200" max="21200">9900</value>
</FhighThreshold>
<FlowThreshold>
<switch type="boolean" default="false">false</switch>
<value type="uint32" min="3200" max="21200">9600</value>
</FlowThreshold>
<alarmHoldTime type="uint32">20</alarmHoldTime>
<triggerAlarmOut type="list" count="2"><itemType type="boolean"/>
<item id="0">false</item>
<item id="1">false</item>
</triggerAlarmOut>
<mail type="list" count="0">
<switch type="boolean">false</switch>
<subject type="string" maxLen="63"><![CDATA[]]></subject>
<content type="string" maxLen="255"><![CDATA[]]></content>
</mail>
<ftp type="list" count="0">
<switch type="boolean">false</switch>
</ftp>
<savePicSwitch type="boolean">false</savePicSwitch>
<sdRecSwitch type="boolean">false</sdRecSwitch>
<audioSwitch type="boolean">false</audioSwitch>
</TakeTemperature>
</config>
```

5.SetTakeTemperatureConfig

SetTakeTemperatureConfig	
Description	To set the IP media device’s Tenoerature configuration.
Typical URL	POST http://<host>[:port]/SetTakeTemperatureConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	The same as " GetTakeTemperatureConfig".
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]:	

6.GetWearmaskDetectConfig

GetWearmaskDetectConfig	
Description	To get the IP media device’s wearmask configuration.
Typical URL	POST or GET http://<host>[:port]/GetWearmaskDetectConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None
Successful Response	The wearmask configuration will be included in the entity of the successful response. For example:
<?xml version="1.0" encoding="UTF-8"?>	

```
<config version="1.7" xmlns="http://www.ipc.com/ver10">
<WearmaskDetect>
<switch type="boolean" default="false">false</switch>
<alarmHoldTime type="uint32">20</alarmHoldTime>
<triggerAlarmOut type="list" count="2"><itemType type="boolean"/>
<item id="0">false</item>
<item id="1">false</item>
</triggerAlarmOut>
<mail type="list" count="0">
<switch type="boolean">false</switch>
<subject type="string" maxLen="63"><![CDATA[]]></subject>
<content type="string" maxLen="255"><![CDATA[]]></content>
</mail>
<ftp type="list" count="0">
<switch type="boolean">false</switch>
</ftp>
<savePicSwitch type="boolean">false</savePicSwitch>
<sdRecSwitch type="boolean">false</sdRecSwitch>
<audioSwitch type="boolean">false</audioSwitch>
</WearmaskDetect>
</config>
```

7.SetWearmaskDetectConfig

SetWearmaskDetectConfig	
Description	To set the IP media device's Wearmask configuration.
Typical URL	POST http://<host>[:port]/SetWearmaskDetectConfig[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.

Action name	None
Entity Data	The same as "GetWearmaskDetectConfig".
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]:	

12 Schedule commands

12.1 Schedule

12.1.1 GetScheduleConfig

GetScheduleConfig	
Description	To get the schedule with the action_name attached.
Typical URL	POST or GET http://<host>[:port]/GetScheduleConfig[/channelId]</action_name>
Channel ID	The channel ID starts from 1.
Action name	The action names are defined as follows:

	alarmIn: schedule of alarmIn. In this scenario, the channelId is used as alarmIn ID motion: schedule of motion record: schedule of record snap: schedule of snap cdd: schedule of Crowd Density Detection ipd: schedule of Intruding People Detection tripwire: schedule of Tripwire Detection osc: schedule of Object Status Change perimeter: schedule of Perimeter Environment Assurance vfd: schedule of Video Face Detection vehicle:schedule of Video vehilce Detection aoientry: schedule of Aoi Entry Detection aoileave: schedule of Aoi Leave Detection passlinecount: schedule of Passline Count Detection thermal：schedule of Thermal imaging temperature measurement
Entity Data	None
Successful Response	The schedule information will be included in the entity of the successful response. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
    <types>
        <weekDay>
            <enum>sunday</enum>
            <enum>monday</enum>
            <enum>tuesday</enum>
            <enum>wednesday</enum>
            <enum>thursday</enum>
            <enum>friday</enum>
            <enum>saturday</enum>
        </weekDay>
    </types>
```

```
<schedule>
    <weekly type="list" maxCount="70" count="7">
        <item>
            <startTime type="string"><![CDATA[00:00]]></startTime>
            <endTime type="string"><![CDATA[23:59]]></endTime>
            <day type="weekDay">sunday</day>
        </item>
        <item>
            <startTime type="string"><![CDATA[00:00]]></startTime>
            <endTime type="string"><![CDATA[23:59]]></endTime>
            <day type="weekDay">monday</day>
        </item>
        <item>
            <startTime type="string"><![CDATA[00:00]]></startTime>
            <endTime type="string"><![CDATA[23:59]]></endTime>
            <day type="weekDay">tuesday</day>
        </item>
        <item>
            <startTime type="string"><![CDATA[05:00]]></startTime>
            <endTime type="string"><![CDATA[13:59]]></endTime>
            <day type="weekDay">wednesday</day>
        </item>
        <item>
            <startTime type="string"><![CDATA[02:00]]></startTime>
            <endTime type="string"><![CDATA[21:59]]></endTime>
            <day type="weekDay">thursday</day>
        </item>
        <item>
            <startTime type="string"><![CDATA[00:00]]></startTime>
            <endTime type="string"><![CDATA[23:59]]></endTime>
            <day type="weekDay">friday</day>
        </item>
```

```
                <item>
                    <startTime type="string"><![CDATA[00:00]]></startTime>
                    <endTime type="string"><![CDATA[23:59]]></endTime>
                    <day type="weekDay">saturday</day>
                </item>
            </weekly>
            <yearly type="list" maxCount="31" count="1">
                <item>
                    <startTime type="string"><![CDATA[00:00]]></startTime>
                    <endTime type="string"><![CDATA[23:59]]></endTime>
                    <date type="string"><![CDATA[05-12]]></date>
                </item>
            </yearly>
        </schedule>
</config>
```

