

VAPIX® version 3

HTTP API

Copyright Notice

This document is copyright protected and is the property of Axis Communications AB and may not be copied, reproduced or distributed in any way without the prior written consent of Axis Communications AB.

Terms of Use

The use of the AXIS VAPIX application programming interface (hereinafter referred to as "the INTERFACE" as further specified below, is subject to the terms and conditions of the License Agreement below. By using the INTERFACE and the written specification of the INTERFACE (hereinafter referred to as "the INTERFACE DESCRIPTION"), whether in whole or in part, you agree to be bound by the terms of the License Agreement.

VAPIX LICENSE AGREEMENT

This is a legal agreement (the "License Agreement") between you (either individual or an entity) and Axis Communications AB (hereinafter referred to as "Axis").

1. GRANT OF LICENSE

Axis hereby grants to you the right to use the INTERFACE and the INTERFACE DESCRIPTION for the sole and limited purpose of creating, manufacturing and developing a solution that integrates any unit or portion included in the product range of Axis network cameras, Axis video servers, Axis video encoders and Axis video decoders (as defined by Axis at its discretion) and to market, sell and distribute any such solution.

2. COPYRIGHT

The INTERFACE and the INTERFACE DESCRIPTION are owned by Axis and are protected by copyright laws and international treaty provisions. Any use of the INTERFACE and/or the INTERFACE DESCRIPTION outside the limited purpose set forth in Section 1 above is strictly prohibited.

3. NO REVERSE ENGINEERING

You may not reverse engineer, decompile, or disassemble the INTERFACE except to the extent required to obtain interoperability with other independently created computer programs as permitted by mandatory law.

4. TERMINATION

This License is effective until terminated. Your rights under this License will terminate automatically without notice from Axis if you fail to comply with any term(s) of this License. Upon the termination of this License, you shall cease all use and disposition of the INTERFACE and/or THE INTERFACE DESCRIPTION whether for the purpose set forth in Section 1 above or not.

5. GOVERNING LAW

This agreement shall be deemed performed in and shall be construed by the laws of Sweden. All disputes in connection with this agreement shall be finally settled by arbitration in accordance with the Rules of the Arbitration Institute of the Stockholm Chamber of

Commerce. The place of arbitration shall be Malmö, Sweden. The language of the proceedings, documentation and the award shall be English.

6. DISCLAIMER

- 6.1. THE INTERFACE AND THE INTERFACE DESCRIPTION ARE DELIVERED FREE OF CHARGE AND “AS IS” WITHOUT WARRANTY OF ANY KIND. THE ENTIRE RISK AS TO THE USE, RESULTS AND PERFORMANCE OF THE INTERFACE AND THE INTERFACE DESCRIPTION IS ASSUMED BY THE USER/YOU. AXIS DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, NON-INFRINGEMENT AND PRODUCT LIABILITY, OR ANY WARRANTY ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE WITH RESPECT TO THE INTERFACE AND THE INTERFACE DESCRIPTION.
 - 6.2. YOU ARE YOURSELF RESPONSIBLE FOR EXAMINING WHETHER THE INTERFACE AND THE INTERFACE DESCRIPTION ARE ENCUMBERED BY OR INFRINGES UPON A RIGHT HELD BY A THIRD PARTY. AXIS, WHO HAS NOT UNDERTAKEN ANY SUCH INVESTIGATIONS, HAS NO KNOWLEDGE OF NOR DOES AXIS ACCEPT ANY LIABILITY FOR ANY SUCH ENCUMBRANCES OR INFRINGEMENTS.
 - 6.3. YOU UNDERTAKE NOT TO PURSUE ANY CLAIMS WHATSOEVER AGAINST AXIS OR ITS AFFILIATES RELATING TO OR EMANATING FROM THE INTERFACE AND THE INTERFACE DESCRIPTION.
 - 6.4. AXIS SHALL NOT BE LIABLE FOR LOSS OF DATA, LOSS OF PRODUCTION, LOSS OF PROFIT, LOSS OF USE, LOSS OF CONTRACTS OR FOR ANY OTHER CONSEQUENTIAL, ECONOMIC OR INDIRECT LOSS WHATSOEVER IN RESPECT OF USE OR DISPOSITION OF THE INTERFACE AND THE INTERFACE DESCRIPTION.
 - 6.5. AXIS TOTAL LIABILITY FOR ALL CLAIMS IN ACCORDANCE WITH THE USE OF THE INTERFACE AND THE INTERFACE DESCRIPTION SHALL NOT EXCEED THE PRICE PAID FOR THE INTERFACE AND THE INTERFACE DESCRIPTION.
 - 6.6. YOU SHALL INDEMNIFY AND HOLD AXIS AND ITS AFFILIATES HARMLESS FROM ANY CLAIMS WHATSOEVER FROM ANY THIRD PARTY AGAINST AXIS OR ITS AFFILIATES RELATING TO OR EMANATING FROM YOUR USE OF THE INTERFACE AND THE INTERFACE DESCRIPTION UNDER THIS LICENSE AGREEMENT. THE FOREGOING INDEMNIFICATION INCLUDES BUT IS NOT LIMITED TO ANY AND ALL DAMAGES, COSTS AND EXPENSES (INCLUDING REASONABLE ATTORNEYS’ FEES).
-

Table of Contents

- 1 Overview7
 - 1.1 Description7
 - 1.2 History7
- 2 Prerequisites7
 - 2.1 Identification.....7
- 3 Definitions7
 - 3.1 General notation.....7
 - 3.1.1 General abbreviations7
 - 3.1.2 Style convention and general CGI URL syntax8
 - 3.1.3 Parameter value convention.....9
- 4 Interface Specification9
 - 4.1 Obsolete and removed CGIs.....9
 - 4.1.1 Obsolete.....9
 - 4.1.2 Removed.....10
 - 4.2 HTTP status codes10
 - 4.3 User access rights10
- 5 API Groups10
 - 5.1 General.....10
 - 5.1.1 Parameter management10
 - 5.1.1.1 List parameters request11
 - 5.1.1.2 List parameters response.....12
 - 5.1.1.3 List parameter definitions request.....13
 - 5.1.1.4 List parameter definitions response.....13
 - 5.1.1.5 Update parameters request16
 - 5.1.1.6 Add parameters request16
 - 5.1.1.7 Add parameters response.....18
 - 5.1.1.8 Remove parameters request.....19
 - 5.1.1.9 Remove parameter response.....19
 - 5.1.2 Add, modify and delete users.....20
 - 5.1.3 Factory default21
 - 5.1.4 Hard factory default21
 - 5.1.5 Firmware upgrade.....21
 - 5.1.6 Restart server.....22
 - 5.1.7 Server report.....22

- 5.1.8 Logs.....22
 - 5.1.8.1 System log22
 - 5.1.8.2 Access log23
- 5.1.9 System date and time23
 - 5.1.9.1 Get system date and time23
 - 5.1.9.2 Set system date and time request24
 - 5.1.9.3 Set system date and time response25
- 5.2 Image and Video25
 - 5.2.1 Image size25
 - 5.2.2 Video status26
 - 5.2.3 Bitmap27
 - 5.2.3.1 Bitmap image request.....27
 - 5.2.3.2 Bitmap image response28
 - 5.2.4 JPEG/MJPEG28
 - 5.2.4.1 JPEG image (snapshot) CGI request28
 - 5.2.4.2 JPEG image response29
 - 5.2.4.3 MJPG video CGI request29
 - 5.2.4.4 MJPG video response30
 - 5.2.4.5 JPEG and MJPG image request arguments31
 - 5.2.5 Dynamic text overlay33
- 5.3 PTZ33
 - 5.3.1 PTZ driver update34
 - 5.3.2 PTZ administration35
 - 5.3.3 PTZ control35
 - 5.3.4 PTZ configuration.....41
 - 5.3.5 Set PTZ parameters.....42
 - 5.3.6 PTZ control queue.....43
 - 5.3.7 PTZ control queue response44
- 5.4 Motion Detection Level45
- 5.5 I/O.....47
 - 5.5.1 I/O ports47
 - 5.5.2 I/O ports response49
 - 5.5.3 Virtual I/O control50
 - 5.5.3.1 Input50
- 5.6 Serial Port.....51
 - 5.6.1 Serial port control.....51

This document is copyright protected and is the property of Axis Communications AB and may not be copied, reproduced or distributed in any way without the prior written consent of Axis Communications AB.

- 5.6.2 Open serial port.....52
- 5.7 IP Filter.....53
 - 5.7.1 IP address filter administration.....53
 - 5.7.2 IP address filter administration response55
- 5.8 Audio56
 - 5.8.1 Audio MIME types.....56
 - 5.8.2 Audio data request.....56
 - 5.8.3 Singlepart audio data response.....56
 - 5.8.4 Multipart audio data response57
 - 5.8.5 Transmit audio data.....58
- 6 References.....59

© 2008 Axis Communications AB. AXIS COMMUNICATIONS, AXIS, ETRAX, ARTPEC and VAPIX are registered trademarks of Axis Communications AB. All other company names and products are trademarks or registered trademarks of their respective companies. We reserve the right to introduce modifications without notice.

1 Overview

1.1 Description

This document specifies the external HTTP-based application programming interface of the Axis cameras and video encoders with firmware version 5.00 and above.

The HTTP-based video interface provides the functionality for requesting single and multi-part images and for getting and setting internal parameter values. The image and CGI-requests are handled by the built-in web server.

1.2 History

Version	Date	Comment
3.00	2008-Dec-1	Initial version

2 Prerequisites

2.1 Identification

Supported APIs are product and firmware dependent. Please refer to the Release notes for compliance information.

Property: Properties.API.Version=3

Firmware: 5.00 and above

3 Definitions

This section contains information on general usage of this document.

3.1 General notation

3.1.1 General abbreviations

The following abbreviations are used throughout this document

CGI	Common Gateway Interface – a standardized method of communication between a client (e.g. a web browser) and a server (e.g. a web server).
TBD	To be done/designed – signifies that the referenced section/subsection/entity is intended to be specified, but has not reached a level of maturity to be public at this time.
N/A	Not applicable - the feature/parameter/value is of no use in a specific task.
URL	A Uniform Resource Location (URL) is a compact string representation for a resource available via the Internet. RFC 1738 describes the syntax and semantics for a URL.

