IP Surveillance API
(RaCM Part)
User Guide

Version 2.0

Revision 1

2014-01

http://www.hikvision.com/

Revision History

Revision History	Description	Date	By
v2.0-draft-0920	Initial version	2012-09	Meng Hong
v2.0.1	Update the document	2014-01	Meng Hong
V2.0.2	Add SMD log type	2014-10	Yj

Notices

The information in this documentation is subject to change without notice and does not represent any commitment on behalf of HIKVISION. HIKVISION disclaims any liability whatsoever for incorrect data that may appear in this documentation. The product(s) described in this documentation are furnished subject to a license and may only be used in accordance with the terms and conditions of such license.

Content

Content....3
1 Introduction....7
2 Conformance....7
3 Glossary and Relationship....7
3.1 Glossary of Terms....7
3.2 Relationship of Entities and Terminology....8
3.3 XML Reserved Characters....8
4 General Rules and Guidelines....9
4.1 DVR & NVR Design Considerations....9
4.2 Input Source Management (Remote Camera Configuration)....11
5 ContentMgmt Base Service....13
5.1 /ISAPI/ContentMgmt/sourceSupport....13
5.1.1 Source Support XML Schema Definition....15
5.1.2 Access and Operation of Source Support....15
6 /ISAPI/ContentMgmt/Capabilities....19
7 /ISAPI/ContentMgmt/record....21
7.1 /ISAPI/ContentMgmt/record/storageMounts....21
7.2 /ISAPI/ContentMgmt/record/profile....23
7.2.1 /ISAPI/ContentMgmt/record/profile Schema Definition....23
7.3 /ISAPI/ContentMgmt/record/tracks....24
7.3.1 Custom Configuration Data (Extensions)....25
7.3.2 ISAPI-REST List-Entry <id> Creation method....25
7.3.3 Streaming URL implied in <Track> configuration....25
7.3.4 Recording Source Description....25
7.3.5 Recording Schedule overview....25
7.3.6 Track Description NVP....26
7.3.7 /ISAPI/ContentMgmt/record/tracks....28
7.3.8 /ISAPI/ContentMgmt/record/tracks/<id>....28
7.3.9 Example Track Creation Message Exchange....29
7.3.10 Track List Schema....31
7.3.11 /ISAPI/ContentMgmt/record/tracks/<id>/dailyDistribution....32
7.4 /ISAPI/ContentMgmt/record/control....32
7.4.1 /ISAPI/ContentMgmt/record/control/manual/start/tracks/<ID>....32
7.4.2 /ISAPI/ContentMgmt/record/control/manual/stop/tracks/<ID>....34
7.4.3 /ISAPI/ContentMgmt/record/control/locks....34
8 /ISAPI/ContentMgmt/search....36
8.1 /ISAPI/ContentMgmt/search/profile....36
8.1.1 /ISAPI/ContentMgmt/search/profile Schema Definition....37

8.2 /ISAPI/ContentMgmt/search 37
8.2.1 Search Query Parameter Schema Definition 39
8.2.2 Search Query Results Schema 41
9 /ISAPI/ContentMgmt/logSearch 44
9.1.1 Search Query Parameter Schema Definition 45
9.1.2 Search Query Results Schema 45
10 Metadata Identity String (MIDS; “metaID”) 46
10.1 MIDS Field Definitions 46
10.1.1 Domain: event.hikvision.com 47
10.1.2 Domain: log.hikvision.com 47
10.1.3 Domain: recordType.meta.hikvision.com 51
11 Streaming and Playback 52
11.1 Streaming URIs 52
11.1.1 Live Streams 52
11.1.2 Archive Streams 52
11.1.3 Time-related Streaming 53
11.2 Playback 54
11.2.1 Use of RTSP 54
11.2.2 Initiating Playback 54
12 /ISAPI/ContentMgmt/InputProxy 56
12.1 /ISAPI/ContentMgmt/InputProxy/sourceCapability 56
12.2 /ISAPI/ContentMgmt/InputProxy/search 57
12.3 /ISAPI/ContentMgmt/InputProxy/ipcConfig 58
12.4 ISAPI/ContentMgmt/InputProxy/syncIPCPasswd 59
12.5 /ISAPI/ContentMgmt/InputProxy/customProtocols 59
12.5.1 /ISAPI/ContentMgmt/InputProxy/customProtocols/<ID> 59
12.6 /ISAPI/ContentMgmt/InputProxy/channels 60
12.7 /ISAPI/ContentMgmt/InputProxy/channels/status 61
12.8 /ISAPI/ContentMgmt/InputProxy/channels/<ID> 61
12.9 /ISAPI/ContentMgmt/InputProxy/channels/<ID>/password 62
12.10 /ISAPI/ContentMgmt/InputProxy/channels/<ID>/netParam 62
12.11 /ISAPI/ContentMgmt/InputProxy/channels/<ID> /status 63
12.12 /ISAPI/ContentMgmt/InputProxy/channels/<ID>/video 64
12.13 /ISAPI/ContentMgmt/InputProxy/channels/<ID>/video/overlays 64
12.13.1 ./InputProxy/channels/<ID>/video/overlays/text 64
12.13.2 ./InputProxy/channels/<ID>/video/overlays/text/<ID> 65
12.13.3 ./InputProxy/channels/<ID>/video/overlays/image 65
12.13.4 ./InputProxy/channels/<ID>/video/overlays/image/<ID> 66
12.14 /ISAPI/ContentMgmt/InputProxy/channels/<ID>/video/privacyMask 66
12.14.1 ./InputProxy/channels/<ID>/video/privacyMask/regions 66
12.14.2 ./InputProxy/channels/<ID>/video/privacyMask/regions/<ID> 67
12.15 /ISAPI/ContentMgmt/InputProxy channels/<ID>/video/tamperDetection 67

12.15.1 ./InputProxy/channels/<ID>/video/tamperDetection/regions.....68
12.15.2 ./InputProxy/channels/<ID>/video/tamperDetection/regions/<ID>.....68
12.16 /ISAPI/ContentMgmt/InputProxy /channels/<ID>/video/motionDetection.....68
12.16.1 ./InputProxy/channels/<ID>/video/motionDetection/layout 69
12.16.2 /ISAPI/ContentMgmt/InputProxy/channels/ID/video/smartDetection....69
13 / ISAPI/ContentMgmt/IOProxy.....70
13.1 /ISAPI/ContentMgmt/IOProxy/status 70
13.2 / ISAPI/ContentMgmt/IOProxy/inputs 70
13.3 / ISAPI/ContentMgmt/IOProxy/inputs/ID 71
13.4 / ISAPI/ContentMgmt/IOProxy/inputs/ID/status.....72
13.5 / ISAPI/ContentMgmt /IOProxy/outputs 72
13.6 / ISAPI/ContentMgmt/IOProxy/outputs/ID 72
13.7 / ISAPI/ContentMgmt /IOProxy/outputs/ID/trigger 73
13.8 /ISAPI/ContentMgmt/IOProxy/outputs/ID/status.....73
14 /ISAPI/ContentMgmt/ZeroVideo.....73
14.1 /ISAPI/ContentMgmt/ZeroVideo/channels.....74
14.2 /ISAPI/ContentMgmt/ZeroVideo/channels/<ID>.....74
14.3 /ISAPI/ContentMgmt/ZeroVideo/channels/<ID>/enlarge 75
14.4 /ISAPI/ContentMgmt/ZeroVideo/channels/<ID>/switchScreen.....75
14.5 /ISAPI/ContentMgmt/ZeroVideo/channels/<ID>/previewCfg.....76
15 /ISAPI/ContentMgmt/ImageProxy.....76
16 /ISAPI/ContentMgmt/SnapshotProxy.....77
17 /ISAPI/ContentMgmt/PTZCtrlProxy 77
18 /ISAPI/ContentMgmt/StreamingProxy.....78
19 /ISAPI/ContentMgmt/ZeroStreaming.....78
19.1 /ISAPI/ContentMgmt/ZeroStreaming/status.....78
19.2 /ISAPI/ContentMgmt/ZerStreaming/channels.....78
19.3 /ISAPI/ContentMgmt/ZeroStreaming/channels/<ID> 79
19.4 /ISAPI/ContentMgmt/ZeroStreaming/channels/<ID>/status 80
20 /ISAPI/ContentMgmt/Storage 81
20.1 /ISAPI/ContentMgmt/Storage/hdd.....81
20.2 /ISAPI/ContentMgmt/Storage/hdd/<ID>.....81
20.3 /ISAPI/ContentMgmt/Storage/hdd/<ID>/format 82
20.4 /ISAPI/ContentMgmt/Storage/hdd/<ID>/formatStatus.....82
20.5 /ISAPI/ContentMgmt/Storage/hdd/<ID>/SMARTTest/start 83
20.6 /ISAPI/ContentMgmt/Storage/hdd/<ID>/SMARTTest/status 83
20.7 /ISAPI/ContentMgmt/Storage/hdd/SMARTTest/config.....84
20.8 /ISAPI/ContentMgmt/Storage/hdd/<ID>/BadSectorsTest/start.....84
20.9 /ISAPI/ContentMgmt/Storage/hdd/<ID>/BadSectorsTest/status.....85
20.10 /ISAPI/ContentMgmt/Storage/hdd/<ID>/BadSectorsTest/pause.....86
20.11 /ISAPI/ContentMgmt/Storage/hdd/<ID>/BadSectorsTest/resume 86
20.12 /ISAPI/ContentMgmt/Storage/hdd/<ID>/BadSectorsTest/stop 86

20.13 /ISAPI/ContentMgmt/Storage/nas 87
20.14 /ISAPI/ContentMgmt/Storage/nas/<ID> 87
20.15 /ISAPI/ContentMgmt/Storage/nas/<ID>/format 88
20.16 /ISAPI/ContentMgmt/Storage/nas/<ID>/formatStatus 88
20.17 /ISAPI/ContentMgmt/Storage/nas/search 88
20.18 /ISAPI/ContentMgmt/Storage/quota 89
20.19 /ISAPI/ContentMgmt/Storage/quota/<ID> 89
20.20 /ISAPI/ContentMgmt/Storage/extract 90
20.21 /ISAPI/ContentMgmt/Storage/diskGroup 90
20.22 /ISAPI/ContentMgmt/Storage/diskGroup/<ID> 91
20.23 /ISAPI/ContentMgmt/Storage/extension 91
20.24 /ISAPI/ContentMgmt/Storage/cloud/URL?type=OneDrive 92
20.25 /ISAPI/ContentMgmt/Storage/cloud/URL/capabilities 92
20.26 /ISAPI/ContentMgmt/Storage/cloud 92
20.27 /ISAPI/ContentMgmt/Storage/cloud/capabilities 93
20.28 /ISAPI/ContentMgmt/Storage/cloud/channels/ID/uploadStrategy 93
20.29 /ISAPI/ContentMgmt/Storage/cloud/channels/ID/uploadStrategy/capabilities 94
20.30 /ISAPI/ContentMgmt/FlashStorage/remove/channels/<ID> 95
21 / ISAPI/ContentMgmt/download 95
22 /ISAPI/ContentMgmt/channels/ID/cloudStorage 96
22.1 /ISAPI/ContentMgmt/channels/ID/cloudStorage/ID/capabilities 96
22.2 /ISAPI/ContentMgmt/channels/ID/cloudStorage/ID 97
22.3 /ISAPI/ContentMgmt/channels/ID/cloudStorage/test 97
23 /ISAPI/ContentMgmt/channels/ID/liteStorage 98
23.1 /ISAPI/ContentMgmt/channels/ID/liteStorage/capabilities 99
24 /ISAPI/ContentMgmt /workmode 99
24.1 /ISAPI/ContentMgmt/workmode/capabilities 100
25 /ISAPI/ContentMgmt/defaultPwdRemain 100
26 Appendix A: Codec Type Dictionary 102

1 Introduction

This document specifies an interface that enables physical security and video management systems to communicate with a Recording and Content Management (RaCM) device in a standardized way.

2 Conformance

The RaCM Device will host ISAPI compliant services and adhere to the ISAPI Service Model.

3 Glossary and Relationship

3.1 Glossary of Terms

- ***Channel*** = Handle/tag for an input source (port or stream); see IPMD spec for usage. For DVR, "channel" is the identifier, or handle, used to identify a local input stream (which may also be accessible to remote entities via a local /ISAPI/Streaming/channel/<id> Resource).
  The "stream" may contain Video, Audio, and/or Metadata. For NVR, "channel" is used to identify remote media stream which, if from an IPMD, should come from the remote device's /ISAPI/Streaming/channels/<id> Resource. The term 'channel' is used as a guiding principle and logical construct.
- ***Track* =** Virtual storage container for a set of contents (e.g. VIDEO, AUDIO, METADATA, etc.). *[Channels and tracks are kept separate to allow the ability to mix and match channels totracks].* **RESTRICTION:** At this time, a Track's configuration contains only 1 <SourceDescriptor> which is intended to describe the source for the recorded media-stream. This media-stream will be construed as the equivalent of the input "channel" for the track, which means that each Track can only record one input stream. The media-stream however can be Multi-Media, if the source delivers such a stream (to be found at the <SrcUrl>), as is the case with IPMD.
- ***Source* =** Any input media device is a 'source', whether the input is a hardware oriented 'port' (e.g. NTSC/PAL, audio input jack, etc.), or an IP-based media device. All 'sources' are identified by ISO/IEC 9834-8:2005 128-bit UUIDs/GUIDs to guarantee uniqueness within any given system. Using the RaCM nomenclature specified herein, sources send their data to RaCM devices on via 'streams' which are mapped to 'channels'. Channels are handles used to identify the specific input streams for a RaCM device. Channels that are recorded are then mapped onto 'tracks' for recording. See the following diagram for more details. **Within the RaCM Device, such 'sources' are not explicitly configured. These logical 'sources' are visible (Read-Only fashion) via the '/ISAPI/ContentMgmt/status/sources' Resource.** It is also possible to search based on these 'sources' via "/ISAPI/ContentMgmt/search" Resource.
- ***Metadata* =** All streaming data except video and audio. This includes events, alarms, etc.
- ***URI* =** A virtual path that specifically identifies a REST resource; this path must follow the

service/resource hierarchy (see ISAPI Service Model specification).

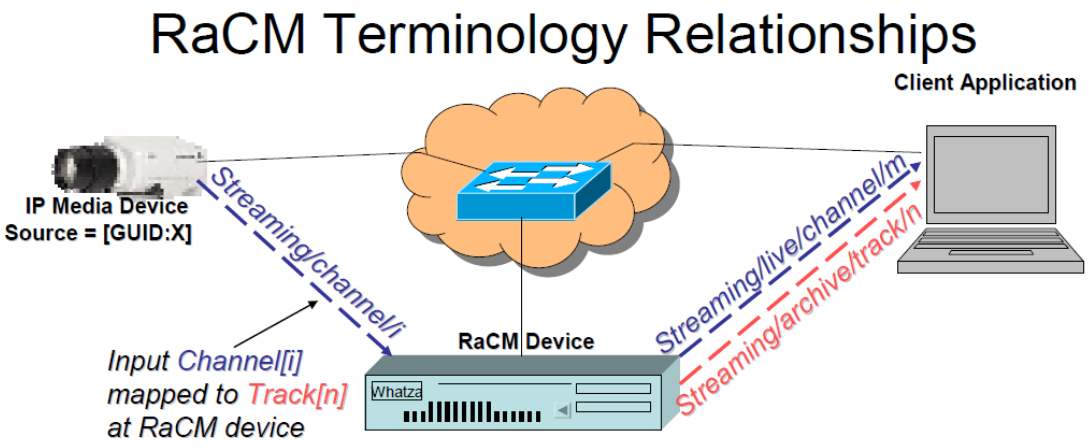
- ***Segment* =** A general term addressing a single, contiguous portion of a media track, or (in some cases) a stream. Basically, a 'media clip' that is less than the whole of a respective media track or stream, yet is individually accessible via one of the mechanisms specified in this document.

3.2 Relationship of Entities and Terminology

Below are the basic relationships between the terms defined above.

- **Source** = The device is the original point for input to a RaCM device. This is usually a camera, an encoder (video and/or audio) or a metadata generator. All devices have their own ISO/IEC 9834-8/ITU X.667, 128-bit UUID/GUID as their based identity.
- **Channel** = Incoming media stream (input identifier). This is a handle/identifier for a specific input stream.
- **Stream** = In general, a network-based output media connection (output identifier).
- **Track** = A recorded channel. Since input characteristics may change over time, 'tracks' are the virtual containers for recorded content associated with a 'channel'. Channels may be extant or extinct with respect to a track. That is why tracks are separate from channels. Channels are mapped to tracks for recording.

The following diagram depicts the relationships between the streams, channels and tracks.



IP Media Device advertises its Streams by channels. Based on configuration the RaCM device selects the IPMD's stream channel ID (i) which is indigenous to the IP Media Device. This input stream becomes Channel 'm' on the RaCM device (since it has multiple inputs) for this particular live stream. Via configuration it is mapped to track 'n' for recording. The track ID 'n' may, or may not, be equal to 'm'. When a client requests a live stream from the RaCM device, it uses the advertised Channel ID 'm'. When it requests a stream from the archived track, it uses the Advertised track ID 'n'..

3.3 XML Reserved Characters

Within an XML document, some characters are reserved for language use. If these characters appear in data values, they should be replaced with their Entity Reference equivalents (akin to ANSI Escapes) to avoid parsing errors.

Character	Description	Entity Reference	Comments

<	Less than	&lt;	May never appear in
&	Ampersand	&amp;	May never appear in
>	Greater than	&gt;	Replace as best
“	Double quote	&quot;	Replace as best
‘	Single quote	&apos;	Replace as best
%	Percent	&#37;	Replace as best
Note that &#0; (null) is not permitted.			

For example, the URL: **“rtsp://144.70.13.92:554/ISAPI/Streaming/tracks/27?starttime=20130731T092241&endtime=20130731T093000”** would appear as follows in XML:

```
<playbackURI>rtsp://144.70.13.92:554/ISAPI/Streaming/tracks/27?starttime=20130731T092241&amp;
endtime=20130731T093000</playbackURI>
```

4 General Rules and Guidelines

The following guidelines and requirements apply to those parties implementing this specification:

- All RaCM devices shall comply with the guidelines, formats, syntax, and base protocol definitions contained in the ISAPI Service Model specification.
- All RaCM devices shall implement the following ISAPI REST Resource hierarchies, as outlined on the ISAPI IP Media Device (IPMD) Specification:
    - For DVRs , the “/ISAPI/System/Video/…” REST resources and services as is pertinent for the hardware capabilities for a given RaCM device.
    - For DVRs that support PTZ commands, the “/ISAPI/PTZ/…” REST resources and services are to be implemented, as outlined in ***IPMD***, where the PTZ capabilities in the IPMD specification are functionally compatible for the DVR device.
    - For DVRs and NVRs, the “/ISAPI/System/…” REST resource hierarchies must be implemented as defined in ***IPMD Sections***, for the configurable system based devices and I/O capabilities (network, serial, I/O, …) supported by a given RaCM device.
    - For DVRs, the “/ISAPI/Streaming/Channels…” REST resource hierarchy for each hardware channel it supports. The “picture” and “requestKeyFrame” resources are not required.
    - NVRs and hybrid DVRs must comply with the management guidelines outlined in the following ***Section*** regarding the management of external IP media devices.
    - All RaCM devices shall support the “/ISAPI/System/time/…” REST resources for setting/reporting local time on their devices unless they use a network domain controller to access network time.

Though the resource hierarchy described herein does cite specific resources in the IPMD and Service Model specifications, it does not preclude RaCM device implementers from incorporating other ISAPI resources that are relevant. The objective of this specification is to provide the design details and intent for the base protocol implementation.

4.1 DVR & NVR Design Considerations

Digital Video Recorder (DVR) devices have design issues unique to their product classes. Unlike NVRs, DVRs have onboard video and (in many cases) audio codec hardware as their input sources. Since this hardware is intrinsic to the device, i.e. it is not dynamically assignable to the unit like NVR input

sources, DVRs must comply with the following design operational items, in addition to the general requirements listed in the above section of this document:

- All video codec hardware must be listed as pre-configured (i.e. with default settings) input 'channels' in the following ISAPI resource hierarchies:
  - For DVR video input hardware that provides image setting parameters, such as brightness, contrast, sharpness, etc., these settings must be accessible in the "/ISAPI/System/Video/inputs/channels…" resource hierarchy. This means that the 'VideoInputChannelList' XML schema document returned by a GET to the "/ISAPI/System/Video/inputs/channels" resource must list all of the existing input channels as 'VideoInputChannel' elements. The IDs for each element must be set for each channel by the DVR and cannot be allowed to be changed by external entities.
  - DVR video codec hardware must be listed as pre-configured (i.e. using default values) input 'channels' in the "/ISAPI/Streaming/channels…" resource. The returned 'StreamingChannelList" XML schema document must be pre-populated with all of the codec channels that are available for configuration and use. The channel IDs in the schema must be preset by the DVR.
- All audio codec hardware present on a DVR must be listed as existing, pre-configured (i.e. using default values) hardware input 'channels' in the "/ISAPI/System/Audio/channels…" resource hierarchy. The channel IDs must be pre-populated in the 'AudioChannelList' XML schema document returned by GETs to the "/ISAPI/System/Audio/channels…" resource. The channels IDs assigned by a DVR are immutable by external entities.
- Multiple "/ISAPI/Streaming/channels/<id>" may have the same <videoInputChannelID>. Multiple "/ISAPI/ContentMgmt/StreamingProxy/channels/<id>" may have the same <dynVideoInputChannelID>. In consideration of the relationship between stream and track, video input id, dynVideo input id, stream id, dynStream id use the following encoding:
  (1) To ensure the compatibilities of different devices, please use unified id No.in /ISAPI/System/Video/inputs/channels/<ID> and /ISAPI/ContentMgmt/InputProxys/channels/<ID>/video. For example, if a device supports 16 channels, the ID No. should be 1-16 in Video/inputs/channels/<ID>, in this case, the ID No.should start with 17 in InputProxys/channels/<ID>/video.
  (2) <ID> of dynamic stream adopts binary coded decimal, the former 6 digits stand for channel number, the last 2 digits stand for stream number, for example, if channel 1 of a Hybrid DVR supports 9-ch streams, the ninth stream should be 109; for the 99-ch stream of the channle100, the number should be10099.
- Tracks will record from their respective <SourceDescriptor>'s.
  - Thus, each active Track implicitly creates an input 'channel'.
  - In addition, /ISAPI/ContentMgmt/InputProxy/channels can also be used to create an input 'channel'. It is recommended to use the service to create an input 'channel'. This service can also manage/configure some corresponding paramters related with input 'channel', such as image parameter, compression parameter, OSD, Motion Detection and so on.
  - The <id> for these 'channels' are determined by the RaCM Device automatically.

