Physical Security Interoperability Alliance
IP Media Device API Specification

Verision 1.0

Revision 4.4

2010-04

http://www.hikvision.com/

Revision History	Description	Date
Version 1.0 Revision 1	Initial version	2009-6
Version 1.0 Revision 2	Finished the Mandatory services	2009-8
Version 1.0 Revision 3	Corrections, expanded services	2009-10
Version 1.0 Revision 4	Corrections, updates services and resources	2009-11
Version 1.0 Revision 4.1	Correct some resources and descriptions	2010-01
Version 1.0 Revision 4.2	Add a detailed description of the resolution, amend the DDNS related resources	2010-01
Version 1.0 Revision 4.3	Update resolution description	2010-02
Version 1.0 Revision 4.4	The <Zeroconf> tag is supported in the block XML of "/System/Network/interfaces /ID/discovery". The <pulseDuration> tag is supported in the block XML of "IO/outputs/ID". The <pulseDuration> tag is not supported in the block XML of "/IO/outputs/ID/trigger". Modify some parameter values in Audio Service. The <enabled> can be configured in the <Audio> of the block XML "/Streaming /channels/ID"	2010-04

HIKVISION

http://www.hikvision.com/

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.

This documentation is issued in strict confidence and is to be used only for the purposes for which it is supplied. It may not be reproduced in whole or in part, in any form, or by any means or be used for any other purpose without prior written consent of HIKVISION and then only on the condition that this notice is included in any such reproduction. No information as to the contents or subject matter of this documentation, or any part thereof, or arising directly or indirectly therefrom, shall be given orally or in writing or shall be communicated in any manner whatsoever to any third party being an individual, firm, or company or any employee thereof without the prior written consent of HIKVISION. Use of this product is subject to acceptance of the HIKVISION agreement required to use this product. HIKVISION reserves the right to make changes to its products as circumstances may warrant, without notice.

This documentation is provided “as-is,” without warranty of any kind. Please send any comments regarding the documentation to:
overseabusiness@hikvision.com

Find out more about HIKVISION at www.hikvision.com

Contents

1 Overview

The Physical Security Interoperability Alliance specifies an interface that enables physical security and video management systems to communicate with various IP media devices in a standardized way. This eliminates the need for device driver customization in order to achieve interoperability among products from different manufacturers. The intent of the specification is to improve the interoperability of IP-based physical security products from different vendors. As the member of the PSIA, Hikvision joined actively and was compliant with the definition of the specification. This document base on the PSIA-IPMD-V1 and expand some definitions as some features optimized by Hikvision.
This document references the PSIA Service Model so it is suggested that the Service Model be required reading before attempting to implement this specification. The Service Model please refer to the Physical Security Interoperability Alliance Service Model Version 1.0 Revision 1.2 19 February 2009

About PSIA
The Physical Security Interoperability Alliance is a global consortium of nearly 50 physical security manufacturers and integrators that are focused on promoting interoperability of IPenabled security devices across every segment of the security industry. Any information about PSIA please visit http://www.psialliance.org/

With regard to Media Streaming, please refer to “develop API of RTSP protocol”.

2 Problem Definition

Security and/or network management applications require the ability to change configurations and control the behaviors of IP media devices － cameras, encoders, decoders, recorders, etc. This functionality can be achieved by sending a standard HTTP(S) request to the unit. The scope of this specification is to define all HTTP(S) application programming interfaces (APIs) for media devices and their functionality; namely, for setting/retrieving various configurations, and controlling device behaviors.

3 Conformance

This document conforms to the PSIA Service model, which describes the methods used for service discovery and introspection. The mandatory service and resources requirements defined by this model are implied in addition to any requirements defined herein.

The following requirement services are the base service for the PSIA specification.And the optional services are the special definition for the Hikvision IPMD.

3.1 Service Requirements

The following table describes the service requirements of the PSIA Service Model.

REQ	Service URL	Notes
√	/	
√	/System	
√	/System/Network	
	/System/IO	
	/System/Audio	
	/System/Video	
	/System/Serial	
√	/Security	
	/Security/AAA	
	/Streaming	
	/Custom/MotionDetection	
	/Custom/Event	
	/Custom/HIK/System/Network	
	/Custom/HIK/System/TwowayAudio	
	/Custom/HIK/System/Video	
	/Custom/HIK/System/Serial	

	/Custom/HIK/Security/AAA	
	/Custom/HIK/PTZ	
	/Custom/HIK/ShelterAlarm	

3.2 Resource Requirements

The following resources are required for the implemented services.

3.2.1 Root Service

REQ	Command	GET	PUT	POST	DEL
√	index	√			
√	indexr	√			
√	description	√			
√	capabilities（Not support now）	√			

3.2.2 /System

REQ	Command	GET	PUT	POST	DEL
√	reboot		√		
√	updateFirmware		√		
√	configurationData	√	√		
√	factoryReset		√		
√	deviceInfo	√	√		
√	supportReport（Not support now）	√			
√	status	√			
√	time	√	√		
√	time/localTime	√	√		
√	time/timeZone	√	√		
√	time/ntpServers	√	√	√	√
√	time/ntpServers/<ID>	√	√		√

3.2.3 /System/Network

REQ	Command	GET	PUT	POST	DEL
√	interfaces	√			
√	interfaces/<ID>	√	√		

√	interfaces/<ID>/ipAddress	√	√		
√	interfaces/<ID>/discovery	√	√		

3.2.4 /System/IO

REQ	Command	GET	PUT	POST	DEL
√	status	√			
√	inputs	√	√		
√	inputs/<ID>	√	√		
√	inputs/<ID>/status	√			
√	outputs	√			
√	outputs/<ID>	√	√		
√	outputs/<ID>/trigger		√		
√	outputs/<ID>/status	√			

3.2.5 /System/Audio

REQ	Command	GET	PUT	POST	DEL
√	channels	√			
√	channels/<ID>	√	√		

3.2.6 /System/Video

REQ	Command	GET	PUT	POST	DEL
√	inputs	√			
√	inputs/channels	√			
√	inputs/channels/<ID>	√	√		
√	inputs/channels/<ID>/privacyMask	√	√		
√	inputs/channels/<ID>/privacyMask/regions	√	√	√	√
√	inputs/channels/<ID>/privacyMask/regions/<ID>	√	√		√

3.2.7 /System/Serial

REQ	Command	GET	PUT	POST	DEL
√	ports	√			
√	ports/<ID>	√	√		

√	ports/<ID>/command		√		

3.2.8 /Security

3.2.9 /Security/AAA

REQ	Command	GET	PUT	POST	DEL
√	users	√	√	√	√
√	users/<ID>	√	√		√

3.2.10 /Streaming

REQ	Command	GET	PUT	POST	DEL
√	status	√			
√	channels	√	√	√ ?	√ ?
√	channels/<ID>	√	√		√ ?
√	channels/<ID>/status	√			
√	channels/<ID>/http（Not support now）	√		√	
√	channels/<ID>/picture	√		√	
	channels/<ID>/requestKeyFrame		√		

3.2.11 /Custom/MotionDectection

REQ	Command	GET	PUT	POST	DEL
		√			
	<ID>	√	√		
	<ID>/regions	√	√	√	√
	<ID>/regions/<ID>	√	√		√

3.2.12 /Custom/Event

REQ	Command	GET	PUT	POST	DEL
		√	√		
	triggers	√	√	√	√
	triggers/<ID>	√	√		√

	triggers/<ID>/notifications	√	√	√	√
	triggers/<ID>/notifications/<ID>	√	√		√
	schedule	√	√		
	notification	√	√		
	notification/mailing	√	√	√	√
	notification/mailing/<ID>	√	√		√
	notification/alertStream	√			

3.2.13 /Custom/HIK/System/Network

REQ	Command	GET	PUT	POST	DEL
	interfaces/<ID>/pppoe	√	√		
	interfaces/<ID>/ddns	√	√		

3.2.14 /Custom/HIK/System/TwowayAudio

REQ	Command	GET	PUT	POST	DEL
	receive	√			
	send		√		
	audioActivate		√		
	audioInActivate		√		

3.2.15 /Custom/HIK/System/Video

REQ	Command	GET	PUT	POST	DEL
	inputs/channels/<ID>/osdDatetime	√	√		
	inputs/channels/<ID>/overlays/text	√	√	√	√
	inputs/channels/<ID>/overlays/text/<ID>	√	√		√

3.2.16 /Custom/HIK/System/Serial

REQ	Command	GET	PUT	POST	DEL
	ports/<ID>/transcommopen		√		
	ports/<ID>/transcommclose		√		
	ports/<ID>/transcommsenddata		√		
	ports/<ID>/transcommrecvdata	√			

3.2.17 /Custom/HIK/Security/AAA

REQ	Command	GET	PUT	POST	DEL
√	users	√	√	√	√
√	users/<ID>	√	√		√

3.2.18 /Custom/HIK/PTZ

REQ	Command	GET	PUT	POST	DEL
√	channels	√	√	√	√
√	channels/<ID>	√	√		√
√	channels/<ID>/patrol	√	√	√	√
√	channels/<ID>/patrol/<ID>	√	√		√
√	channels/<ID>/PTZControl		√		

3.2.19 /Custom/HIK/ShelterAlarm

REQ	Command	GET	PUT	POST	DEL
		√			
	<ID>	√	√		

4 Common Data Types

The XML Data Blocks described in this document contains annotations that describe the properties of the field. For a complete definition, see the XML schema definitions.

The following information is inserted into the comments to describe the data carried in the field:

Annotation	Description
req	Required field.
opt	Optional field. For data uploaded to the device, if the field is present but the device does not support it, it should be ignored.
dep	This field is required depending on the value of another field.
ro	Read-only. For XML data that is both read and written to the device, this field is only present in XML returned from the device. If this field is present in XML uploaded to the device, it should be ignored.
wo	Write-only. This field is only present in XML that can be uploaded to the device.

	This field should never be present in data returned from the device. [This is used for uploading passwords].
xs:<type>	A type defined in XML Schema Part 2: Datatypes Second Edition, see http://www.w3.org/TR/xmlschema-2

Note that XML structures that are optional may have required fields. This means that the entire XML block is optional, however if it is present the required fields are mandatory.

4.1 Built-in Types

Type	Description
BaudRate	A positive numerical value indicating the data transmission rate in symbols per second. Value is >=0. Example: 9600
Color	RGB triplet in hexadecimal format (3 bytes) without the preceding "0x". Example: “FF00FF”
Coordinate	A positive numerical value in pixels. A coordinate pair of 0,0 (x,y) indicates the bottom-left corner of the video image. Value is >=0. Maximum value is dependent on video resolution.
FPS	Frame rate multiplied by 100. Example: 2500 [PAL]
ID	ID from service model.
IPv4 Address	Notation is xxx.xxx.xxx.xxx Example: 3.137.217.220
Ipv6 Address	Notation is xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx using CIDR notation. Example: 2001:0db8:85a3:0000:0000:8a2e:0370:7334
MAC	MAC Address Notation is aa:bb:cc:dd:ee:ff with 6 hex bytes.
TTL	A positive numerical value indicating the number of hops (routers) that traffic is permitted to pass through before expiring. Value is >=0.

4.2 ReceiverAddress

```
<ReceiverAddress>
  <addressingFormatType>
    <!-- req, xs:string, “ipaddress,hostname” -->
  </addressingFormatType>
  <hostName>    <!-- dep, xs:string -->     </hostName>
  <ipAddress>     <!-- dep, xs:string -->     </ipAddress>
```

```
   <ipv6Address>   <!-- dep, xs:string -->       </ipv6Address>
   <portNo>      <!-- opt, xs:integer -->      </portNo>
</ReceiverAddress>
```

Notes:

- Depending on the value of <addressingFormatType>, either the <hostName> or the IP address fields will be used to locate the NTP server.
- Use of IPv4 or IPv6 addresses depends on the value of the <ipVersion> field in /System/Network/interfaces/ID/ipAddress.

4.3 TimeBlockList

<TimeBlockList> holds a set of <TimeBlock> XML that define a set of time ranges.

```
<TimeBlockList version=“1.0” xmlns=“urn:psialliance-org”>
   <TimeBlock>
      <dayOfWeek>
         <!-- opt, xs:integer, ISO8601 weekday number, 1=Monday, … -->
      </dayOfWeek>
      <TimeRange>          <!-- opt -->
         <beginTime>      <!-- req, xs:time, ISO8601 time -->   </beginTime>
         <endTime>      <!-- req, xs:time, ISO8601 time -->   </endTime>
      </TimeRange>
      <bitString>      <!-- opt, xs:string, Hour 0..24, 1/0 per hour -->   </bitString>
   </TimeBlock>
</TimeBlockList>
```

Notes:

- If <dayOfWeek> is not present the time block is valid every day. No two <TimeBlock> in the same list should provide the same <dayOfWeek>.
- If the <bitString> tag in is provided, <TimeRange> should not be provided, and vice versa.
- The <bitString> field can be used to reduce the amount of required, transferable XML. The field is a string of 24 bits, where each bit specifies an hour of the day. The left-most bit is hour 0, and the right-most bit is hour 24. A “1” indicates that the specified hour is enabled for event detection and triggering, and a “0” indicates that it is not. Thus, all <bitString> fields must be 24 bits in length.
- Now it only supports one TimeBlock every day and not support the <bitString>.

5 Service Command Details

5.1 /System

URI	/System		Type	Service
Function	System services.			
Methods	Query String(s)	Inbound Data	Return Result	
Notes				

5.1.1 /System/reboot

URI	/System/reboot		Type	Resource
Function	Reboot the device.			
Methods	Query String(s)	Inbound Data	Return Result	
PUT			<ResponseStatus>	
Notes	The <ResponseStatus> XML data is returned before the device proceeds to reboot.			

5.1.2 /System/updateFirmware

URI	/System/updateFirmware		Type	Resource
Function	Update the firmware of the device.			
Methods	Query String(s)	Inbound Data	Return Result	
PUT			<ResponseStatus>	
Notes	After successful completion of this API, the <ResponseStatus> XML data is returned, and the device proceeds to reboot.			

5.1.3 /System/configurationData

URI	/System/configurationData		Type	Resource
Function	The function is used to get or set the configuration data for the device. This is opaque data that can be used to save and restore the device configuration.			
Methods	Query String(s)	Inbound Data	Return Result	
GET			Opaque data	
PUT		Opaque data	<ResponseStatus>	

Notes	Configuration data is device-dependant – it may be binary or any other format. Client may use the HTTP Accept: header field to inform server what formats are expected. May reboot device after configuration data is applied.

5.1.4 /System/factoryReset

URI	/System/factoryReset		**Type**	Resource
Function	This function is used to reset the configuration for the device to the factory default.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
PUT	mode		<ResponseStatus>	
Notes	Two factory reset modes are supported: “full” resets all device parameters and settings to their factory values. “ basic” resets all device parameters and settings except the values in /System/Network and /System/Security. The default mode is “full”. The device may be rebooted after it is reset.			

