zeevee
Intelligent AV Distribution

API

Application Programming Interface

User Manual
Updated December, 2023
API Release 3.2x

Contacting ZeeVee

Support

Contact us for installation and technical support, repairs, and warranty service:

+1 (877) 4-ZEEVEE (1.877.493.3833)

support@zeevee.com

Sales

North America:

+1 (347) 851-7364 Phone

sales@zeevee.com

EMEA:

+44 1494 956677 Phone

EMEAsales@zeevee.com

DACH:

+49 171 3620083 Phone

europe@zeevee.com

Features and Package Contents

Features

- Pre-configured Linux O/S is maintenance-free and includes upgrades and support.
- Plug & Play operation will discover and enable labeling and control of any number
  of ZyPer4K, ZyPerUHD or ZyPerUHD60 encoders and decoders.
  - **Note:** Release 2.3.x was the final release to support the ZyPerHD
- Interface allows the independent routing of video, audio and control signals.
- The feature-rich API makes ZyPer4K / ZyPerUHD / ZyPerUHD60 the perfect add-on to existing distribution systems without the time and dollars usually required for custom programming.
- Presets enable signal routing and scheduling of saved, pre-defined source-display settings for easy duplication and recall.
- Real time system monitoring includes generating alerts for offline or disconnected ZyPer4K / ZyPerUHD / ZyPerUHD60 devices, sources and displays.
- Auto detection/discovery of additional encoders and decoders make system scaling
  a snap.
- Easily create and manage video walls of any pattern or configurations up to a 15x15 array.
- Create and manage Multi-view displays with up to 19 sources. (ZyPer4K only)

New in Release 3.2

New Features 3.2

- Added support for ZyPerUHD60 2E, 2EA, 2D, 2DA
- Added specific commands related to Dante enabled UHD60 encoders and decoders

New Features 3.1

- Added support for ZyPer4K 12G SDI Encoder

New Features 3.0

- Many account and security related features
- Flash LED lights from GUI
- Set Low Power mode from GUI
- See Icron USB IP addresses in Display/Source Grid
- Set Server IP address from GUI

New in Release 2.3 / 2.4 / 2.5

New Features 2.3 / 2.4

- Additional Multiview preset patterns. (ZyPer4K only)
- Clock added to Preset Calendar.
- Copy/Clone command added for Multiview. (ZyPer4K only)
- License count only applies to ZyPer4K units. Non ZyPer4K units do not count against license limit.
- Channel up/down command added for Multiview windows. (ZyPer4K only)
- Update ZyPerUHD "No Source Found" background from ZMP GUI.
- Ability to enable or disable viewing of IP address and firmware version in ZyPerUHD "No Source Found" screen. (Release 2.3.37261 and newer)
- Updated help search features for API
- Release 2.4 includes update to Linux version 20.04 on the new NUC form factor Management Platform. (See hardware specifications in Section 1)

New Features 2.5

- Update ZyPerUHD "No Source Found" background from ZMP API.
- Maximum supported video wall size increased to 15x15 for ZyPerUHD
- Ability to disable 5V HDMI line on ZyPer4K-XS/XR decoders when no video routed to the decoder.
- Ability to issue a channel up or channel down command to ZyPer4K decoder via a ZeeVee IR remote control or ZyPer Trigger. Requires ZV IR RX unit.

New Features 2.5.3

- Support for the new ZyPerUHD60

ethernetManagementPort changed to utilityPort

The 1Gb port on ZyPer4K units was referred to as the ethernetManagementPort in previous releases of the API. With release 2.3 this is now changed to utilityPort.

New in Release 3.2

New/Updated Commands 3.2 (See API Command Listing)

- help
- join
- join dante
- set encoder danteAudioOut
- set decoder danteAudioOut
- set device security
- set device utilityPort
- set device videoPort

New/Updated Commands 3.0 (See API Command Listing)

- authenticate username
- create account
- delete account
- delete role
- generate tls
- help
- load account
- load tls
- logout
- set account
- set decoder autoAudioConnections hdmiAudioFollowVideo
- set device control authentication
- set multiview allowMainStream
- set role
- set server security
- set server timezone
- set tls
- show account
- show device config
- show files
- show logs
- show server ip duplicates
- show tls
- sign tls
- troubleReport

Removed in Release 3.0

Support for older ZMP NUC Devices

- First generation ZMP NUC devices are not supported with the 3.0 release of ZMP API. These devices are running an incompatible version of the Linux Operating System and were last shipped by ZeeVee back in 2017. These units can be easily identified as they have the brand name “GigaByte” written on the underside of the unit.
- Customers using this older NUC that wish to upgrade to the 3.0 ZMP API release should contact the ZeeVee sales team (sales@zeevee.com) to purchase an updated ZMP Hardware device.