[Tips]:

12.1.2 SetScheduleConfig

SetScheduleConfig	
Description	To set the schedule with the action_name attached.
Typical URL	POST http://<host>[:port]/SetScheduleConfig[/channelId]</action_name>
Channel ID	The channel ID starts from 1.
Action name	The same as "GetScheduleConfig".
Entity Data	The whole "schedule" elements in the "GetScheduleConfig" should be included in entity of this message.
Successful Response	The standard successful result response that described in 1.3.5.
[Tips]:	

12.1.3 SetScheduleConfigEx

SetScheduleConfigEx	
Description	To set the schedule in batches.
Typical URL	POST or GET http://<host>[:port]/SetScheduleConfigEx[/channelId]
Channel ID	Optional. If none channel ID included in the URL, the default channel ID is 1.
Action name	None
Entity Data	None

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="1.0" xmlns="http://www.ipc.com/ver10">
    <types>
        <weekDay>
            <enum>sunday</enum>
            <enum>monday</enum>
            <enum>tuesday</enum>
            <enum>wednesday</enum>
            <enum>thursday</enum>
            <enum>friday</enum>
            <enum>saturday</enum>
        </weekDay>
        <scheduleObject>
            <enum>cdd</enum>
            <enum>ipd</enum>
            <enum>tripwire</enum>
            <enum>osc</enum>
            <enum>perimeter</enum>
            <enum>vfd</enum>
```

```
                <enum>record</enum>
                <enum>snap</enum>
                <enum>motion</enum>
                <enum>sensor1</enum>
                <enum>sensor2</enum>
                <enum>sensor3</enum>
                <enum>sensor4</enum>
                <enum>sensor5</enum>
                <enum>sensor6</enum>
                <enum>sensor7</enum>
                <enum>vehicle</enum>
                <enum>aoientry</enum>
                <enum>aoileave</enum>
                <enum>passlinecount</enum>
                <enum>thermal</enum>
        </scheduleObject>
</types>
<schedule>
        <object type="list" count="3">
                <item type="scheduleObject">cdd</item>
                <item type="scheduleObject">cpc</item>
                <item type="scheduleObject">vfd</item>
        </object>
        <weekly type="list" maxCount="70" count="7">
                <item>
                        <startTime type="string"><![CDATA[00:00]]></startTime>
                        <endTime type="string"><![CDATA[23:59]]></endTime>
                        <day type="weekDay">sunday</day>
                </item>
                <item>
                        <startTime type="string"><![CDATA[00:00]]></startTime>
                        <endTime type="string"><![CDATA[23:59]]></endTime>
```