5.1.5 /System/deviceInfo

URI	/System/deviceInfo		**Type**	Resource
Function	This function is used to get or set device information.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<DeviceInfo>	
PUT		<DeviceInfo>	<ResponseStatus>	
Notes	Some fields are read-only and may not be set. If these fields are present in the inbound XML block, they are ignored. For the <DeviceInfo> uploaded to the device during a PUT operation, all fields are considered optional and any fields that are not present in the inbound XML are not changed on the device. This allows setting of the fields individually without having to load the entire XML block to the device. <deviceDescription> is a description of the device as defined in RFC1213. <deviceLocation> is the location of the device as defined in RFC1213 <systemContact> is the contact information for the device as defined in RFC1213. <systemObjectID> is the System Object Identifier defined in RFC1213.			

DeviceInfo XML Block

```
<DeviceInfo version="1.0" xmlns="urn:psialliance-org">
   <deviceName>          <!-- req, xs:string -->           </deviceName>
   <deviceID>             <!-- req, xs:string -->           </deviceID>
   <deviceDescription>    <!-- opt, xs:string -->       </deviceDescription>
   <deviceLocation>       <!-- opt, xs:string -->       </deviceLocation>
   <systemContact>       <!-- opt, xs:string -->         </systemContact>
     <!-- Note: The following are read-only parameters -->
   <model>                   <!-- ro, req, xs:string -->       </model>
   <serialNumber>         <!-- ro, req, xs:string -->       </serialNumber>
   <macAddress>           <!-- ro, req, xs:string;      -->    </macAddress>
   <firmwareVersion>       <!-- ro, req, xs:string -->       </firmwareVersion>
   <firmwareReleasedDate>   <!-- ro, opt, xs:string -->       </firmwareReleasedDate>
   <logicVersion>         <!-- ro, opt, xs:string -->       </logicVersion>
   <logicReleasedDate>    <!-- ro, opt, xs:string -->       </logicReleasedDate>
   <bootVersion>           <!-- ro, opt, xs:string -->       </bootVersion>
   <bootReleasedDate>       <!-- ro, opt, xs:string -->       </bootReleasedDate>
   <rescueVersion>       <!-- ro, opt, xs:string -->       </rescueVersion>
   <rescueReleasedDate>   <!-- ro, opt, xs:string -->       </rescueReleasedDate>
   <hardwareVersion>       <!-- ro, opt, xs:string -->       </hardwareVersion>
   <systemObjectID>       <!-- ro, opt, xs:string -->       </systemObjectID>
</ DeviceInfo>
```

Notes:

- <deviceID> value range is 1-255;
- <deviceDescription>, <deviceLocation>, <systemContact> is read-only;
- <logicVersion>, <logicReleasedDate>, <rescueVersion>, <rescueReleasedDate>, <systemObjectID> are not support now.

5.1.6 /System/supportReport(Not support now)

URI	/System/supportReport	**Type**	Resource
Function	This function is used to get a compressed archive of support information for the device. The archive must contain at least the device’s current configuration and log files. Other items that might also be packaged include syslog and operating system information, statistics, etc.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			SupportData
Notes	The format of the archive is device-dependent (could be tar, zip, etc.). Use http Accept: header field to inform server what formats are accepted by client.		

5.1.7 /System/status

URI	/System/status	**Type**	Resource
Function	This function is used to get the status of the device.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<DeviceStatus>
Notes	Not all fields of <DeviceStatus> may be present.		

DeviceStatus XML Block

```
<DeviceStatus version="1.0" xmlns="urn:psialliance-org">
   <currentDeviceTime>    <!-- req, xs:datetime -->          </currentDeviceTime>
   <deviceUpTime>         <!-- req, xs:integer, seconds -->    </deviceUpTime>
   <TemperatureList>       <!-- req -->
      <Temperature>
         <tempSensorDescription>    <!-- req, xs:string -->    </tempSensorDescription>
         <temperature>               <!-- req, xs:float -->        </temperature>
      </Temperature>
   </TemperatureList>
   <FanList>               <!-- opt -->
      <Fan>
         <fanDescription>           <!-- req, xs:string -->        </fanDescription>
         <speed>                     <!-- req, xs:integer -->        </speed>
      </Fan>
   </FanList>
   <PressureList>         <!-- opt -->
      <Pressure>
         <pressureSensorDescription>    <!-- req, xs:string --></pressureSensorDescription>
         <pressure>                    <!-- req, xs:int -->    </pressure>
      </Pressure>
   </PressureList>
   <TamperList>           <!-- opt -->
      <Tamper>
         <tamperSensorDescription>      <!-- req, xs:string -->    </tamperSensorDescription>
         <tamper>                      <!-- req, xs:boolean -->    </tamper>
      </Tamper>
   </TamperList>
   <CPUList>               <!-- req -->
      <CPU>
         <cpuDescription>    <!-- req, xs:string -->               </cpuDescription>
         <cpuUtilization>    <!-- req, xs:integer, percentage 0..100 -->    </cpuUtilization>
      </CPU>
   </CPUList>
   <MemoryList>          <!-- req -->
      <Memory>
         <memoryDescription>   <!-- req, xs:string -->          </memoryDescription>
         <memoryUsage>          <!-- req, xs:float, in MB -->   </memoryUsage>
         <memoryAvailable>       <!-- req, xs:float, in MB-->   </memoryAvailable>
      </Memory>
   </MemoryList>
   <openFileHandles>       <!-- opt, xs:integer -->   </openFileHandles>
</DeviceStatus>
```

Notes:

<TemperatureList>, <FanList>, <PressureList>, <TamperList>, <openFileHandles> are not support now.

5.1.8 /System/time

URI	/System/time	**Type**	Resource
Function	Access the device time information.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<Time>
PUT	timeMode localTime timeZone	<Time>	<ResponseStatus>
Notes	If the “localTime” query string with a value is specified, the <Time> XML block is not required as inbound data. If <timeMode> is set to “local” the <localTime> and <timeZone> fields are required. The <LocalTime> block sets the device time. If <timeMode> is set to “NTP”, only the <timeZone> field is required. The device time is set by synchronizing with NTP.		

Time XML Block

```
<Time version=“1.0” xmlns=“urn:psialliance-org”>
   <timeMode>     <!-- req, xs:string, “NTP, manual” -->       </timeMode>
   <localTime>       <!-- req, xs:datetime -->                </localTime>
   <timeZone>     <!-- req, xs:string, POSIX time zone string; see below -->       </timeZone>
</Time>
```

5.1.9 /System/time/localTime

URI	/System/time/localTime	**Type**	Resource
Function	Access the device local time information.		

Methods	Query String(s)	Inbound Data	Return Result
GET			ISO 8601 Date-Time String
PUT		ISO 8601 Date-Time String	<ResponseStatus>
Notes	An ISO 8601 Date/Time string is accepted and returned. If the date/time value has a time zone, the time is converted into the device‟s local time zone. If the device time mode is set to “ntp” setting this value has no effect.		

5.1.10 /System/time/timeZone

URI	/System/time/timeZone		Type	Resource
Function	Access the device time zone.			
Methods	Query String(s)	Inbound Data	Return Result	
GET			Time zone string	
PUT		Time zone string	<ResponseStatus>	
Notes	Time zones are defined by POSIX 1003.1 section 8.3 time zone notations. Note that the value following the +/- is the amount of time that must be added to the local time to result in UTC. Example: EST+5EDT01:00:00,M3.2.0/02:00:00,M11.1.0/02:00:00 Defines eastern standard time as “EST” with a GMT-5 offset. Daylight savings time is called “EDT”, is one hour later and begins on the second Sunday of March at 2am and ends on the first Sunday of November at 2am. CET-1CEST01:00:00,M3.5.0/02:00:00,M10.5.0/03:00:00 Defines central European time as GMT+1 with a one-hour daylight savings time (“CEST”) that starts on the last Sunday in March at 2am and ends on the last Sunday in October at 3am.			

5.1.11 /System/time/ntpServers

URI	/System/time/ntpServers	Version	1.0	Type	Resource
Function	Access the NTP servers configured for the device.				
Methods	Query String(s)	Inbound Data	Return Result		

GET			<NtpServerList>
PUT		<NtpServerList>	<ResponseStatus>
POST		<NtpServer>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	When the <timeMode> is set to "NTP", the servers in this list are used to synchronize the device‟s system time.		

NtpServerList XML Block

```
<NTPServerList version="1.0" xmlns="urn:psialliance-org">
   <NTPServer> <!—opt -->
</ NTPServerList>
```

Notes:

Now only support one NTPServer.

5.1.12 /System/time/ntpServers/<ID>

URI	/System/time/ntpServers/ID	**Type**	Resource
Function	Access an NTP server configured for the device.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<NtpServer>
PUT		<NtpServer>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	Depending on the value of <addressingFormatType>, either the <hostName> or the IP address fields will be used to locate the NTP server. Use of IPv4 or IPv6 addresses depends on the value of the <ipVersion> field in /System/Network/interfaces/ID/ipAddress.		

NtpServer XML Block

```
<NTPServer version="1.0" xmlns="urn:psialliance-org">
   <id>   <!-- req, xs:string;id -->       </id>
   <addressingFormatType>
      <!-- xs:string, "ipaddress,hostname" -->
   </addressingFormatType>
   <hostName>        <!-- dep, xs:string -->       </hostName>
   <ipAddress>         <!-- dep, xs:string -->       </ipAddress>
   <ipv6Address>      <!-- dep, xs:string -->       </ipv6Address>
   <portNo>          <!-- opt, xs:integer -->       </portNo>
</NTPServer>
```

Notes:

- <id> value can only be 1;
- <portNo> is read-only.

5.2 /System/Network

URI	/System/Network		Type	Service
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
Notes	System network configuration.			

5.2.1 /System/Network/interfaces

URI	/System/Network/interfaces		Type	Resource
Function	Access the device network interfaces.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<NetworkInterfaceList>	
Notes	As hardwired system resources, network interfaces cannot be created or destroyed.			

NetworkInterfaceList XML Block

```
<NetworkInterfaceList version=“1.0” xmlns=“urn:psialliance-org”>
   <NetworkInterface/>    <!-- opt -->
</NetworkInterfaceList>
```

Notes:

Now only support one network interface.

5.2.2 /System/Network/interfaces/<ID>

URI	/System/Network/interfaces/ID		Type	Resource
Function	Access a particular network interface.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<NetworkInterface>	
PUT		<NetworkInterface>	<ResponseStatus>	
Notes				

NetworkInterface XML Block

```
<NetworkInterface version=“1.0” xmlns=“urn:psialliance-org”>
   <id>          <!-- ro, req, xs:string;id -->       </id>
   <IPAddress/>    <!-- req -->
   <Wireless/>       <!-- opt -->
   <IEEE802_1x/>    <!-- opt -->
   <IPFilter/>       <!-- opt -->
   <SNMP/>           <!-- opt -->
   <QoS/>           <!-- opt -->
   <Discovery/>    <!-- opt -->
   <Syslog/>       <!-- opt -->
</NetworkInterface>
```

Notes:

- <id> value can only be 1;
- <Wireless>, <IEEE802_1x>, <IPFilter>, <SNMP>, <QoS>, <Syslog> are not support now.

5.2.3 /System/Network/interfaces/<ID>/ipAddress

URI	/System/Network/interfaces/ID/ipAddress		**Type**	Resource
Function				
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<IPAddress>	
PUT		<IPAddress>	<ResponseStatus>	
Notes	If <addressingType> is dynamic, fields below it need not be provided. If <addressingType> is dynamic, a DHCP client is used for the device. If <addressingType> is static the device IP address is configured manually and the gateway and DNS fields are optional. If <addressingType> refers to APIPA, the device IP address is automatically configured without DHCP. In this case the gateway and DNS fields are optional. Use of <ipAddress> or <ipv6Address> in fields is dictated by the <ipVersion> field. If <ipVersion> is "v4" the <ipAddress> fields are used; if <ipVersion> is "v6" the <ipv6Address> fields are used. <subnetMask> notation is "xxx.xxx.xxx.xxx". <IPV6Address> is "xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx" using CIDR notation.			

IPAddress XML Block

```
<IPAddress version=“1.0” xmlns=“urn:psialliance-org”>
   <ipVersion>          <!-- req, xs:string, “v4,v6” -->    </ipVersion>
   <addressingType>    <!-- req, xs:string, “static,dynamic,apipa” -->    </addressingType>
   <ipAddress>          <!-- dep, xs:string -->                          </ipAddress>
   <subnetMask>       <!-- dep, xs:string, subnet mask for IPv4 address -->    </subnetMask>
   <ipv6Address>       <!-- dep, xs:string -->                          </ipv6Address>
   <bitMask>          <!-- dep, xs:integer,     bitmask IPv6 address -->    </bitMask>
   <DefaultGateway>    <!-- dep -->
      <ipAddress>       <!-- dep, xs:string -->       </ipAddress>
      <ipv6Address>    <!-- dep, xs:string -->       </ipv6Address>
   </DefaultGateway>
   <PrimaryDNS>       <!-- dep -->
      <ipAddress>       <!-- dep, xs:string -->       </ipAddress>
```

```
    <ipv6Address>   <!-- dep, xs:string -->       </ipv6Address>
  </PrimaryDNS>
  <SecondaryDNS>     <!-- dep -->
    <ipAddress>      <!-- dep, xs:string -->       </ipAddress>
    <ipv6Address>   <!-- dep, xs:string -->       </ipv6Address>
  </SecondaryDNS>
</IPAddress>
```

Notes：

- <ipVersion> only support v4 now.
- <addressingType>only support static, dynamic now.
- <SecondaryDNS>not support now

5.2.4 /System/Network/interfaces/<ID>/discovery

URI	/System/Network/interfaces/ID/discovery		**Type**	Resource
Function	Device discovery settings.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<Discovery>	
PUT		<Discovery>	<ResponseStatus>	
Notes	Use of IPv4 or IPv6 addresses depends on the value of the <ipVersion> field in /System/Network/interfaces/ID/ipAddress. <portNo> is the port number for the multicast discovery address. <ttl> is the time to live for multicast discovery packets.			

Discovery XML Block

```
<Discovery version="1.0" xmlns="urn:psialliance-org">
  <UPnP>              <!-- opt -->
    <enabled>      <!-- req, xs:boolean -->    </enabled>
  </UPnP>
  <Zeroconf>        <!-- opt -->
    <enabled>      <!-- req, xs:boolean -->    </enabled>
  </Zeroconf>
  <MulticastDiscovery>    <!-- opt -->
    <enabled>      <!-- req, xs:boolean -->    </enabled>
    <ipAddress>      <!-- req, xs:string -->    </ipAddress>
    <ipv6Address>   <!-- req, xs:string -->    </ipv6Address>
    <portNo>      <!-- req, xs:integer -->    </portNo>
    <ttl>          <!-- req, xs:integer -->    </ttl>
  </MulticastDiscovery>
</Discovery>
```

Notes：

Support <UPnP> and <Zeroconf> now.