These API Commands have been removed

- set server ssh password

1 Advanced Operation

Accessing the API 2
  Using Telnet....2
  Getting Help 2
  Setting the Time Zone 5
EDID Management 6
  Auto EDID Mode.... 6
  Using Custom EDID Data.... 6
Using AJAX/JSON 8
  Login Authentication.... 8
  Command Request / Response.... 9
Fast-Switched vs. Genlocked Mode 11
API Command Listing 12

2 Modules / Plug-Ins

Disclaimers....218
ZeeVee Control Modules / Plug-Ins....219
Configuration Tips....220
Troubleshooting.... 224

3 Appendix

Updating the Software....227
  Using Mac OS X 227
  Using Windows.... 229
  Using ZyPer Management Platform 230
Redundancy Configuration Instructions 232

1 Application Programming Interface

Accessing the API

Using Telnet

Telnet is a popular protocol that can be used on both Windows® and Mac OS® operating systems to connect to the programming shell. On a Windows operating system, a Telnet client, such as "PuTTY", must be installed. From a Unix or Mac OS command line, use the `telnet` command followed by the IP address of the Management Platform:

```
telnet 192.168.1.6
```

> *Instead of specifying the IP address of the Management Platform, the following identifier can also be used:* `zyper.local`
>
> *Example:* `telnet zyper.local`

Telnet will use port 23 by default and once connected, the API prompt will be displayed:

```
Zyper$
```

Getting Help

To make it easier to find commands, help now supports groups.

- `help` – lists all groups
- `help <group>` -- lists commands within a group.

  Note: The same command may appear in more than one group.
- `help all byGroup` – lists all groups and all commands in each group
- `help all alphabetical` – list all commands in alphabetical order

Help is available in two forms. Typing `help` or `?` at the prompt will list all available commands:

```
Zyper$ help all alphabetical

 Help All Commands Alphabetical
    add device ipAddress <ip>
    add snmp trapServer v2cTrap ipAddress <address:ip> community
    <string>
    ...
    ...
    update device <deviceNamePart>|all|encoders|decoders <filename>
    update server <filename>
Success
Zyper$
```

```
Zyper$ help
 Help Groups
    Audio
    CEC
    Data
    Decoder/Display
    Device
    Diagnostics
    EDID
    Encoder/Source
    Events
    HDCP
    Join
    Multicast
    Multiview
    Preset
    PreviewStreams
    Redundancy
    SNMP
    Script
    Serial/IR
    Server
    Status/Config
    USB
    Video
    VideoWall
    Zone
Enter 'help <group>', or 'help all byGroup', or 'help all
alphabetical'
Success
```

In addition, a partial list of commands can be listed by specifying the first word of each command. The first part of the command must be specified *before* the `help` command. For example, the following will only list command with the join prefix.

```
Zyper$ join help
join <encoderMac|encoderName>|none <decoderMac|decoderName|zoneNa
me[.zoneName]> analogAudio
join <encoderMac|encoderName>|videoSource|none <decoderMac|decoderN
ame|zoneName[.zoneName]> hdmiAudio
join <encoderMac|encoderName|multiviewName>|none
<decoderMac|decoderName> multiview
join <encoderMac|encoderName>|none <decoderMac|decoderName|zoneNa
me[.zoneName]> video|fastSwitched|genlocked|genlockedScaled
join <encoderMac|encoderName>|none <wallName> videoWall
join <encoderMac|encoderName>|none <decoderMac|decoderName>|none
usb
join <encoderMac|encoderName>|none <decoderMac|decoderName> window
viewportSource <x:int> <y:int> <sizeX:int> <sizeY:int> viewportDest
<x:int> <y:int> <sizeX:int> <sizeY:int>
Zyper$
```