The above design items are described here to aid in providing commonality in the implementation of this specification, and provide clarity for those developing to the interfaces defined herein. Hybrid DVR/NVR products must still follow the above guidelines for the onboard codec hardware that is present on their respective devices. Please note that these design guidelines pertain to the definition and configuration of the video/audio input hardware. The information related to track configuration, etc., is addressed later in this document.

4.2 Input Source Management (Remote Camera Configuration)

NVRs and hybrid DVRs support external IP network devices as their input sources. These devices, usually IP cameras, may, or may not, be ISAPI protocol compliant. This specification does not require the external input sources to be ISAPI compliant in order to be supported by a RaCM device. However, within the industry there are different methods for managing the support for, configuration and status of, input IP media devices. ISAPI RaCM devices must fall into one of two possible categories regarding the management of external IP media sources. The management categories are listed below:

Management Mode	Mode Description
Simple	This mode describes as RaCM device that manages track configuration, but does not manage the settings related to video/audio codecs and streaming at the external source camera/encoder. This means that management entities such as VMS applications are required to manage the settings on the RaCM devices, and the external source devices, separately. I.e. the RaCM device does not change the settings on a source device when a codec or streaming related parameter is changed on a track. A Simple RaCM device will attempt to open a new session to the respective source with the new codec/streaming settings, but it will NOT modify the codec/streaming configuration settings on an external camera/encoder. In some cases, this is sufficient since codec settings on some cameras can be modified on-the-fly via the session setup parameters; in other cases the VMS management application will have to modify the codec/streaming settings at the source prior to making track setting changes. In all cases in Simple management mode, the management application (VMS, etc.) is responsible for the settings at all the devices and for the synchronization of those settings between the sources and consumers. The recommendation in this mode is that management applications SHOULD always change the settings at the source device (camera, encoder) and then modify the appropriate track settings on the dependent RaCM device(s). Please note that: this can produce race conditions between the source and recoding device in some cases.
Proxy	RaCM devices that have the ability to remotely manage other source device's codec/streaming parameters are called 'proxy managers'. Modifications to codec/streaming settings on a proxy managing RaCM device will also be performed at the external source device by the RaCM device for those devices the RaCM unit knows how to configure. Basically, in proxy management mode the RaCM device receives the settings that affect streams and/or tracks, and performs, based on the sources behavior, all of the necessary configuration adjustments on behalf of the management application.

In order to aid interoperability, and to support the use of Proxy management mode, three new RaCM REST resources have been created: “/ISAPI/ContentMgmt/sourceSupport”, “/ISAPI/ContentMgmt/InputProxy”, “/ISAPI/ContentMgmt/StreamingProxy”.

“/ISAPI/ContentMgmt/sourceSupport” identifies which mode of management and level of interoperability a RaCM device provides for each advertised camera/encode device it claims support for (see */ISAPI/ContentMgmt/sourceSupport*).

Fundamentally, “/ISAPI/ContentMgmt/sourceSupport” resource provides a list of IP cameras and encoders, by manufacture and model, that are supported (i.e. compatible) along with the management mode for each.

5 ContentMgmt Base Service

The ‘ContentMgmt’ base service is the ‘root’ for all Recording and Content Management (RaCM) device function related to the recording and management of multimedia data. This service is the base node in the REST resource hierarchy for all active functions provided by a RaCM device.

As mentioned in the previous section, the RaCM device must also comply with the ISAPI Service Model specification and also implement the required IPMD Services.

5.1 /ISAPI/ContentMgmt/sourceSupport

For RaCM devices that support external IP cameras and encoders as input devices, the ‘sourceSupport’ REST resource MUST be supported in order to advertise the types and models of devices that a unit is compatible with. ***Section Input Source Management*** of this specification describes the management modes a RaCM device may support for managing external IP input sources.

GETs from the “/ISAPI/ContentMgmt/sourceSupport” resource return a schema that describes the manufacturers, makes and models of IP media devices that a RaCM device is compatible with. The following table provides the base details for the ‘sessionSupport’ resource.

/ISAPI/ContentMgmt/sourceSupport	**General Resource v1.0**
GET	
Description	Description of the IP media devices, in mfgr, make and model, that a RaCM devices supports as input sources
Query	Optional: “send=head” “send=top”, “send=middle” “send=bottom” or… “mfgr=<mfgrName>”
Inbound Data	None
Success Return	**<CMSourceSupport>**
Notes:	The schema and element definitions for this resource follow.
Example XML	<?xml version="1.0" encoding="UTF-8"?> <CMSourceSupport version=”1.0” xmlns=”http://www.isapi.org/ver20/XMLSchema”> < mediaDeviceListSize>8</ mediaDeviceListSize> < supportedMediaDeviceSourceList> <MediaDeviceSource> <MediaDeviceMfgr>HIKVISION</MediaDeviceMfgr> <MediaDeviceMgmtMode>Proxy</MediaDeviceMgmtMode> <MediaDeviceModel>DS-2CD876MF</MediaDeviceModel> <MediaDeviceModel>DS-2CD877MF</MediaDeviceModel> <MediaDeviceModel>DS-2CD855MF</MediaDeviceModel> </MediaDeviceSource> <MediaDeviceSource> <MediaDeviceMfgr> GB28181</MediaDeviceMfgr> <MediaDeviceMake>MaxView Series</MediaDeviceMake> <MediaDeviceMgmtMode>Simple</MediaDeviceMgmtMode> <MediaDeviceModel>GB28181</MediaDeviceModel> <MediaDeviceModel> GB28181</MediaDeviceModel> <MediaDeviceModel> GB28181</MediaDeviceModel> </MediaDeviceSource> <MediaDeviceSource> <MediaDeviceMfgr>ClearView</MediaDeviceMfgr> <MediaDeviceMake>MaxView Series</MediaDeviceMake> <MediaDeviceMgmtMode>Simple</MediaDeviceMgmtMode> <MediaDeviceModel>MCV-100F</MediaDeviceModel> <MediaDeviceModel>MCV-250PTZ</MediaDeviceModel> <MediaDeviceModel>MCV-500</MediaDeviceModel> </MediaDeviceSource> <MediaDeviceSource> <MediaDeviceMfgr>ISAPI</MediaDeviceMfgr> <MediaDeviceMake>Any ISAPI compliant IP Media device</MediaDeviceMake> <MediaDeviceMgmtModel>Standard</MediaDeviceMgmtModel> <MediaDeviceMgmtModel>PTZ</MediaDeviceMgmtModel> </MediaDeviceSource> </ supportedMediaDeviceSourceList> </CMSourceSupport> The above example represents a RaCM device that supports 9 different camera models from 3 different manufacturers. Please note that the ‘make’ element is not used on every manufacturer. Also, the ‘ISAPI’ IP Media Device support is listed as a generic manufacturer with Standard (i.e. non-PTZ) and a PTZ generic model support. This example is only referential.

Accesses to the 'session Support' resource returns a 'CMSourceSupport' schema document instance. This schema document is primarily a list of the types and models of IP media devices that a RaCM device supports as compatible input sources. Each element in a list element is comprised of the following parameters:

Element name	Requirement	Description
MediaDeviceMfgr	Mandatory	XML string field that lists the manufacturer of the supported IP Media Device. This field is treated as ***case insensitive*** to aid in searching and matching (e.g. Luminati = luminati, effectively).
MediaDeviceMake	Optional	Optional XML string field that lists the make of an IP Media device. Some manufacturers come out with 'lines' (i.e. product lines) of cameras that share protocol attributes. This field is to aid in identifying products that belong to a certain product line (where/when that information is pertinent and relevant).
MediaDeviceModel	Mandatory	XML string that identifies the model of a certain IP Media device.
MediaDeviceMgmtMode	Mandatory	XML type restricted to "Simple" or "Proxy".

Please note that all of the values contained in the above fields are treated in a case insensitive manner to aid in better interoperability between management applications, clients and the RaCM devices. The purpose of the "CMSourceSupport" schema is to provide consumers with a well ordered list of the media devices that it can record and stream. It is also acceptable for 'generic' media device support to be listed where the RaCM device has an industry (or mfgr) standards-based driver that supports compliant cameras generically (i.e. the "MediaDeviceMfgr" does not have to contain a literal manufacturer; it could contain a value like "PSIA" or "ONVIF", etc.). The full schema definition follows.

5.1.1Source Support XML Schema Definition

XSD File: cmSourceSupport.xsd

This schema definition is basically a list of elements that define the types of IP media devices that are supported, or are compatible with, a RaCM device. The first element is a count of the number of IP media device ***models*** (i.e. "MediaDeviceModel" entries) that are listed in a "CMSourceSupport" document instance. This is important since the size of the schema can be huge, in some cases, and due to the fact that consumers can ask for only the 'count' of the supported media devices (see following section). Please note that each list element, which has one manufacturer value, can support multiple model numbers/strings per that manufacturer value. The manufacturer name can also identify standards organizations, not specific manufacturers.

5.1.2Access and Operation of Source Support

The REST URI structure of the "/ISAPI/ContentMgmt/sourceSupport" resource allows consumers to ask for portions, or all, of the schema document information (see table in above ***Section 8.4***). A consumer accessing source support by a simple GET to the "/ISAPI/ContentMgmt/sourceSupport" resource will get the entire data list of all supported IP media devices in the "CMSourceSupport" schema instance. However, due to

the fact that the size of this XML list could be prohibitively large, RaCM devices MUST support the following options for getting portions of the source support information without requiring the entire data set. There are two query parameters that can be added to the REST URI for getting portions of the overall information set. They are:

- "**send=**…" The send designator identifies that the consumer only wants a portion of the overall source support information. A value is supplied with the "send=" string as an NVP that designates the portion of the information to be sent. This is discussed in more detail below.
- "**mfgr=**…" The data designator specifies a 'filter' for the information to be returned. Basically, a consumer specifies what type of information it is looking for. This is described in detail below.

Please note that the 'send' and 'mfgr' query parameters are mutually exclusive; they cannot be used in the same HTTP/REST GET message since they each counter-actively affect what, and how much, of the source support information is returned by a RaCM device. Consumers of source information are to use these query string parameters to govern the amount of source information that is transferred since this information base can be very large. Each of the query string parameters is described below, in detail

5.1.2.1 "send=…" Query String Parameter

The "send=…" query string parameter (QSP) enables consumers to specify, or control, (in a coarse manner) the amount of source support information that they desire to receive. If the "send=…" QSP is not present when a GET to the "/ISAPI/ContentMgmt/sourceSupport" resource is issued, ALL the source support information will be returned in the CMSourceSupport schema instance. In order to prevent data overrun, the "send=…" QSP is provided such that the consumer can ask for the source support list information in 'chunks'. The amount, and type, of information is specified by the value tag supplied with the "send=…" QSP. The value tags that can be supplied are listed below with descriptions.

Tag Value	URI Example	Description
"head"	GET /ISAPI/ContentMgmt/sourceSupport?send=head	This tag value indicates that the consumer only wants the returned schema instance to contain the count of the number of source support *models* in its list; No element data should be returned. This allows consumers to gauge the approximate size of the source support information base, in total.
"top"	GET /ISAPI/ContentMgmt/sourceSupport?send=top	This tag value indicates that the consumer wants the top 1/3rd (roughly) of the source support list. RaCM devices receiving this QSP should provide, approximately, the first one-third of the "CMSourceSupport" list.

“middle”	GET /ISAPI/ContentMgmt/sourceSupport?send=middle	This tag value indicates that the consumer wants the middle 1/3rd (roughly) of the source support list. RaCM devices receiving this QSP should provide, approximately, the middle one-third of the “CMSourceSupport” list data.
“bottom”	GET /ISAPI/ContentMgmt/sourceSupport?send=middle	Similar to the two above tags, this tag value indicates that the consumer wants the last 1/3rd (roughly) of the source support list. RaCM devices receiving this QSP should provide, approximately, the final one-third of the “CMSourceSupport” list data. Please note that it is up to the RaCM device on how to apportion the source support info into ‘chunks’.

When a RaCM device receives a “send=head” request, the returned schema should only contain the “MediaDeviceModelCount” element and its value. For all of the other QSP strings, the RaCM device is to provide portions of the schema document that align on list element boundaries (i.e. manufacturer/make boundaries). This apportioning scheme prohibits the use of the “mfgr=…” QSP, which is described in the following section. The following examples are provided as additional descriptive information. These examples are based on the example XML schema instance listed in the table at the top of ***Section /ISAPI/ContentMgmt/sourceSupport***.

Send ‘head’ Example:

(request)
GET /ISAPI/ContentMgmt/sourceSupport?send=head HTTP/1.1
… (response) HTTP/1.1 200 OK
Content-type:application/xml
Content-length :165

```
<?xml version="1.0" encoding="UTF-8"?>
<CMSourceSupport version=”1.0” xmlns=”http://www.isapi.org/ver20/XMLSchema”>
        <MediaDeviceModelCount>8</MediaDeviceModelCount>
</CMSourceSupport>
```

Send ‘top’ Example:

(request)
GET /ISAPI/ContentMgmt/sourceSupport?send=top HTTP/1.1
… (response) HTTP/1.1 200 OK

Content-type:application/xml
Content-length :<nnn>

```
<?xml version="1.0" encoding="UTF-8"?>
<CMSourceSupport version=”1.0” xmlns=”http://www.isapi.org/ver20/XMLSchema”>
        <MediaDeviceModelCount>3</MediaDeviceModelCount>
        <SupportedMediaDeviceSourceList>
          <MediaDeviceSource>
              <MediaDeviceMfgr>HIKVISION</MediaDeviceMfgr>
              <MediaDeviceMgmtMode>Proxy</MediaDeviceMgmtMode>
              <MediaDeviceModel>DS-2CD876MF</MediaDeviceModel>
```



```
                <MediaDeviceModel>DS-2CD877MF</MediaDeviceModel>
                <MediaDeviceModel>DS-2CD855MF</MediaDeviceModel>
            </MediaDeviceSource>
        </SupportedMediaDeviceList>
</CMSourceSupport>
```

Send ‘middle’ Example:

```
(request)
GET /ISAPI/ContentMgmt/sourceSupport?send=middle HTTP/1.1

… response)HTTP/1.1 200 OK
Content-type:application/xml
Content-length :<nnn>
<?xml version="1.0" encoding="UTF-8"?>
<CMSourceSupport version=”1.0” xmlns=”http://www.isapi.org/ver20/XMLSchema”>
        <MediaDeviceModelCount>3</MediaDeviceModelCount>
        <SupportedMediaDeviceSourceList>
            <MediaDeviceSource>
                <MediaDeviceMfgr>ClearView</MediaDeviceMfgr>
                <MediaDeviceMake>MaxView Series</MediaDeviceMake>
                <MediaDeviceMgmtMode>Simple</MediaDeviceMgmtMode>
                <MediaDeviceModel>MCV-100F</MediaDeviceModel>
                <MediaDeviceModel>MCV-250PTZ</MediaDeviceModel>
                <MediaDeviceModel>MCV-500</MediaDeviceModel>
            </MediaDeviceSource>
        </SupportedMediaDeviceList>
</CMSourceSupport>
```

Send ‘bottom’ Example:

```
(request)
GET /ISAPI/ContentMgmt/sourceSupport?send=bottom HTTP/1.1

… (response) HTTP/1.1 200 OK
Content-type:application/xml
Content-length :<nnn>
<?xml version="1.0" encoding="UTF-8"?>
<CMSourceSupport version=”1.0” xmlns=”http://www.isapi.org/ver20/XMLSchema”>
        <MediaDeviceModelCount>2</MediaDeviceModelCount>
            <MediaDeviceSource>
                <MediaDeviceMfgr>ISAPI</MediaDeviceMfgr>
                <MediaDeviceMake>Any ISAPI compliant IP Media device</MediaDeviceMake>
                <MediaDeviceMgmtModel>Standard</MediaDeviceMgmtModel>
                <MediaDeviceMgmtModel>PTZ</MediaDeviceMgmtModel>
            </MediaDeviceSource>
</CMSourceSupport>
```

The above examples are for reference. The data set they contain is hypothetical and relatively small for the sake of simplicity. Please note that in these examples the RaCM device made its own arbitrary decision as to where to subdivide the CMSourceSupport information of ‘top’ versus ‘middle’ versus ‘bottom’. Also note that the “MediaDeviceModelCount” size varies since the number of IP media device models varied per response. The next section describes the ability for consumers to ask for specific forms of information.

5.1.2.2“mfgr=” Query String Parameter

The “mfgr=…” QSP enables consumers to ask for supported IP media device information related to a specific manufacturer or standards organization. Consumers using this capability can ask for supported model information related to a specific manufacturer or standard. An example request would look like:

GET /ISAPI/ContentMgmt/souceSupport?mfgr=GoodVision

This type of request instructs the RaCM to only return a “CMSourceSupport” schema instance that lists the supported models for the manufacturer “GoodVision”. The following is an example of what is expected to be returned:

```
…(request)
GET /ISAPI/ContentMgmt/sourceSupport?mfgr=HIKVISION HTTP/1.1
… (response) HTTP/1.1 200 OK
Content-type:application/xml
Content-length :<nnn>
<?xml version="1.0" encoding="UTF-8"?>
<CMSourceSupport version=”1.0” xmlns=”http://www.isapi.org/ver20/XMLSchema”>
        <MediaDeviceListSize>3</MediaDeviceListSize>
          <SupportedMediaDeviceSourceList>
            <MediaDeviceSource>
                <MediaDeviceMfgr>HIKVISION</MediaDeviceMfgr>
                <MediaDeviceMgmtMode>Proxy</MediaDeviceMgmtMode>
                <MediaDeviceModel>DS-2CD876MF</MediaDeviceModel>
                <MediaDeviceModel>DS-2CD877MF</MediaDeviceModel>
                <MediaDeviceModel>DS-2CD855MF</MediaDeviceModel>
            </MediaDeviceSource>
          </SupportedMediaDeviceList>
</CMSourceSupport>
```

Please note that the manufacturer name string is treated in a case insensitive manner. Also, manufacturer strings that contain spaces must use the W3C special character replacement scheme. For example, a fictitious vendor named “Volcano Vision” would have a QSP that looks like “...mfgr=Volcano%20Vision…”. RaCM devices must parse out, and convert, the special character symbols just like the processing associated with a standard URI. Additionally, a consumer may place multiple “mfgr=…” QSPs in a request. For example:

GET /ISAPI/ContentMgmt/souceSupport?mfgr=HIKVISION&mfgr=ONVIF is a valid REST URI for the sourceSupport resource. The recommendation is that a consumer should ***not*** send more than 3 manufacturer IDs per request. If a RaCM device cannot process the number of ‘mfgr’ QSPs sent in a single request, it should return an HTTP status code of ‘503 Bad Request’.