5.2.5 Examples

Example: Getting the Network Settings

```
GET /System/Network/interfaces HTTP/1.1
…
HTTP/1.1 200 OK
Content-Type: application/xml; charset=“UTF-8”
Content-Length: xxx

<?xml version=“1.0” encoding=“UTF-8”?>
<NetworkInterfaceList version=“1.0” xmlns=“urn:psialliance-org”>
   <NetworkInterface>
      <id>1</id>
      <IPAddress>
         <ipVersion>v4</ipVersion>
         <addressingType>static</addressingType>
         <ipAddress>3.137.217.220</ipAddress>
         <subnetMask>255.255.255.0</subnetMask>
         <DefaultGateway>
            <ipAddress>3.137.217.0</ipAddress>
         </DefaultGateway>
         <PrimaryDNS>
            <ipAddress>3.137.218.37</ipAddress>
         </PrimaryDNS>
      </IPAddress>
   </NetworkInterface>
   <Discovery>
      <UPnP>
         <enabled>true</enabled>
      </UPnP>
   </Discovery>
</NetworkInterfaceList
```

Example: Setting the IP Address

```
PUT /System/Network/interfaces/1/ipAddress HTTP/1.1
…
HTTP/1.1 200 OK
Content-Type: application/xml; charset=“UTF-8”
```

```
Content-Length: xxx

<?xml version="1.0" encoding="UTF-8"?>
<IPAddress version="1.0" xmlns="urn:psialliance-org">
    <ipVersion>v4</ipVersion>
    <addressingType>static</addressingType>
    <ipAddress>3.137.217.220</ipAddress>
    <subnetMask>255.255.255.0</subnetMask>
    <DefaultGateway>
        <ipAddress>3.137.217.0</ipAddress>
    </DefaultGateway>
    <PrimaryDNS>
        <ipAddress>3.137.218.37</ipAddress>
    </PrimaryDNS>
</IPAddress>
```

5.3 /System/IO

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

IOPortList XML Block

```
<IOPortList version="1.0" xmlns="urn:psialliance-org">
    <IOInputPortList/>        <!-- opt -->
    <IOOutputPortList/>    <!-- opt -->
</IOPortList>
```

Notes:

Just now only support one Input and one Output. Input ID is 1, and output ID is 2.

5.3.1 /System/IO/status

URI	/System/IO/status	**Type**	Resource
Function			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<IOPortStatusList>
Notes	<ioPortID> refers to /System/IO/inputs/ID or /System/IO/outputs/ID. The port IDs are guaranteed to be unique across input and output ports. <ioState> indicates whether the input port is active or inactive. In most		

	applications, a high signal is considered active.

IOPortStatus XML Block

```
<IOPortStatusList version="1.0" xmlns="urn:psialliance-org">
   <IOPortStatus>      <!-- req -->
      <ioPortID>      <!-- req, xs:string;id -->                          </ioPortID>
      <ioPortType>    <!-- req, xs:string, "input,output" -->           </ioPortType>
      <ioState>       <!-- req, xs:string, "active,inactive" -->       </ioState>
   </IOPortStatus>
</IOPortStatusList>
```

5.3.2 /System/IO/inputs

URI	/System/IO/inputs		**Type**	Resource
Function				
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<IOInputPortList>	
Notes	IO inputs are hardwired, meaning that the inputs are statically allocated by the device and cannot be created or deleted.			

IOInputPortList XML Block

```
<IOInputPortList version="1.0" xmlns="urn:psialliance-org">
   <IOInputPort/>      <!-- opt -->
</IOInputPort>
```

5.3.3 /System/IO/inputs/<ID>

URI	/System/IO/inputs/ID		**Type**	Resource
Function				
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<IOInputPort>	
PUT		<IOInputPort>	<ResponseStatus>	
Notes	<triggeringType> indicates the signal conditions to trigger the input port. Rising/Falling refer to a rising/falling edge of a signal. High/Low will continuously trigger for the duration of the high/low input signal.			

IOInputPort XML Block

```
<IOInputPort version="1.0"    xmlns="urn:psialliance-org">
   <id>           <!-- req, xs:string;id -->                          </id>
   <triggering>    <!-- req, xs:string, “high,low,rising,falling” -->    </triggering>
</IOInputPort>
```

Notes：

<triggering>only support high,low now.

5.3.4 /System/IO/inputs/<ID>/status

URI	/System/IO/inputs/ID/status		Type	Resource
Function				
Methods	Query String(s)	Inbound Data	Return Result	
GET			<IOPortStatus>	
Notes	See /System/IO/status for an explanation of the fields.			

5.3.5 /System/IO/outputs

URI	/System/IO/outputs		Type	Resource
Function				
Methods	Query String(s)	Inbound Data	Return Result	
GET			<IOOutputPortList>	
Notes	IO outputs are hardwired, meaning that the inputs are statically allocated by the device and cannot be created or deleted.			

IOOutputPortList XML Block

```
<IOOutputPortList version=“1.0” xmlns=“urn:psialliance-org”>
   <IOOutputPort/>    <!-- opt -->
</IOOutputPort>
```

5.3.6 /System/IO/outputs/<ID>

URI	/System/IO/outputs/ID		Type	Resource
Function				
Methods	Query String(s)	Inbound Data	Return Result	
GET			<IOOutputPort>	
PUT		<IOOutputPort>	<ResponseStatus>	
Notes	<PowerOnState> defines the output port configuration when the device is powered on. <defaultState> is rhe default output port signal when it is not being triggered.			

	<outputState> is the output port signal when it is being triggered. Pulse will cause the output port to send a signal (opposite of the <defaultState>) for a duration specified by the <pulseDuration> tag. <pulseDuration> is the duration of a pulse output port signal when it is being triggered. It must be provided if the <outputState> is "pulse". <actionMapping> is used in interfaces that allow configuration of "On" and "Off" for "High" and "Low" signals.

IOOutputPort XML Block

```
<IOOutputPort version="1.0" xmlns="urn:psialliance-org">
  <id>             <!-- req, xs:string;id -->              </id>
  <PowerOnState>      <!-- req -->
    <defaultState>    <!-- req, xs:string, "high,low" -->          </defaultState>
    <outputState>      <!-- req, xs:string, "high,low,pulse" -->   </outputState>
    <pulseDuration>   <!-- opt, xs:integer, milliseconds -->       </pulseDuration>
  </PowerOnState>
  <ManualControl>      <!-- opt -->
    <actionMapping>
      <!-- req, xs:string, "high,low": ON maps to high / ON maps to low -->
    </actionMapping>
  </ManualControl>
</IOOutputPort>
```

Notes：

- <defaultState>, <outputState> is read-only;
- <outputState> only support "pulse" now;
- <ManualControl>not support now

5.3.7 /System/IO/outputs/<ID>/trigger

URI	/System/IO/outputs/ID/trigger		**Type**	Resource
Function				
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
PUT	outputState pulseDuration	<IOPortData>	<ResponseStatus>	
Notes	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

```
<IOPortData xmlns="urn:psialliance-org">
  <outputState>      <!-- req, xs:string, "high,low,pulse" -->   </outputState>
  <pulseDuration>   <!-- req, xs:integer, milliseconds -->       </pulseDuration>
</IOPortData>
```

Notes:

- <outputState> only support high, low now;
- <pulseDuration> not support now.

5.3.8 /System/IO/outputs/<ID>/status

URI	/System/IO/outputs/ID/status		Type	Resource
Function	Query the status of an output port.			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<IOPortStatus>	
Notes	See /System/IO/status for an explanation of the fields.			

5.4 /System/Audio

URI	/System/Audio		Type	Service
Methods	Query String(s)	Inbound Data	Return Result	
Notes	Audio service.			

5.4.1 /System/Audio/channels

URI	/System/Audio/channels		Type	Resource
Function				
Methods	Query String(s)	Inbound Data	Return Result	
GET			<AudioChannelList>	
Notes	Since inputs are resources that are defined by the hardware configuration of the device, audio channels cannot be created or deleted. ID numbering or values should be considered arbitrary and device-dependent.			

AudioChannelList XML Block

```
<AudioChannelList version=“1.0” xmlns=“urn:psialliance-org”>
  <AudioChannel/>   <!-- opt -->
</AudioChannelList>
```

5.4.2 /System/Audio/channels/<ID>

URI	/System/Audio/channels/ID		Type	Resource
Function				
Methods	Query String(s)	Inbound Data	Return Result	
GET			<AudioChannel>	
PUT		<AudioChannel>	<ResponseStatus>	
Notes	<audioMode> is the duplex mode for audio transmission between the client and media device. <microphoneSource> indicates whether the device microphone is internal or external. <microphoneVolume> Volume control percentage for device microphone. 0 is mute. <speakerVolume> Volume control percentage for device speaker. 0 is mute.			

AudioChannel XML Block

```
<AudioChannel version=“1.0” xmlns=“urn:psialliance-org”>
  <id>        <!-- req, xs:string -->                    </id>
  <enabled>     <!-- req, xs:boolean -->                   </enabled>
  <audioMode>
    <!-- req, xs:string, “listenonly,talkonly,talkorlisten,talkandlisten” -->
  </audioMode>
  <microphoneEnabled>  <!-- req, xs:boolean -->              </microphoneEnabled>
  <microphoneSource>    <!-- req, xs:string, “internal,external” -->    </microphoneSource>
  <microphoneVolume>    <!-- req, xs:integer, 0..100 -->          </microphoneVolume>
  <speakerEnabled>     <!-- req, xs:boolean -->              </speakerEnabled>
  <speakerVolume>     <!-- req, xs:integer, 0..100 -->          </speakerVolume>
</AudioChannel>
```

Notes：

- <enabled> must be true。
- <audioMode> support “talkonly”, “talkandlisten” now.
- <microphoneSource>only support external now.
- The content for his XML data block is read-only.

5.5 /System/Video

URI	/System/Video		Type	Service
Methods	Query String(s)	Inbound Data	Return Result	
Notes	Video service. Video outputs (i.e. decoding) will be covered in a future IPMD specification.			

5.5.1 /System/Video/inputs

URI	/System/Video/inputs		Type	Resource
Function	Access the video inputs on an IP media device.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<VideoInput>	
Notes	An IP media device may contain a set of video inputs. These inputs are hardwired by the device, meaning that the IDs can be discovered but not created or deleted. ID numbering or values should be considered arbitrary and device-dependent.			

VideoInput XML Block

```
<VideoInput version="1.0" xmlns="urn:psialliance-org">
   <VideoInputChannelList/>    <!-- opt -->
</VideoInput>
```

5.5.2 /System/Video/inputs/channels

URI	/System/Video/inputs/channels		Type	Resource
Function				
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<VideoInputChannelList>	
Notes	Since video input channels are resources that are defined by the hardware configuration of the device, they cannot be created or deleted.			

VideoInputChannelList XML Block

```
<VideoInputChannelList version="1.0" xmlns="urn:psialliance-org">
   <VideoInputChannel/>        <!-- opt -->
</VideoInputChannelList>
```

5.5.3 /System/Video/inputs/channels/<ID>

URI	/System/Video/inputs/channels/ID		Type	Resource
Function	Set video input channel properties.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<VideoInputChannel>	
PUT		<VideoInputChannel>	<ResponseStatus>	

Notes	<powerLineFrequencyMode> is used to adjust/correct video image based on different power frequencies. <whiteBalanceMode> indicates the white balance operational mode. <whiteBalanceLevel> indicates the white balance percentage value when whiteBalanceMode refers to manual 0 is 'cool', 100 is 'hot'. <exposureMode> indicates the exposure operational mode. <exposureTarget> the target exposure for manual or auto-exposure. <exposureAutoMin> minimum exposure when <exposureMode> is set to auto. <exposureAutoMax> maximum exposure when <exposureMode> is set to auto. <GainWindow> defines the coordinates of the window used to determine the auto-gain statistics, if smaller than the entire window. <gainLevel> indicates the gain level percentage value when <exposureMode> refers to Manual. 0 is low gain, 100 is high gain. <irisMode> indicates the iris operational mode. Only applicable for auto-iris lens modules. Override will put lens module into manual mode until the scene changes, at which point operation is switched to the auto mode. <focusMode> indicates the focus operational mode. Only applicable for auto-focus lens modules. Override will put lens module into manual mode until the scene changes, at which point operation is switched to the auto mode. In <DayNightFilter>, <beginTime> and <endTime> are only used if <switchScheduleEnabled> is true.

VideoInputChannel XML Block

```
<VideoInputChannel version=“1.0” xmlns=“urn:psialliance-org”>
  <id>              <!-- req, xs:string -->           </id>
  <inputPort>           <!-- req, xs:string -->           </inputPort>
  <powerLineFrequencyMode>        <!--    opt,    xs:string    “50hz,    60hz”    -->
  </powerLineFrequencyMode>
  <whiteBalanceMode>
    <!-- opt, xs:string,
      “manual,auto,indoor/incandescent,fluorescent/white,
      fluorescent/yellow,outdoor,black&white”
    -->
  </whiteBalanceMode>
  <whiteBalanceLevel>    <!-- opt, xs:integer, 0..100 -->       </whiteBalanceLevel>
  <exposureMode>         <!-- opt, xs:string, “manual, auto” -->   </exposureMode>
  <Exposure>            <!-- opt -->
    <exposureTarget>     <!-- req, xs:integer, microseconds -->     </exposureTarget>
    <exposureAutoMin>     <!-- req, xs:integer, microseconds -->     </exposureAutoMin>
    <exposureAutoMax>     <!-- req, xs:integer, microseconds -->     </exposureAutoMax>
  </Exposure>
  <GainWindow>            <!-- opt -->
```

```
    <RegionCoordinatesList>    <!-- opt -->
      <RegionCoordinates>    <!-- opt -->
        <positionX>          <!-- req, xs:integer;coordinate -->    </positionX>
        <positionY>          <!-- req, xs:integer;coordinate -->    </positionY>
      </RegionCoordinates>
    </RegionCoordinatesList>
  </GainWindow>
  <gainLevel>                 <!-- dep, xs:integer, 0..100 -->       </gainLevel>
  <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>
  <gammaCorrectionEnabled>    <!-- opt, xs:boolean -->              </gammaCorrectionEnabled>
  <gammaCorrectionLevel>       <!-- opt, xs:integer, 0..100    -->       </gammaCorrectionLevel>
  <WDREnabled>              <!-- opt, xs:boolean -->              </WDREnabled>
  <WDRLevel>                <!-- opt, xs:integer, 0..100 -->       </WDRLevel>
  <LensList>                <!-- opt -->
    <Lens>                  <!-- opt -->
      <lensModuleName>    <!-- opt, xs:string -->              </lensModuleName>
      <irisMode>
        <!-- opt, xs:string, “manual,auto,override” -->
      </irisMode>
      <focusMode>
        <!-- opt, xs:string, “manual,auto,autobackfocus,override” -->
      </focusMode>
    </Lens>
  </LensList>
  <DayNightFilter>          <!-- opt -->
    <dayNightFilterType>
      <!-- opt, xs:string, “day,night,auto” -->
    </dayNightFilterType>
    <switchScheduleEnabled><!-- opt, xs:boolean -->              </switchScheduleEnabled>
    <beginTime>             <!-- dep, xs:time -->                </beginTime>
    <endTime>             <!-- dep, xs:time -->                </endTime>
  </DayNightFilter>
<VideoInputChannel>
```

Notes：

- <inputPort>, <whiteBalanceLevel>, <exposureMode>, <Exposure>, <GainWindow>, <sharpnessLevel>, <hueLevel>, <gammaCorrectionEnabled>, <gammaCorrectionLevel>, <WDREnabled>, <WDRLevel>, <LensList>and <switchScheduleEnabled>, <beginTime>, <endTime> in <DayNightFilter>not support now;
- <whiteBalanceMode>only support manual, auto, indoor/incandescent now

5.5.4 /System/Video/inputs/channels/<ID>/privacyMask

URI	/System/Video/inputs/channels/ID/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

```
<PrivacyMask version=“1.0” xmlns=“urn:psialliance-org”>
   <enabled>                <!-- req, xs:boolean -->          </enabled>
   <PrivacyMaskRegionList/>    <!-- opt -->
</PrivacyMask>
```

5.5.5 /System/Video/inputs/channels/<ID>/privacyMask/regions

URI	/System/Video/inputs/channels/ID/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

```
<PrivacyMaskRegionList version=“1.0”    xmlns=“urn:psialliance-org”>
   <PrivacyMaskRegion/>    <!-- opt -->
</PrivacyMaskRegionList>
```

Notes：

PrivacyMask support up to 4 regions now.