URI A Uniform Resource Identifier (URI) is a compact string of characters for identifying an abstract or physical resource. [RFC 3986](#) describes the generic syntax of URI.

3.1.2 Style convention and general CGI URL syntax

In URL syntax and in descriptions of CGI arguments, text in italics within angle brackets denotes content that should be replaced with either a value or a string. When replacing the text string, the angle brackets must also be replaced. For example, the name of the camera or video encoder is denoted by *<servername>* in the URL syntax description. In the URL syntax examples *<servername>* is replaced by the string myserver.

Note: This does not apply to responses in XML format where a text string within angle brackets (including the brackets) is a tag (start-tag or end-tag). XML response descriptions use text in italics inside square brackets to denote content that is replaced by the server. For example, [*int*] is replaced by an integer.

CGI URLs are written in lower-case. CGI arguments are written in lower-case and as one word. When the CGI request includes internal camera parameters, the internal parameters must be written exactly as named in the camera or video encoder. For the POST method, the parameters must be included in the body of the HTTP request. The CGIs are organized in function-related directories under the axis-cgi directory. The file extension is required.

Note: The HTTP protocol is used in syntax descriptions and examples throughout this document. It is also possible to use the HTTPS protocol. To enable HTTPS, set the HTTPS.Enabled parameter to yes.

URL syntax is written with the word "**Syntax:**" in bold face, followed by a box with the referred syntax, as shown below. The name of the camera/video encoder is written as *<servername>*. This is intended to be replaced with the name of the actual camera or video encoder. The name can either be a name, e.g. "thecam" or "thecam.adomain.net" or the associated IP number for the server, e.g. 10.10.2.139. Text within square brackets denotes content that can be omitted.

Syntax:

```
http://<servername>/axis-cgi/<subdir>[</subdir>...]/<cgi>.<ext>
[?<argument>=<value>[&<argument>=<value>...]]
```

Example: List the Network parameters.

```
http://myserver/axis-cgi/param.cgi?action=list&group=Network
```

A description of the returned data is written with "**Return**" in bold face, followed by the HTTP status code, header fields and a box with the HTTP body. Carriage Return and Line Feed (CRLF) are not explicitly printed. The HTTP status codes are described in section 4.2.

Return:

HTTP Code: *<code>*

Content-Type: *<type>*

Body:

```
<text>
```

URL syntax examples are written with "**Example:**" in bold face, followed by a short description and a box with the example.

Example: Request default image.

```
http://myserver/axis-cgi/jpg/image.cgi
```

Examples of what can be returned by the server from a request are written with "**Example:**" in bold face, followed by a short description and a box with an example of the returned data.

Example: Successful request, default image returned.

HTTP Code: 200 OK

Content-Type: image/jpeg

Content-Length: 12345

Body:

```
<JPEG image data>
```

Note: Response examples are examples only. The returned data will differ depending on product model and configuration.

3.1.3 Parameter value convention

In tables defining CGI arguments and supported values, the default value for optional arguments is system configured.

4 Interface Specification

4.1 Obsolete and removed CGIs

The differences between VAPIX version 2 and VAPIX version 3 are described in the Migration Guide available at http://www.axis.com/techsup/cam_servers/dev/cam_http_api_index.php

4.1.1 Obsolete

Some CGI requests, arguments and values in this document may be obsolete and are provided for backward compatibility. These might not be supported in the future.

The /admin, /operator and /view paths are obsolete. User access rights are instead determined by user group membership, see section 4.3. The default access rights remain unchanged from VAPIX version 2.

4.1.2 Removed

The HTTP API version 1 (VAPIX 1) is no longer supported. The 3 CGIs `getparam.cgi`, `setparam.cgi` and `audio/getparam.cgi` are removed and replaced by `param.cgi`.

The `buffer/command.cgi` has been removed.

4.2 HTTP status codes

The built-in Web server returns standard HTTP status codes. See [RFC 1945](#) and [RFC 2616](#).

4.3 User access rights

User access rights for CGI requests are determined by group membership.

Security level	Description
viewer	Users with viewer, operator or administrator rights can access this functionality.
operator	Users with operator or administrator rights can access this functionality.
admin	Users with administrator rights can access this functionality.

5 API Groups

Most API requests described in this document are product or release dependent. To help developers to get an idea of which API requests that are supported for different products, the requests are grouped together. Information about supported groups can be found in the product-specific Release notes document, available for download at Axis web site.

5.1 General

All video products with firmware version 5.00 and above support the requests specified in the General section.

5.1.1 Parameter management

Most features in Axis network video products can be configured using camera parameters. The camera parameters are managed with the general parameter handling API `param.cgi`.

Notes:

- Supported parameters are product/release dependent.
- Parameters have different security levels as described in the Parameter specification and API functionality descriptions, available at http://www.axis.com/techsup/cam_servers/dev/cam_http_api_index.php
- The URL must follow the standard way of writing a URL, ([RFC 3986](#): Uniform Resource Identifiers (URI) Generic Syntax); that is, spaces and other reserved characters (";", "/", "?", ":", "@", "&", "=", "+", ",", "\$") within a *<argument>* or a *<value>* must be replaced with %<ASCII hex>. The ISO/IEC 8859-1 character set is supported. For example,

in the string ©Axis Communications the copyright symbol must be replaced with %A9 and the space with %20, i.e. %A9Axis%20Communications.

Security level: The CGI can be accessed by all users, but some requests require operator or admin rights. See the following sections for details.

Method: GET/POST

Syntax:

```
http://<servername>/axis-cgi/param.cgi?
<argument>=<value>[&<argument>=<value>...]
```

with the following arguments and values

Argument	Valid values	Description
action=<string>	add, remove, update, list, listdefinitions	Add, remove, update or list parameters. See the following subsections for more information and examples.
Additional arguments depending on the selected action, see the following sections.		

5.1.1.1 List parameters request

List parameters and their values. See also section 5.1.1.3.

Security level: Parameter dependent. The CGI can be accessed by all users, but a user can only list parameters that are accessible to that user (determined by the parameter security level).

Syntax:

```
http://<servername>/axis-cgi/param.cgi?action=list[&<argument>=<value>...]
```

with the following arguments and values

Argument	Valid values	Description
group=<string>[,<string>...]	<group[.name]>[, <group[.name]>...]	Returns the value of the camera parameter named <group>.<name>. If <name> is omitted, all the parameters of the <group> are returned. The camera parameters must be entered exactly as they are named in the camera or video encoder. Wildcard (*) can be used when listing parameters. See example below. If this parameter is omitted, all parameters in the device are returned.
responseformat	rfc	Get the HTTP response format according to standard. See the response section below.

Argument	Valid values	Description
usergroup	anonymous, viewer, operator, admin	Forces a certain user access level. This might be necessary if the browser has cached the credentials or if “anonymous viewer login” is enabled.

Example 1: List the Network parameters.

```
http://myserver/axis-cgi/param.cgi?action=list&group=Network
```

Example 2: List the names of all Event parameters (this request requires operator access).

```
http://myserver/axis-cgi/param.cgi?action=list&group=Event.*.Name
```

5.1.1.2 List parameters response

1. Success

A list of parameter-value pairs is returned.

Return:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
<parameter>=<value>  
<parameter>=<value>  
...
```

Example: Properties query response (only a small part of the response is shown here).

```
root.Properties.API.HTTP.Version=3  
root.Properties.API.HTTP.AdminPath=/operator/basic.shtml  
root.Properties.Audio.Audio=yes
```

2. Error

If the CGI request includes an invalid parameter, the server returns an error message.

Return:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
# Error: <description>
```

5.1.1.3 List parameter definitions request

List parameter definitions. The response includes parameter name, value, security level, nice name and valid values (where applicable).

Security level: Parameter dependent. The CGI can be accessed by all users, but a user can only list parameters that are accessible to that user (determined by the parameter security level).

Syntax:

```
http://<servername>/axis-cgi/param.cgi?action=listdefinitions
[&<argument>=<value>...]
```

with the following arguments and values

Argument	Valid values	Description
listformat=<string>	xmlschema	Response format.
group=<string>[, <string>...]	<group[.name]>[, <group[.name]>...]	Returns the camera parameter named <group>.<name>. If <name> is omitted, all the parameters of the <group> are returned. The camera parameters must be entered exactly as they are named in the camera or video encoder. Wildcard (*) can be used when listing parameters. See example below. If this parameter is omitted, all parameters in the device are returned.

Example: List the Properties parameters.

```
http://myserver/axis-cgi/param.cgi?action=listdefinitions
&listformat=xmlschema&group=Properties
```

5.1.1.4 List parameter definitions response

Successful request: Response in XML format

Return:

HTTP Code: 200 OK

Content-Type: text/xml

Body:

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<parameterDefinitions [attributes] >
  <model> [string] </model>
  <firmwareVersion> [int.int] </firmwareVersion>
  <group name="[string]">
    [additional group name start-tags]
    parameter
    parameter
    ...
  [additional group name end-tags]
```

```

    </group>
    ...
</parameterDefinitions>
where
parameter=
<parameter name="[string]" value="[value]" securityLevel="[int]"
    niceName="[string]">
    <type [attributes] >
        ...
    </type>
</parameter>

```

Parameter information is contained in the <parameter> element. The attribute “value” contains the current parameter value. All valid values and their type (integer, bool, enum, string) are listed within the <type> element. See examples below.

Note: Attribute securityLevel will be replaced by accessControl in future HTTP API updates.

The security level (attribute securityLevel) consists of 4 integers in order create, delete, read and write. To perform an action on a parameter a user must have an access right equal to or higher than the corresponding security level of that parameter. See also examples below. The following integers are used:

Security level	Description
0	Unprotected, but it is not possible to access the camera/encoder from outside without at least viewer rights.
1	Viewer access
4	Operator access
6	Administrator access
7	Root access. Internal parameters that can be changed by firmware applications or by root editing the configuration files directly.

Example 1: The Properties.API.HTTP parameter. Security level 7707 means that all users can read this parameter, but root access is required to create, delete and write the parameter.

HTTP Code: 200 OK

Content-Type: text/xml

Body:

```

<?xml version="1.0" encoding="iso-8859-1" ?>
<parameterDefinitions
  xmlns="http://www.axis.com/ParameterDefinitionsSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.axis.com/ParameterDefinitionsSchema
  http://10.13.24.132/pub/parameterdefinitions.xsd"
  version="1.0">
  <model>AXIS P1311</model>
  <firmwareVersion>5.00</firmwareVersion>
  <group name="Properties">
    <group name="API">

```