6 /ISAPI/ContentMgmt/Capabilities

/ISAPI/ ContentMgmt /capabilities		**General Resource v2.0**
GET		
Description	It is used to get device capability.	
Query	None	

Inbound Data	None
Success Return	<RacmCap>
Notes:	

RacmCap XML Block

```
< RacmCap version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
   <isSupportZeroChan> <!-- opt, xs:boolean --> </isSupportZeroChan >
   <inputProxyNums><!-- opt, xs:integer -->< /inputProxyNums>
   <eSATANums> <!-- opt, xs:integer --> </eSATANums>
   <miniSASNums> <!-- opt, xs:integer --> </miniSASNums>
   <nasNums> <!-- opt, xs:integer --> </nasNums>
   <ipSanNums> <!-- opt, xs:integer --> </ipSanNums>
   <isSupportRaid> <!-- opt, xs:boolean --> </isSupportRaid>
   <isSupportExtHdCfg> <!-- opt, xs:boolean --> </isSupportExtHdCfg>
   <isSupportTransCode> <!-- opt, xs:boolean --> </isSupportTransCode>
   <isSupportIpcImport> <!-- opt, xs:boolean --> </isSupportIpcImport>
   <NasMountType> <!—opt -->
      <isNFSSupportAuthentication/> <!-- opt, xs:boolean -->
      <isCIFSSupportAuthentication/> <!-- opt, xs:boolean -->
   </NasMountType>
   <isSupportIpcStreamType/>    <!-- opt, xs:boolean -->
   <isSupportIOInputProxy/> <!-- opt, xs:boolean -->
   <isSupportIOOutputProxy/> <!-- opt, xs:boolean -->
   <isSupportPTZRs485Proxy/> <!-- opt, xs:boolean -->
   <isSupportSrcIDSearch/><! -- opt , xs:boolean -- >
   <isSupportReversePlayback/><!-- opt ,xs:boolean-->
   <pictureSearchType
opt="AllEvent,CMR,MOTION,ALARM,EDR,ALARMANDMOTION,Command,pir,wlsensor,callhelp,facedetection
,FieldDetection,scenechangedetection,LineDetection,regionEntrance,regionExiting,loitering,group,rapidMove,pa
rking,unattendedBaggage,attendedBaggage,vehicleDetection,manual,manualSnapShot,playSnapShot,allPic"/>
<!-- opt, xs:string -->
   <isSupportMainAndSubRecord/><! -- opt , xs:boolean -->
   <isSupportSyncIPCPassword><! -- opt , xs:boolean --></isSupportSyncIPCPassword>
   <isSupportTransferIPC><! -- opt , xs:boolean --></isSupportTransferIPC>
   <isSupportPOS><! -- opt , xs:boolean --></isSupportPOS>
</RacmCap>
```

XSD File: cmRacmCap.xsd

7 /ISAPI/ContentMgmt/record

This section describes the REST Interfaces to configure the logical storage used for media archival, along with the actual recording session configurations. A recording session is also known as a “track” in the RaCM Device. A track archives media (Audio, Video, and Metadata) from a Source. The Source can be local (e.g. DVR recording from a local analog port) or remote (e.g. IP Network Media Device).

7.1 /ISAPI/ContentMgmt/record/storageMounts

This REST Interface is used to configure the total storage available to the RaCM Device to be used to archive media. This storage is described as a List of Mount points, along with root directories and sizes. There is no affinity specified, at this time, with regards to assigning tracks or range of tracks to a particular Disk or Mount.

URI	/ISAPI/ContentMgmt/record/storageMounts		**Type**	Resource
RequirementLevel	Basic			
Function	Description of the REST method parameters and formats available to functionally manipulate the ‘record/storage’ resource.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET	None	None	<MountList>	
PUT	None	<MountList>	<ResponseStatus>	
POST	None	<Mount>	<ResponseStatus>	
DELETE	None	None	<ResponseStatus>	
Notes	This resource is used to manage the total storage allocation and logical mounts of the Recorder. It is allowable to DELETE the entire list, though any implementation is free to return an error for that operation if that capability is undesirable. Low level drive configuration should be done through “/ISAPI/System/storage”.			

This resource manages the <MountList> XML object and follows the same scheme used for <TrackList> manipulation and other similar examples from IPMD that manage a list-based XML resource (see “ISAPI-REST List-Entry <id> Creation method”, below.). As such, GET and PUT methods are used to access the entire <MountList>. POST is used (with a “dummy” <id> of 0) to create an individual entry within the list; where the newly created <id> is returned in the <ResponseStatus> given for the POST request.

Once an <Mount> entry is created, it can be accessed by its <id>, using this resource:

URI	/ISAPI/ContentMgmt/record/storageMounts/<id>		Type	Resource
Function	Description of the REST method parameters and formats available to access a single <Mount> entry.			
Methods	Query String(s)	Inbound Data	Return Result	
GET	None	None	<Mount>	
PUT	None	<Mount>	<ResponseStatus>	
POST	N/A	N/A	<ResponseStatus w/ error code>	
DELETE	None	None	<ResponseStatus>	
Notes	POST (i.e. Create) is not allowed for individual <Mount> entry, with given explicit <id>.			

Example XML <MountList>:

```
<?xml version="1.0" encoding="UTF-8"?>
<MountList version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <Mount>
    <id>1</id>
    <path>/dev/hda</path>
    <dir>/racm1/record_tracks</dir>
    <size>200000000000</size>
    <descr>master ide</descr>
  </Mount>
  <Mount>
    <id>2</id>
    <path>/dev/sda</path>
    <dir>/racm1/record_tracks</dir>
    <size>500000000000</size>
    <descr>first scsi</descr>
  </Mount>
  <Mount>
    <id>3</id>
    <path>d:</path>
    <dir>/racm1/record_tracks</dir>
    <size>100000000000</size>
    <descr>win-dos drive</descr>
  </Mount>
</MountList>
```

XSD File: cmRacmMount.xsd

7.2 /ISAPI/ContentMgmt/record/profile

RaCM devices have the ability to support more than one type of 'track'. Since a track is nothing more than a named handle to a virtual content container, ISAPI does not specify how the content is actually recorded, indexed, etc. However, the attributes and properties that define how a track is configured, how a track's content is understood, and how its contents may be played, are critical to interoperability. The definitions and descriptions of track attributes and track types are defined in following ***Sections***. The REST resource that advertises these recording attributes (i.e. track types) is defined here.

URI	/ISAPI/ContentMgmt/record/profile		**Type**	Resource
Function	Description of the REST resource that advertises the track types supported by a RaCM device.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET	None	None	<CMRecordProfile>	
PUT	None	None	<ResponseStatus w/Error Code>	
POST	None	None	<ResponseStatus w/Error Code>	
DELETE	None	None	<ResponseStatus w/Error Code>	
Notes	This resource is ***read-only***.			
Example	<?xml version="1.0" encoding="UTF-8"?> <CMRecordProfile version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema"> <trackType>standard</trackType> <trackType>polymorphic</trackType> <trackType>polytemporal</trackType> </CMRecordProfile>			
	The above example XML response instance indicates that the RaCM device supports all of track types defined by RaCM specifications. See "/ISAPI/ContentMgmt/record (Recorder Configuration & Control)" for details about the attributes and behaviors of track types.			

The "/ISAPI/ContentMgmt/record/profile" REST resource is a read-only advertisement to management applications (VMSs, etc.) of the supported track types which, in turn, governs their configuration options. All RaCM devices are required to support at least one of the sanctioned track types described in this specification. The "CMRecordProfile" schema definition also enables vendors to provide their own extensions to this information using a structured extension mechanism. The schema is described below.

7.2.1/ISAPI/ContentMgmt/record/profile Schema Definition

XSD File: cmRecordProfile.xsd

RaCM devices list the track types they support which affects the configuration characteristics of the tracks (i.e. polymorphic tracks can have video and audio sources mixed (for example) whereas standard tracks contain only one form of media data). For vendors that desire to add their own track type extensions, this is provided for in the schema with the following conditions:

- All RaCM devices MUST support at least one of the sanctioned track types (standard, polymorphic, polytemporal). Any custom track type extensions must be in addition to this requirement. This is to ensure that all RaCM have a base level of interoperability.
- All custom extensions listed in a “CMRecordProfile” MUST have the following contents:
    - The listed track type of “other” for the custom/extended track type.
    - In the “profileExtension” element:
        - A URI link to where the formal definition of the extension information can be accessed.
        - A brief description, embedded in the document instance, of what the extension information means.
        - The extension information itself.

7.3 /ISAPI/ContentMgmt/record/tracks

This REST interface is used to configure a recording session or “track”. A track is treated virtually with regards to how the implementation may actually store the archived media on disk.

The track is generally accessed or referred to by its <id>. The <Stream> number is a logical (outbound streaming channel) number for the track and is not tightly coupled to the <id>, which is treated as an index into the <TrackList>.

Note on GUIDs

All UUIDs/GUIDs MUST be universally unique. They can be assigned by a central VMS Server or GUID broker or auto-generated locally. However, all UUIDs/GUIDs MUST be compliant with the ISO/IEC 9834-8 / ITU X.667 formats and definitions.

Note on Recording Modes

Currently, those recording modes are defined:

- CMR – “Continuous Mode Recording” which implies recording media as it is available.
- EDR – “Alarm or Motion-Driven Recording” which implies recording media when I/O alarms or video motion are detected.
- ALARM–“alarm-Driven Recording” which implies recording media when I/O alarms are detected.
- MOTION–“alarm or motion-Driven Recording” which implies recording media when io alarms or video motion are detected.
- ALARMANDMOTION–“alarm or motion-Driven Recording” which implies recording media when I/O alarms or video motion are detected.
- COMMAND–“alarm or motion-Driven Recording” which implies recording media when I/O alarms or video motion are detected.
- SMART–“SMART Event Driven Recording” which implies recording media when intelligent events are detected or triggered.

Track Size

The Track’s size is determined by the <Size> value. However, optionally, some flexibility is allowed for best-effort by the implementation to honor the <Duration> value.

7.3.1Custom Configuration Data (Extensions)

Custom configuration information can be added using <CustomExtensionList>, but for interoperability, the name is assigned to each extension object which MUST be registered with ISAPI, and a schema (XSD) MUST be provided for the “xs:any” object(s).

7.3.2ISAPI-REST List-Entry <id> Creation method

Track <id>’s, along with most other list-based <id>’s, are managed (SET) by the target RaCM Device. The method for track creation (and return of <id> value is in accordance with other list- based XML object examples in IPMD). See the “/ISAPI/ContentMgmt/record/tracks” Resource Description, **below.**

7.3.3Streaming URL implied in <Track> configuration

There is an implied relationship between the REST URL’s and RTSP URL’s.
Within the <Track> configuration, the <Channel> value is used for “live” viewing of the media stream being recorded to that track, using the URL described in section Live Streams. The Track’s REST <id> value is used for recorded (i.e. “archive”) media streaming (see below).

7.3.4Recording Source Description

The source for a recording track is logically described by the <SourceDescriptor>, which is part of the track configuration. The <SourceDescriptor> contains two important tags which help uniquely identify the media source: <SrcGUID> and <SrcUrl>. The <SrcGUID> is a GUID/UUID for a stream source, which may also provide multiple channels of output. The <SrcChannel> value allows for more specific description of the input media-stream at a logical level.

For a local port, the <SrcUrl> should contain a symbolic reference to a local stream (encoded from a local /System/Video/inputs/channels) as follows:

rtsp://localhost/ISAPI/Streaming/channels/<id>

Also, for both local and remote sources, the <SrcDriver> provides an optional, vendor specific, hint with regards to the name of an executable/driver to use for stream acquisition.

7.3.5Recording Schedule overview

The <TrackSchedule> defines the recording schedule for the Track. It generally contains either external-references to schedules in the schedule database (/ISAPI/ContentMgmt/schedules) or an embedded sequence of <ScheduleBlock>’s (typically just one). A <ScheduleBlock> is a single logical schedule, identified by GUID. The default (embedded) <ScheduleBlock>, in the example, contains the required identifiers <ScheduleBlockGUID> and <ScheduleBlockType>, along with a sequence of <ScheduleAction>’s, which are used to build a day-of-week schedule.

This default, day-of-week schedule contains <Actions> to perform during the defined period. A period is defined by a start-time and end-time, with each time expressed as Day-of-Week and Time- of-Day. The

Day-of-Week value is identified by a restricted name string. The Time-of-Day is expressed in local time, in order to allow for a more human-intuitive definition of time from the local administrators perspective and also account for Day-Light-Savings Time. Thus, for a time period expressed as “midnight to 8 am” local time, the intended elapsed time is 8 hours **normally**.

However, if Day-Light-Savings Time is enabled, on the morning of transition (in the Spring and assuming the switch occurs at 2:00 am), the actual elapsed time would represent just 7 hours of time, during this morning of transition. In the fall, the reverse would be true, and the actual elapsed time, for just that morning, would be 9 hours.

Example XML fragment (“lunch-time” period):

```
<ScheduleActionStartTime>
  <DayOfWeek>Monday</DayOfWeek>
  <!-- inclusive -->
  <TimeOfDay>12:00:00</TimeOfDay>
</ScheduleActionStartTime>
<ScheduleActionEndTime>
  <DayOfWeek>Monday</DayOfWeek>
  <!-- exclusive -->
  <TimeOfDay>1:00:00</TimeOfDay>
</ScheduleActionEndTime>
<ScheduleDSTEnable>true</ScheduleDSTEnable>
```

The start-time is inclusive of the time specified. The end-time is exclusive to allow aggregation of time period definitions to create a continuum without overlap/conflict.

<ExternalScheduleBlockReferences> allow for reduction in size and simplification of the <Track> configuration object, by enabling references to shared schedules in a database, as opposed to embedding them within each <Track> configuration.

7.3.6Track Description NVP

One of the key elements in each track’s parameter base is the “<Description>” element. This field is a Comma Separated Variable (CSV) string which contains a list of Name-Value-Pairs (NVP of form “Name=Value”. With this scheme, the comma (‘,’) and equal (‘=’) are treated as reserved characters; however, if a Name or Value string must contain and these characters, the XML encoding standard can be used to embed them if necessary (i.e. replace the ‘=’ with &61, and replace the ‘,’ with &44).

Parameter Name	Parameter Values	Comments
trackType	➢ “standard” = normal, single content base track ➢ “polymorphic” = multi-content type track ➢ “polytemporal” = multi-time segmented track	Required Field
sourceTag	Manufacturer specific device-type string (e.g. make/model)	Optional

contentType	➢ "video" ➢ "audio" ➢ "metadata" ➢ "text NOTE: For polymorphic tracks this indicates the primary, or predominant, content type.	Required Field
codecType	See Appendix A.	Required Field
resolution	Required for audio/video. Field indicating resolution of the data elements in a datastream. For video, this is the horizontal by vertical resolution in a 'Horizontal x Vertical' format where ASCII 'x' separates the horizontal and vertical integer numbers. The assumed video format is progressive (i.e. frame based). For video streams that are interlaced (i.e. field based) and ASCII lower-case 'i' needs to be. For audio, it is the bit-width of the samples. If a text protocol is enabled for double-byte characters, this field should be used to indicate "2B" character sets.	Required for Audio/Video
framerate	Frame rate of encoder output as a floating point number.	Required for Video
bitrate	Bit rate of datastream (bps or kbps integer).	Optional

Examples

<Description>trackType=standard,sourceTag=AXIS210a,contentType=video,codecType=MPEG4-SP,resolution=640x480,framerate=25.0,bitrate=3200 kbps</Description>

7.3.7/ISAPI/ContentMgmt/record/tracks

URI	/ISAPI/ContentMgmt/record/tracks		Type	Resource
Function	Description of the REST method parameters and formats available to functionally manipulate the 'record/storage/tracks' resource.			
Methods	Query Strin (s)	Inbound Data	Return Result	
GET	None	None	<TrackList>	
PUT	N/A	<TrackList>	<ResponseStatus>	
POST	N/A	<Track>	<ResponseStatus>	
DELETE	N/A	N/A	<ResponseStatus w/error code>	
Notes	**Track Creation**: POST (Create) will expect, as HTTP Payload, an individual <Track> object instead of the <TrackList>. For the Create operation, the <id> tag, within the <Track> XML, must contain a "dummy" value of 0 (zero). To lessen possibility for ambiguity, it is not permissible for the Client (xMS) to set the <id> during track creation, though it is possible for the Client to update (PUT) the entire <TrackList> (**with each entry containing valid <id>'s already set by the target**). This is in alignment with other such list-based REST Resources in IPMD.			

Reference Example <ResponseStatus> from Service Model Specification:

```
<?xml version="1.0" encoding="UTF-8"?>
<ResponseStatus version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <requestURL>/Streaming/Channels</requestURL>
  <statusCode>1</statusCode>
  <!-- O=1-OK, 2-Device Busy, 3-Device Error, 4-Invalid Operation, 5-Invalid XML Format, 6-Invalid XML Content; 7-Reboot Required -->
  <statusString>OK</statusString>
  <ID>1</ID>
</ResponseStatus>
```

7.3.8/ISAPI/ContentMgmt/record/tracks/<id>

Once created, individual Tracks are managed via:

URI	/ISAPI/ContentMgmt/record/tracks/<id>		Type	Resource
RequirementLevel	Basic			
Function	Resource to address (Read/Update) single <Track> by <id>.			
Methods	Query Strin (s)	Inbound Data	Return Result	
GET	None	None	<Track>	
PUT	None	<Track>	<ResponseStatus>	
POST	N/A	N/A	<ResponseStatus w/error code>	

DELETE	None	None	<ResponseStatus>

7.3.9Example Track Creation Message Exchange

A. Client attempts track creation with “POST /ISAPI/ContentMgmt/record/tracks” containing:

```
<?xml version="1.0" encoding="UTF-8"?>
<Track version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
    <!-- new dummy value: -->
    <id>0</id>
    <Channel>12345</Channel>
    <Enable>true</Enable>
    <Description>trackType=standard,sourceTag=AXIS210a,contentType=video,codecType=MPEG
4- SP,resolution=640x480,framerate=20.0,bitrate=6000 kbps</Description>
    <TrackGUID>{A01AAAAA-BBBB-CCCC-DDDD-033595353625}</TrackGUID>
    <Size>4000000000</Size>
    <Duration>P10DT15H</Duration>
    <DefaultRecordingMode>POS</DefaultRecordingMode>
    <LoopEnable>true</LoopEnable>
    <!-- … REST OF OBJ NOT INCLUDED… -->
  </Track>
```

B. If creation is successful, RaCM Device responds with:

```
<?xml version="1.0" encoding="UTF-8"?>
<ResponseStatus version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <requestURL>/ISAPI/ContentMgmt/record/tracks</requestURL>
  <statusCode>1</statusCode>
  <statusString>OK</statusString>
  <ID>777</ID>
</ResponseStatus>
```

Please note that the <ID> tag is uppercase. This is to match the example in Service Model Specification (Section 10.1.4). The returned Track <id> value is 777, which will be used in the following Track Deletion example.

Example Track Deletion Message Exchange

A. Client attempts track deletion with “DELETE /ISAPI/ContentMgmt/record/tracks/777” (no payload), using <id> previously given by the creation example response, above.

B. If deletion is successful, RaCM Device responds with:

```
<?xml version="1.0" encoding="UTF-8"?>
<ResponseStatus version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <requestURL>/ISAPI/ContentMgmt/record/tracks</requestURL>
  <statusCode>1</statusCode>
  <statusString>OK</statusString>
  <ID>777</ID>
</ResponseStatus>
```


More detailed Single Track XML example (<Schedule> incomplete)

```
<?xml version="1.0" encoding="UTF-8"?>
<TrackList version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <Track>
    <id>1</id>
    <Channel>12345</Channel>
    <Enable>true</Enable>
   <Description>trackType=standard,sourceTag=AXIS210a,contentType=video,codecType=MPEG4-SP,
resolution=640x480,frameRate=20 fps,bitrate=6000 kbps</Description>
    <TrackGUID>{A01AAAAA-BBBB-CCCC-DDDD-033595353625}</TrackGUID>
    <Size>4000000000</Size>
    <Duration>P10DT15H</Duration>
    <DefaultRecordingMode>CMR</DefaultRecordingMode>
    <LoopEnable>true</LoopEnable>
    <SourceDescriptor>
      <SrcGUID>{E800A543-9D53-4520-8BB8-9509062C692D}</SrcGUID>
      <SrcChannel>1</SrcChannel>
      <StreamHint>video, mp4, 640x480, 20 fps, 6000 kbps</StreamHint>
      <SrcDriver>RTP/RTSP</SrcDriver>
      <SrcType>mp4 video</SrcType>
      <SrcUrl>rtsp://10.3.2.26/mpeg4/media.amp</SrcUrl>
      <SrcUrlMethods>DESCRIBE, SETUP, PLAY, TEARDOWN</SrcUrlMethods>
      <SrcLogin>admin:admin</SrcLogin>
    </SourceDescriptor>
    <TrackSchedule>
      <ExternalScheduleBlockReferences>
        <ScheduleBlockReference>
          <ScheduleBlockGUID>{F018AD02-BC04-4520-8BB8-123409AC5678}</ScheduleBlockGUID>
        </ScheduleBlockReference>
        <ScheduleReference>
          <ScheduleBlockGUID>{C2F37123-DD19-4520-8BB8-444307DB5565}</ScheduleBlockGUID>
        </ScheduleReference>
      </ExternalScheduleBlockReferences>
        <ScheduleBlock>
          <ScheduleBlockGUID>{ABC12345-CDEF-4520-8BB8-7135789C8790}</ScheduleBlockGUID>
          <ScheduleBlockType>/ISAPI/recording/schedule/default</ScheduleBlockType>
          <ScheduleAction>
            <id>1</id>
            <ScheduleActionStartTime>
              <DayOfWeek>Monday</DayOfWeek>
              <!-- inclusive -->
              <TimeOfDay>00:00:00</TimeOfDay>
            </ScheduleActionStartTime>
            <ScheduleActionEndTime>
```



```
                <DayOfWeek>Monday</DayOfWeek>
                <!-- exclusive -->
                <TimeOfDay>08:00:00</TimeOfDay>
              </ScheduleActionEndTime>
              <ScheduleDSTEnable>true</ScheduleDSTEnable>
              <Description>PreMorning (Midnight to 8am, local time)</Description>
              <Actions>
                <!-- alarm or motion detection triggers a recording-->
                <ActionRecordingMode>EDR</ActionRecordingMode>
                <PreRecordTimeSeconds>30</PreRecordTimeSeconds>
                <PostRecordTimeSeconds>30</PostRecordTimeSeconds>
              </Actions>
            </ScheduleAction>
            <ScheduleAction>
              <id>2</id>
              <ScheduleActionStartTime>
                <DayOfWeek>Monday</DayOfWeek>
                <!-- inclusive -->
                <TimeOfDay>08:00:00</TimeOfDay>
              </ScheduleActionStartTime>
              <ScheduleActionEndTime>
                <DayOfWeek>Monday</DayOfWeek>
                <!-- exclusive -->
                <TimeOfDay>12:00:00</TimeOfDay>
              </ScheduleActionEndTime>
              <ScheduleDSTEnable>true</ScheduleDSTEnable>
              <Description>Morning (8am to noon, local time) </Description>
              <Actions>
                <ActionRecordingMode>CMR</ActionRecordingMode>
                <PreRecordTimeSeconds>30</PreRecordTimeSeconds>
                <PostRecordTimeSeconds>30</PostRecordTimeSeconds>
                </Actions>
              </ScheduleAction>
            </ScheduleBlock>
          </ScheduleBlockList>
      </TrackSchedule>
    </Track>
  </TrackList>
```

7.3.10 Track List Schema

XSD File: cmTrackList.xsd

7.3.11 /ISAPI/ContentMgmt/record/tracks/<id>/dailyDistribution

URI	/ISAPI/ContentMgmt/record/tracks/<id>/dailyDistribution		Type	Resource
Function	get the track record segments's daily distribution			
Methods	Query String(s)	Inbound Data	Return Result	
GET		<trackDailyParam>	<trackDailyDistribution>	
POST		<trackDailyParam>	<trackDailyDistribution>	
Notes	<monthOfYear> month of year, start from 1 <dayOfMonth> day of month, start from 1 <record>detect if someday has video recording <recordType> record type, time: normal record type; event: event record type			

XSD File: cmTrackList.xsd

7.4 /ISAPI/ContentMgmt/record/control

This resource is used to send explicit control command to the "record" service.