5.5.6 /System/Video/inputs/channels/<ID>/privacyMask/regions/<ID>

URI	/System/Video/inputs/channels/ID/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. Ordering of <PrivacyMaskRegion> blocks is insignificant			

PrivacyMaskRegion XML Block

```
<PrivacyMaskRegion version=“1.0” xmlns=“urn:psialliance-org”>
  <id>                  <!-- req, xs:string -->               </id>
  <enabled>              <!-- req, xs:boolean -->              </enabled>
  <RegionCoordinatesList>   <!-- req -->
    <RegionCoordinates>   <!-- req, at least one if list is defined -->
      <positionX>         <!-- req, xs:integer;coordinate -->      </positionX>
      <positionY>         <!-- req, xs:integer;coordinate   -->     </positionY>
    </RegionCoordinates>
  </RegionCoordinatesList>
</PrivacyMaskRegion>
```

Notes:

- <id> value range is 1-4;
- <enabled>not support now.
- Only support the rectangular region which will be “drawn” from four coordinates. The four points is clockwise direction, and the beginning point is the low-left point.

5.6 /System/Serial

URI	/System/Serial		**Type**	Service
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
Notes	Serial line service.			

5.6.1 /System/Serial/ports

URI	/System/Serial/ports		Type	Resource
Function	List of serial ports supported by the device.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<SerialPortList>	
Notes	Since serial ports are resources that are defined by the hardware configuration of the device, they cannot be created or deleted.			

SerialPortList XML Block

```
<SerialPortList version=“1.0” xmlns=“urn:psialliance-org”>
   <SerialPort/>    <!-- opt -->
</SerialPortList>
```

5.6.2 /System/Serial/ports/<ID>

URI	/System/Serial/ports/ID		Type	Resource
Function	Serial port			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<SerialPort>	
PUT		<SerialPort>	<ResponseStatus>	
Notes	Access to the serial port parameters. <serialPortType> set the type of port; RS232, RS485, etc. <direction> indicates whether the port is bidirectional. <duplexMode> indicates whether the serial port runs in full or half duplex mode.			

SerialPort XML Block

```
<SerialPort version=“1.0” xmlns=“urn:psialliance-org”>
   <id>              <!-- req, xs:string -->                         </id>
   <enabled>           <!-- req, xs:boolean -->                         </enabled>
   <serialPortType>    <!-- req, xs:string, “RS485,RS422,RS232” -->          </serialPortType>
   <duplexMode>       <!-- req, xs:string, “half,full” -->                </duplexMode>
   <direction>          <!-- req, xs:string, “monodirectional,bdirectional” --> </direction>
   <baudRate>         <!-- req, xs:integer -->                          </baudRate>
   <dataBits>         <!-- req, xs:integer -->                          </dataBits>
   <parityType>       <!-- req, xs:string, “none,even,odd,mark,space” --> </parityType>
   <stopBits>         <!-- req, xs:string, “1,1.5,2” -->                 </stopBits>
</SerialPort>
```

Notes：

- <id> value can only be 1 and 3. When <id> value is 1, <serialPortType> value is “RS485”.

When <id> value is 3, <serialPortType> value is “RS232”. <serialPortType> value can not set directly;

- <enabled>is read-only and be true now;
- <duplexMode>, <direction>not support now;
- <parityType> only supports “none”, “even” and “odd” now.

5.6.3 /System/Serial/ports/<ID>/command

URI	/System/Serial/ports/ID/command		**Type**	Resource
Function	Send a command to a serial port.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
PUT	chainNo	<SerialCommand> Raw Data	<ResponseStatus>	
Notes	If the IP device is an analog-to-digital encoder and is connected to analog PTZ-enabled camera(s), it is the device‟s responsibility to relay the request to the appropriate serial interface based on the <chainNo> tag or query string. If the IP device is itself a PTZ-enabled digital camera, it is the device‟s responsibility to address the correct serial interface for the corresponding PTZ command. The serial command can either be encapsulated in the <command> field, in which case the data should be encoded in hexadecimal notation, or the data can be uploaded directly as the HTTP payload, in which case the content type should be application/octet-stream.			

SerialCommand XML Block

```
<SerialCommand version=“1.0”    xmlns=“urn:psialliance-org”>
  <chainNo>     <!-- req, xs:string -->   </chainNo>
  <command>     <!-- req, xs:string, bytes in hexadecimal -->   </command>
</SerialCommand>
```

5.7 /Securiy

URI	/Security		**Type**	Service
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
Notes	Security service.			

5.8 /Security/AAA

URI	/Security/AAA	Type	Service
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
Notes	Authentication, authorization and auditing service.		

5.8.1 /Security/AAA/users

URI	/Security/AAA/users	Type	Resource
Function	Access the device user list.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<UserList>
PUT		<UserList>	<ResponseStatus>
POST		<User>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	It is possible to add, remove and update users entries in the list. Passwords can only be uploaded - they are never revealed during GET operations.		

UserList XML Block

```
<UserList version="1.0" xmlns="urn:psialliance-org">
   <User/>        <!-- opt -->
</UserList>
```

Notes：

- Up to 16 users now.
- A default user account ,"admin", must be provided. Its default password is "12345". Its ID is 1.

5.8.2 /Security/AAA/users/<ID>

URI	/Security/AAA/users/ID	Type	Resource
Function	Authentication user settings.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<User>
PUT		<User>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	Each <protocolID> tag, if <ProtocolList> is provided, must match a corresponding <id> tag in /Security/adminAccesses. Note: <password> is a write-only field.		

User XML Block

```
<User version="1.0" xmlns="urn:psialliance-org">
   <id>                <!-- req, xs:string;id -->          </id>
   <userName>            <!-- req, xs:string -->          </userName>
   <password>            <!-- wo, req, xs:string -->       </password>
   <ProtocolList>        <!-- opt -->
      <Protocol>        <!-- opt -->
         <protocolID>    <!-- req, xs:string;id -->          </protocolID>
      </Protocol>
   </ProtocolList>
</User>
```

Notes：

- <id> value range is 1-16;
- <userName> maximum length is 31, and <password> maximum length is 15;
- <ProtocolList>not support now.

5.9 /Streaming

URI	/Streaming		Type	Service
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
Notes	Streaming service.			

5.9.1 /Streaming/status

URI	/Streaming/status		Type	Resource
Function	Query the device streaming status.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<StreamingStatus>	
Notes	This command accesses the status of all device streaming sessions.			

StreamingStatus XML Block

```
<StreamingStatus version="1.0" xmlns="urn:psialliance-org">
   <totalStreamingSessions>          <!-- req, xs:integer -->    </totalStreamingSessions>
   <StreamingSessionStatusList/>       <!-- dep, only if there are sessions -->
</StreamingStatus>
```

5.9.2 /Streaming/channels

URI	/Streaming/channels		Type	Resource
Function	Streaming channels.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<StreamingChannelList>	
PUT		<StreamingChannelList>	<ResponseStatus>	
POST		<StreamingChannel>	<ResponseStatus>	
DELETE			<ResponseStatus>	
Notes	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 possible to dynamically create streaming channels, check the defined HTTP methods in /Streaming/channels/description			

StreamingChannelList XML Block

```
<StreamingChannelList version="1.0" xmlns="urn:psialliance-org">
   <StreamingChannel/>    <!-- opt -->
</StreamingChannelList>
```

Notes：

- Support 2 different configuration channles for each Hikvision IPMD.
- Streaming channels is hardwired, so POST and DELETE methods are not supported.

5.9.3 /Streaming/channels/<ID>

URI	/Streaming/channels/ID		Type	Resource
Function				
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<StreamingChannel>	
PUT		<StreamingChannel>	<ResponseStatus>	
DELETE			<ResponseStatus>	
Notes	<ControlProtocolList> identifies the control protocols that are valid for this type of streaming. <Unicast> is for direct unicast streaming. <Multicast> is for direct multicast streaming. <videoSourcePortNo> and <audioSourcePortNo> are the source port numbers for the outbound video or audio streams. <videoInputChannelID> refers to /System/Video/inputs/channel/ID. <audioInputChannelID> refers to /System/Audio/channels/ID. It must be configured as an input channel. Use of IPv4 or IPv6 addresses depends on the value of the <ipVersion> field in /System/Network/interfaces/ID/ipAddress.			

	<Security> determines whether SRTP is used for stream encryption. <audioResolution> is the resolution for the outbound audio stream in bits.

StreamingChannel XML Block

```
<StreamingChannel version=“1.0” xmlns=“urn:psialliance-org”>
  <id>          <!-- req, xs:string;id -->       </id>
  <channelName>   <!-- req, xs:string -->       </channelName>
  <enabled>       <!-- req, xs:boolean -->       </enabled>
  <Transport>      <!-- req -->
    <rtspPortNo>              <!-- opt, xs:integer -->       </rtspPortNo>
    <maxPacketSize>           <!-- opt, xs:integer -->       </maxPacketSize>
    <audioPacketLength>       <!-- opt, xs:integer -->       </audioPacketLength>
    <audioInboundPacketLength>   <!-- opt, xs:integer -->       </audioInboundPacketLength>
    <audioInboundPortNo>       <!-- opt, xs:integer -->       </audioInboundPortNo>
    <videoSourcePortNo>       <!-- opt, xs:integer -->       </videoSourcePortNo>
    <audioSourcePortNo>       <!-- opt, xs:integer -->       </audioSourcePortNo>
    <ControlProtocolList>       <!-- req -->
      <ControlProtocol>       <!-- opt -->
        <streamingTransport>
          <!-- req, xs:string, “HTTP,RTSP” -->
        </streamingTransport>
      </ControlProtocol>
    </ControlProtocolList>
    <Unicast>                  <!-- opt -->
      <enabled>              <!-- req, xs:boolean -->          </enabled>
      <interfaceID>          <!-- opt, xs:string -->          </interfaceID>
      <rtpTransportType>
        <!-- opt, xs:string, “RTP/UDP,RTP/TCP” -->
      </rtpTransportType>
    </Unicast>
    <Multicast>                  <!-- opt -->
      <enabled>              <!-- req, xs:boolean -->          </enabled>
      <userTriggerThreshold>   <!-- opt, xs:integer -->          </userTriggerThreshold>
      <destIPAddress>       <!-- opt, xs:string -->          </destIPAddress>
      <videoDestPortNo>       <!-- opt, xs:integer -->          </videoDestPortNo>
      <audioDestPortNo>       <!-- opt, xs:integer -->          </audioDestPortNo>
      <destIPv6Address>       <!-- opt, xs:string -->          </destIPv6Address>
      <ttl>                  <!-- opt, xs:integer -->          </ttl>
    </Multicast>
    <Security>               <!-- opt -->
      <enabled>              <!-- req, xs:boolean -->          </enabled>
    </Security>
  </Transport>
```

```
<Video>
  <enabled>              <!-- req, xs:boolean -->         </enabled>
  <videoInputChannelID>     <!-- req, xs:string;id -->        </videoInputChannelID>
  <videoCodecType>
    <!-- opt, xs:string, “MPEG4,MJPEG,3GP,H.264,MPNG” -->
  </videoCodecType>
  <videoScanType>
    <!-- opt, xs:string, “progressive,interlaced” -->
  </videoScanType>
  <videoResolutionWidth>     <!-- opt, xs:integer -->         </videoResolutionWidth>
  <videoResolutionHeight>   <!-- opt, xs:integer -->         </videoResolutionHeight>
  <videoPositionX>         <!-- opt, xs:integer -->         </videoPositionX>
  <videoPositionY>         <!-- opt, xs:integer -->         </videoPositionY>
  <videoQualityControlType>
    <!-- req, xs:string, “CBR,VBR” -->
  </videoQualityControlType>
  <constantBitRate>   <!-- opt, xs:integer, in kbps -->         </constantBitRate>
  <fixedQuality>    <!-- opt, xs:integer, percentage, 0..100 -->      </fixedQuality>
  <vbrUpperCap>      <!-- opt, xs:integer, in kbps -->         </vbrUpperCap>
  <vbrLowerCap>      <!-- opt, xs:integer, in kbps -->         </vbrLowerCap>
  <maxFrameRate>     <!-- req, xs:integer, maximum frame rate x100 -->   </maxFrameRate>
  <keyFrameInterval>   <!-- opt, xs:integer, milliseconds -->   </keyFrameInterval>
  <rotationDegree>   <!-- opt, xs:integer, degrees, 0..360 --></rotationDegree>
  <mirrorEnabled>   <!-- opt, xs:boolean -->              </mirrorEnabled>
  <snapShotImageType><!-- opt, xs:string, “JPEG,GIF,PNG” -->   </snapShotImageType>
</Video>
<Audio>
  <enabled>              <!-- req, xs:boolean -->         </enabled>
  <audioInputChannelID>     <!-- req, xs:string;id -->        </audioInputChannelID>
  <audioCompressionType>
    <!-- opt, xs:string,
      “G.711alaw,G.711ulaw,G.726,G.729,G.729a,G.729b,PCM,MP3,AC3,AAC,ADPCM”
     -->
  </audioCompressionType>
  <audioInboundCompressionType>
    <!-- opt, xs:string,
      “G.711alaw,G.711ulaw,G.726,G.729,G.729a,G.729b,PCM,MP3,AC3,AAC,ADPCM”
     -->
  </audioInboundCompressionType>
  <audioBitRate>       <!-- opt, xs:integer, in kbps -->      </audioBitRate>
  <audioSamplingRate>   <!-- opt, xs:float, in kHz -->      </audioSamplingRate>
  <audioResolution>      <!-- opt, xs:integer, in bits -->      </audioResolution>
</Audio>
```

```
</StreamingChannel>
```