```
<group name="HTTP">
  <parameter name="Version" value="3" securityLevel="7707"
    niceName="Version">
    <type readonly="true" const="true">
      <int />
    </type>
  </parameter>
</group>
</group>
</group>
</parameterDefinitions>
```

Example 2: The AudioSource.A0.AudioEncoding parameter. This parameter has 3 valid values. Security level 7714 means that root access is required to create/delete the parameter, reading requires viewer rights and writing requires operator rights.

HTTP Code: 200 OK

Content-Type: text/xml

Body: (only a portion of the body is shown here)

```
<group name="AudioSource">
  <group name="A0">
    <parameter name="AudioEncoding" value="g711" securityLevel="7714"
      niceName="Audio encoding">
      <type>
        <enum>
          <entry value="g711" niceValue="G711 &micro;-law" />
          <entry value="g726" niceValue="G726" />
          <entry value="aac" niceValue="AAC" />
        </enum>
      </type>
    </parameter>
  </group>
</group>
```

Example 3: The Image.IO.Appearance.Compression parameter is an integer with valid values between 0 and 100.

HTTP Code: 200 OK

Content-Type: text/xml

Body: (only a portion of the body is shown here)

```
<group name="Image">
  <group name="IO">
    <group name="Appearance">
      <parameter name="Compression" value="50" securityLevel="7744"
        niceName="Compression">
        <type>
          <int min="0" max="100" maxlen="3" />
        </type>
      </parameter>
    </group name="Appearance">
  </group name="IO">
</group name="Image">
```

5.1.1.5 Update parameters request

Set new parameter values.

Security level: Parameter dependent. The CGI can be accessed by users with operator or admin rights, but operators can only update parameters that are accessible to operators (determined by the parameter security level).

Security level: operator

Syntax:

```
http://<servername>/axis-cgi/param.cgi?action=update  
[&<argument>=<value>...]
```

with the following arguments and values

Argument	Valid values	Description
<string>=<string>	<group.name>=<value>	Assigns <value> to the parameter <group.name>. The <value> must be URL-encoded when it contains non-alphanumeric characters. The camera parameters must be entered exactly as named in the camera or the video encoder.

Example 1: Set the default image resolution to 320x240 pixels.

```
http://myserver/axis-cgi/param.cgi?action=update  
&Image.I0.Resolution=320x240
```

Example 2: Set the maximum number of viewers to 5.

```
http://myserver/axis-cgi/param.cgi?action=update&Image.MaxViewers=5
```

5.1.1.6 Add parameters request

Add new parameters.

Note: Only applicable for dynamic parameter groups such as events, motion detection windows and stream profiles.

Security level: operator

Syntax:

```
http://<servername>/axis-cgi/param.cgi?action=add  
[&<argument>=<value>...]
```

with the following arguments and values

Argument	Valid values	Description
template=<string>	<template> ¹	Use the specified <template> when creating the new group. The template is a file describing all parameters for this group. See examples below.

Argument	Valid values	Description
group=<string>	<group>	Specifies the parent group. The parent group defines where in the parameter structure the new group will be created. For example, if adding an event (template=event) and specify group=Event the new group will be available as Event.E<number>. Where <number> is the unique number for the group (see return values below). The character before <number> is generated from the last section of the group name. E.g. Event will generate the character E and Event.Notification will generate the character N.
<string>=<string>	<group.name>=<value>	Set a parameter in the newly created group. As the group number is not known before the group is created, the id-number is simply left out, see the examples below. The new group number is created dynamically and can be any number. This is why all parameters are specified to set without any group number. The base path to the parameter is specified as <group>.<uppercase first letter of group>.<parameter name>.
force	yes	The force parameter can be used to exceed limits set for adding dynamic parameter groups. Example: Axis products can be configured for up to 10 event types. The force parameter can be used to exceed this maximum number of events.

¹ Product/release-dependent. Check the product's Release notes.

Example 1: Create a new event under the group Event; use H.264, set the name to “My Event” and the Enabled parameter to yes.

```
http://myserver/axis-cgi/param.cgi?action=add
&group=Event&template=event
&Event.E.Name=My%20Event
&Event.E.Enabled=yes
&Event.E.VideoFormat=h264
```

Example 2: A listing of the new group will output the following.

```
root.Event.E0.Name=My Event
root.Event.E0.Type=T
root.Event.E0.Enabled=yes
root.Event.E0.Priority=1
root.Event.E0.Image=0
root.Event.E0.HWInputs=xxxx
root.Event.E0.SWInput=
root.Event.E0.Weekdays=1111111
root.Event.E0.Starttime=00:00
```

```
root.Event.E0.Duration=24:00
root.Event.E0.MinimumTriggerInterval=00:00:00
root.Event.E0.MinimumTriggerTimePeriod=00:00:00
root.Event.E0.ImageURLSettingsEnabled=no
root.Event.E0.ImageURLSettings=
root.Event.E0.IncludePreTrigger=no
root.Event.E0.PreTriggerDuration=0
root.Event.E0.PreTriggerDurationUnit=s
root.Event.E0.IncludePostTrigger=no
root.Event.E0.PostTriggerDuration=0
root.Event.E0.PostTriggerDurationUnit=s
root.Event.E0.IncludeBestEffort=no
root.Event.E0.BestEffortDuration=0
root.Event.E0.BestEffortDurationUnit=s
root.Event.E0.IncludeAudio=no
root.Event.E0.MPEGPreTriggerDuration=5
root.Event.E0.MPEGPostTriggerDuration=2
root.Event.E0.VideoFormat=h264
root.Event.E0.FrameRate=25fps
```

Note that in this example the id is E0. This can be any number, depending on if other events were added before. Parameters that are not specified in the request will have their default values.

5.1.1.7 Add parameters response

The add action produces one of the following server responses:

1. Success

Return:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
<entry> OK
```

2. Failure – No group created

The group could not be created due to missing or erroneous CGI arguments.

Return:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
<additional error information>
# Request failed: <error message>
```

3. Failure – Parameters could not be set

The group was created, but the specified parameters could not be set.

Return:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
<additional error information>
# Error: <error message>
<entry> OK
```

5.1.1.8 Remove parameters request

Delete parameters.

Note: Only applicable for dynamic parameter groups such as the event parameters.

Security level: operator

Syntax:

```
http://<servername>/axis-cgi/param.cgi?action=remove
[&<argument>=<value>...]
```

with the following arguments and values

Argument	Valid values	Description
group=<string>[,<string>...]	<group>[,<group>]	Deletes the specified group(s).

Example: Delete event groups E2 and E4.

```
http://myserver/axis-cgi/param.cgi?action=remove
&group=Event.E2,Event.E4
```

5.1.1.9 Remove parameter response

The remove action produces one of the following server responses:

1. Success

Return:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
OK
```

2. Failure

Return:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
<additional error information>
# Request failed: <error message>
```

5.1.2 Add, modify and delete users

Add a new user with password and group membership, modify the information and remove a user.

Security level: admin

Method: GET/POST

Syntax:

```
http://<servername>/axis-cgi/pwdgrp.cgi?
<argument>=<value>[&<argument>=<value>...]
```

with the following arguments and values

Argument	Valid values	Description
action=<string>	add, update, remove, get	add = create a new user account. update = change account information of specified parameters if the account exists. remove = remove an existing account if it exists. get = get a list of the users which belong to each group defined.
user=<string>	A string	The user account name, a non-existing user name. Valid characters are a thru z, A thru Z and 0 thru 9.
pwd=<string>	A string	The unencrypted password (1-8 characters) for the account. All ascii characters are valid.
grp=<string>	A string	An existing primary group name of the account.
sgrp=<string>:[<string>...]]	<string>[,<string>...]	Colon separated existing secondary group names of the account. See example below.
comment=<string>	A string	The comment field of the account.

Example 1: Create a new administrator account.

```
http://myserver/axis-cgi/pwdgrp.cgi?action=add
&user=joe&pwd=foo&grp=users&sgrp=admin:operator:viewer&comment=Joe
```

Example 2: Change the password of an existing account.

```
http://myserver/axis-cgi/pwdgrp.cgi?action=update&user=joe&pwd=bar
```

Example 3: Remove an account.

```
http://myserver/axis-cgi/pwdgrp.cgi?action=remove&user=joe
```

Example 4: List groups and users.

```
http://myserver/axis-cgi/pwdgrp.cgi?action=get
```

5.1.3 Factory default

Reload factory default. All parameters except Network.BootProto, Network.IPAddress, Network.SubnetMask, Network.Broadcast and Network.DefaultRouter are set to their factory default values.

Security level: admin

Method: GET

Syntax:

```
http://<servername>/axis-cgi/factorydefault.cgi
```

5.1.4 Hard factory default

Reload factory default. All parameters are set to their factory default value.

Security level: admin

Method: GET

Syntax:

```
http://<servername>/axis-cgi/hardfactorydefault.cgi
```

5.1.5 Firmware upgrade

Upgrade the firmware version.

Security level: admin

Method: POST

Syntax:

```
http://<servername>/axis-cgi/firmwareupgrade.cgi[?<argument>=<value>]
```

with the following arguments and values

Argument	Valid values	Description
type=<string>	normal, factorydefault	Specifies the type of firmware upgrade. normal = Upgrade and restore old settings. factorydefault = Upgrade and discard all settings. Default: normal.

The file content is provided in the HTTP body according to the format given in [RFC 1867](#). The body is created automatically by the browser if using HTML form with input type "file".

```
POST /axis-cgi/firmwareupgrade.cgi?type=normal HTTP/1.0
Content-Type: multipart/form-data; boundary=AsCg5y
Content-Length: <content length>
```

```
--AsCg5y
Content-Disposition: form-data; name="firmware.bin";
filename="firmware.bin"
Content-Type: application/octet-stream

<firmware file content>

--AsCg5y--
```

5.1.6 Restart server

Restart server.

Security level: admin

Method: GET

Syntax:

```
http://<servername>/axis-cgi/restart.cgi
```

5.1.7 Server report

This CGI request generates and returns a server report. This report is useful as an input when requesting support. The report includes product information, parameter settings and system logs.