In addition, help can be searched by keyword. help search <string>

(**Note**: Feature added to release 2.3.37234 and newer)

```
Zyper$ help search layer
set multiview <multiviewName> windowNumber <int> encoderName
<encoderName>|none percentPositionX <float> percentPositionY <float>
percentSizeX <float> percentSizeY <float> layer <int>
set multiview <multiviewName> windowNumber <int> encoderName
<encoderName>|none pixelPositionX <int> pixelPositionY <int>
pixelSizeX <int> pixelSizeY <int> layer <int>
set multiview <multiviewName> windowNumber <int> layer <int>
Success

Zyper$ help search audio
join <encoderMac|encoderName>|none <decoderMac|decoderName|zoneNa
me[.zoneName]> analogAudio
join <encoderMac|encoderName>|videoSource|none <decoderMac|decoderN
ame|zoneName[.zoneName]> hdmiAudio
set encoder <encoderMac|encoderName> analogAudioOut source
none|hdmiAudioDownmix
set encoder <encoderMac|encoderName> edid audio onlyPcm|allowCompre
ssed|serverDefault
set decoder <decoderMac|decoderName> analogAudioOut source
analogAudio|hdmiAudioDownmix
set decoder <decoderMac|decoderName> hdmiAudioOut source analogAudi
o|hdmiPassthroughAudio|hdmiAudio|hdmiAudioDownmix
set multiview <multiviewName> audioSource windowNumber <int>|none
set server encoderDefault edid audio onlyPcm|allowCompressed
start encoder <encoderMac|encoderName> stream video|videoScaled|hdm
iAudio|analogAudio
stop encoder <encoderMac|encoderName> stream video|videoScaled|hdmi
Audio|analogAudio
Success

Zyper$ help search create
create multiview <newMultiviewName>
create presetNew <newPresetName> commands existingConnections|empty
create presetSchedule <presetName> schedule <newPresetScheduleName>
create videoWall <newWallName>
create zone <[zoneName.]newZoneName>
Success
```

Setting the Time Zone

The Management Platform can use the Network Time Protocol (NTP) to set the date and time. However, the time zone will need to be specified. Alternately the date and time can be set manually.

1. Telnet to the Management Platform.

```
telnet 192.168.1.6
```

2. After the connection has been established, use the `set server timezone` command to set the time zone.

The time zone must be specified in POSIX format and is case-sensitive. Refer to the following link for more information:

http://wikipedia.org/wiki/List_of_tz_database_time_zones.

```
Zyper$ set server timezone America/New_York
Success
Zyper$
```

3. To set date/time manually use the `set server timezonedate manual` command.

```
Zyper$ set server date manual  month 4 day 1 year 2021 hour 15
minute 1
Success
Zyper$
```

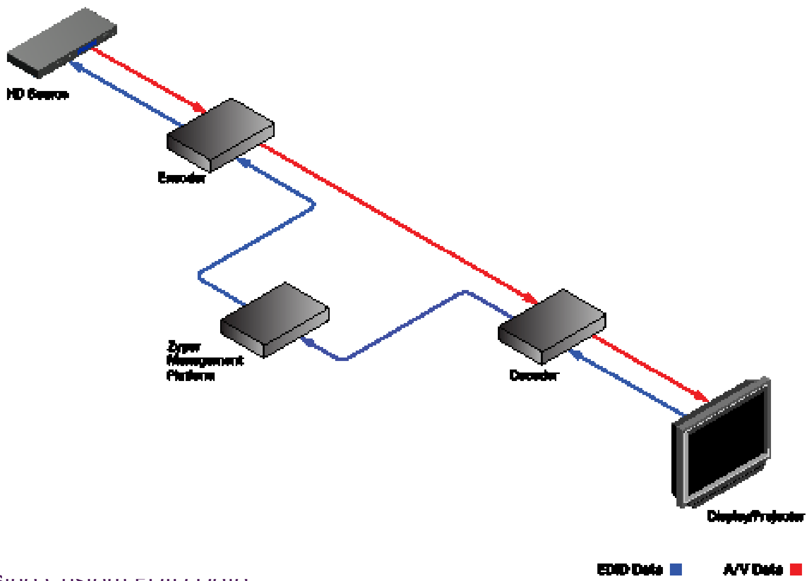
Use the `show server info` command to verify the correct time zone has been set.

```
Zyper$ show server info
server(192.168.0.22);
  server.gen; hostname=zyper.local, serverType=NUC,
version=3.0.38847, previousVersion=2.5.3.38647, master=true
  server.gen; uptime=1d:1h:20m:31s, freeMem= 6.74GB, \
sdvoeVersion=3.5.0.0, bootCount=204, serialNumber=ZZM1K400011D
  server.gen; macAddress=94:c6:91:a0:47:fc, managementMacAddress=
  server.ipActive; ipServerAddr=192.168.0.22,
ipManagementAddr=192.168.0.22, gatewayAddr=192.168.0.1, dnsAddr=0.0.0.0
  server.ipActive; managementGatewayAddr=0.0.0.0, managementDnsAddr=0.0.0.0
  server.time; time="Fri May 12 14:14:31 2023", timezone=EST
  server.license; productID=F9188182-AF72-C6C8-92C6-94C691A047FC,
license=none
  server.license; Zyper4KLimit=24, Zyper4KDevices=9, allDevices=25,
allDevicesUp=7, Zyper4KDevicesExceeded=0
  server.deviceUpdates; active=0
  server.activeDeviceVersions; num_1.5.0.1=1, num_1.7.2.0=2,
num_1.7.4.1=2, num_2.0.4.18=1, num_4.1.2.9=1
lastChangeIdMax(40);
Success
```