Notes：

- <id> value can only be 1 and 2. When <id> value is 1, stream is the main stream. When <id> is 2, stream is sub stream;
- <channelName>, <enabled> is read-only. <enabled> value must be true.
- In<Transport>,the<audioPacketLength><audioInboundPacketLength><audioInboundPortNo><videoSourcePortNo><audioSourcePortNo> not support now. In <Unicast>, the <interfaceID> not support now . <Multicast>, <Security>not support now
- In<Video>,the<videoPositionX><videoPositionY><vbrUpperCap><vbrLowerCap><rotationDegree><mirrorEnabled>not support now.
- In<Audio>,the<audioInboundCompressionType>not support now
- In<Transport>,the<maxPacketSize>is read-only and the value is 1000;<streamingTransport>is read-only and the value is RTSP ；and in<Unicast>the<enabled> is read-only and the value is true；<rtpTransportType> is read -only and the value is RTP/TCP.
- In<Video>the<enabled> is read-only and the value is true; <videoInputChannelID> is read-only and the value is 1; <videoCodecType> is read-only and the value is H.264; <videoScanType> is read-only and the value is progressive; <videoResolutionWidth> and <videoResolutionHeight> are constrained. <snapShotImageType> is read-only and the value is JPEG.
- For DS-2CD802 series, DS-2CD812 series, DS-2CD832 series, DS-2CD892 series, DS-2CD702 series, DS-2CD712 series, DS-2CD732 series and DS-2CD792 series, 176*144, 352*288, 704*288, 528*384 and 704*576 resolution are supported on PAL devices; 176*120, 352*240, 704*240, 528*320 and 704*480 resolution are supported on NTSC devices.
- For DS-2CD852F series, DS-2CD852MF series and DS-2CD752MF series, 176*144, 352*288, 704*288, 528*384, 704*576, 640*480, 800*600, 1280*720, 1600*912 and 1600*1200 resolution are supported on PAL devices; 176*120, 352*240, 704*240, 528*320, 704*480, 640*480, 800*600, 1280*720, 1600*912 and 1600*1200 resolution are supported on NTSC devices.
- For DS-2CD862MF series and DS-2CD762MF series, 640*480, 1280*720 and 1280*960 are supported on the devices.
- The three description information above is for main stream. For sub stream, 176*144 and 352*288 resolution are supported on PAL devices; 176*120 and 352*240 resolution are supported on NTSC devices.
- In<Audio>the<audioInputChannelID>is read-only and the value is 1; <audioCompresssionType>is read-only and the value is G.711ulaw; <audioBitRate>is read-only and the value is 8; <audioSamplingRate> is read-only and the value is 16; <audioResolution> is read-only and the value is 16.
- Streaming channels is hardwired, so DELETE method is not supported.

Example: Getting Streaming Channel Properties

The following is an example of a GET on the streaming parameters of a particular channel that has been preconfigured by the IP media device. Depending on the device, some streaming channels may be already preconfigured or the device while other may require that channels be manually configured before use.

```
GET /Streaming/channels/1 HTTP/1.1
…
HTTP/1.1 200 OK
Content-Type: application/xml; charset=“UTF-8”
Content-Length: xxx

<?xml version=“1.0” encoding=“UTF-8”?>
<StreamingChannel version=“1.0” xmlns=“urn:psialliance-org”>
    <id>1</id>
    <channelName>Input 1 H.264</channelName>
    <enabled>true</enabled>
    <Transport>
        <rtspPortNo>554</rtspPortNo>
        <maxPacketSize>1000</maxPacketSize>
        <ControlProtocolList>
            <ControlProtocol>
                <streamingTransport>RTSP</streamingTransport>
            </ControlProtocol>
        </ControlProtocolList>
        <Unicast>
            <enabled>true</enabled>
            <rtpTransportType>RTP/TCP</rtpTransportType>
        </Unicast>
    </Transport>
    <Video>
        <enabled>true</enabled>
        <videoInputChannelID>1</videoInputChannelID>
        <videoCodecType>H.264</videoCodecType>
        <videoScanType>progressive</videoScanType>
        <videoResolutionWidth>1600</videoResolutionWidth>
        <videoResolutionHeight>1200</videoResolutionHeight>
        <videoQualityControlType>CBR</videoQualityControlType>
        <constantBitRate>3072</constantBitRate>
        <fixedQuality>8</fixedQuality>
        <maxFrameRate>2500</maxFrameRate>
        <keyFrameInterval>25</keyFrameInterval>
        <snapShotImageType>JPEG</snapShotImageType>
    </Video>
    <Audio>
```

```
        <enabled>false</enabled>
        <audioInputChannelID>1</audioInputChannelID>
        <audioCompressionType>G.711ulaw</audioCompressionType>
        <audioBitRate>8</audioBitRate>
        <audioSamplingRate>16</audioSamplingRate>
        <audioResolution>16</audioResolution>
    </Audio>
</StreamingChannel>
```

Example: Getting Streaming Capabilities

```
GET /Streaming/channels/1/capabilities HTTP/1.1
…
HTTP/1.1 200 OK
Content-Type: application/xml; charset="UTF-8"
Content-Length: xxx

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

<StreamingChannel version="1.0" xmlns="urn:psialliance-org">
    <id opt="1, 2">1</id>
    <channelName min="0" max="64">Input 1 H.264</channelName>
    <enabled opt="true">true</enabled>
    <Transport>
        <rtspPortNo min="0" max="65535" def="554">554</rtspPortNo>
        <maxPacketSize opt="1000">1000</maxPacketSize>
        <ControlProtocolList>
            <ControlProtocol>
                <streamingTransport opt="RTSP/RTP">RTSP</streamingTransport>
            </ControlProtocol>
        </ControlProtocolList>
        <Unicast>
            <enabled opt="true">true</enabled>
            <rtpTransportType opt="RTP/TCP">RTP/TCP<rtpTransportType>
        </Unicast>
    </Transport>
    <Video>
        <enabled opt="true">true</enabled>
        <videoInputChannelID opt="1">1</videoInputChannelID>
        <videoCodecType opt="H.264">H.264</videoCodecType>
        <videoScanType opt="progressive">progressive</videoScanType>
        <videoResolutionWidth opt="176,352,528,640,704,800,1280,1600">1600
        </videoResolutionWidth>
        <videoResolutionHeight opt=”144,288,384,480,576,600,720,912,1200”>1200
        </videoResolutionHeight>
        <videoQualityControlType opt=“CBR,VBR”>CBR</videoQualityControlType>
        <constantBitRate min=“32” max=“4000”>3072</constantBitRate>
        <fixedQuality opt=”4,8,10,12,14,16”>8</fixedQuality>
        <maxFrameRate opt=“2500,2200,1800,1600,1200,1000,800,600,400,200,100,50,25,12,6”>
            2500</maxFrameRate>
        <keyFrameInterval min=“1”, max=“400”>25</keyFrameInterval>
        <snapShotImageType opt=“JPEG” def=“JPEG”>JPEG</snapShotImageType>
    </Video>
    <Audio>
        <enabled opt=“true,false”>true</enabled>
        <audioInputChannelID opt=“1”>1</audioInputChannelID>
        <audioCompressionType opt=“G.711ulaw” def=“G.711ulaw”> G.711ulaw
        </audioCompressionType>
        <audioBitRate opt=“8” def=“8” >8</audioBitRate>
        <audioSamplingRate opt=“16” >16</audioSamplingRate>
        <audioResolution opt=“16”>16</audioResolution>
    </Audio>
</StreamingChannel>
```

5.9.4 /Streaming/channels/<ID>/status

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

StreamingSessionStatusList XML Block

```
<StreamingSessionStatusList version=“1.0” xmlns=“urn:psialliance-org”>
  <StreamingSessionStatus version=“1.0” xmlns=“urn:psialliance-org”>
    <clientAddress>   <!-- req -->
      <ipAddress>      <!-- dep, xs:string -->            </ipAddress>
      <ipv6Address>   <!-- dep, xs:string -->            </ipv6Address>
    </clientAddress>
    <clientUserName>   <!-- opt, xs:string -->            </clientUserName>
    <startDateTime>   <!-- opt, xs:datetime -->          </startDateTime>
    <elapsedTime>      <!-- opt, xs:integer, seconds -->      </elapsedTime>
```

```
        <bandwidth>            <!-- opt, xs:integer, in kbps -->       </bandwidth>
    </StreamingSessionStatus>
</StreamingSessionStatusList>
```

Notes：

<clientUserName>, <startDateTime>, <elapsedTime>, <bandwidth>not support now.

5.9.5 /Streaming/channels/<ID>/http（Not support now）

URI	/Streaming/channels/<ID>/http	**Type**	Resource
Function	Access a live stream via http.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET	videoCodecType videoScanType videoResolutionWidth videoResolutionHeight videoPositionX videoPositionY videoQualityControlType constantBitRate fixedQuality vbrUpperCap vbrLowerCap maxFrameRate keyFrameInterval rotationDegree mirrorEnabled snapShotImageType		Stream over HTTP
POST		<Video>	
Notes	This function is used to request a stream from the device using HTTP or HTTPS. This API uses HTTP server-push with the MIME type multipart/x-mixed-replace. HTTP streaming must be enabled on the channel. To determine the format of the video returned, either the parameters in <Video> or the query string values are used, depending on the capabilities of the encoder. For supported values, query /Streaming/channels/ID/http/capabilities.		

Example:

```
GET /Streaming/channels/777/http?videoCodecType=MJPEG HTTP/1.1
…
HTTP/1.1 200 OK
Content-Type:    multipart/x-mixed-replace; boundary=<boundary>
--<boundary>
Content-Type: image/jpeg
Content-Length: xxx
```

```
Image data for a single frame
--<boundary>
…
```

5.9.6 /Streaming/channels/<ID>/picture

URI	/Streaming/channels/ID/picture	**Type**	Resource
Function	Get a snapshot of the current image.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET	videoResolutionWidth videoResolutionHeight videoPositionX videoPositionY rotationDegree mirrorEnabled snapShotImageType		Picture over HTTP
POST		<Video>	
Notes	All devices must support <snapShotImageType> of "JPEG". To determine the format of the picture returned, either the parameters in <Video> or the query string values are used, or, if the Accept: header field is present in the request and the server supports it, the picture is returned in that format. For supported values, query /Streaming/channels/ID/picture/capabilities. Examples: GET /Streaming/channels/123456/picture?snapShotImageType=JPEG POST /Streaming/channels/123456/picture … <?xml version="1.0" encoding="UTF-8"?> <Video>…</Video> GET /Streaming/channels/123456/picture Accept: image/jpeg		

Notes：

- The parameters videoPositionX, videoPositionY, rotationDegree, mirrorEnabled for Query String(s) are not support now, snapShotImageType must be JPEG.
- POST method not support now.
- Only support the main stream channel snapshot.

5.9.7 /Streaming/channels/<ID>/requestKeyFrame

URI	/Streaming/channels/ID/requestKeyFrame		Type	Resource
Function	Request that the device issue a key frame on a particular channel.			
Methods	Query String(s)	Inbound Data	Return Result	
PUT			<ResponseStatus>	
Notes	The key frame that is issued should include everything necessary to initialize a video decoder, i.e. parameter sets for H.264 or VOS for MPEG-4.			

5.10 /Custom/MotionDetection

URI	/Custom/MotionDetection		Type	Service
Function	Motion detection configuration for all video input channels.			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<MotionDetectionList>	
Notes	If motion detection is supported by the device, a motion detection ID will be allocated for each video input channel ID. The motion detection ID must correspond to the video input channel ID.			

MotionDetectionList XML Block

```
<MotionDetectionList version="1.0" xmlns="urn:psialliance-org">
   <MotionDetection/>          <!-- opt -->
</MotionDetectionList >
```

5.10.1 /Custom/MotionDetection/<ID>

URI	/Custom/MotionDetection/ID		Type	Resource
Function	Motion detection configuration for a video input channel.			
Methods	Query String(s)	Inbound Data	Return Result	
GET			<MotionDetection>	
PUT		<MotionDetection>	<ResponseStatus>	
Notes	Note that the ID used here MUST correspond to the video input ID. The interface supports both grid-based and region-based motion detection. The actual types supported can be determined by looking at the result of a GET of /Custom/MotionDetection/ID/capabilities and looking at the options available for the <regionType> field. Grid-based motion detect divides the image into a set of fixed "bins" that delimit			

	the motion detection area boundaries. ROI-based motion detection allows motion areas or regions of interest to be defined based on pixel coordinates.

MotionDetection XML Block

```
<MotionDetection version="1.0" xmlns="urn:psialliance-org">
   <id>                 <!-- req, xs:string -->                     </id>
   <enabled>              <!-- req, xs:boolean -->                      </enabled>
   <samplingInterval>       <!-- req, xs:integer, number of frames -->    </samplingInterval>
   <startTriggerTime>       <!-- req, xs:integer, milliseconds -->       </startTriggerTime>
   <endTriggerTime>       <!-- req, xs:integer, milliseconds -->       </endTriggerTime>
   <directionSensitivity>
      <!-- opt, xs:string, "left-right,right-left,up-down,down-up" -->
   </directionSensitivity>
   <regionType>           <!-- req, xs:string, "grid,roi" -->           </regionType>
   <minObjectSize>
      <!-- opt, xs:integer, min number of pixels per object -->
   </minObjectSize>
   <maxObjectSize>
      <!-- opt, xs:integer, max number of pixels per object -->
   </maxObjectSize>
   <Grid>              <!-- dep, required if <motionType> is "grid" -->
      <rowGranularity>       <!-- req, xs:integer -->    </rowGranularity>
      <columnGranularity>    <!-- req, xs:integer -->    </columnGranularity>
   </Grid>
   <ROI>               <!-- dep, required if <motionType> is "roi" -->
      <minHorizontalResolution> <!-- req, xs:integer -->    </minHorizontalResolution>
      <minVerticalResolution>    <!-- req, xs:integer -->    </minVerticalResolution>
   </ROI>
   <MotionDetectionRegionList/>    <!-- req -->
</MotionDetection>
```

Notes：

- <regionType> is read-only. Its value is "grid".
- <minObjectSize>, <maxObjectSize>, <samplingInterval>, <startTriggerTime>, <endTriggerTime>, <directionSensitivity>not support now.
- In <Grid>, <rowGranularity>,<columnGranularity> are read-only. The <rowGranularity> value is 18, <columnGranularity> value is 22.

5.10.2 /Custom/MotionDetection/<ID>/regions

URI	/Custom/MotionDetection/ID/regions	**Type**	Resource
Function	Access the list of regions for motion detection on a particular video input channel.		

Methods	Query String(s)	Inbound Data	Return Result
GET			<MotionDetectionRegionList>
PUT		<MotionDetectionRegionList>	<ResponseStatus>
POST		<MotionDetectionRegion>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	Each motion detection region has its own detection threshold and sensitivity level. It is possible to define mask regions that are subtracted from other regions, allowing non-rectangular motion areas to be configured.		

MotionDetectionRegionList XML Block

```
<MotionDetectionRegionList version="1.0" xmlns="urn:psialliance-org">
  <MotionDetectionRegion/>   <!-- opt -->
</MotionDetectionRegionList>
```

Notes:

MotionDetection supports up to 16 regions now.