7.4.1/ISAPI/ContentMgmt/record/control/manual/start/tracks/<ID>

URI	/ISAPI/ContentMgmt/record/control/manual/start/tracks/<id>		Type	Resource
RequirementLevel	Basic			
Function	Description of the REST method parameters and formats available to functionally manipulate the 'record/control/manual/start' resource.			
Methods	Query String(s)	Inbound Data	Return Result	
GET	N/A	N/A	<ResourceDescription>	
PUT	N/A	None	<ResponseStatus>	
POST	N/A	N/A	<ResponseStatus w/error code>	
DELETE	N/A	N/A	<ResponseStatus w/error code>	

Notes	This resource is used to manually Start the recording track, regardless of recording mode. To Enable or Disable (i.e. permanent Stop) the track, the configuration interface should be used to update the track configuration object to set the enable/disable value accordingly.

7.4.2/ISAPI/ContentMgmt/record/control/manual/stop/tracks/<ID>

URI	/ISAPI/ContentMgmt/record/control/manual/stop/tracks/<id>		Type	Resource
RequirementLevel	Basic			
Function	Description of the REST method parameters and formats available to functionally manipulate the 'record/control/manual/stop' resource.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET	N/A	N/A	<ResourceDescription>	
PUT	N/A	None	<ResponseStatus>	
POST	N/A	N/A	<ResponseStatus w/error code>	
DELETE	N/A	N/A	<ResponseStatus w/error code>	
Notes	This resource is used to manually stop the recording track and regardless of recording mode. To enable or disable (i.e. permanent Stop) the track, the configuration interface should be used to update the track configuration object to set the enable/disable value accordingly.			

7.4.3/ISAPI/ContentMgmt/record/control/locks

URI	/ISAPI/ContentMgmt/record/control/locks		Type	Resource
Requirement Level	Basic			
Function	Description of the REST method parameters and formats available to functionally manipulate the '/ISAPI/ContentMgmt/record/control/locks' resource.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET	None	None	<RecordingLockList>	
PUT	None	<RecordingLockList>	<ResponseStatus>	
POST	None	<RecordingLock>	<ResponseStatus>	
DELETE	N/A	N/A	<ResponseStatus>	
Notes	Used to manage the list of recording locks. (Please refer to "ISAPI-REST List-Entry <id> Creation method above.)			

URI	/ISAPI/ContentMgmt/record/control/locks/<id>		Type	Resource
RequirementLevel	Basic			
Function	Resource used to manage a single <RecordingLock> entry.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET	None	None	<RecordingLock>	
PUT	None	<RecordingLock>	<ResponseStatus>	
POST	N/A	N/A	<ResponseStatus w/error code>	
DELETE	None	None	<ResponseStatus>	
Notes				

XSD File: cmTrackLockList.xsd

8 /ISAPI/ContentMgmt/search

This section of the specification defines the operation and parameters associated with the 'search' service within the ISAPI Content Management hierarchy. Tables, examples and schemas are provided for defining and explaining the search functions.

8.1 /ISAPI/ContentMgmt/search/profile

Due to the complexity of the functions entailed in a recording and content management (RaCM) device, all RaCM devices MUST provide a 'profile' resource (schema instance) such that entities accessing them may determine their functional level and basic attributes. Details are outlined below.

URI	/ISAPI/ContentMgmt/search/profile		**Type**	Resource
Function	RaCM Mandatory REST resource/object that publishes the functional profile/level of a RaCM device and its operable service/resource structure.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET	None		<CMSearchProfile>	
PUT	N/A	N/A	<ResponseStatus w/error code>	
POST	N/A	N/A	<ResponseStatus w/error code>	
DELETE	N/A	N/A	<ResponseStatus w/error code>	
Notes	The 'GET' request issued to retrieve an instance of the 'CMCapabilities' XML schema.			

Example(s)	<?xml version="1.0" encoding="UTF-8"?> <CMSearchProfile version="1.0" xmlns="ISAPIllianxce.org:resourcedescription"> <searchProfile>full</searchProfile> <textSearch>true</textSearch> <maxSearchTimespans>2</maxSearchTimespans> <maxSearchTracks>40</maxSearchtracks> <maxSearchSources>40</maxSearchSources> <maxSearchMetadatas>16</maxSearchMetadatas> <maxSearchMatchResults>100</maxSearchMatchResults> <maxSearchTimeout>120</maxSearchTimeout> <maxConcurrentSearches>8</maxConcurrentSearches> </CMSearchProfile> The above example represents a hypothetical RaCM device that supports a 'Full' profile for search its functionality (see below for more details). The succeeding "textSearch" parameter indicates that this device performs raw text string searches on recorded text, and text-based metadata such as Point-of-Sale, or Automated Teller Machine output. The next set of optional parameters (though recommended for Full devices) indicates the maximum number of specific search parameters that can be part of a single instance of a search criterion. The following parameter, "maxSearchResults" indicates the maximum number of results the RaCM will pass back per search instance. The optional "maxSearchTimeout" parameter indicates that the RaCM devices will timeout searches that exceed two minutes (120 seconds) to execute. The final parameter, which is optional, indicates the maximum number of concurrent search operations the RaCM device can support.

8.1.1/ISAPI/ContentMgmt/search/profile Schema Definition

The ContentMgmt/search/profile schema is used to define the types of searches a RaCM device supports. A device with a search profile of 'Basic' only performs searches with one timespan per search request (see "*/ISAPI/ContentMgmt/search*"). Devices that support the 'Full' search profile must outline their parameter limits, as described in the following schema.

XSD File: cmSearchProfile.xsd

8.2 /ISAPI/ContentMgmt/search

The Content Management 'Search' service is the primary component for conducting searches of content bases managed by a ISAPI recording device. Fundamentally, searches are initiated using a parameter-based criteria set which is conveyed by the initiator to the device via the 'CMSearchDescription' XML schema. *Since not all programming languages allow content bodies with HTTP GET methods, both 'GET' and 'POST' are supported as message types for initiating searches*. The responding device passes back the results in a 'CMSearchResult' XML schema instance for search requests that had valid syntax. If a search request is syntactically invalid (i.e. no payload, malformed schema instance, etc.), an HTTP response with status code 400 (Bad Request) and a corresponding 'Response Status' are returned to the requester. Please note that a syntactically correct search, that has no matching criteria, returns a 'CMSearchResult' schema instance with a 'NO MATCHES' status string.

Essentially, most searches are conducted based on time and/or track and/or source related search parameters. Full profile Content Management devices also support the potential for metadata search related parameters. More details, and examples, follow.

URI	/ISAPI/ContentMgmt/search/		**Type**	Service
Function	Mandatory description of the REST method parameters and formats available to functionally manipulate the 'search' resource/object.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET	None	<CMSearchDescription>	<CMSearchResult or ResponseStatus w/error code>	
PUT	N/A	N/A	<ResponseStatus w/error code>	
POST	None	<CMSearchDescription>	<CMSearchResult or ResponseStatus w/error code>	
DELETE	N/A	N/A	<ResponseStatus w/error code>	
Notes	The 'GET' or 'POST' message requires a "CMSearchDescription" XML document to engage a search. An example XML document instance follows.			

Example(s)	<?xml version="1.0" encoding="UTF-8"?> <CMSearchDescription version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema"> <searchID>{812F04E0-4089-11A3-9A0C-0305E82C2906}</searchID> <trackIDList> <trackID>9</trackID> <trackID>22</trackID> <trackID>43</trackID> </trackIDList> <timeSpanList> <timeSpan> <startTime>2013-06-10T12:00:00Z</startTime> <endTIme>2013-06-10T13:30:00Z</endTime> <timeSpan> </timeSpanList> <contentTypeList> <contentType>video</contentType> </contentTypeList> <maxResults>40</maxResults> <metadataList> <metadataDescriptor>recordType.meta.hikvision.com/motion</metadataDescriptor> <SearchProperity> <plateSearchMask><!-- opt, xs:string,1-31 --></plateSearchMask> <stateOrProvince><!-- opt, xs:interger--></stateOrProvince> <country><!--opt, xs:string, Regional, and national mutex , 0- Algorithm library does not support the national identification card ,1-(CZ - Czech Republic),2-(FRA - France),3-(DE - Germany), 4-(E - Spain),5-(IT - Italy),6-(NL - Netherlands),7-(PL - Poland), 8-(SVK - Slovakia), 9-(BY - Belorussia), 10-(MDA - Moldova), 11-(RU - Russia),12-(UA - Ukraine) , 0xff-(All) --></country> </SearchProperity> </metadataList> </CMSearchDescription> The above example is for a search of tracks 9, 22, and 43, between twelve noon and 1:30PM on June 10th, 2013 for Video Motion events. The requester does not want more than 40 results passed back in the search response, and, that matching results should only be video segments (based on the above “contentTypeList”).

8.2.1Search Query Parameter Schema Definition

The following XML schema definition defines the search parameters that are provided to the ContentMgmt/search service. Basically, an entity can search based on time and/or source and/or tracks

and/or metadata. Basic profile devices are not required to support metadata in their search criteria. Each search must be given a unique ID (“searchID”) by the initiator. This ID is an ISO/IEC 9834-8/ITU X.667 compliant UUID/GUID. The responder echoes this value back in the search response (see next section) to correlate the results to the corresponding query. Basic profile devices are only required to support one query at-a-time. Full profile devices MUST advertise the number of concurrent search queries they can support (via the “/ISAPI/ContentMgmt/search/profile” Resource).

The ‘cmSearchDescription’ schema allows searches based on: track lists, track state lists, source ID lists, channel ID lists, timespan lists, and metadata lists. Only one of these criteria is required for a valid search.

Optional information includes the ability for an initiator to specify that it only wants to receive a maximum number of responses to a search criteria (i.e. limit the potential number of responses). This is specified by the “maxResults” element. When this limit is specified in a search request, the responding device must not send more than the requested number of responses (matches). If the querying entity specifies a “maxResult” limit that is less than the number of total results that matched the original search criteria, it must re-issue the search query with the “search ResultsPosition” parameter indicating the number of prior results the inquirer has already received. For example, a client application searches for Video Motion events from the prior night and indicates it only wants a maximum of ten results passed back. The ContentMgmt device, while performing the search, discovers 23 matches. However, the device is only allowed to pass back the first ten results based on the inquirer’s “maxResults” value. The device does so, but indicates in the search response schema instance a status string of “MORE”. After this, if the inquirer desires to retrieve the next set of matching results, it must do so with the same search criteria and a “searchResultsPosition” value that indicates the cumulative number of results received by the inquirer. In other words, the “searchResultsPosition” is a walking index used to indicate where an inquirer desires the searching entity to resume the search.

Another optional search parameter that governs, or conditions, the search result contents is the “contentTypeList”. This element allows one or more “contentType” designators. These designators instruct the searching Content Manager to only provide matching result segments that correspond to the designated content types. The supported content types are: “video”, “audio”, “metadata”, “text”, “mixed” and “other”. If a search specifies one, or more content types, the searching Content Manager will ignore matches to the search criteria if the content segment/track does not match the specified content type(s).

If a RaCM device supports raw text searches, searches are allowed to pass in “searchText” as a search criteria, but only for “text” and/or “metadata” content types. The following conditions and restrictions apply regarding text searching:

- The ‘content type’ MUST be specified when conducting searches that involve text. The applicable content types are “text” and “metadata”.
- There is an inherent limitation of one text string per search instance.
- The search string may not exceed 128 characters.
- ALL text searches are performed in a ***case insensitive*** manner such that any combination of upper and lower casing can be matched (‘open matching’).
- Any search string that contains whitespace, or special characters, MUST be started and ended with double-quotes (“).

Since raw text searches are compute intensive, requesters, and RaCM devices, that honor search timeouts should either: A) ignore timeouts for text searches, or B) timeouts should be lengthened significantly for text searches.

Finally, RaCM devices that support search timeout limits must allow requesters to dynamically/optionally specify ‘desired’ timeout limits on a per-search basis. Please note that this is a hint to the RaCM device to restrict, in a best effort manner, the search operation duration to the time limit specified in the XML parameter set. If a RaCM device cannot honor the timeout it will process the search as best it can. Additionally, a RaCM device that does not support search timeouts should ignore a timeout if it errantly receives one on search request instance.

XSD File: cmSearchDescription.xsd

8.2.2Search Query Results Schema

The response to a ContentMgmt/search operation, using the “CMSearchDescription” schema, is defined by the “CMSearchResult” schema definition below. Fundamentally, the response echoes the initiator’s search ID and an overall status value. The status consists of 2 parts: A) a boolean, called “responseStatus”, indicates overall success or failure, which is followed by B) the “responseStatusString” which indicates a readable status string that further qualifies the overall status (e.g. TRUE + “OK”, or TRUE + “MORE” [indicating more matches were found than the initiator desired to receive], or FALSE + “NO MATCHES [search failed to find matches to specified criteria], or FALSE + “INVALID TRACK ID” [indicating an invalid track ID had been provided]). The combination of an overall Boolean status indicator, plus a qualifying string, allows a requester to quickly determine if a search was successful and then have information detailing the operation.

After the response status, the responder provides the number of matches found, and a list of “matchElements” that corresponding the each matching multimedia segment. Each “matchElement” contains the following information:

- The source ID of the input source that corresponds to the matching segment;
- The track ID of the multimedia track that corresponds to the matching segment;
- The timespan of the matching segment (since it is probably a subset of the overall search’s timespan);
- A media segment descriptor that the search initiator can directly use to playback the corresponding match segment. It contains the following:
    - oThe content type of the segment (video, audio, metadata, text, other);
    - oThe codecType of the content in the segment (e.g. “MPEG4-SP” or G.726);
    - o The rate attribute of the segment, if applicable;
    - o The playback URI for the match segment (i.e. a URI of the responder’s definition that contains all the information necessary to ‘play’ the segment, via RTSP, without the responder having to recommit another search).

Optionally, the responder can include the following information in a “matchElement” where applicable and/or supported:

- The channel ID of the source corresponding the match segment, if the input channel is still active (i.e. ‘live’);
- For metadata searches, the fully qualified ISAPI Domain/Class/Type REST URI(s) of the corresponding events that correlate to the match segment.

8.2.2.1 Search Response Examples

The following example XML schema instances describe some of the potential responses to search queries. Please note that these examples are provided for reference but there are still many potential scenarios not directly addressed.


```
<?xml version="1.0" encoding="UTF-8"?>
<CMSearchResult version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <searchID>{812F04E0-4089-11A3-9A0C-0305E82C2906}</searchID>
  <responseStatus>true</responseStatus>
  <responseStatusStrg>OK</responseStatusStrg>
  <numOfMatches>1</numOfMatches>
  <matchList>
    <matchElement>
      <sourceID>{b049902e72-0049-1158-c0d2-7e330680d93c}</sourceID>
      <trackID>27</trackID>
      <timeSpan>
        <startTime>2013-05-18T10:31.26</startTime>
        <endTime>2013-05-18T10:32:54</endTime>
      </timeSpan>
      <mediaSegmentDescriptor>
        <contenType>video</contenType>
        <codecType>MPEG4-SP</codecType>
        <rateType>”3 Mbps, 30 fps”</rateType>

<playbackURI>rtsp://144.70.13.92:554/ISAPI/Streaming/tracks/27?offset=a07724&amp;
endtime=2013-05-18T10:31.25</playbackURI>
      </mediaSegmentDescriptor>
      <metadataMatches>

<metadataDescriptor>recordType.meta.hikvision.com/motion</metadataDescriptor>
      </metadataMatches>
    </matchElement>
  </matchList>
</CMSearchResult>
```

The above example is in response to a search query for video motion events, with respect to a set of specific sources, i.e. b049902e72-0049-1158-c0d2-7e330680d93c and b049902e72-0049-1158-c0d2-7e330680755e, for video motion events between 10:30 and 10:45AM on May 18th, 2013. One match (“matchElements”) was found that matched the search criteria. The status combination of “true” and “OK” indicate a complete response. If there had been more matches than the requester had allowed for reply, the status would have been “true” and “MORE” in the “responseStatus” and “responseStatusStrg” fields respectively. The “mediaURIDescriptor”s provided is exemplary only.

The playback URIs within the match elements are fictional examples that are syntactically correct. Match element playback URIs are opaque to the requester; they are only required to contain whatever information the responder needs to playback the corresponding media segment via RTSP.


```
<?xml version="1.0" encoding="UTF-8"?>
<CMSearchResult version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
   <searchID>{812F04E0-4089-11A3-9A0C-0305E82C2906}</searchID>
   <responseStatus>false</responseStatus>
   <responseStatusStrg>NO MATCHES</responseStatusStrg>
   <numOfMatches>0</numOfMatches>
</CMSearchResult>
```

In this example, a valid search query had no matches for the original criteria provided.

8.2.2.2 Search Result Schema Definition

XSD File: cmSearchResult.xsd

9 /ISAPI/ContentMgmt/logSearch

The services defined log search protocol. The service from the implementation of RaCM protocol in serach, in addition to the request and the corresponding XML content is different, the log search mechanism and RaCM in the same search services.

URI	/ISAPI/ContentMgmt/logSearch		**Type**	Service
RequirementLevel	- All Profiles -			
Function	Mandatory description of the REST method parameters and formats available to functionally manipulate the 'logSearch' resource/object.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET	None	<CMSearchDescription>	<CMSearchResult or…ResponseStatus w/error code>	
PUT	N/A	N/A	<ResponseStatus w/error code>	
POST	None	<CMSearchDescription>	<CMSearchResult or…ResponseStatus w/error code>	
DELETE	N/A	N/A	<ResponseStatus w/error code>	
Notes	The 'GET' or 'POST' messages require a "CMSearchDescription" XML document to engage a search. An example XML document instance follows.			

Example(s)	<?xml version="1.0" encoding="UTF-8"?> <CMSearchDescription version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema"> <searchID>{812F04E0-4089-11A3-9A0C-0305E82C2906}</searchID> <timeSpanList> <timeSpan> <startTime>2013-06-10T12:00:00Z</startTime> <endTIme>2013-06-10T13:30:00Z</endTime> <timeSpan> </timeSpanList> <metaID>log.hikvision.com/Alarm/motionstart</metaID> <searchResultPostion> 20 </searchResultPostion> <maxResults> 40 </maxResults> </CMSearchDescription> The above example is for a search of log of video motion detection, between twelve noon and 1:30PM on June 10th, 2013 for Video Motion events. The requester does not want more than 40 results passed back in the search response.

9.1.1Search Query Parameter Schema Definition

XSD File: cmSearchDescription.xsd

9.1.2Search Query Results Schema

XSD File: cmSearchResult.xsd

10 Metadata Identity String (MIDS; "metaID")

In order to have a large variety of metadata types, that can be commonly processed, and yet allow flexibility in designing and developing metadata product components, a hierarchical namespace, forming a metadata taxonomy, is employed. This notation is based on a URI structure. The format is:

<domain>/<class>/<type>[/attribute/LID][/TransID][/…]

Definitions for the above URI fields are:

10.1 MIDS Field Definitions

Field/Name	Requirement Level	Comments
Domain	Mandatory	The 'virtual domain' name of the ordaining body for the ***format and definitions*** that are used for the associated metadata/event information. The domain determines the format, and thus the processing and interpretation, of metadata/event instance data.
Class	Optional	Domain-specific 'Class' of the metadata/event information. Some examples are: "Alarm", "Exception", etc. If Class is not contained, it stands for all of classes
Type	Optional	Class-dependent type of metadata/event information. For example, within a class called "Alarm" there would be types such as: "motionStart", "motionStop", etc. If Type is not contained, it stands for all of types
Attribute/LID ('Local ID')	Dependent / Optional	Free-form field that is available for use as additional descriptive information using the following rules: > The convention is that this field MUST be used as the 'Local ID' field for all metadata/event occurrences that are related to, or associated with, a channel/port/stream ID. > For metadata/event occurrences that have no correlation to a channel or port (etc.), this field is optional.