Security level: admin

Method: GET

Syntax:

```
http://<servername>/axis-cgi/serverreport.cgi
```

Return:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
<server report>
```

5.1.8 Logs

5.1.8.1 System log

Retrieve system log information. The level of information included in the log is set in the Log.System parameter group.

Security level: admin

Method: GET

Syntax:

```
http://<servername>/axis-cgi/systemlog.cgi
```

Return:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
<system log information>
```

5.1.8.2 Access log

Retrieve client access log information. The level of information included in the log is set in the Log.Access parameter group.

Security level: admin

Method: GET

Syntax:

```
http://<servername>/axis-cgi/accesslog.cgi
```

Return:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
<access log information>
```

5.1.9 System date and time

Get or set the system date and time.

Security level: admin

Method: GET/POST

Syntax:

```
http://<servername>/axis-cgi/date.cgi?<argument>=<value>
```

with the following argument and values

Argument	Valid values	Description
action=<string>	get, set	get = get the current date and time. set = set the current date and/or time. See the following sections for more information.

5.1.9.1 Get system date and time

Security level: admin

Syntax:

```
http://<servername>/axis-cgi/date.cgi?action=get
```

This document is copyright protected and is the property of Axis Communications AB and may not be copied, reproduced or distributed in any way without the prior written consent of Axis Communications AB.

Return:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
<month> <day>, <year> <hour>:<minute>:<second>
```

Example:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
Oct 03, 2008 15:16:04
```

5.1.9.2 Set system date and time request

Change the system date and time.

Note: The settings made here require that the camera or video encoder is not configured to synchronize with an NTP server or with the computer time.

Security level: admin

Syntax:

```
http://<servername>/axis-cgi/date.cgi?action=set[&<argument>=<value>...]
```

with the following arguments and values

Argument	Valid values	Description
year=<int>	1970 ... 2031	Current year.
month=<int>	1 ... 12	Current month.
day=<int>	1 ... 31	Current day.
hour=<int>	0 ... 23	Current hour.
minute=<int>	0 ... 59	Current minute.
second=<int>	0 ... 59	Current second.
timezone=<string>	GMT	Specifies the time zone that the new date and/or time is given in. The camera translates the time into local time using whichever time zone has been specified through the web configuration. If omitted the new date and/or time is assumed to be in local time. Currently only GMT is considered valid input. The rest of the time zones are subject to future expansion.

Example: Set the date to October 5, 2008.

```
http://myserver/axis-cgi/date.cgi?action=set&year=2008&month=10&day=5
```

5.1.9.3 Set system date and time response

The set action produces one of the following server responses:

1. Success

Return:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
OK
```

2. Failure

Settings or syntax are probably incorrect.

Return:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
Request failed: <error message>
```

5.2 Image and Video

5.2.1 Image size

Retrieve the actual image size with default image settings or with given parameters.

Security level: viewer

Method: GET/POST

Syntax:

```
http://<servername>/axis-cgi/imagesize.cgi?  
<argument>=<value>[&<argument>=<value>...]
```

with the following arguments and values

Argument	Valid values	Description
camera=<string>	1 ... , ¹ quad ¹	Video source number or the quad stream. Note: This parameter is required.
Any image argument affecting image size.		See 5.2.4.5 for additional image CGI arguments.

¹ Product-dependent. Check the product's Release notes.

This document is copyright protected and is the property of Axis Communications AB and may not be copied, reproduced or distributed in any way without the prior written consent of Axis Communications AB.

Return: Image height and width in pixels.

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
image width = <value>
image height = <value>
```

Example 1: Request image size of default settings from camera 1.

```
http://myserver/axis-cgi/imagesize.cgi?camera=1
```

Example 2: Request image size with supplied parameters for camera 1.

```
http://myserver/axis-cgi/imagesize.cgi?camera=1&resolution=QCIF
&rotation=90&squarepixel=1
```

Example 3: Returned data after a successful request.

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
image width = 176
image height = 144
```

5.2.2 Video status

Video encoders only. Check the status of one or more video sources.

Security level: viewer

Method: GET

Syntax:

```
http://<servername>/axis-cgi/videostatus.cgi?<argument>=<value>
```

with the following argument and values

Argument	Valid values	Description
status=<int>[[,<int>],...]	1 ... ¹	Check status of the listed video sources.

¹ Product-dependent. Check the product's Release notes.

Return:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
Video 1 = <information>
...
```

Example 1: Request video status from input 1, 2, 3 and 4.

```
http://myserver/axis-cgi/videostatus.cgi?status=1,2,3,4
```

Example 2: Returned data after a successful request.

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
Video 1 = video
Video 2 = no video
Video 3 = no video
Video 4 = video
```

5.2.3 Bitmap

Support for bitmap images is product-dependent. Use the list action in param.cgi (see section 5.1.1.1) on the parameter Properties.Image.Format to check supported image formats.

5.2.3.1 Bitmap image request

Request a bitmap image.

Security level: viewer

Method: GET

Syntax:

```
http://<servername>/axis-cgi/bitmap/image.bmp
[?<argument>=<value>[&<argument>=<value>...]]
```

with the following arguments and values

Argument	Valid values	Description
resolution=<string>	A string ¹	Resolution of the returned image. Check the product's Release notes for supported resolutions. Examples: 320x240, 4CIF
camera=<string>	1 ... , ¹ quad ¹	Select a video source or the quad stream.
squarepixel=<int>	0, 1	Enable/disable square pixel correction. Applies only to video encoders.

¹ Product/release-dependent. Check the product's Release notes.

Example 1: Request a bitmap image from the default camera using default settings.

```
http://myserver/axis-cgi/bitmap/image.bmp
```

Example 2: Request a bitmap image containing four video sources using default settings.

```
http://myserver/axis-cgi/bitmap/image.bmp?camera=quad
```

Example 3: Request a bitmap image from camera 1 with a resolution of 320x240.

```
http://myserver/axis-cgi/bitmap/image.bmp?resolution=320x240&camera=1
```

5.2.3.2 Bitmap image response

When a bitmap image is requested, the server either returns the specified bitmap image file or an error.

1. Success

Return:

HTTP Code: 200 OK

Content-Type: image/bitmap

Content-Length: <image size>

Body:

```
<bitmap image data>
```

2. Failure – Bad request

If the specified parameter values are invalid, the server returns 400 Bad Request.

Return:

HTTP Code: 400 Bad Request

Body:

```
<body>
```

5.2.4 JPEG/MJPEG

The requests specified in the JPEG/MJPEG section are supported by those video products that use JPEG and MJPG encoding.

5.2.4.1 JPEG image (snapshot) CGI request

Request a JPEG image (snapshot) with specified properties.

Security level: viewer

Method: GET

Syntax:

```
http://<servername>/axis-cgi/jpg/image.cgi  
[?<argument>=<value>[&<argument>=<value>...]]
```

with the following arguments and values

Argument	Valid values	Description
JPEG and MJPEG image arguments, see section 5.2.4.5.		

Example 1: Request a JPEG image from camera 1 with a resolution of 320x240 and compression of 25.

```
http://myserver/axis-cgi/jpg/image.cgi?resolution=320x240
&camera=1&compression=25
```

Example 2: Request a JPEG image from camera 2 with the text My Camera displayed.

```
http://myserver/axis-cgi/jpg/image.cgi?camera=2
&text=1&textstring=My%20Camera
```

5.2.4.2 JPEG image response

When a JPEG image is requested, the server returns either the specified JPEG image file or an error.

1. Successful request

Return:

HTTP Code: 200 OK

Content-Type: image/jpeg

Content-Length: <image size>

Body:

```
<JPEG image data>
```

2. Failure – Bad request

If the specified parameter values are invalid, the server returns 400 Bad Request.

Return:

HTTP Code: 400 Bad Request

Body:

```
<body>
```

5.2.4.3 MJPG video CGI request

Request a Multipart-JPEG image stream (video) with specified properties. The properties can be specified explicitly, or a pre-defined stream profile can be used. Image settings saved in a stream profile can be overridden by specifying new settings after the stream profile argument.

Security level: viewer

Method: GET

Syntax:

```
http://<servername>/axis-cgi/mjpg/video.cgi
[?<argument>=<value>[&<argument>=<value>...]]
```

with the following arguments and values

Argument	Valid values	Description
streamprofile=<string>	<stream profile name>	Use a predefined stream profile. Supported stream profile names are stored in the StreamProfile.S#.Name parameters.
duration=<int> ¹	An unsigned integer	Specifies for how many seconds the video will be generated and pushed to the client. 0 = unlimited.
nbrofframes=<int>	An unsigned integer	Specifies how many frames the server will generate and push. 0 = unlimited.
fps=<int>	An unsigned integer	Using fps it is possible to specify the frame rate from the server. 0 = unlimited.

Additional JPEG and MJPG image arguments, see section 5.2.4.5.

¹ Product/release-dependent. Check the product's Release notes.

Example 1: Request a Multipart JPEG image stream from camera 1 with a resolution of 320x240 and compression of 25.

```
http://myserver/axis-cgi/mjpg/video.cgi?resolution=320x240
&camera=1&compression=25
```

Example 2: Request a Multipart JPEG image stream from camera 1 with a frame rate of 5.

```
http://myserver/axis-cgi/mjpg/video.cgi?fps=5
```

Example 3: Request a Multipart JPEG image stream using the Bandwidth stream profile but with a lower resolution.