5.10.3 /Custom/MotionDetection/<ID>/regions/<ID>

URI	/Custom/MotionDetection/ID/regions/<ID>	**Type**	Resource
Function	Access the list of regions for motion detection.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<MotionDetectionRegion>
PUT		<MotionDetectionRegion>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	The region detection coordinate space depends on the value of <motionType>.		

MotionDetectionRegion XML Block

```
<MotionDetectionRegion version="1.0" xmlns="urn:psialliance-org">
  <id>              <!-- req, xs:string -->   </id>
  <enabled>            <!-- req, xs:boolean -->   </enabled>
  <maskEnabled>          <!-- req, xs:boolean -->   </maskEnabled>
  <sensitivityLevel>       <!-- req -->
    <!-- req, xs:integer, 0..100, 0 is least sensitive -->
  </sensitivityLevel>
  <detectionThreshold>       <!-- req -->
    <!-- req, xs:integer, 0..100, percentage-->
  </detectionThreshold>
  <RegionCoordinatesList>   <!-- req -->
    <RegionCoordinates>   <!-- Note: at least two coordinates are required -->
      <positionX>          <!-- req, xs:integer -->   </positionX>
```

```
        <positionY>          <!-- req, xs:integer -->    </positionY>
      </RegionCoordinates>
    </RegionCoordinatesList>
</MotionDetectionRegion>
```

Notes：

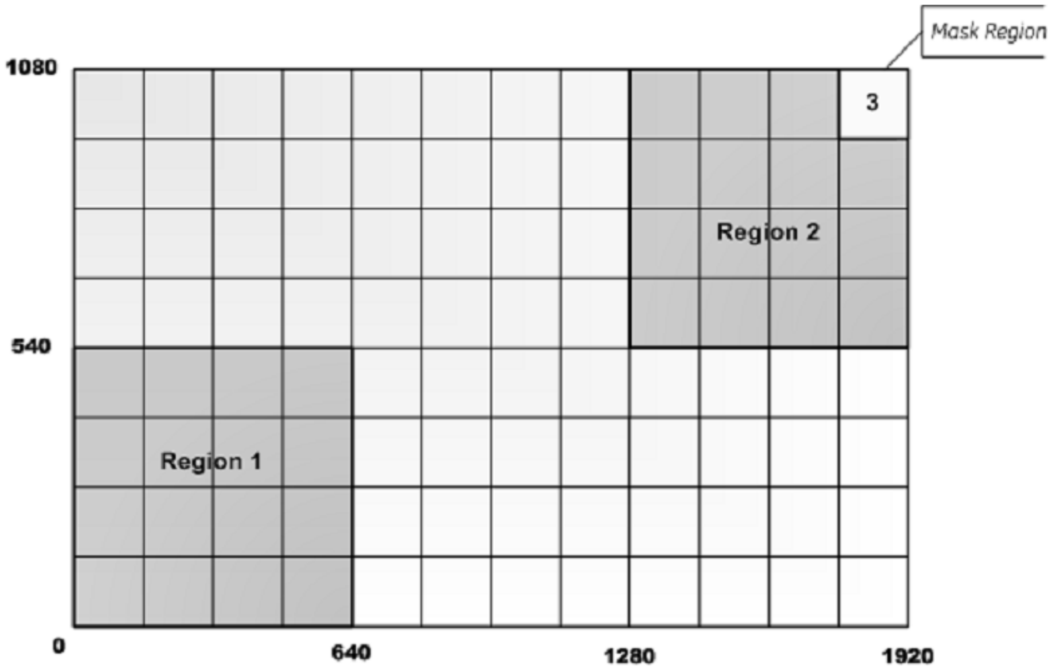
- <id> value range is 1-16;
- <sensitivityLevel> value range is 0-5. All regions share a <sensitivityLevel>, and its value is the last set of values;
- Only support the rectangular region which will be “drawn” from four coordinates. The four points is clockwise direction, and the beginning point is the low-left point;
- <detectionThreshold>not support now.

5.10.4 Motion Detection Example

Set up Motion Detection

The following command configures two rectangular detection regions, with one “masked” region on video input channel ID 1. Example assumes a resolution of 1920x1080 and a grid motion detection algorithm:

- Motion detection is enabled with a granularity of a 12x8 grid – this means the detection region coordinates will ultimately be defined by a grid of 96 regions. For a resolution of 1920x1080, this means that each “granule” will be 160x135 pixels (1920/12 x 1080/8). (If a coordinate doesn’t exactly match the configured granularity, it should be mapped internally to the nearest possible point)
- A sample will be taken every 2 frames for motion detection and motion must be detected for at least one second before triggering an event notification (motion must be stopped for at least one second to stop the triggering).
- Two detection regions are defined, the second containing an inner/overlapping region that is disabled. Region 1 occupies the bottom-left 8 granules. Region 2 occupies the top-right 8 granules, with the top-right-most corner granule (region 3) disabled by use of the <maskEnabled> tag.



```
PUT /Custom/MotionDetection/1 HTTP/1.1
Content-Type: application/xml; charset=“UTF-8”
Content-Length: xxx

<?xml version=“1.0” encoding=“UTF-8”?>
<MotionDetection version=“1.0” xmlns=“urn:psialliance-org”>
    <enabled>true</enabled>
    <regionType>grid</regionType>
    <Grid>
        <rowGranularity>8</rowGranularity>
        <columnGranularity>12</columnGranularity>
    </Grid>
    <MotionDetectionRegionList>
        <MotionDetectionRegion>
            <enabled>true</enabled>
            <sensitivityLevel>50</sensitivityLevel>
            <RegionCoordinatesList>
                <RegionCoordinates>
                    <positionX>0</positionX>
                    <positionY>0</positionY>
                </RegionCoordinates>
                <RegionCoordinates>
                    <positionX>0</positionX>
                    <positionY>4</positionY>
                </RegionCoordinates>
                <RegionCoordinates>
        <positionX>4</positionX>
        <positionY>4</positionY>
      </RegionCoordinates>
      <RegionCoordinates>
        <positionX>4</positionX>
        <positionY>0</positionY>
      </RegionCoordinates>
    </RegionCoordinatesList>
  </MotionDetectionRegion>
  <MotionDetectionRegion>
    <enabled>true</enabled>
    <sensitivityLevel>20</sensitivityLevel>
    <RegionCoordinatesList>
      <RegionCoordinates>
        <positionX>8</positionX>
        <positionY>4</positionY>
      </RegionCoordinates>
      <RegionCoordinates>
        <positionX>8</positionX>
        <positionY>8</positionY>
      </RegionCoordinates>
      <RegionCoordinates>
        <positionX>12</positionX>
        <positionY>8</positionY>
      </RegionCoordinates>
      <RegionCoordinates>
        <positionX>12</positionX>
        <positionY>4</positionY>
      </RegionCoordinates>
    </RegionCoordinatesList>
  </MotionDetectionRegion>
  <MotionDetectionRegion>
    <maskEnabled>true</maskEnabled>
    <RegionCoordinatesList>
      <RegionCoordinates>
        <positionX>11</positionX>
        <positionY>7</positionY>
      </RegionCoordinates>
      <RegionCoordinates>
        <positionX>11</positionX>
        <positionY>8</positionY>
      </RegionCoordinates>
      <RegionCoordinates>
```

```
                <positionX>12</positionX>
                <positionY>8</positionY>
            </RegionCoordinates>
            <RegionCoordinates>
                <positionX>12</positionX>
                <positionY>7</positionY>
            </RegionCoordinates>
        </RegionCoordinatesList>
    </MotionDetectionRegion>
  </MotionDetectionRegionList>
</MotionDetection>
```

5.11 /Custom/Event

URI	/Custom/Event		**Type**	Service
Function	Access and configure the device event behavior, scheduling and notifications.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<EventNotification>	
PUT		<EventNotification>	<ResponseStatus>	
Notes	The event trigger list defines the set of device behaviors that trigger events. The event schedule defines when event notifications are active. The event notification methods define what types of notification (HTTP, FTP, e-mail) are supported.			

EventNotification XML Block

```
<EventNotification version="1.0" xmlns="urn:psialliance-org">
  <EventTriggerList/>            <!-- opt -->
  <EventSchedule/>               <!-- opt -->
  <EventNotificationMethods/>    <!-- opt -->
</EventNotification>
```

5.11.1 /Custom/Event/triggers

URI	/Custom/Event/triggers		**Type**	Resource
Function	Access the list of event triggers.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<EventTriggerList>	
PUT		<EventTriggerList>	<ResponseStatus>	

POST		<EventTrigger>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	Event triggering defines how the device reacts to particular events, such as video loss or motion detection.		

EventTriggerList XML Block

```
<EventTriggerList version="1.0" xmlns="urn:psialliance-org">
   <EventTrigger/>    <!-- opt -->
</EventTriggerList>
```

Notes:

PUT method is not support now.

5.11.2 /Custom/Event/triggers/<ID>

URI	/Custom/Event/triggers/ID	**Type**	Resource
Function	Access a particular event trigger.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<EventTrigger>
PUT		<EventTrigger>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	An event trigger determines how the device reacts when a particular event is detected. The following types are supported: IO: trigger when an input IO port changes state. VMD: trigger on video motion detection. Video loss: trigger when the input video signal cannot be detected. Disk failure: trigger when a disk fails. Recording failure: trigger when recording fails: either there is a problem with the disk, or the storage volume is full, or the volume is corrupt. Bad video: trigger when the input video is bad. POS: trigger when a point-of-sale event is detected. Analytics: trigger on a general analytics event. Currently analytics events apart from VMD, which has its own event trigger, are not supported. Fan failure: trigger when a fan fails. Overheat: trigger when the temperate threshold of a particular sensor is exceeded. Device vendors can add additional event types and advertise these using the capabilities query on /Custom/Event/triggers. <inputIOPortID> is only required if <eventType> is "IO".		

EventTriggerList XML Block

```
<EventTrigger version="1.0" xmlns="urn:psialliance-org">
   <id>                    <!-- req, xs:string -->              </id>
   <eventType>                  <!-- req -->
      <!-- req, xs:string,
         "IO,VMD,videoloss,diskfailure,recordingfailure,
          badvideo,POS,analytics,fanfailure,overheat"
      -->
   </eventType>
   <eventDescription>          <!-- req, xs:string -->              </eventDescription>
   <inputIOPortID>          <!-- req, xs:string -->              </inputIOPortID>
   <intervalBetweenEvents>    <!-- req, xs:integer, seconds -->       </intervalBetweenEvents>
   <EventTriggerNotificationList/>    <!-- opt -->
</EventTrigger>
```

Notes：

- <eventType> could support the IO,VMD,videoloss that define in the PSIA specification and support the shelteralarm that expanded by Hikvision;
- <id> value range is 1-4. When <id> value is 1, <eventType> value is "IO". When <id> value is 2, <eventType> value is "VMD". When <id> value is 3, <eventType> value is "videoloss". When <id> value is 4, <eventType> value is "shelteralarm". <eventType> value can not set directly;
- <eventDescription>, <inputIOPortID> is read-only;
- <intervalBetweenEvents> is not supported now.
- PUT method is not support now.

5.11.3 /Custom/Event/triggers/<ID>/notifications

URI	/Custom/Event/triggers/ID/notifications	**Type**	Resource
Function	List of notification methods and behaviors.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<EventTriggerNotificationList>
PUT		<EventTriggerNotificationList>	<ResponseStatus>
POST		<EventTriggerNotification>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	This section determines the kinds of notifications that are supported for a particular event trigger and their recurrences and behaviors		

EventTriggerNotificationList XML Block

```
<EventTriggerNotificationList version="1.0" xmlns="urn:psialliance-org">
   <EventTriggerNotification/>    <!-- opt -->
```

```
</EventTriggerNotificationList>
```

Notes:

PUT method is not support now.

5.11.4 /Custom/Event/triggers/<ID>/notifications/<ID>

URI	/Custom/Event/triggers/ID/notifications/ID	**Type**	Resource
Function	Access and configure a particular notification trigger.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<EventTriggerNotification>
PUT		<EventTriggerNotification>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	<outputIOPortID> is only required if the <notifiocationMethod> is “IO”.		

EventTriggerNotification XML Block

```
<EventTriggerNotification version=“1.0” xmlns=“urn:psialliance-org”>
   <id>                    <!-- req, xs:string -->                  </id>
   <notificationMethod>       <!-- req -->
      <!-- req, xs:string, “email,IM,IO,syslog,HTTP,FTP” -->
   </notificationMethod>
   <notificationRecurrence>    <!-- req -->
      <!-- req, xs:string, “beginning,beginningandend,recurring” -->
   </notificationRecurrence>
   <notificationInterval>       <!-- req, xs:integer, milliseconds -->    </notificationInterval>
   <outputIOPortID>           <!-- dep, xs:string -->                 </outputIOPortID>
</EventTriggerNotification>
```

Notes：

- <notificatonMethod>only support email, IO now;
- <id> value range is 1-2. When <id> value is 1, <notificationMethod> value is “email”. When <id> value is 2, <notificationMethod> value is “IO”. <notificationMethod> value can not set directly;
- <notificationRecurrence>only support beginning now;
- <notificationInterval> is not supported;
- <outputIOPortID> is read-only.
- PUT method is not supported now.

5.11.5 /Custom/Event/schedule

URI	/Custom/Event/schedule	**Type**	Resource

Function	Event schedules.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<EventSchedule>
PUT		<EventSchedule>	<ResponseStatus>
Notes	Defines the schedule. The schedule is defined as a date-time range and a set of time blocks that define when the events are active. If <DateTimeRange> is not present, the schedule is always valid.		

EventSchedule XML Block

```
<EventSchedule version="1.0" xmlns="urn:psialliance-org">
   <DateTimeRange>       <!-- opt -->
      <beginDateTime>    <!-- req, xs:datetime -->      </beginDateTime>
      <endDateTime>       <!-- req, xs:datetime -->      </endDateTime>
   </DateTimeRange>
   <TimeBlockList/>       <!-- req -->
</EventSchedule>
```

Notes：

- <DateTimeRange> is not supported now.
- Now it only supports one TimeBlock every day.

5.11.6 /Custom/Event/notification

URI	/Custom/Event/notification		Type	Resource
Function	Configure notifications.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<EventNotificationMethods>	
PUT		<EventNotificationMethods>	<ResponseStatus>	
Notes	The following notification types are supported: HTTP: the device connects to a given address and port and issues an HTTP GET/POST with the given parameters. FTP: a video clip or snapshot is uploaded to an FTP server. E-mail: a mail with the video clip or snapshot is sent in an e-mail to a list of servers. <MediaFormat> determines the type of snapshot, video clip and the video clip pre and post recording times.			