TransID (Transaction ID)	Optional	A string field that uniquely identifies this occurrence instance to the source. If a source entity requires a transactional level acknowledgement, then this field MAY be used as an identifier for expressly acknowledging a specific metadata/event instance. Please note that the source UUID/GUID and timestamp of a metadata/event instance are the standard fields used for uniqueness. Additional fields are optional.

In this hierarchical namespace scheme, the Domain is REQUIRED. The Class and Type fields Attribute/LID and TransID fields are optional. To provide consistent parsing and decoding, the above described fields are 'positonal' within an MIDS URI. Empty slots after the Domain/Class/Type need not be present.

10.1.1 Domain: event.hikvision.com

Domain:event.hikvison.com：

Class	Type	Attribute/LID ('Local ID')	TransID (Transaction ID)
VideoMotion	motion	video input port/dynamical video input port	
	motionStart		
	motionStop		
Intrusion	alarmIn	alarm In port	

10.1.2 Domain: log.hikvision.com

Domain:log.hikvison.com：

Class	Type	Attribute/LID ('Local ID')	TransID (Transaction ID)
Alarm	alarmIn	alarm in port	
	alarmOut	alarm out port	
	motionStart	video input port	
	motionStop	video input port	
	hideStart	video input port	
	hideStop	video input port	
	vcaStart	video input port	
	vcaStop	video input port	
	lineDetectionStart	video input port	
	lineDetectionStop	video input port	
	fieldDetectionStart	video input port	
	fieldDetectionStop	video input port	

	audioInputExceptionStart	video input port	
	audioInputExceptionStop	video input port	
	soundIntensityMutationStart	video input port	
	soundIntensityMutationStop	video input port	
	faceDetectionStart	video input port	
	faceDetectionStop	video input port	
	defocusDetectionStart	video input port	
	defocusDetectionStop	video input port	
	sceneChangeDetectionStart	video input port	
	sceneChangeDetectionStop	video input port	
	regionEntranceStart	video input port	
	regionEntranceStop	video input port	
	regionExitingStart	video input port	
	regionExitingStop	video input port	
	loiteringStart	video input port	
	loiteringStop	video input port	
	groupStart	video input port	
	groupStop	video input port	
	rapidMoveStart	video input port	
	rapidMoveStop	video input port	
	parkingStart	video input port	
	parkingStop	video input port	
	unattendedBaggageStart	video input port	
	unattendedBaggageStop	video input port	
	attendedBaggageStart	video input port	
	attendedBaggageStop	video input port	
	vehicleDetectionStart	video input port	
	vehicleDetectionStop	video input port	
	pirStart		
	pirStop		
Exception	videoLost	video input port	
	videoException	video input port	
	videoFormatMismatch	video input port	
	illlegealAccess		
	hdError		
	hdFull		
	netBroken		
	recordError	video input port	
	ipcDisconnect	video input port	
	ipcIpConfilict	video input port	
	ipConflict	video input port	
	poePowerException	video input port	
	syncIPCPasswd		
	ezvizOffline		
	uploadDataCsException		
Operation	devicePowerOn		

	devicePowerOff		
	deviceRecycle		
	stopAbnormal		
	localLogin		
	localLogOut		
	localCfgPara		
	localUpdate		
	localStartRec		
	localStopRec		
	localCtrlPtz		
	localLockFile		
	localUnlockFile		
	localManulAlarm		
	localFormatDisk		
	localAddIpc		
	localDelIpc		
	localSetIpc		
	localPlayByFile		
	localPlayByTime		
	localDownloadCfgFile		
	localUploadCfgFile		
	localDownloadRecFile		
	localDownloadPicFile		
	localAddNas		
	localDelNas		
	localSetNas		
	localConfRebRaid		
	localConfSpareRaid		
	localAddRaid		
	locaDelRaid		
	localMigRaid		
	localRebRaid		
	localQuickConfRaid		
	localAddVd		
	localDelVd		
	localStartPicRec		
	localStopPicRec		
	localSetSnmp		
	localResetPasswd		
	localTagOperation		
	localLock		
	localUnlock		
	localHdTest		
	localDownloadHeatMapFile		
	localDownloadCountingFile		
	remotePowerOff		

	remotePowerRecycle		
	remoteLogin		
	remoteLogout		
	remoteCfgPara		
	remoteUpgrade		
	remoteStartRec		
	remoteStopRec		
	remoteCtrlPtz		
	remoteLockFile		
	remoteUnlockFile		
	remoteManulAlarm		
	remoteFormatHd		
	remoteAddIpc		
	remoteDelIpc		
	remoteSetIpc		
	remotePlayByFile		
	remotePlayByTime		
	remoteDownloadCfgFile		
	remoteUploadCfgFile		
	remoteDownloadRecFile		
	remoteGetPara		
	remoteGetStatus		
	remoteStartTransChan		
	remoteStopTransChan		
	startVoiceTalk		
	stopVoiceTalk		
	remoteArm		
	remoteDisArm		
	remoteAddNas		
	remoteDelNas		
	remoteSetNas		
	remoteConfRebRaid		
	remoteConfSpareRaid		
	remoteAddRaid		
	remoteDelRaid		
	remoteMigRaid		
	remoteRebRaid		
	remoteQuickConfRaid		
	remoteAddVd		
	remoteDelVd		
	remoteRpVd		
	remoteUpgradeRaid		
	remoteStartPicRec		
	remoteStopPicRec		
	remotePicBackUp		
	remoteSetSnmp		

	remoteTagOperation		
	remoteDelHdisk		
	remoteLoadHdisk		
	remoteUnloadHdisk		
	localExpandVd		
	localStopRaid		
	remoteExpandVd		
	remoteStopRaid		
	remoteEnableCloudStorage		
	remoteDisableCloudStorage		
	remoteCreateCloudStoragePool		
	remoteDeleteCloudStoragePool		
	remoteModCloudStorageParam		
	remoteModCloudStorageVolume		
	remoteDeletePic		
	remoteDeleteRecord		
	remoteSetSIPServer		
	localSetSIPServer		
	localExportBlackWhiteListFile		
	localImportBlackWhiteListFile		
	remoteExportBlackWhiteListFile		
Information	hddInfo		
	smartInfo		
	startRec		
	stopRec		
	delExpiredRec		
	nasInfo		
	raidInfo		
	runStatusInfo		

10.1.3 Domain: recordType.meta.hikvision.com

Domain:recordType.meta.hikvison.com：

Class	Type	Attribute/LID ('Local ID')	TransID (Transaction ID)
Timing			
Motion			
Alarm			
Manual			
motionOrAlarm			
motionAndAlarm			
Smart			

11 Streaming and Playback

RaCM devicea must offer ISAPI compliant streaming services for the playback of recorded media information. For RaCM devices that also offer the ability to serve live streams to clients, and other multimedia consumers, as a proxy server, these streams must also be provided in an ISAPI compliant manner. The requirements for multimedia streaming are specified in the “ISAPI IP Media Device API Specification”. These sections of the IP Media Device specification cover just live streaming. Additionally, a RaCM device has the ability to stream recorded data. Due to these unique functional differences, the following exceptions, qualifications, and additions to the IP Media Device specification

11.1 Streaming URIs

11.1.1 Live Streams

The IP Media Device specification specifies the following URI structure for a client/consumer to initiate and RTSP Streaming session:

rtsp://<addr>:<port>/ISAPI/Streaming/Channels/<id>

This is compatible with channel definitions for RaCM devices as well. All input streams, port or network-based, are mapped to RaCM device ‘channels’ identified by channel IDs. Each track, in its configuration, information also contains the corresponding channel ID for its input stream. So, channel IDs can be obtained by reading track configuration information (see “*/ISAPI/ContentMgmt/record/tracks (Recording Session Configuration)*”).

rtsp://10.1.2.55/ISAPI/Streaming/channels/701

In the above example, a client desires to retrieve an RTSP description of channels 701. Please note that this RTSP URI construct will only work for the RTSP DESCRIBE message/method.

11.1.2 Archive Streams

RaCM devices record multimedia information onto ‘tracks’ which are accessible for RTSP Streaming via track IDs. The URI structure for Streaming this media information is:

rtsp://<addr>:<port>/ISAPI/Streaming/tracks/<id>

The above 2 URI constructs are direct derivatives of the ISAPI REST resource hierarchy for media information and match the RaCM notations described herein. Additionally, a client/consumer that establishes an RTSP session to a RACM device and issues a “DESCRIBE” (see RFC 2326), for a channel or track, only receives an SDP description of the media information for that specific, channel or track, not for an entire “presentation” (see RFC 2326). For the use of time as a parameter in the management of streaming sessions, please see “Time-related streaming” below

11.1.3 Time-related Streaming

For archive streams, there usually is a time component that indicates the desired time range with respect to a track. In the cases where a consumer needs to specify the specific time range associated with a streaming session, URI request line parameters are employed for defining the specific time range. The following parameter tags are used:

- "starttime": This parameter tag indicates that it will be followed by and ISO 8601 timestamp indicating the start time of the media stream the consumer is targeting for description, setup, or playing.
- "endtime": This parameter tag indicates that it will be followed by an ISO 8601 time stamp indicating the ending time the consumer desires to be the termination point for a media stream. This parameter field is optional. If it is not present, the stream is to proceed from the start time until the session is manually terminated, or paused, by the consumer.

The format of the time stamps is ISO 8601 as specified in Section 3.7 of RFC 2326, "Absolute Time.' This format is almost identical to XML 'dateTime' except there are no dashes to separate the fields. Basically, the format is: YYYYMMDD"T"HHmmSS.fraction"Z" where Y=year, M=month, D=day, "T" is the time separator, H=hour, m=minutes, S=seconds, and "Z" is the optional field indicating Zulu (GMT) time. A time stamp example is: "20130526T134258Z" which represents May 26th, 2013 at 1:42.58 PM GMT.

Given the above formatting information, a client using RTSP session management would append the time stamps to the end of its URI (either track or source-based) as a way of designating the target timeframe associated with a streaming session. A track-based example follows:

rtsp://10.1.2.39:554/ISAPI/Streaming/tracks/18?starttime=20130731T092241Z&endtime=20130731T093000Z

In the above example, a requester desires to describe/setup/play a media stream that spans the time range on July 31st, 2013 from 9:22:48 AM GMT to 9:30AM GMT.

The appending of timestamps as parameters to a URI request line is pertinent for all URIs, even those returned by the 'SearchResponse' since a requester may not desire an entire time segment. Please reference RFC 2326 for more details on URI and parameter formats.

11.2 Playback

11.2.1 Use of RTSP

The replay protocol is based on RTSP [RFC 2326].
In addition, we make the following stipulations on the usage of RTSP:

1. Interleaved mode ([RFC 2326] section 10.12) MUST be supported by the RACM DEVICE.
2. Clients should use either interleaved mode or RTP/TCP (if supported) .
3. The RACM DEVICE MAY elect not to send RTCP packets during replay. In typical usage, at least with an ONVIF-aware client, RTCP packets are not required, because usually a reliable transport will be used, and because absolute time information is sent within the stream, making the timing information in RTCP sender reports redundant.

11.2.2 Initiating Playback

Playback is initiated by means of the RTSP PLAY method. For example:

PLAY rtsp://192.168.0.1/path/to/recording RTSP/1.0
CSeq: 123
Session: 12345678
Range: clock=20090615T114900.440Z- Rate-Control: no

Reverse playback is indicated using the Scale header field with a negative value. For example to play in reverse without any data loss a value of –1.0 would be used.

PLAY rtsp://192.168.0.1/path/to/recording RTSP/1.0
CSeq: 123
Session: 12345678
Range: clock=20090615T114900.440Z- Rate-Control: no
Scale: -1.0

11.2.2.1 Range header field

The Range field MUST be expressed using absolute times only; the other formats defined by [RFC 2326] shall not be used.

Either open or closed ranges may be used. In the case of a closed range, the range is increasing (end time later than start time) for forward playback and decreasing for reverse playback. The direction of the range MUST correspond to the value of the Scale header.

In all cases, the first point of the range indicates the starting point for replay. Examples:

PLAY rtsp://192.168.0.1/path/to/recording RTSP/1.0
CSeq: 123
Session: 12345678
Range: clock=20090615T114900.440Z-20090615T115000
Rate-Control: no

PLAY rtsp://192.168.0.1/path/to/recording RTSP/1.0

CSeq: 123
Session: 12345678
Range: clock=20090615T115000.440Z-20090615T114900
Rate-Control: no
Scale: -1.0

11.2.2.2 Rate-Control header field

The Rate-Control field is an ONVIF extension and may be either “yes” or “no”. If this field is not present, “yes” is assumed, and the stream is delivered in real time using standard RTP timing mechanisms. If this field is “no”, the stream is delivered as fast as possible, using only the flow control provided by the transport to limit the delivery rate.

12 /ISAPI/ContentMgmt/InputProxy

URI	/ISAPI/ContentMgmt/InputProxy		Type	Service
Methods	Query String(s)	Inbound Data	Return Result	
Notes	Dynamical input service.			

In this document, expanding the InputProxy and StreamingProxy interfaces to realize the configuration of the source parameters.

Dynamic video channel generates dynamic stream. When the dynamic video channel is deleted, all the related dynamic streams will also be deleted.

12.1 /ISAPI/ContentMgmt/InputProxy/sourceCapability

URI	/ISAPI/ContentMgmt/InputPorxy/sourceCapability		Type	Resource
Function	get the source's capability			
Methods	Query String(s)	Inbound Data	Return Result	
GET		<sourceDescriptor>	<sourceCapability>	
POST		<sourceDescriptor>	<sourceCapability>	
Notes				

XSD File: cmInputProxy.xsd

sourceDescriptor XML Block

```
<sourceDescriptor version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <adminProtocol>
    <!--req, xs:string "HIKVISION, SONY, ISAPI, ONVIF …">
  </adminProtocol>
  <addressingFormatType>
    <!-- req, xs:string, "ipaddress,hostname"-->
  </addressingFormatType>
  <hostName>  <!-- dep, xs:string -->  </hostName>
  <ipAddress>  <!-- dep, xs:string -->  </ipAddress>
  <ipv6Address> <!-- dep, xs:string -->  </ipv6Address>
  <adminPortNo>  <!-- req, xs:integer --> </adminPortNo>
  <userName>  <!-- req, xs:string -->  </userName>
  <password>  <!-- req, xs:string -->  </password>
</sourceDescriptor>
```


sourceCapability XML Block

```
<sourceCapability version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
   <videoInputNums>  <!-- req, xs:integer --> </videoInputNums>
   <audioInputNums>  <!-- opt, xs:integer --> </audioInputNums>
</sourceCapability>
```

12.2 /ISAPI/ContentMgmt/InputProxy/search

URI	/ISAPI/ContentMgmt/InputProxy/search		Type	Resource
Function	serach proxy inputs			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<VideoSourceList>	
Notes	<adminProtocol> <adminPort>			

VideoSourceList XML Block

```
<VideoSourceList version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
   <VideoSourceDescriptor> <!-- opt -->
      <id><!-- req, xs:string;id -->      </id>
      <proxyProtocol> <!--req, xs:string "HIKVISION, SONY, ISAPI, ONVIF, GB28181 …">
</proxyProtocol>
      <addressingFormatType>
         <!-- req, xs:string, "ipaddress,hostname"-->
      </addressingFormatType>
      <hostName> <!-- dep, xs:string -->  </hostName>
      <ipAddress> <!-- dep, xs:string -->  </ipAddress>
      <subnetMask>    <!-- opt, xs:string, subnet mask for IPv4 address -->    </subnetMask>
      <ipv6Address>    <!-- dep, xs:string -->  </ipv6Address>
      <bitMask>    <!-- opt, xs:integer,  bitmask IPv6 address -->   </bitMask>
      <serialNumber>   <!--opt, xs:string -->    </serialNumber>
     <macAddress>      <!--opt, xs:string;  -->  </macAddress>
     <firmwareVersion><!-- opt, req, xs:string -->   </firmwareVersion>
      <managePortNo><!-- opt, xs:integer --> </managePortNo>
      <userName>          <!-- opt, xs:string -->       </userName>
      <password>          <!-- opt, xs:string -->       </password>
      <srcInputPortNums> <!-- req, xs: integer--> </srcInputPortNums>
      <deviceID> <!-- dep, xs:string,depend on procxyProtocal GB28181 --> </deviceID>
   </VideoSourceDescriptor>
<VideoSourceList>
```


12.3 /ISAPI/ContentMgmt/InputProxy/ipcConfig

/ISAPI/ContentMgmt/InputProxy/ipcConfig			**General Resource v2.0**
GET			
Description	Export ipc's configuration data.		
Query	None		
Inbound Data	None		
Success Return	Opaque Data		
PUT			
Description	Import ipc's configuration data.		
Query	None		
Inbound Data	Opaque Data		
Success Return	<ImportIpcCfgError >		
Error Status Code	**statusCode**	**subStatusCode**	**description**
	2	upgrading	device upgrading
	2	noMemory	noMemory
	6	importErrorData	importErrorData
	2	configOperating	device importing or exporting
Notes: Configuration file is device-dependant – it may be binary or any other format. May reboot device after configuration file is applied.			

ImportErrorStatus XML Block

```
<ImportIpcCfgError version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <existError> <!-- req, xs:boolean --> </existError>
  <errorCode>  <!-- opt, xs:string, chanNumReachLimit, configOperating, badDevType，
  badLanguage，importErrorData，importFail> </errorCode>
  <IpcErrorList/> <!-- opt -->
</ImportIpcCfgError>
```

```
<IpcErrorList version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <IpcError> <!-- opt -->
<id>  <!-- req, xs:string -->  </id>
<errorRowNo>  <!-- req, xs:integer -->  </errorRowNo>error number
<errorType>  <!-- req, xs:string, channelNoInvalid、channelNoConflict、channel IP/Domain
invalid、channel IP/Domain conflict、channel IP conflict with local IP、protocolError、
adminPortError、channelError、UserNameInvalid、passwordInvalid、transProtocalError -->
</errorType>
</IpcError>
</IpcErrorList>
```


12.4 ISAPI/ContentMgmt/InputProxy/syncIPCPasswd

URI	/ISAPI/ContentMgmt/InputProxy/syncIPCPasswd		Type	Resource
Function	Sync the IPC's password with NVR			
Methods	Query String(s)	Inbound Data	Return Result	
PUT	NONE	NONE	< ResponseStatus >	
Notes				

12.5 /ISAPI/ContentMgmt/InputProxy/customProtocols

/ISAPI/ContentMgmt/InputProxy/customProtocols	General Resource v2.0
GET	
Description	It is used to get the configurations of customProtocols.
Query	None
Inbound Data	None
Success Return	customProtocolList
PUT	
Description	It is used to set the configurations of customProtocols.
Query	None
Inbound Data	customProtocolList
Success Return	ResponseStatus
Notes:	

customProtocalList XML Block

```
<customProtocolList version="1.0" xmlns:="http://www.isapi.org/ver20/XMLSchema">
   <customProtocol/> <!--opt-->
</customProtocolList>
```

12.5.1 /ISAPI/ContentMgmt/InputProxy/customProtocols/<ID>

/ISAPI/ContentMgmt/InputProxy/customProtocols/ID	General Resource v2.0
GET	

Description	It is used to get the configurations of customProtocols.
Query	None
Inbound Data	None
Success Return	customProtocol
PUT	
Description	It is used to set the configurations of customProtocols.
Query	None
Inbound Data	customProtocol
Success Return	ResponseStatus
Notes:	

customProtocol XML Block

```
<customProtocol    version="1.0" xmlns:="http://www.isapi.org/ver20/XMLSchema">
   <id> <!--req,xs:string --> </id>
   <protocalName> <!-- req,xs:string --> </protocalName>
   <streamingList/> <!-- req --> <streamingList>
</customProtocol>
```

streamingList XML Block

```
<streamingList>
   <streaming> <!-- opt -->
     <id> <!--req,xs:string --> </id>
     <enableStream> <!-- req, xs:boolean --> </enableStream>
     <streamProType> <!-- req , xs:string,RTSP --> </streamProType>
     <streamTransMode> <!-- req , xs:string,UDP, --> </streamTransMode>
     <streamPort> <!-- req , xs:integer --> </streamPort>
     <streamPath> <!-- req , xs:string --> </streamPath>
   </streaming>
</streamingList>
```

12.6 /ISAPI/ContentMgmt/InputProxy/channels

URI	/ISAPI/ContentMgmt/InputProxy/channels	**Type**	Resource
Function	Access dynamical video input channels.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET		None	<InputProxyChannelList>
PUT		<InputProxyChannelList>	<ResponseStatus>
POST		<InputProxyChannel>	<ResponseStatus>
Notes	Dynamical video inputport can be created or deleted.		