```
                <day type="weekDay">monday</day>
            </item>
            <item>
                <startTime type="string"><![CDATA[00:00]]></startTime>
                <endTime type="string"><![CDATA[23:59]]></endTime>
                <day type="weekDay">tuesday</day>
            </item>
            <item>
                <startTime type="string"><![CDATA[05:00]]></startTime>
                <endTime type="string"><![CDATA[13:59]]></endTime>
                <day type="weekDay">wednesday</day>
            </item>
            <item>
                <startTime type="string"><![CDATA[02:00]]></startTime>
                <endTime type="string"><![CDATA[21:59]]></endTime>
                <day type="weekDay">thursday</day>
            </item>
            <item>
                <startTime type="string"><![CDATA[00:00]]></startTime>
                <endTime type="string"><![CDATA[23:59]]></endTime>
                <day type="weekDay">friday</day>
            </item>
            <item>
                <startTime type="string"><![CDATA[00:00]]></startTime>
                <endTime type="string"><![CDATA[23:59]]></endTime>
                <day type="weekDay">saturday</day>
            </item>
        </weekly>
        <yearly type="list" maxCount="31" count="1">
            <item>
                <startTime type="string"><![CDATA[00:00]]></startTime>
                <endTime type="string"><![CDATA[23:59]]></endTime>
```

<table>
<tr><td colspan="2"><date type="string"><![CDATA[05-12]]></date>
</item>
</yearly>
</schedule>
</config></td></tr>
<tr><td>Successful Response</td><td>The standard successful result response that described in 1.3.5.</td></tr>
<tr><td colspan="2">[Tips]:
2、 The "GetDeviceDetail" includes how many sensors the device supported.
2、The "types" is defined by this document to constrain how the "schedule.object" is filled out, it can not be included in this message.</td></tr>
</table>

Annex A

A.1 Change Log

Date	Version	Note
2017-11-22	1.7	1. add "2.1.6 GetDeviceDetail" section 2. "5.3.1GetAlarmStatus" section, add status of smart alarm 3. add "5.4 AlarmTrigger" section 4. add "11 Smart commands" section 5. add "12 Schedule commands" section

Date	Version	Note
2019-10-21	1.8	1. Modify “2.1.6 GetDeviceDetail” add supportVfdMatch supportvehicle supportAoiEntry supportAoiLeave supportPassLineCount supportAudioAlarmOut supportWhiteLightAlarmOut 2. Modify “3.1.1GetStreamCaps” encodeType add h264plus h265plus h264smart h265smart 3. Modify “3.3.1GetAudioStreamConfig” add audioInSwitch audioInput audioOutput loudSpeaker 4. Modify“3.3.3GetVideoStreamConfig”encodeType add h264plus h265plus h264smart h265smart 5. GetPtzConfig 6. SetPtzConfig 7. Modify “4.3.1PtzGetPresets” presetInfo maxCount 255 -> 360 8. itemType maxLen 11 -> 10 9. Modify “5.4.1GetAlarmTriggerConfig” Action name add vehicle aoientry aoileave passlinecount 10. Add “5.5 Sound-Light Alarm” section 11. Add “5.6 Alarm PIR” section 12. Modify “11.1 Face Detect & Face Comparison” section 13. Modify “11.5 Line Crossing” section 14. Modify “11.6Intrusion” section 15. Add “11.9 License Plate Recognition” section 16. Add “11.10 Region Entrance” section 17. Add “11.11 Region Entrance” section 18. Add “11.12 Target Counting” section 19. Modify “12.1 GetScheduleConfig” Action name add vehicle aoientry aoileave passlinecount 20. Modify “12.3 SetScheduleConfigEx” scheduleObject add vehicle aoientry aoileave passlinecount.

Date	Version	Note
2020-05-06	1.9	1. Add “11.13 Thermographic Temperature Measurement” 2. Add “11.14 Infrared temperature control” 3. Modify “2.1.6 **GetDeviceDetail**” add `supportThermal` 4. Modify “3.2.1 **GetImageConfig**” add node “backLightAdjust” 5. Modify “5.5.1 **GetAudioStreamConfig**” add enum “Abnormal temperature alarm” 6. Modify “11.1.8 **SearchSnapFaceByKey**” node matchInfo add “temperature ” 7. Add “11.12.3 GetPassLineCountStatistics” 8. Modify “12.1.1 GetScheduleConfig” Action name add “thermal” 9. Modify “12.1.3 **SetScheduleConfigEx**” node scheduleObject add enum “thermal”