EDID Management

Auto EDID Mode

By default, Auto EDID mode is *enabled*. This means that the Management Platform will compare the encoder EDID with the decoder EDID. If they are different, then the EDID from the decoder (sink) will be used by the encoder (source). Setting the EDID Mode affects all join modes: fast-switched, genlocked, and video-wall. Refer to the `join` command in the API Command Listing (page 94) section for more information.



Using Custom EDID Data

There may be some instances where a custom EDID is desired. One example is when using a single encoder with multiple displays, such as a *video wall*. In such a case, follow the steps below to save and load a custom EDID to the Management Platform.

1. Telnet to the Management Platform.

```
telnet 192.168.1.6
```

2. Disable Auto EDID mode by entering the following command:

```
zyper$ set server auto-edid-mode disabled
```

3. Use the `save deviceEdid` command to save the EDID of the sink device (attached to the decoder) to the Management Platform, using the following convention:

   ```
   save device-edid [id] [filename]
   ```

   Make sure to replace `[id]` with the identifier of the sink device containing the EDID you wish to capture. You can specify either a MAC address or a name identifier. Follow the identifier with the name of the EDID file. For example:

   ```
   zyper$ save device-edid SonyXBR4 myEDID
   ```

4. After executing this command, two files will be created under the following directory:

   ```
   /srv/ftp/files/myEDID
   /srv/ftp/files/myEDID.txt
   ```

   `myEDID` is a binary EDID data file in standard format. `myEDID.txt` contains the decoded EDID in standard ASCII text.

   These files must remain in this directory when disabling Auto EDID mode.

5. To force a ZyPer encoder to use the saved EDID you need to have the MP load the binary EDID file onto the desired encoder.

   ```
   zyper$ load encoder-edid [id] saved [filename]
   ```

   Make sure to replace `[id]` with the identifier of the source device you want to load the EDID onto. You can specify either a MAC address or a name identifier. Follow the identifier with the name of the EDID file. For example:

   ```
   zyper$ load encoder-edid BlueRay1 saved myEDID
   ```

6. To return to Auto EDID mode, for any reason, enter the following command at the prompt:

```
zyper$ load encoder-edid BlueRay1 auto

or

    zyper$ set server auto-edid-mode enabled
```

Using AJAX/JSON

The AJAX/JSON programming interface allows developers to control the Management Platform within browser-based applications. All calls to the server are asynchronous post/receive operations using Javascript and do not require any specific HTML or CSS code. We will present two examples in this section: Login authentication and command request/response.

Login Authentication

There are two methods to authenticate with the server. The first and recommended method is to pass the username and password to `rcLogin.php`. The second method is to pass the username and password in every AJAX request.

Once the server accepts the username and password, it will generate a secure cookie called "userToken". This cookie will expire one hour after the last AJAX command is received by the server. After the cookie expires, all other AJAX requests will result in a failed authentication until `rcLogin.php` is called again. The following code excerpt is from the `zyperLogin()` function within `zyper.html`:

```
...
...
xmlhttp=new XMLHttpRequest();
xmlhttp.onreadystatechange = function(){
   if (xmlhttp.readyState == 4 && xmlhttp.status == 200){
      procLoginResp(xmlhttp.responseText);
      }
   }
postdata = "";
postdata += encodeURIComponent("serverSocketName") + '=' +
            encodeURIComponent(socketName) + '&' +
            encodeURIComponent("username") + '=' +
            encodeURIComponent(username) + '&' +
            encodeURIComponent("password") + '=' +
            encodeURIComponent(password) + '&';
xmlhttp.open("POST", url, true);
xmlhttp.setRequestHeader("Content-Type", "application/x-www-form-
urlencoded");
xmlhttp.send(postdata);
}
```