InputProxyChannelList XML Block


```
<InputProxyChannelList version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <InputProxyChannel/>  <!-- opt -->
</InputProxyChannelList>
```

12.7 /ISAPI/ContentMgmt/InputProxy/channels/status

URI	/ISAPI/ContentMgmt/InputProxy/channels/status		Type	Resource
Function	Access dynamical video input channels status.			
Methods	Query String(s)	Inbound Data	Return Result	
GET		None	<InputProxyChannelStatusList>	
Notes				

InputProxyChannelStatusList XML Block

```
<InputProxyChannelStatusList version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <InputProxyChannelStatus/> <!-- opt -->
</InputProxyChannelStatusList>
```

12.8 /ISAPI/ContentMgmt/InputProxy/channels/<ID>

URI	/ISAPI/ContentMgmt/InputProxy/channels/*ID*		Type	Resource
Function	Access dynamical input channel properties.			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<InputProxyChannel>	
PUT		<InputProxyChannel>	<ResponseStatus>	
DELETE			<ResponseStatus>	
Notes	<sourceInputPortDescriptor> creates channel by content from the tag. <adminProtocol>protocol of the IPC <adminPort>adminPOrt <srcInputPort> specify inputPort\ <srcLogin> username and password of the channel			

InputProxyChannel XML Block

```
<InputProxyChannel version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <id>  <!-- req, xs:string;id -->  </id>
  <name>  <!-- opt, xs:string>  </name>
  <sourceInputPortDescriptor> <!-- req -->
    <adminProtocol> <!--req, xs:string "HIKVISION, SONY, ISAPI, ONVIF ,GB28181 …">
</adminProtocol>
    <addressingFormatType><!-- req, xs:string, "ipaddress,hostname" When setting does not
resolve GB28181 -->
```



```
    </addressingFormatType>
      <hostName> <!-- dep, xs:string ,When setting does not resolve GB28181 --> </hostName>
      <ipAddress> <!-- dep, xs:string , When setting does not resolve GB28181 -->
      </ipAddress>
      <ipv6Address>     <!-- dep, xs:string , When setting does not resolve GB28181 -->
      </ipv6Address>
      <adminPortNo>    <!-- req, xs:integer --> </adminPortNo>
      <srcInputPort> <!-- req, xs:string; id --> </srcInputPort>
      <userName>          <!-- req, xs:string -->       </userName>
      <password>          <!--_req,wo, xs:string -->       </password>
      <connMode>    <!-- opt, xs:string “plugplay, manual” -->    </connMode>
      <streamType> <!-- opt, xs:string; “auto, tcp, udp”--> </streamType>
      <deviceID> <!-- dep, xs:string --> </deviceID>
    </sourceInputPortDescriptor>
    <enableAnr>      <!-- opt, xs:boolean -->     </enableAnr>
</InputProxyChannel>
```

12.9 /ISAPI/ContentMgmt/InputProxy/channels/<ID>/password

URI	/ISAPI/ContentMgmt/InputProxy/channels/<ID>/password		Type	Resource
Function	set channel password			
Methods	Query String(s)	Inbound Data	Return Result	
PUT		None	<InputProxyChannelPassword>	
Notes	<oldPassword> <newPassword>			

InputProxyChannelPassword XML Block

```
<InputProxyChannelPassword version=“1.0” xmlns=“http://www.isapi.org/ver20/XMLSchema”>
   <oldPassword> <!-- req, xs:string --> </oldPassword>
   <newPassword> <!--req, xs:string --> </newPassword>
</InputProxyChannelPassword>
```

12.10 /ISAPI/ContentMgmt/InputProxy/channels/<ID>/netParam

URI	/ISAPI/ContentMgmt/InputProxy/channels/<ID>/netParam		Type	Resource
Function	set netparam for spec channel			
Methods	Query String(s)	Inbound Data	Return Result	

PUT		None	<DynVideoInputNetParam>
Notes	<ipAddress> <ipv6Address> <managePortNo>		

InputProxyChannelNetParam XML Block

```
<InputProxyChannelNetParam version="1.0"
xmlns="http://www.isapi.org/ver20/XMLSchema">
   <ipAddress>          <!-- opt, xs:string -->   </ipAddress>
   <ipv6Address> <!-- opt, xs:string -->   </ipv6Address>
   <managePortNo>   <!-- opt, xs:integer --> </managePortNo>
</InputProxyChannelNetParam>
```

12.11 /ISAPI/ContentMgmt/InputProxy/channels/<ID> /status

URI	/ISAPI/ContentMgmt/InputProxy/channels/ID/status		Type	Resource
Function	Access dynamical input channels status.			
Methods	Query String(s)	Inbound Data	Return Result	
GET		None	<InputProxyChannelStatusList>	
Notes	<online> the channel online or outline <StreamingProxyChannelIdList> the channel dynamic streaming <relatedIOProxy>channel IO			

InputProxyChannelStatus XML Block

```
<InputProxyChannelStatus version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
   <id> <!-- req, xs:string; id --> </id>
   <sourceInputPortDescriptor>   <!-- req --> </sourceInputPortDescriptor>
   <online> <!-- req, xs:boolean --> </online>
   <supportCreateStream/> <!-- opt, xs:boolean -->
   <streamingProxyChannelIdList> <!-- req -->
      <streamingProxyChannelId> <!-- req, xs:string; id -->   </streamingProxyChannelId>
   </streamingProxyChannelIdList>
   <relatedIOProxy> <!-- opt -->
      <inputProxyPortIdList> <!--opt-->
        <inputProxyPortId/> <!-- opt -->
      </inputProxyPortIdList>
      <outputProxyPortIdList><!-- opt-->
        <outputProxyPortId/> <!-- opt -->
      </outputProxyPortIdList>
   </relatedIOProxy>
   <chanDetectResult>     <!-- opt, xs:string, "connect, overSysBandwidth, domainError,
     ipcStreamFail, connecting, chanNoError, ipAddrConflictWithDev, ipAddrConflicWithIpc,
```



```
    errorUserNameOrPasswd, netUnreachable, unknownError，notExist，
    ipcStreamTypeNotSupport, ipcResolutionNotSupport,
    userLocked,userNotExist,ipcUnregistered, ipcNotActivated, poePortDetecting” -->
     </chanDetectResult>
</InputProxyChannelStatus>
```

12.12 /ISAPI/ContentMgmt/InputProxy/channels/<ID>/video

URI	/ISAPI/ContentMgmt/InputProxy/channels/*ID*/video		Type	Resource
Function	Access the special dynamical video input.			
Methods	Query String(s)	Inbound Data	Return Result	
PUT	None	<VideoInputChannel>	<VideoInputChannel>	
Notes				

12.13 /ISAPI/ContentMgmt/InputProxy/channels/<ID>/video/overlays

URI	/ISAPI/ContentMgmt/InputProxy/channels/*ID/video*/overlays		Type	Resource
Function	Configure and access text and image overlays.			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<VideoOverlay>	
PUT		<VideoOverlay>	<ResponseStatus>	
DELETE			<ResponseStatus>	
Notes	IP media devices can overlay additional information on the encoded video stream. These overlays can be either text information or a set of images. Overlays are composited together in ID-order when displayed in the video.			

VideoOverlay XML Block (refer to IPMD)

12.13.1 ./InputProxy/channels/<ID>/video/overlays/text

URI	/ISAPI/ContentMgmt/InputProxy/channels/*ID*/video/overlays/text		Type	Resource
Function	Access and configure text overlays for a particular video channel.			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<TextOverlayList>	
PUT		<TextOverlayList>	<ResponseStatus>	

POST		<TextOverlay>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	A set of text overlays is managed. They are composited over the video signal in increasing ID-order.		

TextOverlayList XML Block (refer to IPMD)

12.13.2 ./InputProxy/channels/<ID>/video/overlays/text/<ID>

URI	/ISAPI/ContentMgmt/InputProxy/channels/*ID*/video/overlays/text/*ID*	**Type**	Resource
Function	Access and configure a particular text overlay for a video channel.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			TextOverlay>
PUT		<TextOverlay>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	A text overlay can contain time information and static text with color and transparency information.		

TextOverlay XML Block (refer to IPMD)

12.13.3 ./InputProxy/channels/<ID>/video/overlays/image

URI	/ISAPI/ContentMgmt/InputProxy/channels/*ID*/video/overlays/image	**Type**	Resource
Function	Access and configure image overlays for a particular video channel.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<ImageOverlayList>
PUT		<ImageOverlayList>	<ResponseStatus>
POST		<ImageOverlay>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	A set of image overlays is managed. They are composited over the video signal in increasing ID-order.		

ImageOverlayList XML Block (please refer to IPMD)

12.13.4 ./InputProxy/channels/<ID>/video/overlays/image/<ID>

URI	/ISAPI/ContentMgmt/InputProxy/channels/*ID/video*/overlays/image/*ID*	Type	Resource
Function	Access and configure a particular image overlay for a video channel.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<ImageOverlay>
PUT		<ImageOverlay>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	An image overlay can contain time information and static text with color and transparency information. In order to enable image overlay, an image must have been previously uploaded to the device using the /ISAPI/System/Video/overlayImages command.		

ImageOverlay XML Block (please refer to IPMD)

12.14 /ISAPI/ContentMgmt/InputProxy/channels/<ID>/video/privacyMask

URI	/ISAPI/ContentMgmt/InputProxy/channels/*ID*/video/privacyMask	Type	Resource
Function	Access and configure privacy masking.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<PrivacyMask>
PUT		<PrivacyMask>	<ResponseStatus>
Notes	Privacy masking can be enabled and the region list configured per channel.		

PrivacyMask XML Block (please refer to IPMD)

12.14.1 ./InputProxy/channels/<ID>/video/privacyMask/regions

URI	/ISAPI/ContentMgmt/InputProxy/channels/*ID*/Video/privacyMask/regions	Type	Resource
Function	Access and configure privacy mask regions.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<PrivacyMaskRegionList>

PUT		<PrivacyMaskRegionList>	<ResponseStatus>
POST		<PrivacyMaskRegion>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	Privacy masking consists of a set of regions that are combined to grey or black out areas of a video input.		

PrivacyMaskRegionList XML Block (please refer to IPMD)

12.14.2 ./InputProxy/channels/<ID>/video/privacyMask/regions/<ID>

URI	/ISAPI/ContentMgmt/InputProxy/channels/ID/video/privacyMask/regions/ID	**Type**	Resource
Function	Access and configure a particular privacy mask region.		

Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<PrivacyMaskRegion>
PUT		<PrivacyMaskRegion>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	Region coordinates are dependent on video resolution. Regions will be “drawn” from the coordinates provided in a top-down fashion. At least three <RegionCoordinates> blocks must be provided for a single PrivacyMaskRegion> block.		

PrivacyMaskRegion XML Block (please refer to IPMD)

12.15 /ISAPI/ContentMgmt/InputProxy channels/<ID>/video/tamperDetection

URI	/ISAPI/ContentMgmt/InputProxy/channels/ID/video/tamperDetection	**Type**	Service
Function	Tamper detection configuration for a dynamic video input channel.		

Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<TamperDetectionChannel>
PUT		<TamperDetectionChannel>	<ResponseStatus>
Notes	<coordinateCapabilities> coordinate for every regional		

TamperDetectionChannel XML Block (please refer to IPMD)

12.15.1 ./InputProxy/channels/<ID>/video/tamperDetection/regions

URI	/ISAPI/ContentMgmt/InputProxy/channels/*ID*/video/tamperDetection/regions		Type	Resource
Function	Access the list of regions for tamper detection on a particular video input channel.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<TamperDetectionRegionList>	
PUT		<TamperDetectionRegionList>	<ResponseStatus>	
POST		<TamperDetectionRegion>	<ResponseStatus>	
DELETE			<ResponseStatus>	
Notes				

TamperDetectionRegionList XML Block (please refer to IPMD)

12.15.2 ./InputProxy/channels/<ID>/video/tamperDetection/regions/<ID>

URI	/ISAP/Custom/SelfExt/TamperDetection/channels/*ID*/regions/*ID*		Type	Resource
Function	/ISAPI/ContentMgmt/InputProxy/channels/*ID*/video/tamperDetection/region/ID			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<TamperDetectionRegion>	
PUT		<TamperDetectionRegion>	<ResponseStatus>	
DELETE			<ResponseStatus>	
Notes	Region coordinates are dependent on video resolution. Regions will be “drawn” from the coordinates provided in a top-down fashion. At least three <RegionCoordinates> blocks must be provided for a single <TamperDetectionRegion> block. Ordering of <TamperDectectionRegion> blocks is insignificant.			

TamperDetectionRegion XML Block (please refer to IPMD)

12.16 /ISAPI/ContentMgmt/InputProxy /channels/<ID>/video/motionDetection

URI	/ISAPI/ContentMgmt/InputProxy/channels/ID/video/motionDetection	Type	Service
Function	Tamper detection configuration for a dynamic video input channel.		

Methods	Query String(s)	Inbound Data	Return Result
GET			<**MotionDetection**>
PUT		<**MotionDetection**>	<ResponseStatus>
Notes			

MotionDetection XML Block (please refer to IPMD)

12.16.1 ./InputProxy/channels/<ID>/video/motionDetection/layout

URI	/ISAPI/ContentMgmt/InputProxy/channels/*ID*/video/motionDetection/layout			Type	Resource
Function	Access the list of regions for tamper detection on a particular video input channel.				
Methods	**Query String(s)**	**Inbound Data**	**Return Result**		
GET			< **MotionDetectionGridLayout** >		
PUT		< **MotionDetectionGridLayout** >	<ResponseStatus>		
Notes					

MotionDetectionGridLayout XML Block (please refer to IPMD)

12.16.2 /ISAPI/ContentMgmt/InputProxy/channels/ID/video/smartDetection

When some smart events are detetected or triggered, these devices support to pop up an alarm image, audio alarm, Email alarm, trigger alarm output, trigger recording on a channel or capture a picture, etc.

/ISAPI/ContentMgmt/InputProxy/channels/ID/video/smartDetection	**General Resource v2.0**
GET	
Description	It is used to get a particular smart detection configuration.
Query	None
Inbound Data	None
Success Return	**SmartDetection**
PUT	
Description	It is used to update a particular smart detection configuration.
Query	None
Inbound Data	**SmartDetection**
Success Return	**ResponseStatus**
Notes:	

SmartDetection XML Block


```
<SmartDetection version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <enabled>          <!-- req, xs:boolean -->          </enabled>
</SmartDetection>
```

13 / ISAPI/ContentMgmt/IOProxy

URI	/IISAPI/ContentMgmt/IOProxy		Type	Service
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			< IOProxyPortList>	
Notes	The allocation of IDs between dynamic input and output ports must be unique.			

IOProxyPortList XML Block

```
<IOProxyPortList version="1.0" xmlns= "http://www.isapi.org/ver20/XMLSchema">
  <IOProxyInputPortList/>   <!-- opt -->
  <IOProxyOutputPortList/> <!-- opt -->
</IOProxyPortList>
```

13.1 /ISAPI/ContentMgmt/IOProxy/status

URI	/ISAPI/ContentMgmt/IOProxy/status		Type	Resource
Function	Query the dynamic IO status			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<IOProxyPortStatusList>	
Notes	The allocation of IDs between dynamic input and output ports must be unique.			

IOProxyPortStatusList XML Block (please refer to IPMD)

13.2 / ISAPI/ContentMgmt/IOProxy/inputs

URI	/ISAPI/ContentMgmt/IOProxy/inputs		Type	Resource
Function	Access dynamic input ports			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<hik:IOProxyInputPortList>	
PUT		<hik:IOProxyInputPortList>	<ResponseStatus>	

POST		<hik:IOProxyInputPort>	<ResponseStatus>
Notes	The resource is or not support POST method,the device support create IO.		

IOProxyInputPortList XML Block

```
<IOProxyInputPortList version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
   <IOProxyInputPort/>        <!-- opt -->
</IOProxyInputPortList>
```

13.3 / ISAPI/ContentMgmt/IOProxy/inputs/ID

URI	/ISAPI/ContentMgmt/IOProxy/inputs/ID	**Type**	Resource
Function	Access the particular dynamic input port		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<hik:IOProxyInputPort>
PUT		<hik:IOProxyInputPort>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes			

IOInputPort XML Block

```
<IOProxyInputPort version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
   <id>   <!-- req, xs:string -->   </id>
   <enabled> <!--req, xs:boolean--> </enabled>
   <IODescriptor>   <!--req, xs:string -->
   <proxyProtocol>
  <!—req xs:enumeration "HIKVISION, AXIS, PANASONIC, BOSCH, PELCO, SONY…"-->
   </proxyProtocol>
   <userName>   <!—req,wo, xs:string   -->   <userName>
   <password>   <!—req,wo, xs:string   -->   <password>
   <addressingFormatType>     <!-- dep -->
  <!-- req, xs:enumeration, "ipaddress, hostname, …" -->
   </addressingFormatType>
   <hostName>           <!-- dep, xs:string -->       </hostName>
   <ipAddress>          <!-- dep, xs:string -->       </ipAddress>
   <ipv6Address>       <!-- dep, xs:string -->       </ipv6Address>
   <managePortNo>     <!--req, xs:integer -->     </managePortNo>
   <innerIOPortID> <!--req, xs:string; id --> </innerIOPortID>
  <!-- ##any other protocol address-->
   </IODescriptor >
   <triggering>   <!-- req, xs:string, "high,low, rising, falling" -->   </triggering>
  <name> <!-- opt, xs:string --> </name>
</IOProxyInputPort>
```


13.4 / ISAPI/ContentMgmt/IOProxy/inputs/ID/status

URI	/ISAPI/ContentMgmt/IOProxy/input/ID/status		Type	Resource
Function	Query the status of a dynamic input port			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<IOPortStatus>	
Notes				

IOPortStatus XML Block (please refer to IPMD)

13.5 / ISAPI/ContentMgmt /IOProxy/outputs

URI	/ISAPI/ContentMgmt/IOProxy/outputs		Type	Resource
Function	Access dynamic input ports			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<hik:IOProxyOutputPortList>	
PUT		<hik:IOProxyOutputPortList>	<ResponseStatus>	
POST		<hik:IOProxyOutputPort>	<ResponseStatus>	
Notes				

IOProxyOutPortList XML Block

```
<IOProxyOutputPortList version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
   <IOProxyOutputPort/>      <!-- opt -->
</IOProxyOutputPortList>
```

13.6 / ISAPI/ContentMgmt/IOProxy/outputs/ID

URI	/ISAPI/ContentMgmt/IOProxy/outputs/ID		Type	Resource
Function	Access the particular dynamic output port			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<hik:IOProxyOutputPort>	
PUT		<hik:IOProxyOutputPort>	<ResponseStatus>	
DELETE			<ResponseStatus>	
Notes				

IOProxyOutputPort XML Block


```
<IOProxyOutputPort version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <id>  <!-- req, xs:string -->  </id>
  <IODescriptor/>  <!--req, -->
  <PowerOnState>  <!-- req -->
    <defaultState> <!-- req, xs:string, "high,low" --> </defaultState>
    <outputState> <!-- req, xs:string, "high,low,pulse" --> </outputState>
    <pulseDuration> <!-- dep, xs:integer, milliseconds --> </pulseDuration>
  </PowerOnState>
  <name> <!-- opt, xs:string --> </name>
</IOProxyOutputPort>
```

13.7 / ISAPI/ContentMgmt /IOProxy/outputs/ID/trigger

URI	/ISAPI/ContentMgmt/IOProxy/outputs/ID/trigger		Type	Resource
Function	Manually trigger an dynamical output port.			
Methods	Query String(s)	Inbound Data	Return Result	
PUT	outputState pulseDuration	<IOPortData>	<ResponseStatus>	
Notes	Either the inbound data or query string values are used. The IO output port is toggled to a high or low signal accordingly. If the <outputState> refers to pulse, then the <pulseDuration> tag must be provided and the output port will be triggered to the specified state for the duration specified by <pulseDuration>.			

IOPortData XML Block (please refer to IPMD)

13.8 /ISAPI/ContentMgmt/IOProxy/outputs/ID/status

URI	/ISAPI/ContentMgmt/IOProxy/outputs/ID/status		Type	Resource
Function	Query the status of a dynamic output port			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<IOPortStatus>	
Notes	See /ISAPI/ContentMgmt/IOProxy/status for an explanation of the fileds.			

IOPortStatus XML Block (please refer to IPMD)

14 /ISAPI/ContentMgmt/ZeroVideo

URI	/ISAPI/ContentMgmt/ZeroVideo	Type	Service

Methods	Query String(s)	Inbound Data	Return Result
Notes	Zero Video service.		

14.1 /ISAPI/ContentMgmt/ZeroVideo/channels

URI	/ISAPI/ContentMgmt/ZeroVideo/channels	Type	Resource
Function	Access zero video channels.		
Methods	Query String(s)	Inbound Data	Return Result
GET		None	<ZeroVideoChannelList>
Notes	Since zero video input channels are resources that are defined by the hardware configuration of the device, they cannot be created or deleted.		

ZeroVideoChannelList XML Block

```
<ZeroVideoChannelList version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <ZeroVideoChannel/>    <!-- opt -->
</ZeroVideoChannelList>
```

14.2 /ISAPI/ContentMgmt/ZeroVideo/channels/<ID>

URI	/ISAPI/ContentMgmt/ZeroVideo/channels/*ID*	Type	Resource
Function	Access zero video input channel properties.		
Methods	Query String(s)	Inbound Data	Return Result
GET			<ZeroVideoChannel>
PUT		<ZeroVideoChannel>	<ResponseStatus>
Notes	None		

ZeroVideoChannel XML Block

```
<ZeroVideoChannel version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <id>  <!-- req, xs:string;id -->  </id>
  <enabled> <!--req, xs:Boolean --> </eanbled>
  <inputPort>  <!-- req, xs:string, id-->  </inputPort>
  <brightnessLevel>  <!-- opt, xs:integer, 0..100 -->  </brightnessLevel>
  <contrastLevel>  <!-- opt, xs:integer, 0..100 -->  </contrastLevel>
  <sharpnessLevel>  <!-- opt, xs:integer, 0..100 -->  </sharpnessLevel>
  <saturationLevel>  <!-- opt, xs:integer, 0..100 -->  </saturationLevel>
  <hueLevel>  <!-- opt, xs:integer, 0..100 -->  </hueLevel>
<ZeroVideoChannel>
```


14.3 /ISAPI/ContentMgmt/ZeroVideo/channels/<ID>/enlarge

URI	/ISAPI/ContentMgmt/ZeroVideo/channels/ID/enlarge		Type	Resource
Function	Get or set zero chan video input enlarge configuration			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<ZeroVideoEnlarge>	
PUT		<ZeroVideoEnlarge>	<ResponseStatus>	
Notes	mousePosition: the device uses this element to decide which sub screen should be enlarged.			