```
http://myserver/axis-cgi/mjpg/video.cgi?
streamprofile=Bandwidth&resolution=CIF
```

5.2.4.4 MJPG video response

1. Successful request

If the request was successful, the server returns a continuous flow of JPEG files. The content type is "multipart/x-mixed-replace" and each image ends with a boundary string <boundary>. The returned image and HTTP data is equal to the request for a single JPEG image.

Return:

HTTP Code: 200 OK

Content-Type: multipart/x-mixed-replace; boundary=<boundary>

Body:

```
--<boundary>
<image>

where the returned <image> field is

Content-Type: image/jpeg
Content-Length: <image size>

<JPEG image data>
--<boundary>
<image>
```

2. Failure – Bad request

If the specified parameter values are invalid, the server returns 400 Bad Request.

Return:

HTTP Code: 400 Bad Request

Body:

```
<body>
```

Example: Requested Multipart JPEG image.

HTTP Code: 200 OK

Content-Type: multipart/x-mixed-replace; boundary=myboundary

Body:

```
--myboundary
Content-Type: image/jpeg
Content-Length: 15656

<JPEG image data>
--myboundary
Content-Type: image/jpeg
Content-Length: 14978

<JPEG image data>
--myboundary
Content-Type: image/jpeg
Content-Length: 15136

<JPEG image data>
--myboundary
```

5.2.4.5 JPEG and MJPG image request arguments

The following arguments and values can be used in JPEG and MJPG CGI requests:

Argument	Valid values	Description
resolution=<string>	A string ¹	Resolution of the returned image. For supported resolutions, Check the product's Release notes. Examples: 320x240, 4CIF

Argument	Valid values	Description
camera=<string>	1 ... , ¹ quad ¹	Selects the source camera or the quad stream.
compression=<int>	0 ... 100 ¹	Adjusts the compression level of the image. Higher values correspond to higher compression, i.e. lower quality and smaller image size. Note: This value is internally mapped and is therefore product-dependent.
clock=<int>	0, 1	Shows/hides the time stamp. 0 = hide, 1 = show.
date=<int>	0, 1	Shows/hides the date. 0 = hide, 1 = show.
text=<int>	0, 1	Shows/hides the text. 0 = hide, 1 = show.
textstring=<string> ¹	A string	The text shown in the image, the string must be URL encoded.
textcolor=<string> ¹	black, white	The color of the text shown in the image.
textbackgroundcolor=<string> ¹	black, white, transparent, semitransparent	The color of the text background shown in the image.
rotation=<int>	0, 90 ¹ , 180, 270 ¹	Rotate the image clockwise.
textpos=<string>	top, bottom	The position of the string shown in the image.
overlayimage=<int> ¹	0, 1	Enable/disable overlay image.
overlaypos=<int>,<int> ¹ overlaypos=<int>x<int> ²	Two unsigned integers	The x and y coordinates defining the position of the overlay image.
squarepixel=<int> ¹	0, 1	Enable/disable square pixel correction. Applies only to video encoders.

¹ Product/release-dependent. Check the product's Release notes.

² Obsolete.

5.2.5 Dynamic text overlay

Set or get dynamic text overlay in the image.

Note: To use this functionality set Image.I#.Text.TextEnabled to 'yes' to enable text overlay and set Image.I#.Text.String to '#D' to use dynamic text in the overlay. The '#' in I# should be replaced by an integer starting from zero. The '#' in #D is a modifier and should not be replaced.

Security level: operator

Method: GET

Syntax:

```
http://<servername>/axis-cgi/dynamicoverlay.cgi?<argument>=<value>
[&<argument>=<value>...]
```

with the following arguments and values

Argument	Valid values	Description
action=<string>	settext, gettext	gettext = get the dynamic text overlay. settext = set the dynamic text overlay.
text=<string>	A string	The overlay text to apply, only applicable with action=settext.
camera=<string>[, <string>,...]	1 ... , ¹ quad ¹	Select video source or the quad stream. Default: default camera.

¹ Product-dependent. Check the product's Release notes.

Example 1: Set the dynamic text overlay to "Test text" for cameras 1, 3 and 4.

```
http://myserver/axis-cgi/dynamicoverlay.cgi?
action=settext&camera=1,3,4&text=Test%20text
```

Example 2: Get the dynamic text overlay for camera 2 and quad view.

```
http://myserver/axis-cgi/dynamicoverlay.cgi?action=gettext
&camera=2,quad
```

5.3 PTZ

The requests in this section are supported by products with Pan/Tilt/Zoom capability.

Note: Installing a PTZ driver is done in three main steps:

1. Upload the driver
2. Associate the driver with a serial port
3. Connect cameras to the serial port

PTZ driver update (section 5.3.1) can accomplish step 1 and 2 in one request. PTZ administration (section 5.3.2) can accomplish step 2 if the driver is already present in the product. Open serial port (section 5.6.2) accomplishes step 3.

5.3.1 PTZ driver update

Installation and removal of PTZ drivers.

Security level: admin

Method: POST

Syntax:

```
http://<servername>/axis-cgi/ptz/ptzupdate.cgi?  
[<argument>=<value>[&<argument>=<value>...]]
```

with the following arguments and values

Argument	Valid values	Description
replacedriver=<string>	<driver name>	Replace the currently installed driver <driver name>. <driver name> is a driver name from the parameter group PTZ.PTZDrivers.Driver# where # is the index of the installed driver. See below for rules concerning driver replacement.
port=<int>	1, ... ¹	Associate the driver with this serial port. See below for rules concerning driver replacement.
force=<string>	yes	Force driver installation even if the driver is potentially incompatible with firmware (if for instance a relevant library has been considerably updated) but may still work well. This argument has no effect on serious incompatibilities such as wrong CPU architecture.

¹ Product-dependent. Check the product's Release notes.

The file content, if any, is provided in the HTTP body according to the format given in [RFC 1867](#). The body is created automatically by the browser if using HTML form with input type "file". Empty file content will result in the removal of the driver only.

The following rules apply for driver replacement, in descending order of priority:

1. If argument replacedriver is provided, the given driver will be uninstalled.
2. If argument port is provided, the driver currently associated with that serial port, if any, is uninstalled.
3. If the maximum number of installed drivers (equals the number of serial ports) is not reached, no driver is uninstalled.
4. Any driver not currently associated with any serial port will be uninstalled.

Example: The driver "Videmech" is currently associated with serial port 1. A client sends the request below to uninstall the driver "Videmech" and install the "Philips" driver. The argument port=1 (for the serial port) and the file content of the installation file Philips-1.0.ptz are provided in the HTTP body. for the new driver. The new driver will be associated with serial port 1.

```
POST /axis-cgi/ptz/ptzupdate.cgi HTTP/1.0  
Content-Type: multipart/form-data; boundary=AaBo3x  
Content-Length: <content length>
```

```
--AaBo3x
Content-Disposition: form-data; name="port"

1
--AaBo3x
Content-Disposition: form-data; name="Upload Driver"; filename="Philips-
1.0.ptz"
Content-Type: text/plain
<file content of Philips-1.0.ptz>
--AaBo3x--
```

5.3.2 PTZ administration

This CGI handles relations between drivers and serial ports. When associating a driver with a serial port, any driver currently associated with that port is first disassociated. The new driver is then activated to control units connected to the serial port. Any disassociated driver is automatically stopped.

Security level: admin

Method: GET/POST

Syntax:

```
http://<servername>/axis-cgi/com/ptzadmin.cgi?
<argument>[=<value>][&<argument>=<value>...]
```

with the following arguments and values

Argument	Valid values	Description
port=<int>	1, ... ¹	Selects the serial port.
ptzdrivername=<string>	<driver name>	Associates a driver name with a serial port. If "none" the current driver (if any) is disassociated from the serial port. <driver name> is a driver name from the parameter group PTZ.PTZDrivers.Driver

¹Product-dependent. Check the product's Release notes.

Example: Associate driver "Videmech" with serial port 2.

```
http://myserver/axis-cgi/com/ptzadmin.cgi?port=2&ptzdrivername=Videmech
```

5.3.3 PTZ control

Control the pan, tilt and zoom behavior of a PTZ unit.

Important:

Some PTZ units automatically reduce pan and tilt movements as the zoom factor increases. Therefore, the actual movement may be less than what is requested of these units.

The PTZ control is device-dependent. For information about supported parameters and actual parameter values, please check the specification of the Axis PTZ driver you intend to use. The following table is only an overview.

Security level: viewer with access to PTZ controls

Method: GET/POST

Syntax:

```
http://<servername>/axis-cgi/com/ptz.cgi?
<argument>=<value>[&<argument>=<value>...]
```

with the following arguments and values

Argument	Valid values	Description
camera=<int>	1 ... ¹	Selects the source camera. If omitted, the default camera is used.
whoami=<string>	A string	Returns the name of the system-configured device driver.
center=<int>,<int>	<x>,<y>	<p>Absolute: Used to send the coordinates for the point in the image where the user clicked. This information is then used by the server to calculate the pan/tilt move required to (approximately) center the clicked point.</p> <p>Relative: Used to send the coordinates for the point in the image where the user clicked. This information is then used by the server to calculate the direction and number of degrees to move. The number of degrees increases with the distance from the center of the image to the point clicked.</p> <p>Digital: Used to send the coordinates for the point in the image where the user clicked. This information is then used by the server to center the clicked point.</p>
areazoom=<int>,<int>,<int>	<x>,<y>,<z>	<p>Absolute: Centers on positions x,y (like the center command) and zooms by a factor of z/100. If z is more than 100 the image is zoomed in (for example; z=300 zooms in to 1/3 of the current field of view). If z is less than 100 the image is zoomed out (for example; z=50 zooms out to twice the current field of view).</p> <p>Relative: N/A</p> <p>Note 1: In some camera models, the precision of areazoom can be strongly improved by calibrating the lens offset parameters.</p>

Argument	Valid values	Description
		Note 2: The HTTP API for area zoom is currently only supported by Axis PTZ and Dome cameras.
imagewidth=<int>	1, ... ¹	Required in conjunction with center and areazoom if the image width displayed is different from the default size of the image, which is product-specific.
imageheight=<int>	1, ... ¹	Needed in conjunction with center and areazoom if the image height is different from the default size, which is product-specific.
move=<string>	home, up, down, left, right, upleft, upright, downleft, downright,	Absolute: Moves the device 5 degrees in the specified direction. Relative: Moves the device approx. 50-90 degrees ² in the specified direction. Digital: Moves the image 25% of the image field width in the specified direction. Note: home is only valid if any home position has been previously set with "home=yes".
pan=<float>	-180.0 ... 180.0	Absolute: Pans the device relative to the (0,0) position. Relative: N/A Digital: Pans the device relative to the (0,0) position.
tilt=<float>	-180.0 ... 180.0	Absolute: Tilts the device relative to the (0,0) position. Relative: N/A Digital: Tilts the device relative to the (0,0) position.
zoom=<int>	1 ... 9999	Absolute: Zooms the device n steps. Relative: N/A Digital: Zooms the device n steps.
focus=<int>	1 ... 9999	Absolute: Move Focus n steps. Relative: N/A Digital: N/A
iris=<int>	1 ... 9999	Absolute: Move iris n steps. Relative: N/A Digital: N/A

Argument	Valid values	Description
brightness=<int> ¹	1 ... 9999	Absolute: Move brightness n steps. Relative: N/A Digital: N/A
rpan=<float>	-360.0 ... 360.0	Absolute: Pans the device n degrees relative to the current position. Relative: Pans the device approx. n degrees relative to the current position. Digital: Pans the device n degrees relative to the current position.
rtilt=<float>	-360.0 ... 360.0	Absolute: Tilts the device n degrees relative to the current position. Relative: Tilts the device approx. n degrees relative to the current position. Digital: Tilts the device n degrees relative to the current position.
rzoom=<int>	-9999 ... 9999	Absolute: Zooms the device n steps relative to the current position. Positive values mean zoom in, negative values mean zoom out. Relative: Zooms the device approx. n steps relative to the current position. Positive values mean zoom in, negative values mean zoom out. Digital: Zooms the device n steps relative to the current position. Positive values mean zoom in, negative values mean zoom out.
rfocus=<int>	-9999 ... 9999	Absolute: Move Focus n steps relative to the current position. Positive values mean focus far, negative values mean focus near. Relative: Move Focus approx. n steps relative to the current position. Positive values mean focus far, negative values mean focus near. Digital: N/A
riris=<int>	-9999 ... 9999	Absolute: Move iris n steps relative to the current position. Positive values mean open iris, negative values mean close iris. Relative: Move iris approx. n steps relative to the current position. Positive values mean open iris, negative values mean close iris. Digital: N/A

Argument	Valid values	Description
rbrightness=<int> ¹	-9999 ... 9999	<p>Absolute: Move brightness n steps relative to the current position. Positive values mean brighter image, negative values mean darker image.</p> <p>Relative: Move brightness approx. n steps relative to the current position. Positive values mean brighter image, negative values mean darker image.</p> <p>Digital: N/A</p>
autofocus=<string>	on, off	Autofocus on/off. Digital: N/A
autoiris=<string>	on, off	Autoiris on/off. Digital: N/A
continuouspantiltmove=<int>,<int>	-100 ... 100, -100 ... 100	Continuous pan/tilt motion. Positive values mean right (pan) and up (tilt), negative values mean left (pan) and down (tilt). "0,0" means stop. Values as <pan speed>,<tilt speed>
continuouszoommove=<int>	-100 ... 100	Continuous zoom motion. Positive values mean zoom in and negative values mean zoom out. "0" means stop.
continuousfocusmove=<int>	-100 ... 100	Continuous focus motion. Positive values mean focus near and negative values mean focus far. "0" means stop. Digital: N/A
continuousirismove=<int>	-100 ... 100	Continuous iris motion. Positive values mean iris open and negative values mean iris close. "0" means stop. Digital: N/A
continuousbrightnessmove=<int> ¹	-100 ... 100	Continuous brightness motion. Positive values mean brighter image and negative values mean darker image. "0" means stop. Digital: N/A
auxiliary=<string>	<function name>	Activates/deactivates auxiliary functions of the device where <function name> is the name of the device-specific function. Digital: N/A

Argument	Valid values	Description
gotoserverpresetname= <string>	<preset name> ³	Move to the position associated with the <preset name>.
gotoserverpresetno=<int>	1, ... ³	Move to the position associated with the specified preset position number.
gotodevicepreset=<int>	<preset pos> ³	Bypasses the presetpos interface and tells the device to go directly to the preset position number <preset pos> stored in the device, where the <preset pos> is a device-specific preset position number. Digital: N/A
speed=<int>	1 ... 100	Sets the head speed of the device that is connected to the specified camera. Digital: N/A
imagerotation=<int>	0, 90, 180, 270	If PTZ command refers to an image stream that is rotated differently than the current image setup, then the image stream rotation must be added to each command with this argument to allow the server to compensate.
ircutfilter=<string> ¹	auto ¹ , on, off	Controls the IR cut filter. auto = automatically switch between on and off depending on the lighting conditions on = apply the filter, i.e. block IR light off = remove the filter, i.e. allow IR light to reach the image sensor
backlight=<string> ¹	on, off	Controls backlight compensation. on = bright mode off = normal mode
query=<string>	speed, position ² , presetposcam, presetposall	Returns the current parameter values.
info=<int>	1	Returns a description of available PTZ commands. No PTZ control is performed.

¹ Product/release-dependent. Check the product's Release notes.

² Driver-specific.

³ Preset positions are configured using com/ptzconfig.cgi, see section 5.3.4.

Example: Request information about which PTZ commands are available for camera 3.

```
http://myserver/axis-cgi/com/ptz.cgi?info=1&camera=3
```

5.3.4 PTZ configuration

Set and configure PTZ preset positions. On Screen Display (OSD) control.

Security level: admin

Method: GET/POST

Syntax:

```
http://<servername>/axis-cgi/com/ptzconfig.cgi?
<argument>=<value>[&<argument>=<value>...]
```

with the following arguments and values

Argument	Valid values	Description
camera=<int>	1 ... ¹	The video source. If omitted, the default camera is used.
osdmenu=<string>	open, close, up, down, left, right, select, back	Commands to control the OSD menu in the camera. Note that support for different commands, and the behavior of the commands, are driver dependent. Digital: N/A
setserverpresetname=<string>	<preset name> ¹	Associates the current position to <preset name> as a preset position in the server.
setserverpresetno=<int>	1, ...	Saves the current position as a preset position number in the server.
home=<string>	yes	Makes the current position the home position for the camera. Used with <i>setserverpresetname</i> or <i>setserverpresetno</i> .
removeserverpresetname=<string>	<preset name> ¹	Removes the specified preset position associated with <preset name>.
removeserverpresetno=<int>	1, ...	Removes the specified preset position.
setdevicepreset=<int>	<preset pos>	Bypasses the presetpos interface and tells the device to save its current position as preset position <preset pos> directly in the device,

Argument	Valid values	Description
		where <i><preset pos></i> is a device-specific preset position number. Digital: N/A

¹*<preset name>* is a string with a maximum of 31 characters, ~ is not allowed.

5.3.5 Set PTZ parameters

Set PTZ parameters. To list the PTZ parameters use param.cgi, see section 5.1.1.

Security level: operator

Method: GET/POST

Syntax:

```
http://<servername>/axis-cgi/com/ptzparam.cgi?
<argument>=<value>[&<argument>=<value>...]
```

with the following arguments and values

Argument	Valid values	Description
port=<int>	1, ... ¹	Selects the serial port. Used together with grpdefault below, for group "Serial".
camera=<int>	1, ... ¹	Selects the camera. If omitted, the default camera is used. Used together with grpdefault below, for all groups but "Serial".
<PTZ parameter> ² = <value>	Any parameter in the groups PTZ.Limit.L#, where # denotes the camera number, e.g. PTZ_Limit_L2_MaxPan	Sets a PTZ limit parameter. Max values are adjusted so that they are greater than or equal to min values, e.g. PTZ.Limit.L2.MaxPan >= PTZ.Limit.L2.MinPan.
pardefault=<string> ²	Any parameter in the group: Serial.Ser# where # denotes the serial port number, and the groups: PTZ.Support.S# PTZ.Limit.L# PTZ.Various.V# PTZ.UserBasic.U# PTZ.UserAdv.U# where # denotes the camera number.	Sets the default value for a parameter. Note: No max value validation is performed as with previous CGI argument.
grpdefault=<string>	Serial, Support,	Sets the default value for a parameter group. The default values are taken from

Argument	Valid values	Description
	Limit, Various, UserBasic, UserAdv	the driver currently associated with the port or camera.

¹ Product-dependent. Check the product's Release notes.

² The path shall have the format root_PTZ..., e.g. root_PTZ_Limit_L1_MaxPan for parameter PTZ.Limit.L1.MaxPan. The reason for this format is related to internal ssi implementation.

In the examples below, assume the driver installed on index 1 is associated with serial port 2 (ttyS1) and camera 3 is mapped to serial port 2.

Example 1: Set PTZ.Limit.L3.MaxPan to 110 and adjust it so that it is >= PTZ.Limit.L3.MinPan.

```
http://myserver/axis-cgi/com/ptzparam.cgi?root_PTZ_Limit_L3_MaxPan=110
```

Example 2: Set PTZ.Limit.L3.MaxPan to the value of PTZ.Driver1.Limit.L0.MaxPan.

```
http://myserver/axis-cgi/com/ptzparam.cgi
?pardefault=root_PTZ_Limit_L3_MaxPan
```

Example 3: Copy the parameter values from group PTZ.Driver1.Limit.L0 to group PTZ.Limit.L3.

```
http://myserver/axis-cgi/com/ptzparam.cgi?camera=3&grpdefault=Limit
```

Example 4: Copy the parameter values from group PTZ.Driver1.Serial.S0 to group Serial.Ser1.

```
http://myserver/axis-cgi/com/ptzparam.cgi?port=2&grpdefault=Serial
```

5.3.6 PTZ control queue

If the PTZ control queuing mechanism is enabled for the camera concerned, control of PTZ units is limited to the client currently possessing the control, if any. This CGI request handles requests concerning the control queue. Note that cookies are enabled by default when enabling PTZ control queue.

Security level: viewer with access to PTZ controls

Method: GET/POST

Syntax:

```
http://<servername>/axis-cgi/com/ptzqueue.cgi?
<argument>=<value>[&<argument>=<value>...]
```

with the following arguments and values

Argument	Valid values	Description
control=<string>	request, drop,	"request" requests PTZ control. "drop" drops the PTZ control or leaves the queue.

Argument	Valid values	Description
	query	"query" reports the current status for the client. For possessing clients with no peers existing in the queue, "request" will reset the control timer. For all other clients, "request" will have the same effect as "query".
camera=<int>	1, ... ¹	The video source number. If omitted, the default camera is used.

¹Product-dependent. Check the product's Release notes.

Example: Request PTZ control for camera 2.