EventNotificationMethods XML Block

```
<EventNotificationMethods version="1.0" xmlns="urn:psialliance-org">
   <MailingNotificationList/>       <!-- opt -->
   <FTPNotificationList/>           <!-- opt -->
   <HttpHostNotificationList/>    <!-- opt -->
   <FTPFormat>
      <uploadSnapShotEnabled>    <!-- req, xs:boolean -->    </uploadSnapShotEnabled>
      <uploadVideoClipEnabled>    <!-- req, xs:boolean -->    </uploadVideoClipEnabled>
   </FTPFormat>
   <EmailFormat>                  <!-- opt -->
      <senderEmailAddress>       <!-- req, xs:string -->    </senderEmailAddress>
      <receiverEmailAddress>       <!-- req, xs:string -->    </receiverEmailAddress>
      <subject></subject>
      <BodySetting>
         <attachedVideoURLEnabled> <!-- req, xs:boolean -->    </attachedVideoURLEnabled>
         <attachedSnapShotEnabled> <!-- req, xs:boolean -->    </attachedSnapShotEnabled>
         <attachedVideoClipEnabled>    <!-- req, xs:boolean -->    </attachedVideoClipEnabled>
      </BodySetting>
   </EmailFormat>
   <MediaFormat>                  <!-- opt -->
      <snapShotImageType>    <!-- req, xs:string -->    </snapShotImageType>
      <videoClipFormatType>    <!-- req, xs:string -->    </videoClipFormatType>
      <preCaptureLength>       <!-- req, xs:integer, milliseconds -->    </preCaptureLength>
      <postCaptureLength>    <!-- req, xs:integer, milliseconds -->    </postCaptureLength>
   </MediaFormat>
</EvenNotificationMethods>
```

Notes：

Only support <MailingNotificationList> and <senderEmailAddress>, <receiverEmailAddress> in the <EmailFomat> now.

5.11.7 /Custom/Event/notification/mailing

URI	/Custom/Event/notification/mailing	**Type**	Resource
Function	E-mail notifications.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<MailingNotificationList>
PUT		<MailingNotificationList>	<ResponseStatus>
POST		<MailingNotification>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	When the notification is triggered, an e-mail with a snapshot or video clip is mailed to the each of the addresses in the mailing list.		

MailingNotificationList XML Block

```
<MailingNotificationList version=“1.0” xmlns=“urn:psialliance-org”>
```

```
  <MailingNotification/>      <!-- opt -->
</MailingNotificationList>
```

Notes:

Now only support one <MailingNotification>.

5.11.8 /Custom/Event/notification/mailing/<ID>

URI	/Custom/Event/notification/mailing/ID		**Type**	Resource
Function	Access a particular e-mail notification.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<MailingNotification>	
PUT		<MailingNotification>	<ResponseStatus>	
DELETE			<ResponseStatus>	
Notes	Depending on the value of <addressingFormatType>, either the <hostName> or the IP address fields will be used to locate the NTP server. <authenticationMode> determines the authentication requirements for sending an email from the device. <portNo> is the port number of the SMTP server entry. <popAddressingFormatType> indicates whether an IP address or hostname is used for the POP server. <accountName> is the user account name for the SMTP server			

MailingNotification XML Block

```
<MailingNotification version="1.0" xmlns="urn:psialliance-org">
  <id>              <!-- req, xs:string -->       </id>
  <authenticationMode>
    <!-- req, xs:string, "none,SMTP,POP/SMTP" -->
  </authenticationMode>
  <addressingFormatType>
    <!-- req, xs:string, "ipaddress,hostname" -->
  </addressingFormatType>
  <hostName>          <!-- dep, xs:string -->       </hostName>
  <ipAddress>           <!-- dep, xs:string -->       </ipAddress>
  <ipv6Address>         <!-- dep, xs:string -->       </ipv6Address>
  <portNo>            <!-- opt, xs:integer -->       </portNo>
  <popAddressingFormatType>
    <!-- opt, xs:string, "ipaddress,hostname" -->
  </popAddressingFormatType>
  <popServerHostName>    <!-- opt, xs:string -->    </popServerHostName>
  <popServerIPAddress>    <!-- opt, xs:string -->    </popServerIPAddress>
  <popServerIPv6Address>    <!-- opt, xs:string -->    </popServerIPv6Address>
  <accountName>          <!-- req, xs:string -->    </accountName>
  <password>            <!-- req, xs:string -->    </password>
</MailingNotification>
```

Now：

- <id> value can only be 1;
- <authenticationMode>only support SMTP now;
- <addressingFormatType>, <hostName>, <ipAddress>, <ipv6Address> fill in the SMTP server information;
- <portNo> is read-only;
- <popAddressingFormatType>, <popServerHostName>, <popServerIPAddress>, <popServerIPv6Address>nort support now.

5.11.9 /Custom/Event/notification/alertStream

URI	/Custom/Event/notification/alertStream		**Type**	Resource
Function	Access the event notification data stream through HTTP server push.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			Stream of <EventNotificationAlert>	
Notes	This function is used to get an event notification alert stream from the media device via HTTP or HTTPS. This function does not require that a client/VMS system be added as an HTTP(S) destination on the media device. Instead, the client/VMS system can call this API to initialize a stream of event information from the device. In other words, a connection is established with the device when this function is called, and stays open to constantly receive event notifications. This API uses HTTP server-push with the MIME type multipart/mixed defined in RFC 2046. <protocol> is the protocol name, i.e. "HTTP" or "HTTPS". <channelID> is present for video and analytics events. <activePostCount> is the sequence number of current notification for this particular event. It starts at 1. Useful for recurring notifications of an event. Each event maintains a separate post count.			

EventNotificationAlert XML Block

```
<EventNotificationAlert version="1.0" xmlns="urn:psialliance-org">
  <ipAddress>         <!-- dep, xs:string -->       </ipAddress>
  <ipv6Address>       <!-- dep, xs:string -->       </ipv6Address>
  <portNo>          <!-- opt, xs:integer -->       </portNo>
  <protocol>        <!-- opt, xs:string -->       </protocol>
  <macAddress>       <!-- opt, xs:string;MAC -->    </macAddress>
  <channelID>         <!-- dep, xs:string -->       </channelID>
  <dateTime>        <!-- req, xs:datetime -->      </dateTime>
  <activePostCount>    <!-- req, xs:integer -->       </activePostCount>
```

```
   <eventType>
      <!-- req, xs:string,
         “IO,VMD,videoloss,raidfailure,recordingfailure,
           badvideo,POS,analytics,fanfailure,overheat”
      -->
   </eventType>
   <eventState>      <!-- req, xs:string, “active,inactive” -->   </eventState>
   <eventDescription>   <!-- req, xs:string -->                  </eventDescription>
   <inputIOPortID>   <!-- dep, xs:string, if <eventType> is “IO” -->      </inputIOPortID>
   <DetectionRegionList>          <!-- dep, if <eventType> is “VMD” -->
      <DetectionRegionEntry>       <!-- req -->
         <regionID>            <!-- req, xs:string -->           </regionID>
         <sensitivityLevel>       <!-- req, xs:integer, 0..100 -->   </sensitivityLevel>
         <detectionThreshold>   <!-- req, xs:integer, 0..100 -->   </detectionThreshold>
         <detectionLevel>       <!-- req, xs:integer, 0..100 -->   </detectionLevel>
      </DetectionRegionEntry>
   </DetectionRegionList>
</EventNotificationAlert>
```

Notes：

- <protocol>only support HTTP now
- <eventType> could support the IO,VMD,videoloss that define in the PSIA specification and support the shelteralarm that expanded by Hikvision.
- In <DetectionRegionEntry>, the<detectionThreshold>, <detectionLevel>not support now.

Example

The following is an example of an HTTP event stream that pushes a VMD event from video channel 1.

```
GET /Custom/Event/notification/alertStream HTTP/1.1
…
HTTP/1.1 200 OK
MIME-Version: 1.0
Content-Type:   multipart/mixed; boundary=“<boundary>”
--<boundary>
Content-Type: application/xml; charset=“UTF-8”
Content-Length: xxx

<?xml version=“1.0” encoding=“UTF-8”?>
<EventNotificationAlert version=“1.0” xmlns=“urn:psialliance-org”>
   <ipAddress>3.137.217.220</ipAddress>
   <portNo>80</portNo>
   <protocol>HTTP</protocol>
   <macAddress>00:14:22:43:D5:D4</macAddress>
```

```
    <channelID>1</channelID>
    <dateTime>2009-03-11T15:27Z</dateTime>
    <activePostCount>1</activePostCount>
    <eventType>VMD</eventType>
    <eventState>active</eventState>
    <eventDescription>Motion alarm</eventDescription>
    <DetectionRegionList>
        <DetectionRegionEntry>
            <regionID>2</regionID>
            <sensitivityLevel>67</sensitivityLevel>
        </DetectionRegionEntry>
    </DetectionRegionList>
</EventNotificationAlert>
--<boundary>
…
```

5.11.10 Event Triggering Examples

Example: Trigger Events on IO Port

The command below enables detection for input port 1. When the input signal is detected according to <inputIOPortID>, two event notification responses are used – output port 2 will be triggered for the duration of the input signal detection, and an SMTP server will be notified with the "E-mail Event Notification Alert". The behavior of this notification is as follows:

- An SMTP notification is sent at detection time, and every 5 seconds after while the signal is present. This is denoted by the <notificationRecurrence> and <notificationInterval> tags. These APIs will have an <eventState> of "active".
- When the input port 1 signal detection stops, one last E-mail notification is sent to the server (again, 5 seconds from the last notification) with an <eventState> of "active".
- After the signal detection stops for input port 1, the device will wait 1 second before starting to detect the signal again for this port (indicated by <intervalBetweenEvents>).

```
POST /Custom/Event/triggers HTTP/1.1
Content-Type: application/xml; charset="UTF-8"
Content-Length: xxx

<?xml version="1.0" encoding="UTF-8"?>
<EventTrigger version="1.0"   xmlns="urn:psialliance-org">
    <eventType>IO</eventType>
    <eventDescription>Input port 1 event detection</eventDescription>
```

```
   <inputIOPortID>1</inputIOPortID>
   <intervalBetweenEvents>1</intervalBetweenEvents>
   <EventTriggerNotificationList>
      <EventTriggerNotification>
         <notificationMethod>IO</notificationMethod>
         <outputIOPortID>2</outputIOPortID>
      </EventTriggerNotification>
      <EventTriggerNotification>
         <notificationMethod>email</notificationMethod>
         <notificationRecurrence>beginning</notificationRecurrence>
         <notificationInterval>5000</notificationInterval>
      </EventTriggerNotification>
   </EventTriggerNotificationList>
</EventTrigger>
```

Example: Schedule event detection and triggering

The command below schedules event detection and triggering from 7:00 am to 5:00 pm. every Tuesday.

```
PUT /Custom/Event/schedule HTTP/1.1
Content-Type: application/xml; charset="UTF-8"
Content-Length: xxx

<?xml version="1.0" encoding="UTF-8"?>
<EventSchedule version="1.0" xmlns="urn:psialliance-org">
   <TimeBlockList>
      <TimeBlock>
         <dayOfWeek>2</dayOfWeek>
         <TimeRange>
            <beginTime>07:00:00</beginTime>
            <endTime>17:00:00</endTime>
         </TimeRange>
      </TimeBlock>
   </TimeBlockList>
</EventSchedule>
```

5.12 /Custom/HIK/System/Network

URI	/Custom/HIK/System/Network	**Type**	Service
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
Notes			

5.12.1 /Custom/HIK/System/Network/interfaces/<ID>/pppoe

URI	/Custom/HIK/System/Network/interfaces/ID/pppoe	**Type**	Resource
Function	Get and configure PPPoE parameter.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<PPPoE>
PUT		<PPPoE>	<ResponseStatus>
Notes	<password> is a write-only field.		

PPPoEConfig XML Block

```
<PPPoE version=“1.0” xmlns=“urn:hikvision-com”>
        <enabled> <!-- req, xs:boolean --> </enabled>
        <userName> <!-- req, xs:string --> </userName>
        <password> <!-- wo, req, xs:string --> </password>
</PPPoE>
```

5.12.2 /Custom/HIK/System/Network/interfaces/<ID>/ddns

URI	/Custom/HIK/System/Network/interfaces/ID/ddns	**Type**	Resource
Function	Get and set the configuration information of DDNS.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<DDNS>
PUT		<DDNS>	<ResponseStatus>
Notes	When <provider> is “IPServer”, <serverIPAddress> is required. When <provider> is “DysDNS”, <serverIPAddress>, <domainName>, <userName>, <password> are required. When <provider> is “PeanutHall”, <domainName>, <userName>, <password> are required. <password> is a write-only field.		

DDNS XML Block

```
<DDNS version=“1.0” xmlns=“urn:hikvision-com”>
    <enabled>      <!-- req, xs:boolean --> </enabled>
    <provider>    <!-- req, xs:string, “IPServer, DynDNS, PeanutHall” -->    </provider>
    <serverIPAddress> <!—dep, xs:string --> </serverIPAddress>
    <portNo> <!-- dep, xs:integer --> </portNo>
    <domainName> <!-- dep, xs:string --> </domainName>
    <userName> <!-- dep, xs:string --> </userName>
    <password> <!-- wo, dep, xs:string --></password>
</DDNS>
```

5.13 /Custom/HIK/System/TwowayAudio

URI	/Custom/HIK/System/TwowayAudio	Type	Service
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
Notes			

5.13.1 /Custom/HIK/System/TwowayAudio/receive

URI	/Custom/HIK/System/TwowayAudio/receive	Type	Resource
Function	Receive the intercom data.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			TwowayAudio Data
Notes			

Example:

```
GET /Custom/HIK/System/TwowayAudio/receive HTTP/1.1

HTTP/1.1 200 OK
…
Content-Type: application/binary; charset=”UTF-8”
Content-Length: xxx
\r\n
TwoWayAudio Data…
```

5.13.2 /Custom/HIK/System/TwowayAudio/send

URI	/Custom/HIK/System/TwowayAudio/send	Type	Resource
Function	Send the intercom data.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
PUT		TwowayAudio Data	<ResponseStatus>
Notes			

Example:

```
PUT /Custom/HIK/System/TwowayAudio/send HTTP/1.1
…
```

```
Content-Type: application/binary; charset=”UTF-8”
Content-Length: xxx
\r\n
TwowayAudio Data…
```

5.13.3 /Custom/HIK/System/TwowayAudio/audioActivate

URI	/Custom/HIK/System/TwowayAudio/audioActivate		**Type**	Resource
Function	Open intercom.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
PUT			<ResponseStatus>	
Notes				

5.13.4 /Custom/HIK/System/TwowayAudio/audioInActivate

URI	/Custom/HIK/System/TwowayAudio/audioInActivate		**Type**	Resource
Function	Stop intercom.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
PUT			<ResponseStatus>	
Notes				

5.14 /Custom/HIK/System/Video

URI	/Custom/HIK/System/Video		**Type**	Service
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
Notes				

5.14.1 /Custom/HIK/System/Video/inputs/channels/<ID>/osdDatetime

URI	/Custom/HIK/System/Video/inputs/channels/ID/osdDatetime		**Type**	Resource
Function	Get and set the designate channel’s information about OSD.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	

GET			<OsdDatetime>
PUT		<OsdDatetime>	<ResponseStatus>
Notes	Type is the type of the year month day and should be: 0: XXXX-XX-XX Y-M-D 1: XX-XX-XXXX M-D-Y 4: XX-XX-XXXX D-M-Y displayWeek means display the week or not attribute the configuration of the OSD,the value should be: 1: transparent,flash 2: transparent, not flash 3: not transparent, flash 4: not transparent, not flash		

OsdDatetime XML Block

```
<OsdDatetime version="1.0" xmlns="urn:hikvision-com">
   <enabled> <!-- req, xs:boolean -->    </enabled>
   <posX> <!-- req, xs:integer;coordinate --> </posX>
   <posY> <!-- req, xs:integer;coordinate --> </posY>
   <type> <!-- req, xs:integer --> </type>
   <displayWeek> <!-- req, xs:boolean --> </displayWeek>
   <attribute> <!-- req, xs:integer --> </attribute>
</OsdDatetime>
```

5.14.2 /Custom/HIK/System/Video/inputs/channels/<ID>/overlays/text

URI	/Custom/HIK/System/Video/inputs/channels/ID/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

```
<TextOverlayList version="1.0" xmlns="urn:hikvision-com">
   <TextOverlay/>      <!-- opt -->
</TextOverlayList>
```

Notes:

<TextOverlayList> supports up to four <TextOveylay> now.