The response is a string value. The variable `resp` can be "Success", "Failed", or "Server not running".

```
function procLoginResp(jsonData) {
       var resp = JSON.parse(jsonData);
...
...
```

Command Request / Response

After login, any further commands are sent to the rcCmd.php

The following code excerpt sends an AJAX request to list all ZyPer encoders and decoders:

```
function zt(){
  xmlhttp = new XMLHttpRequest();
  xmlhttp.open("POST", url, true);
  xmlhttp.setRequestHeader("Content-Type", "application/x-www-form-
urlencoded");
  xmlhttp.onreadystatechange = function(){
     if (xmlhttp.readyState == 4 && xmlhttp.status == 200){
        procResp(xmlhttp.responseText);
     }
  }
xmlhttp.send(encodeURIComponent("commands:show device-status
all"));
}
```

In this example, the `encodeURIComponent` function has two parts: The request type, which is `commands` and the command `show device-status all`. Refer to the `show device status` command for more information. Currently, `commands` is the only request type that is supported and only a single command can be suppplied for each request.

Here, we handle the AJAX response:

```
function procRespTest(jsonData){
   var jsData = JSON.parse(jsonData);
   # jsData.status may have the values:
   # "Success"
   # "Request failed authentication"
   # "Server not running"
   # "no commands provided"
   #
   if (jsData.status == "Success"){
      var element = document.getElementById("responseError");
      element.innerHTML = jsData.responses[0].error;
      element = document.getElementById("responseWarning");
      element.innerHTML = jsData.responses[0].warning;
      element = document.getElementById("numObjectsInResponse");
      element.innerHTML = jsData.responses[0].text.length;
   }
   else{
      // Failed authentication
   }
}
```

The JSON data is decoded using the `JSON.parse()` method. In this example, information about the response data is displayed on the web page (HTML code not shown).

The JavaScript object that is returned is:

```
var jsObj = {
   status: true | false;
   responses: [ {error: “errorText”,
                 warning: “warningText”,
                 text: [ { param1: “val1”, parmN: “paramN” } ]
                 }
              ]
   };
```

The return value is an object that contains two members: `status` and `responses`. If the `status` member is not equal to “Success”, then the `responses` member is not valid. If the request fails authentication, then the `status` value will be “Request failed authentication”. Note that there may be other web-server level failures that can be returned in the `status` string.

The second member in the returned object, `responses`, which is an array of objects. Each of these objects contains three members: `error`, `warning`, and `text`. The `error` and `warning` members are strings. The `text` member is an array of objects with the desired parameters and values. If the `error` string is non-null, then the `warning` and `text` members will be null. If `text` is non-null, then the `warning` string may still be valid.

Currently, the `responses` member is always an array size of 1.

Fast-Switched vs. Genlocked Mode

The ZyPer4K provides two uniquely different modes for joining video/audio between a source (encoder) and display (decoder). The chart below details the differences between these two modes.

Feature	***Fast-Switched***	***Genlocked***
Latency	1-frame of latency. (16-33ms depending on frame rate of source video)	0 frames of latency. Less than 100µs
Transition Appearance	Instantaneous if switching between sources at same resolution and frame rate	Visible blanking of display when switching between sources
Scaling	Automatic scaling up or down to preferred resolution of the display (As determined by display EDID)	Source is not scaled. What comes in at source is presented to display exactly as input. (Note: Special Genlock-scaled mode is available)
HDR	HDR input is automatically reduced to 8-bits at output	HDR input is maintained exactly as input at the output
Color Space	Output from decoder is always RGB	Output from decoder matches the input at encoder
Encoded Audio	AC3 or other encoded audio formats are passed from encoder to decoder	AC3 or other encoded audio formats are passed from encoder to decoder
Video Wall	Video walls are technically not supported in Fast-Switched mode. (Join command for walls defaults to Genlock-scaled)	Video walls are technically always in Genlock-scaled mode
Multiview	Multiview is supported in Fast-Switch mode	Multiview is **not** supported in Genlocked mode
Video Disconnect	Disconnecting video (Join None) will maintain a black screen output. (Video is not technically disconnected)	Disconnecting video (Join None) will disconnect the video stream entirely. No video output from decoder
USB, IR, RS232	None of these items are associated with Fast-Switched or Genlocked mode	

Notes: Video Disconnect refers to the existing join between encoder and decoder. If existing join is Fast-Switched then a “Join None” command simply puts out a fully black video output from the decoder. If the existing join is Genlocked, then a “Join None” command disconnects the video stream and nothing is output from the decoder. This can be verified by looking at the VID light on the ZyPer4K decoder.