```
http://myserver/axis-cgi/com/ptzqueue.cgi?control=request&camera=2
```

5.3.7 PTZ control queue response

The 200 OK response on success for "request" and "query" has a format that enables simple javascript implementations using the DOM API and should be easy to parse for components such as ActiveX as well.

1. Success

Return:

HTTP Code: 200 OK

Content-Length: <length>

Body:

```
<a name="<pos>"></a><a name="<seconds>"></a><a name="<period>"></a>
```

where

<pos> can have a value from 0 to the maximum value of how many clients are allowed in the queue. This value is the given position in the queue. 0 means that the client is not in the queue, 1 means control is possessed. For 0 the other parameters are undefined.

<seconds> is the estimated number of seconds remaining, i.e. for position 1 the remaining control time and for other positions, the time until position 1 is reached. -1 denotes that the time remaining to get control cannot be estimated. This means that a client in the queue has the "TimeoutType" set to infinity.

<period> is the recommended time in seconds when the client should send a new "control=query" requests. To stay active in the queue the client must regularly send PTZ requests to the camera. An inactive client will automatically be removed from the queue.

2. Failure

On failure no anchor elements are provided but simply the error message in plain text.

Return:

HTTP Code: 200 OK

Content-Length: <length>

Body:

```
Error:
<error information>
```

Example: Control requested.

HTTP Code: 200 OK

Content-Length: 63

Body:

```
<a name="3"></a><a name="410"></a><a name="5"></a>
```

This means the client was assigned queue position 3. The expected number of seconds until control is possessed is 410 and the recommended time until the next request is 5 seconds.

5.4 Motion Detection Level

Retrieve the current motion detection level from all or specific motion detection include windows. Alternatively, retrieve all motion detection levels related to a specific motion detection window configuration. A motion detection window is defined by a dynamic parameter group Motion.M<group number>.

Note: Obsolete functionality. This CGI is most likely to be replaced in future VAPIX® versions.

Security level: viewer

Method: GET/POST

Syntax:

```
http://<servername>/axis-cgi/motion/motiondata.cgi[?<argument>=<value>...]
```

with the following arguments and values

Argument	Valid values	Description
group=<int>[,<int>, ...]	<group number>[,<group number>, ...]	Specify the motion detection windows that are of interest. Excluding the group argument will return all motion detection level information from all motion detection windows. Exclude windows are ignored.
<string>=<string>	<parameter name>=<value>	Get the motion detection levels related to a specific window configuration. The last part of the parameters in the parameter group Motion.M# (where # is a number) can be used, e.g. Sensitivity and History.

Return:

HTTP Code: 200 OK

Content-Type: multipart/x-mixed-replace; boundary=<boundary>

Body:

```
--<boundary>
<motion levels>

where the returned <motion levels> part is
Content-Type: text/plain

<motion level for window with lowest group number>
--<boundary>

and <motion level for window with group number n> is
group=<group number n>;level=<motion level for n>;threshold=
<threshold level for n>;
[ <motion level for window n+1> ]
```

Example 1: Get motion detection levels related to motion detection windows defined within Motion.M0 and Motion.M1.

```
http://myserver/axis-cgi/motion/motiondata.cgi?group=0,1
```

The example returns the following.

HTTP Code: 200 OK

Content-Type: multipart/x-mixed-replace; boundary=axismdb

Body:

```
--axismdb
Content-Type:text/plain

group=0;level=28;threshold=45;
group=1;level=43;threshold=25;
--axismdb
Content-Type:text/plain

group=0;level=54;threshold=45;
group=1;level=38;threshold=25;
--axismdb
Content-Type:text/plain

group=0;level=49;threshold=45;
group=1;level=19;threshold=25;
--axismdb
...
```

Example 2: Get motion detection levels related to a specific window configuration. The group number related to this motion detection window configuration will always be X.

```
http://myserver/axis-cgi/motion/motiondata.cgi?
Name=Temp&ImageSource=2&WindowType=include
&Left=3000&Right=5000
&Top=3000&Bottom=5000
&Sensitivity=65&History=50&ObjectSize=35
```

Response:

HTTP Code: 200 OK

Content-Type: multipart/x-mixed-replace; boundary=axismdb

Body:

```
--axismdb
Content-Type:text/plain

group=X;level=16;threshold=35;
--axismdb
Content-Type:text/plain

group=X;level=55;threshold=35;
--axismdb
Content-Type:text/plain

group=X;level=39;threshold=35;
--axismdb
...
```

- If no Motion Detection windows are defined or only exclude windows are defined, HTTP/1.0 204 No Content is returned.
- If any errors are found in the CGI request, HTTP/1.0 400 Bad Request is returned.
- If too many clients try to get motion data, HTTP/1.0 503 Service Unavailable is returned.

5.5 I/O

The requests specified in the I/O section are supported by products that have input/output connectors. The number of I/O ports is product dependent.

For products with configurable ports, each physical port can be configured to act as input or output using the parameters in the IOPort group.

Numbering of I/O ports

The numbering of the I/O ports starts from *one* in port.cgi requests and in all responses except in the response to the monitor request. In the response to the monitor request, port numbering starts from *zero*. In the IOPort.I# parameter groups, port numbering starts from *zero*.

The physical port labeled *n* is thus *n* in port.cgi requests but *n-1* in the response to the monitor request. The corresponding parameter group is IOPort.I(*n-1*).

Note: The port.cgi described below replaces input.cgi and output.cgi from VAPIX version 2. The input.cgi and output.cgi are obsolete but supported for backwards compatibility. For products with configurable ports, port.cgi must be used.

5.5.1 I/O ports

Retrieve information about port status and directions, activate/deactivate and monitor ports.

Port numbering

In port.cgi requests and in all responses *except* the response to the monitor request, port numbering (*id* below) starts from *one* (where one is the physical port labeled '1'). In the *monitor response* port numbering starts from *zero*.

Security level: viewer

Method: GET

Syntax:

```
http://<servername>/axis-cgi/io/port.cgi?
<argument>=<value>[&<argument>=<value>...]
```

with the following arguments and values

Argument	Valid values	Description
check=<int>[,<int>,...]	<id1>[,<id2>,...]	Return the status (1 or 0) of one or more ports numbered <id1>,<id2>, ... 1=active, 0=inactive
checkactive=<int>[,<int>, ...]	<id1>[,<id2>,...]	Return the status (active or inactive) of one or more ports numbered <id1>, <id2>, ... This value depends on the parameters Active for an output and Trig for an input. If the port is an output and Active is configured as closed, then this request will return active if the port state is closed.
checkdirection=<int>[,<int>,...]	<id1>[,<id2>,...]	Return the port direction (input or output) of one or more ports numbered <id1>,< id2>, ...
monitor=<int>[,<int>,...]	<id1>[,<id2>,...]	Return a multipart stream of “check” ports (see return description below). Input and output ports must be monitored separately.
action=<string> ¹	[<id>]:<a>[<wait><a>...]	Set the port relay <id> active or inactive and wait <wait> milliseconds. <id> = Port number. Default: Output 1 <a> = Action character: / or \ /=active, \=inactive <wait> = delay in milliseconds Example: 1:/ sets output 1 active

¹ Valid for output ports only.

Example 1: Check if port 3 is active.

```
http://myserver/axis-cgi/io/port.cgi?checkactive=3
```

Example 2: Activate port 1. (Only valid if port 1 is configured as output.)

```
http://myserver/axis-cgi/io/port.cgi?action=1:/
```

Example 3: For port 1, set two 300 ms pulses with 500 ms delay between the pulses. (Only valid if port 1 is configured as output.)

```
http://myserver/axis-cgi/io/port.cgi?action=1:/300\500/300\
```

Example 4: Monitor data on ports 1 and 2. (Only valid when both ports have the same direction.)

```
http://myserver/axis-cgi/io/port.cgi?monitor=1,2
```

5.5.2 I/O ports response

1. Success: All arguments *except* monitor

Return:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
port<id>=<information>
```

Note: For the action argument, the body is empty.

2. Success: The monitor request

Non-empty boundaries are sent when the port status changes. If there are no changes, empty boundaries are sent at 15-second intervals.

Return:

HTTP Code: 200 OK

Content-Type: multipart/x-mixed-replace; boundary=<boundary>

Body:

```
--<boundary>
<monitor data>

where the returned <monitor data> is
Content-Type: text/plain

<id><port>:<action character>

--<boundary>
<monitor data>
```

Here <id> is the port number and <port> is I for inputs and O for outputs. The action character is / or H for active and \ or L for inactive ports.

3. Examples

Example 1: The check argument

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
port1=1  
port2=0
```

Example 2: The checkdirection argument

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
port1=input  
port2=output
```

Example 3: Monitor data on port 3 (configured as input). Note how port numbering starts from zero.

HTTP Code: 200 OK

Content-Type: multipart/x-mixed-replace; boundary=ioboundary

Body:

```
--ioboundary  
Content-Type: text/plain  
  
2I:/  
  
--ioboundary  
Content-Type: text/plain  
  
2I:H  
  
--ioboundary  
Content-Type: text/plain  
  
2I:\  
  
--ioboundary  
Content-Type: text/plain  
  
2I:L  
  
--ioboundary  
Content-Type: text/plain  
  
...  
  
...
```

5.5.3 Virtual I/O control

5.5.3.1 Input

The virtual input command is used to simulate activation or deactivation of an input connector. This can be used to trigger an event and is also useful for testing purposes.

Security level: viewer

Method: GET

Syntax:

```
http://<servername>/axis-cgi/io/virtualinput.cgi?<argument>=<value>
```

with the following arguments and values

Argument	Valid values	Description
action=<string>	[<id>]:<a>[<wait><a>...]	<p>Simulate activation/deactivation of an input connector by setting the virtual input <id> active or inactive and, optionally, wait <wait> milliseconds before the next action.</p> <p><id> = Input number. If omitted, input 1 is selected.</p> <p><a> = Action character: / or \</p> <p>/ = active, \ = inactive.</p> <p><wait> = delay in milliseconds</p>

Example 1: Set Input 1 active.

```
http://myserver/axis-cgi/io/virtualinput.cgi?action=1:/
```

Example 2: Pulse the Input. First set it active, wait 2 seconds, then set it inactive again.

```
http://myserver/axis-cgi/io/virtualinput.cgi?action=1:/2000\
```

5.6 Serial Port

The requests specified in the Serial Port section are supported by products with PTZ support.

5.6.1 Serial port control

Control serial port.

Security level: viewer

Method: GET/POST

Syntax:

```
http://<servername>/axis-cgi/com/serial.cgi?
<argument>=<value>[&<argument>=<value>... ]
```

with the following arguments and values

Argument	Valid values	Description
port=<int>	1, ... ¹	Select COM port. -1 means disconnect the camera from the serial port.
write=<string>	<bytestring>	<p><bytestring>: hex coded bytes with values {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, a, b, c, d, e, f}</p> <p>Writes the specified data string to the selected serial port.</p>

Argument	Valid values	Description
		Max string length: 128 bytes ¹ .
writestring=<string>	<URL encoded string>	Writes the URL-encoded string to the selected serial port. Max string length: 128 bytes ¹ .
read=<int>	1, ...	Reads n bytes from the selected serial port. The returned data will be hexadecimal coded and placed between #s (e.g. #3A#)
wait=<int>	1 - 9	Specified in seconds. Used together with the "read" parameter. A read is terminated when the specified number of bytes is read or when the wait period has ended.
timeout=<int>	1 - 9000	Specified in milliseconds. Used together with the "read" parameter. A read is terminated when the specified number of bytes is read or the timeout has expired.