ZeroVideoEnlarge XML Block

```
<ZeroVideoEnlarge version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <stat> <!--req, xs:string, "normal, enlarge" --> </stat>
  <mousePosition> <!--wr,dep -->
    <x> <!--req, xs:integer --> </x>
    <y> <!--req, xs:integer --> </y>
  </mousePosition>
<ZeroVideoEnlarge>
```

14.4 /ISAPI/ContentMgmt/ZeroVideo/channels/<ID>/switchScr een

URI	/ISAPI/ContentMgmt/ZeroVideo/channels/ID/switchScreen		Type	Resource
Function	Switch screen			
Methods	Query String(s)	Inbound Data	Return Result	
PUT		<ZeroVideoSwitch>	<ResponseStatus>	
Notes				

ZeroVideoSwitch XML Block

```
<ZeroVideoSwitch version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <mode> <!--req, xs:string, "back, next" --> </mode>
<ZeroVideoSwitch>
```


14.5 /ISAPI/ContentMgmt/ZeroVideo/channels/<ID>/previewCfg

URI	/ISAPI/ContentMgmt/ZeroVideo/channels/ID/previewCfg		Type	Resource
Function	Get or set zero chan video input preview configuration			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<ZeroVideoPreview>	
PUT		<ZeroVideoPreview>	<ResponseStatus>	
Notes	screenMode: sub screen nums per screen subScreenOrder: sub screen order. Attribute 'order' represent sub screen order, for example: order="1,3,5,2,4".			

ZeroVideoPreview XML Block

```
<ZeroVideoPreview version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <screenMode> <!--req, xs:integer --> </screenMode>
  <enAudio> <!-- req, xs:boolean --> </enAudio>
  <switchInterval> <!--req, xs:integer, in sec--> <switchInterval>
  <subScreenOrderList> <!-- opt -->
    <subScreenOrder order="">
      <id> <!--xs:string; id --> </id>
      <screenMode> <!--req, xs:integer -->  </screeMode>
    </subScreenOrder>
  <subScreenOrderList>
<ZeroVideoPreview>
```

15 /ISAPI/ContentMgmt/ImageProxy

URI	/ISAPI/ContentMgmt/ImageProxy		Type	Service
Methods	Query String(s)	Inbound Data	Return Result	
Notes	Image Proxy service			

Image Proxy service (please refer to IPMD)

16 /ISAPI/ContentMgmt/SnapshotProxy

URI	/ISAPI/ContentMgmt/SnapshotProxy		Type	Service
Methods	Query String(s)	Inbound Data	Return Result	
Notes	Snapshot Proxy service			

Snapshot Proxy service (please refer to IPMD)

17 /ISAPI/ContentMgmt/PTZCtrlProxy

URI	/ISAPI/ContentMgmt/PTZCtrlProxy		Type	Service
Methods	Query String(s)	Inbound Data	Return Result	
Notes	Ptz Proxy service			

PTZ Control Proxy service (please refer to IPMD PTZ)

18 /ISAPI/ContentMgmt/StreamingProxy

URI	/ISAPI/ContentMgmt/StreamingProxy		Type	Service
Methods	Query String(s)	Inbound Data	Return Result	
Notes	dynamical Streaming service			

Streaming Proxy service (please refer to IPMD Streaming)

19 /ISAPI/ContentMgmt/ZeroStreaming

URI	/ISAPI/ContentMgmt/ZeroStreaming		Type	Service
Methods	Query String(s)	Inbound Data	Return Result	
Notes	Zero Streaming service			

19.1 /ISAPI/ContentMgmt/ZeroStreaming/status

URI	/ISAPI/ContentMgmt/ZeroStreaming/status		Type	Resource
Function	Query the device zero streaming status.			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<ZeroStreamingStatus>	
Notes	This command accesses the status of all devices zero streaming sessions.			

ZeroStreamingStatus XML Block

```
<ZeroStreamingStatus version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <totalStreamingSessions>  <!-- req, xs:integer -->  </totalStreamingSessions>
  <StreamingSessionStatusList/>  <!-- dep, only if there are sessions -->
</ZeroStreamingStatus>
```

19.2 /ISAPI/ContentMgmt/ZerStreaming/channels

URI	/ISAPI/ContentMgmt/ZeroStreaming/channels	Type	Resource
Function	Zero Streaming channels.		

Methods	Query String(s)	Inbound Data	Return Result
GET			<ZeroStreamingChannelList>
PUT		<ZeroStreamingChannelList>	<ResponseStatus>
POST		<ZeroStreamingChannel>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	Zero Streaming channels may be hardwired, or it may be possible to create multiple streaming channels per input if the device supports it. To determine whether it is possibly to dynamically create streaming channels, check the defined HTTP methods in /ISAPI/ContentMgmt/Streaming/channels/description.		

ZeroStreamingChannelList XML Block

```
<ZeroStreamingChannelList version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <ZeroStreamingChannel/>    <!-- opt -->
</ZeroStreamingChannelList>
```

19.3 /ISAPI/ContentMgmt/ZeroStreaming/channels/<ID>

URI	**/ISAPI/ContentMgmt/ZeroStreaming/channels/*ID***		**Type**	Resource
Function	Access zero streaming channels.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<ZeroStreamingChannel>	
PUT		<ZeroStreamingChannel>	<ResponseStatus>	
DELETE			<ResponseStatus>	
Notes	<videoInputChannelID> refers to /ISAPI/ContentMgmt/ZeroVideo/ /channels/*ID*.			

ZeroStreamingChannel XML Block


```
<ZeroStreamingChannel version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <id>   <!-- req, xs:string;id -->      </id>
  <channelName> <!-- req, xs:string --> </channelName>
  <enabled> <!-- req, xs:boolean -->      </enabled>
  <Video>
    <!-- opt -->
    <enabled><!-- req, xs:boolean --></enabled>
    <videoInputChannelID> <!-- req, xs:string;id --> </videoInputChannelID>
    <videoCodecType>
        <!-- req, xs:string, "MPEG4,MJPEG,3GP,H.264,MPNG" -->
    </videoCodecType>
    <videoResolutionWidth>    <!-- req, xs:integer --> </videoResolutionWidth>
    <videoResolutionHeight> <!-- req, xs:integer -->   </videoResolutionHeight>
    <videoQualityControlType>
        <!-- opt, xs:string, "cbr,vbr" -->
     </videoQualityControlType>
    <constantBitRate> <!-- dep, xs:integer, in kbps -->      </constantBitRate>
    <vbrUpperCap>   <!-- dep, xs:integer, in kbps --> </vbrUpperCap>
    <vbrLowerCap>   <!-- dep, xs:integer, in kbps --> </vbrLowerCap>
    <maxFrameRate>      <!-- req, xs:integer, maximum frame rate x100 --> </maxFrameRate>
  </Video>
</ZeroStreamingChannel>
```

19.4 /ISAPI/ContentMgmt/ZeroStreaming/channels/<ID>/statu s

URI	/ISAPI/ContentMgmt/ZeroStreaming/channels/*ID*/status		Type	Resource
Function	Get the list of zero streaming sessions associated with a particular channel.			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<ZeroStreamingSessionStatusList>	
Notes	Use of IPv4 or IPv6 addresses depends on the value of the <ipVersion> field in /ISAPI/System/Network/interfaces/*ID*/ipAddress.			

StreamingSessionStatus XML Block

```
<ZeroStreamingSessionStatusList version="1.0"
xmlns="http://www.isapi.org/ver20/XMLSchema">
  <StreamingSessionStatus/>
</ZeroStreamingSessionStatusList>
```


20 /ISAPI/ContentMgmt/Storage

URI	/ISAPI/ContentMgmt/storage		Type	Service
Function				
Methods	Query String(s)	Inbound Data	Return Result	
GET			<storage>	
Notes				

storage XML Block

```
<storage version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <hddList>  <!-- opt -->  </hddList >
  <nasList>  <!-- opt -->  </nasList >
  <workMode>  <!-- opt, xs:string, "group, quota, extract" --> <workMode>
</storage>
```

20.1 /ISAPI/ContentMgmt/Storage/hdd

URI	/ISAPI/ContentMgmt/Storage/hdd		Type	Resource
Function	Device HDD management			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<hddList>	
Notes				

hddList XML Block

```
<hddList version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <hdd> <!-- opt -->  </hdd>
</hddList>
```

20.2 /ISAPI/ContentMgmt/Storage/hdd/<ID>

URI	/ISAPI/ContentMgmt/Storage/hdd/<ID>		Type	Resource
Function	set specify hdisk			
Methods	Query String(s)	Inbound Data	Return Result	

GET			<hdd>
PUT		<hdd>	<ResponseStatus>
Notes	<property>hdisk attribute: RW, RO and Redund <group>hdisk group		

hdd XML Block

```
<hdd version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <id>  <!-- ro, req, xs:string;id -->  </id>
  <hddName>  <!-- ro, req, xs:string -->  </hddName>
  <hddPath><!-- ro, opt, xs:string -->  </hddPath>
  <hddType>
    <!-- ro, req, xs:string, "IDE,SATA,eSATA, NFS, iSCSI, Virtual Disk", etc -->
  </hddType>
  <status> <!--ro, req, xs:string "ok, unformatted, error, idle, mismatch, offline, smartFailed, reparing, formating, notexist..." --> </status>
  <capacity> <!-- ro, req, xs:float, in MB -->  </capacity>
  <freeSpace>  <!-- ro, req, xs:float, in MB -->  </freeSpace>
  <property> <!--req, xs:string "RW, RO, Redund"--> </property>
  <group> <!-- opt, xs:string; id --> </group>
</hdd>
```

20.3 /ISAPI/ContentMgmt/Storage/hdd/<ID>/format

URI	**/ISAPI/ContentMgmt/Storage/hdd/<ID>/format**	**Type**	Resource
Function	Format specified hdd disk		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
PUT			<hdd>
Notes			

20.4 /ISAPI/ContentMgmt/Storage/hdd/<ID>/formatStatus

URI	**/ISAPI/ContentMgmt/Storage/hdd/<ID>/formatStatus**	**Type**	Resource
Function			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<formatStatus>
Notes			

formatStatus XML Block


```
<formatStatus version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <formating>  <!-- ro, req, xs:boolean --> </formating>
  <percent> <!-- ro, req, xs:integer "0-100" -->  </percent>
</formatStatus>
```

20.5 /ISAPI/ContentMgmt/Storage/hdd/<ID>/SMARTTest/start

URI	/ISAPI/ContentMgmt/Storage/hdd/<ID>/SMARTTest/start		Type	Resource
Function	Start Hard Disk S.M.A.R.T Test.			
Methods	Query String(s)	Inbound Data	Return Result	
GET	None	HDDSMARTTest	ResponseStatus	
Notes	Notes: Request again when enable detection, device returns Device Busy (statusCode), deviceBusy(subStatusCode) error.			

HDDSMARTTest XML Block

```
<HDDSMARTTest version="2.0" xmlns="http://www.std-cgi.org/ver20/XMLSchema">
    <testType opt="short,expanded,conveyance"> <!-- opt, xs:string, ""--></testType>
</HDDSMARTTest>
```

20.6 /ISAPI/ContentMgmt/Storage/hdd/<ID>/SMARTTest/status

URI	/ISAPI/ContentMgmt/Storage/hdd/<ID>/SMARTTest/status		Type	Resource
Function	Get disk S.M.A.R.T detection information·			
Methods	Query String(s)	Inbound Data	Return Result	
GET	None	None	SMARTTestStatus	
Notes	Notes: The ID in "/hdd/ID"is defined as following declaration: 1: disk1			

SMARTTestStatus XML Block

```
<SMARTTestStatus version="2.0" xmlns="http://www.std-cgi.org/ver20/XMLSchema">
    <id> <!-- req, xs:string, --> </id>
    <temprature> <!-- req, xs:integer, degree--> </temprature>
    <powerOnDay> <!-- req, xs:integer, days --> </powerOnDay>
    <selfEvaluaingStatus opt="ok,error"> <!-- req, xs:string, ""--> </selfEvaluaingStatus>
    <allEvaluaingStatus opt="functional,badsectors,fault "> <!-- req, xs:string, "" -->
</allEvaluaingStatus>
    <selfTestPercent> <!-- req, xs: integer, --> </selfTestPercent>
    <selfTestStatus
opt="ok,aborted,interrupted,failed,unkown,electronic_element_error,servo_error,read_fail
```



```
ed,progress,not_tested,not_recognized"> <!-- req, xs:string, “"--> </selfTestStatus>
    <testType opt="short,expanded,conveyance"> <!-- req, xs:string,”” --> </testType>
    <TestResultList> <!-- req, "Size=30"-->
        <TestResult> <!-- req-->
            <attributeID> <!-- req, xs:string, --> </attributeID>
            <attributeName> <!-- req, xs:string, --> </ attributeName>
            <status opt="ok,illegal"> <!-- req, xs:string, “”--> </status>
            <flags> <!-- req, xs:integer, --> </flags>
            <thresholds> <!-- req, xs: integer, --> </thresholds>
            <value> <!-- req, xs: string, --> </value>
            <worst> <!-- req, xs: integer, --> </worst>
            <rawValue> <!-- req, xs: integer, --> </rawValue>
        </TestResult>
    </TestResultList>
</SMARTTestStatus>
```

20.7 /ISAPI/ContentMgmt/Storage/hdd/SMARTTest/config

/ISAPI/ContentMgmt/Storage/hdd/SMARTTest/config	**General Resource v2.0**
GET	
Description	Get disk S.M.A.R.T detection configuration
Query	None
Inbound Data	None
Success Return	**SMARTTestConfig**
PUT	
Description	Set disk S.M.A.R.T detection configuration
Query	None
Inbound Data	**SMARTTestConfig**
Success Return	**ResponseStatus**
Notes: The ID in “**/hdd/*ID***”is defined as following declaration: 1:disk1	

SMARTTestConfig XML Block

```
<SMARTTestConfig version="2.0" xmlns="http://www.std-cgi.org/ver20/XMLSchema">
    <enable opt="true,false"> <!-- req, xs:string when self-assessment of HDD doesn’t pass, whether to continue using the HDD--> </enable>
</SMARTTestConfig>
```

20.8 /ISAPI/ContentMgmt/Storage/hdd/<ID>/BadSectorsTest/start

/ISAPI/ContentMgmt/Storage/hdd/<ID>/BadSectorsTest/start	**General Resource v2.0**
PUT	

Description	Enable HDD bad block check
Query	None
Inbound Data	**BadSectorsTest**
Success Return	**ResponseStatus**
Notes:	

BadSectorsTest XML Block

```
<BadSectorsTest version="2.0" xmlns="http://www.std-cgi.org/ver20/XMLSchema">
    <testType opt="full,metadata"> <!-- opt, xs:string, “”--></testType>
</BadSectorsTest>
```

20.9 /ISAPI/ContentMgmt/Storage/hdd/<ID>/BadSectorsTest/status

/ISAPI/ContentMgmt/Storage/hdd/<ID>/BadSectorsTest/status	**General Resource**	**v2.0**
GET		
Description	Get HDD bad block check information	
Query	None	
Inbound Data	None	
Success Return	**BadSectorsTestStatus**	
Notes: The ID in “**/hdd/*ID***”is defined as following declaration: 1:disk1		

BadSectorsTestStatus XML Block

```
<BadSectorsTestStatus version="2.0"
xmlns="http://www.std-cgi.org/ver20/XMLSchema">
    <diskID> <!-- req, xs:string --> </diskID>
    <MaskAreaList> <!-- req-->
        <MaskArea> <!-- req -->
            <maskAreaID> <!-- req, xs:integer --> </maskAreaID>
            <startLBA> <!-- opt, xs:integer --> </startLBA>
            <endLBA> <!-- opt, xs:integer --> </endLBA>
        </MaskArea>
    </MaskAreaList>
    <BlockAreaTestStatus>   <!-- opt-->
        <testType opt="full,metadata"> <!-- opt, xs:string, “”--></testType>
        <testStatus opt="none,running,pause,complete,exceed,abort"> <!-- req,
xs:string,"" --> </testStatus>
        <fistBlock> <!-- opt, xs:integer, --> </fistBlock>
        <lastBlock> <!-- opt, xs:integer, --> </lastBlock>
        <currentBlock> <!-- opt, xs:integer, --> </currentBlock>
        <BadSectorsList> <!-- opt-->
            <BadSectors> <!-- req -->
                <id> <!-- req, xs:integer --> </id>
```



```
                    <block> <!-- opt, xs:integer --> </block>
                </BadSectors >
            <BadSectorsList>
        </BlockAreaTestStatus>
</BadSectorsTestStatus>
```

20.10 /ISAPI/ContentMgmt/Storage/hdd/<ID>/BadSectorsTest/pause

/ISAPI/ContentMgmt/Storage/hdd/<ID>/BadSectorsTest/pause	General Resource	v2.0
PUT		
Description	Suspend HDD bad block check	
Query	None	
Inbound Data	**None**	
Success Return	**ResponseStatus**	
Notes:		

20.11 /ISAPI/ContentMgmt/Storage/hdd/<ID>/BadSectorsTest/resume

/ISAPI/ContentMgmt/Storage/hdd/<ID>/BadSectorsTest/resume	General Resource	v2.0
PUT		
Description	Resume HDD bad block check	
Query	None	
Inbound Data	**None**	
Success Return	**ResponseStatus**	
Notes:		

20.12 /ISAPI/ContentMgmt/Storage/hdd/<ID>/BadSectorsTest/stop

/ISAPI/ContentMgmt/Storage/hdd/<ID>/BadSectorsTest/stop	General Resource	v2.0
PUT		
Description	Stop HDD bad block check	
Query	None	
Inbound Data	**None**	
Success Return	**ResponseStatus**	
Notes:		

20.13 /ISAPI/ContentMgmt/Storage/nas

URI	/ISAPI/ContentMgmt/Storage/nas		Type	Resource
Function				
Methods	Query String(s)	Inbound Data	Return Result	
GET			<nasList>	
PUT		<nasList>	<ResponseStatus>	
POST		<nas>	<ResponseStatus>	
Notes				

nasList XML Block

```
<nasList version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
   <nas> <!-- opt -->     </nas>
</nasList>
```

20.14 /ISAPI/ContentMgmt/Storage/nas/<ID>

URI	/ISAPI/ContentMgmt/Storage/nas/<*ID*>		Type	Resource
Function				
Methods	Query String(s)	Inbound Data	Return Result	
GET			<nas>	
PUT		<nas>	<ResponseStatus>	
DELETE			<ResponseStatus>	
Notes	<nasType>support NFS，iSCSI。 <property> attribute :RW, RO and RDD <group>			

nas XML Block

```
<nas version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
   <id>    <!-- req, xs:string; id -->      </id>
   <addressingFormatType>
      <!-- req, xs:string, "ipaddress,hostname"-->
   </addressingFormatType>
   <hostName>     <!-- dep, xs:string -->   </hostName>
   <ipAddress>     <!-- dep, xs:string -->   </ipAddress>
   <ipv6Address> <!-- dep, xs:string -->   </ipv6Address>
   <portNo>   <!-- req, xs:integer --> </portNo>
```



```
    <userName> <!--opt, xs:string --><userName>
    <password><!--opt, xs:string --></password>
    <nasType> <!--req, xs:string, “NFS, iSCSI …” -->    </nasType>
    <path> <!--req, xs:string -->    </path>
    <status> <!--ro, req, xs:string “ok, unformatted, error, idle, mismatch, offline, smartFailed,
reparing, formating, notexist...” --> </status>
    <capacity> <!-- ro, req, xs:float, in MB -->    </capacity>
    <freeSpace>    <!-- ro, req, xs:float, in MB -->    </freeSpace>
    <property> <!--req, xs:string “RW, RO, RDD”--> </property>
    <group> <!-- opt, xs:string; id --> </group>
</nas>
```

20.15 /ISAPI/ContentMgmt/Storage/nas/<ID>/format

URI	/ISAPI/ContentMgmt/nas/<ID>/format		Type	Resource
Function	Remote format specified NFS/ISCSI			
Methods	Query String(s)	Inbound Data	Return Result	
PUT			<hdd>	
Notes				

20.16 /ISAPI/ContentMgmt/Storage/nas/<ID>/formatStatus

URI	/ISAPI/ContentMgmt/Storage/nas/<ID>/formatStatus		Type	Resource
Function	Get format process			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<formatStatus>	
Notes				

formatStatus XML Block(please refer to Chapter 20.4)

20.17 /ISAPI/ContentMgmt/Storage/nas/search

URI	/ISAPI/ContentMgmt/Storage/nas/search		Type	Resource
Function	Get nas directoryList from a specify nas server.			
Methods	Query String(s)	Inbound Data	Return Result	
GET		<nasServerDescriptor>	<nasDirectoryList>	
POST		<nasServerDescriptor>	<nasDirectoryList>	