API Command Listing

Command	Description
`add device`	Manually adds a device to Management Platform
`add snmp`	Add new snmp user or trap server
`add zoneDisplay`	Adds a display or Video wall to an existing zone.
`authenticate username`	Used by browser to authenticate users
`channel`	Cycles up or down through encoders. Used to change channels.
`clone multiview`	Used to clone an existing multiview.
`create account`	create a new user account with password
`create multiview`	Creates a new multiview display (ZyPer4K family only)
`create presetNew`	Creates a new preset
`create presetSchedule`	Creates schedule for existing preset
`create role`	Create a new role with specified access level
`create videoWall`	Creates an empty 2x2 video wall.
`create zone`	Creates a new empty zone.
`dataConnect`	Used Creates a TCP port connection between devices for IR or RS232
`delete account`	delete a user account
`delete allConfiguration`	Deletes all encoder/decoder and server information from the Management Platform
`delete device`	Used Deletes the specified encoder or decoder from the Management Platform database.
`delete devicemultiview`	Deletes the specified multiview from the Management Platform database. (ZyPer4K family only)
`delete multiviewWindow`	Deletes a specific window from an existing multiview (ZyPer4K family only)
`delete preset`	Used to delete a preset, preset runlog or preset schedule
`delete role`	Delete an existing role
`delete snmp`	Delete SNMP user or trap server
`delete videoWall`	Deletes the specified video wall from the Management Platform database.
`delete zone`	Deletes an existing zone
`delete zoneDisplay`	Removes a display from an existing zone
`diagnostics device`	Used Runs a set of diagnostics on device
`dumpusb`	Outputs information about USB devices
`events`	Causes the event mode to be entered
`factoryDefaults device`	Sets the specified encoder/decoder to factory-default settings.

Command	Description
`flashLeds`	Plysically identifies the specified encoder/decoder on the network using LED flashes.
`generate tls ca privKeyPass`	Used to generate a local Transport Layer Stream Certificate Authority private key.
`generate tls server csr privKeyPass`	Used to generate a local Transport Layer Stream server Certificate Signing Request private key.
`help`	Brings up various help options
`join`	Switches audo and/or video from source to display or video wall
`join videoSource`	Selects audio feed to follow a video join
`load account`	Sets GUI pre and post login images and warning text
`load encoderEdid`	Uploads an EDID file to the specified encoder
`load idleImage`	Uploads an image to use as UHD background when no video streamed to decoder (ZyPerUHD only)
`load tls`	Used to load TLS certifications and keys
`logging`	Used to set logging level and add notes to the log
`logout`	Used to logout current session or force logout of any active session
`previewStream`	Used to turn on/off the preview stream viable in the Management Platform GUI (ZyPer4K family and ZyPerUHD only)
`redundancy switchover`	Swap Management Platform Master and Slave
`redundancy delete downServers`	Removes no longer present servers from list of redundant servers
`restart device`	Restarts the specified encoder/decoder
`restore server database`	Restores a saved database
`revert server`	Switch to a previously installeed version of the API
`save deviceEdid`	Saves the EDID from a decoder to a local file
`save server database`	Saves current server database to file
`save system config`	Saves current system configuration to a file
`script`	Executes the specified AJAX/JSON or text script.
`send`	Sends an IR, CEC or RS232 string to the specified device
`set account`	Sets various security features
`set decoder connectionMode`	Changes current connection to decoder to fastSwitched, genlocked or genlockedScaled. (ZyPer4K family only)
`set decoder displayMode`	Sets defaults decoder output to crop, stretch or box
`set decoder displayResolution`	Used to set decoder output size to auto or manual resolution. (Width, Height, FPS)