¹ Product-dependent. Check the product's Release notes.

5.6.2 Open serial port

Open the serial port using the HTTP protocol. Authentication is handled by the Web server. After an initial connect command from the client, the connection is kept alive until the client closes it. Several clients may be connected concurrently to the same serial port. After the connection has been set up, data sent from the client to the Web server is forwarded to the serial port, and incoming serial data is returned to all connected clients.

Syntax:

```
http://<servername>/axis-cgi/com/serial.cgi?
<argument>=<value>[&<argument>=<value>...]
```

with the following arguments and values

Argument	Valid values	Description
port=<int>	1, ... ¹	Select COM port. -1 = disconnect the camera from any serial port.
camera=<int> unit=<int>	1, ... ¹	Selects the source camera or external unit. If omitted, and "port=" command is also omitted, the default camera/unit is used to determine the serial port to use.
connect=<string>	yes	Instructs the server to keep the connection open and serve as a link between the client and the serial port.

¹ Product-dependent. Check the product's Release notes.

Example 1: Open serial port 1 if not already opened and connect camera 2 to it.

```
http://myserver/axis-cgi/com/serial.cgi?port=1&camera=2
```

Example 2: Disconnect camera 2 from any serial port (but keep that port open).

```
http://myserver/axis-cgi/com/serial.cgi?port=-1&camera=2
```

5.7 IP Filter

The requests specified in the IP filter section are supported by products that support IP address filtering.

5.7.1 IP address filter administration

Allow or deny the listed IP addresses to access the Axis device.

Security level: admin

Method: GET

Syntax:

```
http://<servername>/axis-cgi/ipfilter.cgi?
<argument>=<value>[&<argument>=<value>... ]
```

with the following arguments and values

Argument	Valid values	Description
action=<string>	add, remove, removeall, update, list	add = Add new IP address (or addresses). remove = remove an entry in the IP address list. removeall = Remove all IP addresses. The IP address filtering function will automatically be disabled. update = Update settings for the IP address filtering function. list = List the settings for the IP address filtering function.
ipaddress=<string>[%20<string>...]	<IP addresses>	A space separated list of IP addresses and network addresses in the CIDR notation (IP address/netmask bits). Note: If accessing the Axis device via a proxy server, the proxy server's IP address must be added to the list of allowed addresses.
enable=<string>	yes, no	Enable/disable the IP address filtering function.
policy=<string>	allow, deny	Allow or deny access for the addresses in the list. Default: allow Note: The policy used will apply for all the addresses

Argument	Valid values	Description
		in the list!
verify=<string>	yes, no	<p>Verify that you are able to access the Axis device with the new settings. If the verification fails, the old settings will be kept.</p> <p>Default: verify is automatically set to yes for requests that risk to make the device inaccessible, i.e. setting enable to yes, or adding, updating and removing IP addresses when the filter function is enabled. In the same way verify is by default set to no when using requests that do not risk to make the device inaccessible, i.e. listing filter settings or adding, updating and removing IP addresses when the filter function is disabled. This default behavior can be changed by using the verify argument in the request.</p> <p>yes = Use verification. no = Do not use verification.</p> <p>Warning: Setting verify=no may cause the Axis device to be inaccessible.</p>

Example 1: Add a list of IP addresses and enable the IP address filtering function. Verification that the device is still accessible will automatically be done.

```
http://myserver/axis-cgi/ipfilter.cgi?action=add
&ipaddress=10.13.10.12%2010.13.17.0/24&enable=yes
```

Example 2: List settings for the IP address filtering function.

```
http://myserver/axis-cgi/ipfilter.cgi?action=list
```

Example 3: Remove an entry in the list of addresses. Verification will automatically be done if the IP filter function is enabled.

```
http://myserver/axis-cgi/ipfilter.cgi?action=remove&ipaddress=10.13.10.12
```

Example 4: Add 10.13.10.12 to the list of addresses which will be allowed access to the device.

```
http://myserver/axis-cgi/ipfilter.cgi?action=add&ipaddress=10.13.10.12
&policy=allow&enable=yes
```

Example 5: Add 10.13.10.12 to the list of addresses which will be denied access to the device.

```
http://myserver/axis-cgi/ipfilter.cgi?action=add&ipaddress=10.13.10.12
&policy=deny&enable=yes
```

Example 6: Remove all IP addresses and automatically disable the IP address filtering function

```
http://myserver/axis-cgi/ipfilter.cgi?action=removeall
```

5.7.2 IP address filter administration response

1. A successful add, remove, removeall or update.

Return:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
OK
```

2. A successful list

Return:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
Accept addresses: <IP addresses>  
Enabled: <yes/no>
```

3. Verification failed

Settings did not change.

Return:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
Verification failed: IP address "<IP address>" is not an accepted address
```

Example: List settings for the IP address filtering function, set verify to yes to check that the IP addresses are accepted.

```
http://myserver/axis-cgi/ipfilter.cgi?action=list&verify=yes
```

Only the IP address 10.13.10.12 is an accepted address. The computer 10.13.17.245 will not be able to access the device if enabling the IP address filtering function.

Response:

HTTP Code: 200 OK

Content-Type: text/plain

Body:

```
Accept addresses: 10.13.10.12  
Enabled: no  
Verification failed: IP address "10.13.17.245" is not an accepted address
```

5.8 Audio

The requests specified in the audio section are supported by products with audio capability. Audio can be transmitted on its own or together with the video stream. In addition to the HTTP streaming described in this document, audio can be transmitted over RTP/RTSP.

5.8.1 Audio MIME types

Supported MIME types for audio:

MIME	Description
audio/basic	G.711 μ -law 64 kbit/s
audio/G726-24	G.726 24 kbit/s
audio/G726-32	G.726 32 kbit/s
audio/mpeg4-generic	AAC

AAC is usually transferred through RTP/RTSP; transfer via HTTP is supported but not recommended. AAC is not supported for multipart audio.

5.8.2 Audio data request

Request an audio stream.

Security level: viewer

Method: GET

Syntax:

```
http://<servername>/axis-cgi/audio/receive.cgi[?<argument>=<value>]
```

with the following argument and values

Argument	Valid values	Description
httpstype=<string>	singlepart, multipart	Choose streaming method. Some proxies require multipart streaming. Default: As defined by the parameter Audio.A#.HTTPMessageType

Example: Request a singlepart audio stream

```
http://myserver/axis-cgi/audio/receive.cgi?httpstype=singlepart
```

5.8.3 Singlepart audio data response

1. Successful request

If the request was successful, the server returns a continuous flow of audio packets. The content type is only set at the beginning of the connection. When the connection is up and running audio packets will come one after another without any extra information between the packets.

Return:

HTTP Code: 200 OK

Content-Type: <audio MIME>

Body:

```
<Audio data>
```

2. Failure – Bad request

If the specified parameter value is invalid, the server returns 400 Bad Request.

Return:

HTTP Code: 400 Bad Request

Body:

```
<body>
```

Example: Singlepart audio data encoded with G.711 μ -law.

HTTP Code: 200 OK

Content-Type: audio/basic

Body:

```
<Audio data>  
<Audio data>  
<Audio data>  
...
```

5.8.4 Multipart audio data response

1. Successful request

If the request was successful, the server returns a continuous flow of audio packets. The content type is "multipart/x-mixed-replace" and each audio packet ends with a boundary string. The message body contains a block of binary data. The content length provides the size of each block of coded audio which varies for different codecs: G.711 has 512 bytes block size, G.726 32 kbit/s has 256 bytes and G.726 24 kbit/s has 192 bytes. AAC is not supported.

Return:

HTTP Code: 200 OK

Content-Type: multipart/x-mixed-replace; boundary=<boundary>

Body:

```
--<boundary>  
<audio>  
  
where the returned <audio> field is  
  
Content-Type: <audio MIME>  
Content-Length: <audio packet size>  
  
<Audio data>  
--<boundary>  
<audio>
```

2. Failure – Bad request

If the specified parameter values are invalid, the server returns 400 Bad Request.

Return:

HTTP Code: 400 Bad Request

Body:

```
<body>
```

Example: Multipart audio data encoded with G.726 32kbit/s.

HTTP Code: 200 OK

Content-Type: multipart/x-mixed-replace; boundary=myboundary

Body:

```
--myboundary
Content-Type: audio/G726-32
Content-Length: 256

<Audio data>
--myboundary
Content-Type: audio/G726-32
Content-Length: 256

<Audio data>
--myboundary
Content-Type: audio/G726-32
Content-Length: 256

<Audio data>
--myboundary
Content-Type: audio/G726-32
Content-Length: 256

<Audio data>
--myboundary
...
```

5.8.5 Transmit audio data

Transmit a singlepart audio data stream.

Security level: viewer

Method: POST

Syntax:

```
http://<servername>/axis-cgi/audio/transmit.cgi
```

There are no arguments to this CGI.

Example: Singlepart audio data transmit with G.711 μ -law (authorization omitted)

```
POST /axis-cgi/audio/transmit.cgi HTTP/1.0
Content-Type: audio/basic
Content-Length: 9999999
```

```
Connection: Keep-Alive
Cache-Control: no-cache

<Audio data>
<Audio data>
<Audio data>
...
```

6 References

HTTP Protocol

Hypertext Transfer Protocol – HTTP/1.0 <http://www.w3.org/Protocols/rfc1945/rfc1945>

Hypertext Transfer Protocol – HTTP/1.1 <http://www.w3.org/Protocols/rfc2616/rfc2616.html>

RTSP Protocol

Real-Time Streaming Protocol – RTSP/1.0 <http://www.ietf.org/rfc/rfc2326.txt>

VAPIX® http://www.axis.com/techsup/cam_servers/dev/cam_http_api_index.php

VAPIX® Parameter Specification

VAPIX® RTSP API

Axis Video Product Release Notes

Migration Guide