5.14.3 /Custom/HIK/System/Video/inputs/channels/<ID>/overlays/text/<ID>

URI	/Custom/HIK/System/Video/inputs/channels/ID/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				

TextOverlay XML Block

```
<TextOverlay version="1.0" xmlns="urn:hikvision-com">
   <id>              <!-- req, xs:string;id -->        </id>
   <enabled>             <!-- req, xs:boolean -->        </enabled>
   <posX>      <!-- req, xs:integer -->        </posX>
   <posY>      <!-- req, xs:integer -->        </posY>
   <message>      <!-- req, xs:string -->      </message>
</TextOverlay>
```

Notes:

- <id> value range is 1-4.
- <posY> value is a multiple of 16.

5.15 /Custom/HIK/System/Serial

URI	/Custom/HIK/System/Serial		Type	Service
Methods	Query String(s)	Inbound Data	Return Result	
Notes				

5.15.1 /Custom/HIK/System/Serial/ports/<ID>/transcommopen

URI	/Custom/HIK/System/Serial/ports/ID/transcommopen	Type	Resource
Function	Open transparent channel.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
PUT			<ResponseStatus>
Notes			

5.15.2 /Custom/HIK/System/Serial/ports/<ID>/transcommclose

URI	/Custom/HIK/System/Serial/ports/ID/ transcommclose	Type	Resource
Function	Close transparent channel.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
PUT			<ResponseStatus>
Notes			

Notes:

Only support RS485 transparent channel, so ID value can only be 1.

5.15.3 /Custom/HIK/System/Serial/ports/<ID>/transcommsenddata

URI	/Custom/HIK/System/Serial/ports/ID/transcommsenddata	Type	Resource
Function	Send data to the serial port.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
PUT		Raw data	<ResponseStatus>
Notes			

Example：

```
PUT /Custom/HIK/System/Serial/ports/ID/transcommsenddata HTTP/1.1
…
Content-Type: application/binary; charset=”UTF-8”
Content-Length: xxx
\r\n
```

```
Raw data…
```

5.15.4 /Custom/HIK/System/Serial/ports/<ID>/transcommrecvdata

URI	/Custom/HIK/System/Serial/ports/ID/transcommrecvdata	Type	Resource
Function	Receive data from the serial port.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			Raw data
Notes			

Example:

```
GET /Custom/HIK/System/Serial/ports/ID/transcommrecvdata HTTP/1.1

HTTP/1.1 200 OK
…
Content-Type: application/binary; charset=”UTF-8”
Content-Length: xxx
\r\n
Raw data…
```

5.16 /Custom/HIK/Security/AAA

URI	/Custom/HIK/Security/AAA	Type	Service
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
Notes			

5.16.1 /Custom/HIK/Security/AAA/users

URI	/Custom/HIK/Security/AAA/users	Type	Resource
Function	Access the device user list.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<UserList>
PUT		<UserList>	<ResponseStatus>
POST		<User>	<ResponseStatus>

DELETE			<ResponseStatus>
Notes	It is possible to add, remove and update users entries in the list. Passwords can only be uploaded - they are never revealed during GET operations.		

UserList XML Block

```
<UserList version="1.0" xmlns="urn:hikvision-com">
   <User/>       <!-- opt -->
</UserList>
```

Notes：

- Up to 16 users.
- A default user account ,“admin”, must be provided. Its default password is “12345”. Its ID is 1.

5.16.2 /Custom/HIK/Security/AAA/users/<ID>

URI	/Custom/HIK/Security/AAA/users/ID	**Type**	Resource
Function	Authentication user settings.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<User>
PUT		<User>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	<password> is a write-only field. <permission> len is 4 bytes. Every bit means one permission.“1” means own.“0” is not.The different permission and the corresponding values as follow: 0x1: Local control PTZ 0x2: Local manual record 0x4: Local play back 0x8: Local set the configuration 0x10: Local examine status and log 0x20: Local advanced operation(update,reboot) 0x40: Local examine the configuration 0x1000: Remote control PTZ 0x2000: Remote manual record 0x4000: Remote play back 0x8000: Remote set the configuration 0x10000: Remote examine the status 0x20000: Remote advanced operation(update, reboot) 0x40000: Remote launch talkandlisten 0x80000: Remote live preview 0x100000: Remote require uploading alarm and output 0x200000: Remote control local output 0x400000: Remote control serial port		

	0x800000: Remote examine the configuration Use of IPv4 or IPv6 addresses depends on the value of the <ipVersion> field in /System/Network/interfaces/ID/ipAddress.

User XML Block

```
<User version="1.0" xmlns="urn:hikvision-com">
   <id>              <!-- req, xs:string;id -->         </id>
   <userName>           <!-- req, xs:string -->         </userName>
   <password>           <!-- wo, req, xs:string -->       </password>
   <priority> <!-- opt, xs:integer; "0 – low, 1 – middle, 2 – high" --> </priority>
   <ipAddress> <!-- dep, xs:string --> </ipAddress>
   <ipv6Address> <!-- dep, xs:string --> </ipv6Address>
   <macAddress> <!-- opt, xs:string --> </macAddress>
   <permission> <!-- opt, xs:string, 4 bytes --></permission>
   <localPreviewPermission> <!-- opt, xs:boolean --> </localPreviewPermission>
   <networkPreviewPermission> <!-- opt, xs:boolean --> </networkPreviewPermission>
   <localRecordPermission> <!-- opt, xs:boolean --> </localRecordPermission>
   <networkRecordPermission> <!-- opt, xs:boolean --> </networkRecordPermission>
   <localPlaybackPermission> <!-- opt, xs:boolean --> </localPlaybackPermission>
   <networkPlaybackPermission> <!-- opt, xs:boolean --> </networkPlaybackPermission>
   <localBackupPermission> <!-- opt, xs:boolean --> </localBackupPermission>
   <networkBackupPermission> <!-- opt, xs:boolean --> </networkBackupPermission>
   <localPTZControlPermission> <!-- opt, xs:boolean --> </localPTZControlPermission>
   <networkPTZControlPermission> <!-- opt, xs:boolean --> </networkPTZControlPermission>
</User>
```

Notes：

- <id> value range is 1-16;
- <userName> maximum length is 31, and <password> maximum length is 15;
- <localPreviewPermission>, <localBackupPermission> and <networkBackupPermission> apply to DVR.
- <localPreviewPermission>, <networkPreviewPermission>, <localRecordPermission>, <networkRecordPermission>,<localPlaybackPermission>, <networkPlaybackPermission>, <localBackupPermission>, <networkBackupPermission>, <localPTZControlPermission>, <networkPTZControlPermission> is read-only.

5.17 /Custom/HIK/PTZ

URI	/Custom/HIK/PTZ	**Type**	Service
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
Notes	PTZ control service.		

5.17.1 /Custom/HIK/PTZ/channels

URI	/Custom/HIK/PTZ/channels	**Type**	Resource
Function	Access the list of PTZ channels.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<PTZChannelList>
PUT		<PTZChannelList>	<ResponseStatus>
POST		<PTZChannel>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes	PTZ channels may be hardwired, or it may be possible to create channels if the device supports it. To determine whether it is possible to dynamically PTZ channels, check the defined HTTP methods in /Custom/HIK/PTZ/channels/description.		

PTZChannelList XML Block

```
<PTZChannelList version="1.0" xmlns="urn:hikvision-com">
   <PTZChannel/>       <!-- opt -->
</PTZChannelList>
```

5.17.2 /Custom/HIK/PTZ/channels/<ID>

URI	/Custom/HIK/PTZ/channels/<ID>	**Type**	Resource
Function	Access or control a PTZ channel.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
GET			<PTZChannel>
PUT		<PTZChannel>	<ResponseStatus>
DELETE			<ResponseStatus>
Notes			

PTZChannel XML Block

```
<PTZChannel version="1.0" xmlns="urn:hikvision-com">
   <videoInputID>        <!-- req, xs:string; id -->                    </videoInputID>
   <controlProtocol>     <!—req: xs:string --> </controlProtocol>
   <controlAddress>      <!-- req: xs:string -->   </controlAddress>
   <PresetIDList>
      <id> <!-- ro, opt, xs:integer, 0.15 --> </id>
      <enabled> <!-- req, xs:boolean --> </enabled>
   </PresetIDList >
   <PatrolIDList>
      <id> <!-- ro, opt, xs:integer, 0.15 --> </id>
```

```
        <enabled> <!-- req, xs:boolean --> </enabled>
    </PatrolIDList >
    <PatternIDList>
        <id> <!-- ro, opt, xs:integer, 0.15 --> </id>
        <enabled> <!-- req, xs:boolean --> </enabled>
    </PatternIDList>
</PTZChannel>
```

5.17.3 /Custom/HIK/PTZ/channels/<ID>/patrol

URI	/Custom/HIK/PTZ/channels/<ID>/patrol		**Type**	Resource
Function	Access the list of PTZ channel Patrol.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<PTZPatrolList>	
PUT		<PTZPatrolList>	<ResponseStatus>	
POST		<PTZPatrol>	<ResponseStatus>	
DELETE			<ResponseStatus>	
Notes				

PTZPatrolList XML Block

```
<PTZPatrolList version="1.0" xmlns="urn:hikvision-com">
    <PTZPatrol>    <!—opt -->
</ PTZPatrolList >
```

5.17.4 /Custom/HIK/PTZ/channels/<ID>/patrol/<ID>

URI	/Custom/HIK/PTZ/channels/<ID>/patrol/<ID>		**Type**	Resource
Function	Access or control a PTZ channel Patrol.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<PTZPatrol>	
PUT		<PTZPatrol>	<ResponseStatus>	
DELETE			<ResponseStatus>	
Notes	<presetNo> is Preset's series number. <seqSpeed> is Patrol speed. <dwellTime> is the stay time for the patrol point, the unit is second			

PTZPatrol XML Block

```
<PTZPatrol version="1.0" xmlns="urn:hikvision-com">
    <id>     <!-- req, xs:string -->     </id>
```

```
   <presetNo> <!-- req, xs:integer -->    </ presetNo >
   <seqSpeed> <!-- req, xs:integer -->    </seqSpeed>
   <dwellTime> <!-- req, xs:integer -->    </dwellTime>
</ PTZPatrol>
```

5.17.5 /Custom/HIK/PTZ/channels/<ID>/PTZControl

URI	/Custom/HIK/PTZ/channels/<ID>/PTZControl	**Type**	Resource
Function	PTZ control.		
Methods	**Query String(s)**	**Inbound Data**	**Return Result**
PUT	command		<ResponseStatus>
Notes	The value of command is: LIGHT_PWRON: Turn on Light WIPER_PWRON: Turn on Wiper FAN_PWRON: Turn on Fan HEATER_PWRON: Turn on Heater. AUX_PWRON1: Turn on auxiliary equipment 1. AUX_PWRON2: Turn on auxiliary equipment 2 SET_PRESET: Set preset CLE_PRESET: Clear preset. ZOOM_IN: Zoom in the maximum speed. ZOOM_OUT: Zoom out in the maximum speed. FOCUS_NEAR: focus near in the maximum speed. FOCUS_FAR: focus far in the maximum speed. IRIS_OPEN: IRIS is open in the maximum speed IRIS_CLOSE: IRIS is cloesd in the maximum speed TILT_UP: PTZ is tilt up in the maximum speed TILT_DOWN: PTZ is tilt down in the maximum speed PAN_LEFT: PTZ is pan left in the maximum speed PAN_RIGHT: PTZ is pan right in the maximum speed UP_LEFT: PTZ is up-left in the maximum speed UP_RIGHT: PTZ is up-right in the maximum speed DOWN_LEFT: PTZ is down-left in the maximum speed DOWN_RIGHT: PTZ is down-right in the maximum speed PAN_AUTO: PTZ scans pan with maximum speed. FILL_PRE_PATROL: Add preset into patrol. SET_PATROL_DWELL: set the dwell time of preset. SET_PATROL_SPEED: Set the speed of patrol. CLE_PRE_PATROL: Clear preset from patrol. STA_MEM_PATTERN: Start memory pattern. STO_MEM_PATTERN: Stop memory pattern.		

	RUN_ PATTERN: Start pattern. RUN_PATROL: Start patrol. STOP_PATROL: Stop patrol. GOTO_PRESET: Go to preset.

5.18 /Custom/HIK/ShelterAlarm

URI	/Custom/HIK/ShelterAlarm		**Type**	Service
Function	Shelter alarm configuration for all video input channels.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<ShelterAlarmList>	
Notes	If shelter alarm is supported by the device, a shelter alarm ID will be allocated for each video input channel ID. The shelter alarm ID must correspond to the video input channel ID.			

ShelterAlarmList XML Block

```
<ShelterAlarmList version="1.0" xmlns="urn:psialliance-org">
   <ShelterAlarm/>          <!-- opt -->
</ShelterAlarmList>
```

5.18.1 /Custom/HIK/ShelterAlarm/<ID>

URI	/Custom/HIK/ShelterAlarm/ID		**Type**	Resource
Function	Shelter alarm configuration for a video input channel.			
Methods	**Query String(s)**	**Inbound Data**	**Return Result**	
GET			<ShelterAlarm>	
PUT		<ShelterAlarm>	<ResponseStatus>	
Notes	Note that the ID used here MUST correspond to the video input ID.			

ShelterAlarm XML Block

```
<ShelterAlarm version="1.0" xmlns="urn:hikvision-com">
   <id>   <!-- req, xs:string -->   </id>
   <enabled> <!-- req, xs:boolean --> </enabled>
   <posX> <!-- req, xs:integer --> </posX>
   <posY> <!-- req, xs:integer --> </posY>
   <width> <!-- req, xs:integer --> </width>
   <height> <!-- req, xs:integer --> </height>
</ShelterAlarm>
```

Notes：

Only support one shelter region.

Notes:

- For IP Camera, now only support one input channel. <id> associated with the input channel can only be 1.
- /Custom/HIK/PTZ Service is not supported now.