Command	Description
`set decoder analogAudioOut source`	Sets the source of Analog audio output for specified decoder
`set decoder edidPreferMode`	Sets the preferred resolution from the display EDID
`set decoder hdmiAudioOut source`	Sets the source of HDMI audio output for specified decoder
`set decoder hdmi5vControl`	Enables or disables HDMI 5V line (ZyPer4K-XS and ZyPer4K-XR only)
`set decoder osdStatusMode`	Enables or disables OSD feature (ZyPerUHD and ZyPerUHD60 only)
`set decoder powerSave`	Enables or disables power-save feature (ZyPerUHD and ZyPerUHD60 only)
`set device general name`	Sets the name for the specified device.
`set device ip dhcp linkLocal`	Sets the specified device to DHCP or Link-Local mode (ZyPer4K family only)
`set device ip static`	Sets the device to static mode (ZyPer4K family and ZyPerUHD only)
`set device irProcessing`	Configures decoder to process incoming IR commands to issue Channel up/down command (ZyPer4K family only)
`set device rs232`	Sets the RS232 settings for the specified device
`set device security`	Enables security between server and device (ZyPer4K-XS and ZyPer4K-XR only)
`set device sendIpMcastRange`	Sets allowable range of multicast addresses for selected devices (ZyPer4K family only)
`set device sourceDisplay iconImageName`	Sets the icon image for the specified device.
`set device sourceDisplay location`	Sets the location name for the specified device.
`set device sourceDisplay manufacturer`	Sets the manufacturer name for the specified device.
`set device sourceDisplay model`	Sets the model name for the specified device
`set device sourceDisplay serialNumber`	Sets the serial number name for the specified device
`set device usbFilter`	Allows restrictions to USB use on selected device (ZyPer4K family only)
`set device utilityPort`	Enables or disables the 1G Ethernet utility port for the specified device (ZyPer4K family only)
`set device videoPort`	Selects active input port for ZyPer4K units with multiple inputs (ZyPer4K family only)
`set encoder analogAudioOut source`	Sets the source of Analog audio output for specified encoder (ZyPer4K family only)
`set encoder edid audio`	Sets allowable input audio formats
`set encoder hdcpMode`	Sets the HDCP compatibility at the encoder side

Command	Description
`set multiview`	Assigns source to a position and size within a multiview display (ZyPer4K family only)
`set multiview allowMainStream`	Determines if main unscaled stream can be used in a multiview (ZyPer4K family only)
`set multiview audioSource windowNumber`	Selects the input source to provide Audio for multiview display (ZyPer4K family only)
`set multiview canvasSize`	Specifies Multiview canvas for multiview creation. (ZyPer4K family only)
`set multiview newEncoderName`	Used to specify a new encoder for existing multiview window. Can also set to “none”
`set preset commands`	Specifies commands to be used for a preset
`set preset description`	Sets a description for the preset
`set preset schedule eventColor`	Sets the color to be used for a preset schedule in the GUI calendar
`set preset schedule month`	Sets the schedule month/day/time to run preset
`set responses rs232TermChars`	Specifies the RS232 termination string
`set role`	Sets permisison levels for a specific role.
`set server api lineWrap`	Sets number of characters before API command line interface starts a new line
`set server autoEdidMode`	Sets the EDID mode
`set server dataTunnelMode`	Sets server transfer mode to raw or telnet
`set server date`	Used to set server date manually or via ntp server
`set server discoverMode`	Used to set how server discovers ZyPerUHD endpoints. Broadcast or Multicast
`set server encoderDefault edid audio`	Sets the default encoder audio format for HDMI audio input.
`set server ftp mode`	Enables or Disables FTP access to Management Platform
`set server ip`	Sets the IP address of the Management Platform
`set server license`	Sets server license. (Max endpoints)
`set server isaac address`	Sets the domain name of the Isaac server
`set server isaac subsystemId`	Sets the subsystem ID of the Isaac server
`set server redundancy`	Set a virtual IP address/mask for Master and Slave Management Platforms
`set server security`	Set server device Security Key. (ZyPer4K-XS and ZyPer4K-XR only)
`set server telnet mode`	Used to enable or disable telnet access