Notes	

nasServerDescriptor XML Block

```
<nasServerDescriptor version="1.0" xmlns=" http://www.isapi.org/ver20/XMLSchema-xsd">
   <nasType> <!--req, xs:string "NFS, iSCSI…" -->   </nasType>
   <nasServerIP> <!-- req, xs:string -->   </nasServerIP>
   <nasServerPort>     <!-- opt, xs:integer --> </nasServerPort>
</nasServerDescriptor>
```

nasDirectoryList XML Block

```
<nasDirectoryList version="1.0" xmlns=" http://www.isapi.org/ver20/XMLSchema-xsd">
   <nasDirectory> <!-- opt -->
      <id> <!-- req, xs:integet --> </id>
      <directory>  <!-- req, xs:string --> </directory>
   </nasDirectory>
</nasDirectoryList>
```

20.18 /ISAPI/ContentMgmt/Storage/quota

URI	/ISAPI/ContentMgmt/Storage/quota		Type	Resource
Function	Manage disk quota			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<diskQuotaList>	
Notes				

diskQuotaList XML Block

```
<diskQuotaList version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
   <diskQuota>   <!-- opt --> </diskQuota>
</diskQuotaList>
```

20.19 /ISAPI/ContentMgmt/Storage/quota/<ID>

URI	/ISAPI/ContentMgmt/Storage/quota/<ID>		Type	Resource
Function	Manage specify channel quota			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<diskQuota>	
PUT		<diskQuota>	<ResponseStatus>	
Notes				

diskQuota XML Block

```
<diskQuota version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <id>  <!-- req, xs:integer; channel --> </id>
  <useVideoQuota> <!-- ro, integer, MB --> </useVideoQuota>
  <usePictureQuota> <!-- ro, integer, MB --> </usePictureQuota>
  <totalDiskVolume> <!-- ro, integer, MB --> </totalDiskVolume>
  <videoQuota> <!-- req, integer, MB --> </videoQuota>
  <pictureQuota> <!-- opt, integer, MB --> </pictureQuota>
  <type> <!-- opt, xs:string,"volume,ratio" --> </type>  // absent-ratio; otherwise-volume
  <videoQuotaRatio> <!-- dep, integer, 0...100 percentage--> </videoQuotaRatio>
  <pictureQuotaRatio> <!-- dep, 0...100 percentage--> </pictureQuotaRatio>
  <totalVideoVolume> <!-- ro, integer, MB --> </useVideoQuota>
  <totalPictureVolume> <!-- ro, integer, MB --> </usePictureQuota>
  <freeVideoQuota> <!-- ro, integer, MB --> </useVideoQuota>
  <freePictureQuota> <!-- ro, integer, MB --> </usePictureQuota>
</diskQuota>
```

20.20 /ISAPI/ContentMgmt/Storage/extract

URI	/ISAPI/ContentMgmt/Storage/extract		Type	Resource
Function	Manage specify channel extract			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<diskExtract>	
PUT		<diskExtract>	<ResponseStatus>	
Notes				

diskExtract XML Block

```
<diskExtract version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <nomalVideoPercent> <!-- req, integer, 0-100--> </nomalVideoPercent>
  <extractVideoPercent> <!-- req, integer, 0-100 --> </extractVideoPercent>
  <picturePercent> <!-- opt, integer, 0-100 --> </picturePercent>
</diskExtract>
```

20.21 /ISAPI/ContentMgmt/Storage/diskGroup

URI	/ISAPI/ContentMgmt/Storage/diskGroup		Type	Resource
Function				
Methods	Query String(s)	Inbound Data	Return Result	
GET			<diskGroupList>	
Notes				

diskGroupList XML Block

```
<diskGroupList version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <diskGroup>  <!-- opt -->  </diskGroup>
</diskGroupList>
```

20.22 /ISAPI/ContentMgmt/Storage/diskGroup/<ID>

URI	/ISAPI/ContentMgmt/Storage/diskGroup/<ID>		Type	Resource
Function				
Methods	Query String(s)	Inbound Data	Return Result	
GET			<diskGroup>	
PUT		<diskGroup>	<ResponseStatus>	
Notes				

diskGroup XML Block

```
<diskGroup version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <id>  <!-- req, xs:string; id -->  </id>
  <videoInputChannelIDList>  <!-- req -->
    <videoInputChannelID> <!-- opt, xs:string; id --> <videoInputChannelID>
  </videoInputChannelIDList>
</diskGroup>
```

20.23 /ISAPI/ContentMgmt/Storage/extension

URI	/ISAPI/ContentMgmt/Storage/extension		Type	Resource
Function	Get or set the configuration information of network extension			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<storageExtension>	
PUT		<storageExtension>	<ResponseStatus>	
Notes				

BondList XML Block

```
<storageExtension version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <LoopEnable>  <!-- opt, xs: boolean -->  <LoopEnable>
  <enableDormant> <!-- opt,xs:boolean -->  </enableDormant>
  <packDuration> <!--opt, xs:int, unit is minute> </packDuration>
</storageExtension>
```


20.24 /ISAPI/ContentMgmt/Storage/cloud/URL?type=OneDrive

/ISAPI/ContentMgmt/Storage/cloud/URL?type=OneDrive	General Resource v2.0
GET	
Description	
Query	type
Inbound Data	None
Success Return	CloudURL
Notes: cloud type：OneDrive ,GoogleDrive, DropBox get URL through cloud type, user apply authCode through the return URL	

CloudURL XML Block

```
<CloudURL>
    <url min="0" max="256"><!--req, xs:string --></url>
</CloudURL>
```

20.25 /ISAPI/ContentMgmt/Storage/cloud/URL/capabilities

/ISAPI/ContentMgmt/Storage/cloud/URL/capabilities	General Resource v2.0
GET	
Description	It is used to get cloud's URL capability.
Query	None
Inbound Data	None
Success Return	<CloudURL>
Notes: get cloud's URL capability	

20.26 /ISAPI/ContentMgmt/Storage/cloud

/ISAPI/ContentMgmt/Storage/cloud	General Resource v2.0
GET	
Description	
Query	None
Inbound Data	None
Success Return	**<**Cloud**>**
PUT	
Description	
Query	None
Inbound Data	**<**Cloud**>**
Success Return	**ResponseStatus**
Notes: the interface is applicable to get and set cloud paramter	


```
<type> cloud type：OneDrive ，GoogleDrive， DropBox
<status> cloud status，online，offline
<authCode> authcode
<alias> cloud account alias，read only
<tolalCapacity>cloud tolalCapacity，read only
<usedSpace>cloud usedSpace，read only
```

Cloud XML Block

```
<Cloud version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
<enable> <!-- req, xs:boolean --> </enable>
<type opt="OneDrive, GoogleDrive ,DropBox"><!-- req, xs:string --> </type>
<status opt="Online, Offline"> <!-- req, xs: string,ro --> </status>
<authCode min="0" max="64"> <!-- req, xs: string --> </authCode>
<alias min="0" max="32"><!--ro, xs: string></alias>
<tolalCapacity><!--ro, xs:integer(64bit), tolalCapacity（MB） --></tolalCapacity>
<usedSpace><!--ro,xs:integer(64bit), usedSpace（MB） --></usedSpace>
</Cloud>
```

20.27 /ISAPI/ContentMgmt/Storage/cloud/capabilities

/ISAPI/ContentMgmt/Storage/cloud/capabilities	General Resource v2.0
GET	
Description	It is used to get cloud capability.
Query	None
Inbound Data	None
Success Return	<Cloud>
Notes:	

20.28 /ISAPI/ContentMgmt/Storage/cloud/channels/ID/uploadStrategy

/ISAPI/ContentMgmt/Storage/cloud/channels/ID/uploadStrategy	General Resource v2.0
GET	
Description	
Query	None
Inbound Data	None
Success Return	**UploadStrategy**
PUT	
Description	
Query	None
Inbound Data	**UploadStrategy**

Success Return	ResponseStatus
Notes: cloud upload strategy configure <strategyType> strategy type：record，picture <Record>it is used when strategyType is record type <recordType>record type：all-all event，motion-motion detection，alarm-signal alarm，VCA-intelligent video	

UploadStrategy XML Block

```
<UploadStrategy>
    <strategyType opt="record,picture"/><!--req, xs:string -->
    <Record>
        <allEvent> true<!--dep, xs:boolean depend strategyType --></allEvent>
        <motion> true<!--dep, xs:boolean depend strategyType --></motion>
        <alarm> true<!--dep, xs:boolean depend strategyType --></alarm>
        <VCA> true<!--dep, xs:boolean depend strategyType --></VCA>
    </Record>
</UploadStrategy>
```

20.29 /ISAPI/ContentMgmt/Storage/cloud/channels/ID/uploadStrategy/capabilities

/ISAPI/ContentMgmt/Storage/cloud/channels/ID/uploadStrategy/capabilities	General Resource	v2.0
GET		
Description	It is used to get cloud uploadStrategy capabilities.	
Query	None	
Inbound Data	None	
Success Return	<UploadStrategy>	
Notes: get cloud uploadStrategy capabilities		

20.30 /ISAPI/ContentMgmt/FlashStorage/remove/channels/<ID>

/ISAPI/ContentMgmt/FlashStorage/remove/channels/<ID>	General Resource v2.0
PUT	
Description	Remove Flash Storage PDC Data
Query	None
Inbound Data	FlashStorageRemove
Success Return	**ResponseStatus**
Notes: <ID>:Video Channel。	

FlashStorageRemove XML Block

```
<FlashStorageRemove version="2.0"
xmlns="http://www.hikvision.com/ver20/XMLSchema">
    <coutingRemove><!--opt,xs:boolen, --></coutingRemove>
</FlashStorageRemove>
```

21 / ISAPI/ContentMgmt/download

URI	/ISAPI/ContentMgmt/download		Type	Resource
Function	Down load a special record segment			
Methods	Query String(s)	Inbound Data	Return Result	
GET		<downloadRequest >	Record data	
PUT		< downloadRequest>	<ResponseStatus>	
Notes	Playback URI is returned by the search service. In the url, there may be some information about the name or size of the segment. For example, rtsp://<host>/Streaming/tracks/<id>?name=track1segment1&size=1024B			

downloadRequest XML Block

```
<downloadRequest version="1.0" xmlns="http://"http://www.isapi.org/ver20/XMLSchema">
  <playbackURI> <!—req, xs:string --> </playbackURI>
</downloadRequest>
```


22 /ISAPI/ContentMgmt/channels/ID/cloudStorage

/ISAPI/ContentMgmt/channels/ID/cloudStorage	General Resource v2.0
GET	
Description	CloudStorage configuration for all video input channels.
Query	None
Inbound Data	None
Success Return	**CloudStorageList**
PUT	
Description	CloudStorage configuration for all video input channels.
Query	None
Inbound Data	**CloudStorageList**
Success Return	**ResponseStatus**
Notes:	

CloudStorageList XML Block

```
<CloudStorageList version="2.0" xmlns=http://www.isapi.org/ver20/XMLSchema size="">
   <CloudStorage/>     <!-- opt -->
</CloudStorageList>
```

22.1 /ISAPI/ContentMgmt/channels/ID/cloudStorage/ID/capabilities

/ISAPI/ContentMgmt/channels/ID/cloudStorage/ID/capabilities	General Resource v2.0
GET	
Description	It is used to get CloudStorage capability.
Query	None
Inbound Data	None
Success Return	<CloudStorage>
Notes:	

CloudStorage XML Block

```
<CloudStorage version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
      <id><!-- req, xs:string-->     </id>
      <enable> <!-- req, xs:boolean --> </enable>
      <addressingFormatType opt="ipaddress,hostname"> <!-- req, xs:string, "ipaddress"
--> </addressingFormatType>
      <hostName> <!-- dep, xs:string --> </hostName>
      <ipAddress> <!-- dep, xs:string --> </ipAddress>
      <ipv6Address> <!-- dep, xs:string --> </ipv6Address>
```



```
    <port min="" max=""> <!-- req, xs:integer --> </port>
    <userName min="" max=""> <!-- req, xs:string --> </userName>
    <password> <!-- req,wo, xs:string --> </password>
    <postPoolID> <!-- opt, xs:integer --> </postPoolID>
    <illegalPoolID> <!-- opt, xs:integer --> </illegalPoolID>
    <vehicleDetectionID> <!-- opt, xs:integer --> </vehicleDetectionID>
</CloudStorage>
```

22.2 /ISAPI/ContentMgmt/channels/ID/cloudStorage/ID

/ISAPI/ContentMgmt/channels/ID/cloudStorage/ID	General Resource v2.0
GET	
Description	CloudStorage configuration for all video input channels.
Query	None
Inbound Data	None
Success Return	**<**CloudStorage**>**
PUT	
Description	CloudStorage configuration for all video input channels.
Query	None
Inbound Data	**<**CloudStorage**>**
Success Return	**ResponseStatus**
Notes:	

CloudStorage XML Block

```
<CloudStorage version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
   <id>   <!-- req, xs:string -->   </id>
   <enable> <!-- req, xs:boolean --> </enable>
   <addressingFormatType opt=" ipaddress,hostname"> <!-- req, xs:string, "ipaddress" -->
</addressingFormatType>
   <hostName> <!-- dep, xs:string --> </hostName>
   <ipAddress> <!-- dep, xs:string --> </ipAddress>
   <ipv6Address> <!-- dep, xs:string --> </ipv6Address>
   <port min="" max=""> <!-- req, xs:integer --> </port>
   <userName> <!-- req, xs:string --> </userName>
   <password> <!-- req,wo, xs:string --> </password>
   <postPoolID> <!-- opt, xs:integer --> </postPoolID>
   <illegalPoolID> <!-- opt, xs:integer --> </illegalPoolID>
   <vehicleDetectionID> <!-- opt, xs:integer --> </vehicleDetectionID>
</CloudStorage>
```

22.3 /ISAPI/ContentMgmt/channels/ID/cloudStorage/test

/ISAPI/ContentMgmt/channels/ID/cloudStorage/test	General Resource v2.0
GET	
Description	It is used to test the CloudStorage server available or not
Query	None
Inbound Data	**CloudStorageTestDescription**

Success Return	**CloudStorageTestResult**
POST	
Description	It is used to test the CloudStorage server available or not
Query	None
Inbound Data	**CloudStorageTestDescription**
Success Return	**CloudStorageTestResult**
Notes: <errorDescription> ok,connect server fail,no nas directory,no permission,no dns,no gateway,password error,exchange server fail,create directory failed,no write permission	

CloudStorageTestDescription XML Block

```
<CloudStorageTestDescription version="2.0"
xmlns="http://www.isapi.org/ver20/XMLSchema">
    <addressingFormatType>
    <!-- req, xs:string, “ipaddress,hostname”-->
    </addressingFormatType>
    <hostName> <!-- dep, xs:string --> </hostName>
    <ipAddress> <!-- dep, xs:string --> </ipAddress>
    <ipv6Address> <!-- dep, xs:string --> </ipv6Address>
    <portNo> <!-- req, xs:integer --> </portNo>
    <username> <!-- dep, xs:string --> </username>
    <password> <!-- dep, xs:string --> </password>
    <postPoolID> <!-- opt, xs:integer --> </postPoolID>
    <illegalPoolID> <!-- opt, xs:integer --> </illegalPoolID>
    <vehicleDetectionID> <!-- opt, xs:integer --> </vehicleDetectionID>
</CloudStorageTestDescription>
```

CloudStorageTestResult XML Block

```
<CloudStorageTestResult version=”2.0” xmlns=”http://www.isapi.org/ver20/XMLSchema”>
  <errorDescription> <!-- req, xs:string -->.</errorDescription>
</CloudStorageTestResult>
```

23 /ISAPI/ContentMgmt/channels/ID/liteStorage

/ISAPI/ContentMgmt/channels/ID/liteStorage	General Resource v2.0
GET	
Description	LiteStorage configuration for all video input channels.
Query	None
Inbound Data	None
Success Return	**<**LiteStorage**>**
PUT	
Description	LiteStorage configuration for all video input channels.
Query	None

Inbound Data	<LiteStorage>
Success Return	ResponseStatus
Notes:	

LiteStorage XML Block

```
<LiteStorage version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
   <id>    <!-- req, xs:string-->      </id>
   <enabled> <!-- req, xs:boolean --> </enabled>
   <capacity>  <!-- ro, opt, xs:float ,GB-s->  </capacity>
   <storageTime min="1" max="30"> <!-- req, xs:integer --> </storageTime>
   <level><!--opt, xs:string, low,medium,high--></level>
   <DefStorageTime><!--dep, "storage time(days)"-->
      <low><!-- ro, opt, xs:integer, low mode days--></low>
      <medium><!-- ro, opt, xs:integer, medium mode days--></medium>
      <high><!-- ro, opt, xs:integer, high mode days--></high>
   </DefStorageTime>
</LiteStorage>
```

23.1 /ISAPI/ContentMgmt/channels/ID/liteStorage/capabilities

/ISAPI/ContentMgmt/channels/ID/liteStorage/ID/capabilities	General Resource	v2.0
GET		
Description	It is used to get LiteStorage capability.	
Query	None	
Inbound Data	None	
Success Return	<LiteStorage>	
Notes:		

LiteStorage XML Block

```
<LiteStorage version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
      <id><!-- req, xs:string-->      </id>
      <enabled> <!-- req, xs:boolean --> </enabled>
      <capacity>    <!-- ro, opt, xs:float,GB --> </capacity>
      <storageTime> <!-- req, xs:integer,"1…30"--> </storageTime>
      <level opt="low,medium,high" def="medium"><!--req, xs:string, --></level>
      <DefStorageTime><!--dep, "storage time(days)"-->
            <low min="" max=""><!-- ro, opt, xs:integer, low mode days--></low>
            <medium min="" max=""><!-- ro, opt, xs:integer, medium mode days--></medium>
            <high min="" max=""><!-- ro, opt, xs:integer, high mode days--></high>
      </DefStorageTime>
</LiteStorage>
```

24 /ISAPI/ContentMgmt /workmode

/ISAPI/ContentMgmt /workmode	General Resource	v2.0

GET	
Description	
Query	None
Inbound Data	None
Success Return	**WorkMode**
PUT	
Description	
Query	None
Inbound Data	**WorkMode**
Success Return	**ResponseStatus**
Notes:	

WorkMode XML Block

```
<WorkMode version="2.0" xmlns="http://www.std-cgi.org/ver20/XMLSchema">
   <type opt="normal,N+1,cloud,GB28181">   <!-- req, xs:string -->   </type>
</WorkMode>
```

24.1 /ISAPI/ContentMgmt/workmode/capabilities

/ISAPI/ContentMgmt/workmode/capabilities	General Resource v2.0
GET	
Description	It is used to get work mode capability.
Query	None
Inbound Data	None
Success Return	**WorkMode**
Notes:	

WorkMode XML Block

```
<WorkMode version="2.0" xmlns="http://www.std-cgi.org/ver20/XMLSchema">
   <type opt="normal,N+1,cloud,GB28181">   <!-- req, xs:string -->   </type>
</WorkMode>
```

25 /ISAPI/ContentMgmt/defaultPwdRemain

/ISAPI/ContentMgmt /defaultPwdRemain	General Resource v2.0
PUT	
Description	
Query	None
Inbound Data	**LoginInfo**
Success Return	**ResponseStatus**
Notes:	

LoginInfo XML Block


```
<LoginInfo version="2.0" xmlns="http://www.std-cgi.org/ver20/XMLSchema">
   <userName> <!-- req, xs:string --> </userName>
   <password> <!-- req, xs:string --> </password>
</LoginInfo>
```


26 Appendix A: Codec Type Dictionary

The ‘Codec Tag’s below represent the literal ASCII strings employed to identify the specific codec standards listed below.

Codec Tag (Literal)	Codec Tag Description
Audio Codecs	
G.711	ITU G.711 (PCM) audio codec format (a-/u-law determined by SDP or XML
G.711a	ITU G.711 (PCM) audio codec format; a-law encoding
G.711u	ITU G.711 (PCM) audio codec format; u-law encoding
G.726	ITU G.726 (ADPCM) audio codec format (bitrate advertised by SDP or XML)
G.723.1	ITU G.723.1 audio codec format
G.728	ITU G.728 audio codec format

G.722, G.722.1, G.722.2

ITU G.722/.1/.2 audio codec formats (SB-ADPCM)

G.728 ITU G.728 (LD-CELP) audio codec format G.729, G.729.1 ITU G.729/.1 (CS-ACELP) audio codec format MP3 MPEG-1/Layer 3 audio codec format

AAC	MPEG-2/4 Advanced Audio Codec format
Video Codecs	
MPEG4-SP	ISO/IEC 14496-2 MPEG-4 Simple Profile
MPEG4-ASP	ISO/IEC 14496-2 MPEG-4 Advanced Simple Profile
MPEG4-MP	ISO/IEC 14496-2 MPEG-4 Main Profile

H.264-BP ISO/IEC 14496-10/ITU H.264 Baseline Profile H.264-MP ISO/IEC 14496-10/ITU H.264 Main Profile H.264-HP ISO/IEC 14496-10/ITU H.264 High Profile

H.264SVC-BP	ISO/IEC 14496-10/ITU H.264 Scalable Video Codec (SVC), Baseline Profile encoding (Must read s-props/p-props and SDP for embedded stream info)
H.264SVC-MP	ISO/IEC 14496-10/ITU H.264 Scalable Video Codec (SVC), Main Profile encoding (Must read s-props/p-props and SDP for embedded stream info)
MPEG2-MP	ISO/IEC 13818 MPEG-2 Main Profile

MJPEG Motion version (multi-frame) of ISO/IEC JPEG video encoding (see below) JPEG ISO/IEC 10918 JPEG video encoding JPEG2000 ISO/IEC 154