Command	Description
set server telnet password	Used to set telnet password
set server timezone	Sets the time zone
set terminal output	Select either normal or JSON format output from API
set tls	Used to enable web server TLS mode
set videoWall	Modifies an existing wall
set videoWall Decoder	Assigns the specified decoder to a position within the video wall
show account	Shows information about accounts
show dataTunnels	Shows what rs232 or IR data relay ports are opened on the server.
show device capabilities	Shows detailed capabilities of specified device or devices
show device config	Shows detailed configuration information for specified device or devices
show device connections	Shows encoder connections to decoders
show device status	Provides detailed status information for specified device or devices
show device userAdded	Will show a list of all ZyPer endpoints that have been manually added with the add device command
show multiviews config	Lists all created multiviews with source, position and size info (ZyPer4K family only)
show multiviews status	Lists all created multiviews with source, datarate and multicast address info (ZyPer4K family only)
show files	Show various types of files currently stored on Management Server. (EDID, Firmware, Icons, Idle Images)
show logs commands	Shows a listing of last commands send to the Management Server
show logs authentications	Shows listing of recent logon/logoff events
show preset	Shows information and configuration details for a preset
show previewStreams	Lists names of encoders currently generating a preview stream. (ZyPer4K family and ZyPerUHD family)
show responses	Displays the lastChangeId for the specified device
show role	Shows information about a specific role or all roles
show server config	Displays the IP address and EDID mode of the Management Platform
show server info	Displays Management Platform information
show server ip duplicates	Shows cases were an IP address has been duplicated in the system. (Issue needs to be resolved)

Command	Description
`show server redundancy`	Displays information about Master and Slave Management Platforms
`show snmp`	Displays information related to SNMP
`show tls pem`	Show TLS Private Enhanced Mail details
`show tls summary`	Show Transport Layer Security summary
`show values`	Shows information related to encoder, decoder, server and multiviews
`show videoWalls`	Displays a list of all created video walls
`show zones`	Displays a list of zone and displays contained within
`shutdown server`	Shuts down or reboots the Management Platform
`sign tls`	Used with CSR to create signed certifiate
`sleep`	Sets a time delay, in milliseconds
`stop encoder`	Stop a specified stream (ZyPer4K family only)
`start encoder`	Start a specified stream (ZyPer4K family only)
`switch`	Switches IR or RS-232 between devices
`troubleReport`	Generates a trouble report
`update device`	Updates the individual encoder or decoder units
`update server`	Updates the Mangement Platform software. See Updating the Software (page 227) for more information

add Device

Used to manually add a device to the ZyPer Management Platform that are located on a different VLAN/Subnet than the ZMP itself.

A qualified network engineer should be involved in making these configuration updates and the network switch provider may need to be consulted to ensure support of needed features.

Syntax

```
add device ipAddress ip
```

Parameters

i

Type: **IP Address**

The IP address of the device

Example

```
add device ipAddress 192.168.10.81
Success
```

Detailed Example

The ZyPer4K Endpoints are located on VLAN 10 and the 192.168.10.X subnet. The ZyPer Management Platform is on VLAN 20 and the 192.168.20.X subnet.

The ZMP will automatically discover any ZyPer4K endpoints located on VLAN 20. The ZMP will NOT automatically discover any ZyPer4K endpoints located on VLAN 10. However, given the proper circumstances, the ZyPer4K endpoints on VLAN 10 can be manually added to the ZMP for control.

For this to work, the network MUST be configured to route traffic between VLAN 10 and VLAN 20. How to configure the network to allow routing between VLANs is beyond the scope this document and should be done by a qualified network engineer. A simple test to confirm routing is that a device in VLAN 10 can ping a device in VLAN 20.

The ZyPer4K endpoints need to have a known IP Address. The IP Address should either be assigned by a DHCP server or assigned statically.

ZyPer4K endpoints need to be added one at a time.

You can get a listing of all “user added” devices with the “show device userAdded” command.

add snmp

Creates an new SNMP user or trap server. (Please see Section 5 of this manual for additional details on SNMP support)

Syntax

```
add snmp arg name
```

Parameters

arg

Type: **STRING**

Supply one of the following arguments before executing this command.

argument	Description
`trapServer vc2Trap ipAddress <address> community <comm>`	Add new trap server at the specified IP Address
`user v2c accessLevel readOnly community`	Add new SNMP user
`user v3 accessLevel readOnly auth MD5 encrypted no username <name> password <password>`	Add new SNMP user

name

Type: **STRING**

IP address of trapserver or name of new SNMP user (Password must be 8 to 127 characters)

Example

```
add snmp user v3 accessLevel readOnly auth MD5 encrypted no
username john password abc12345
Success

add snmp trapServer v2cTrap ipAddress 192.168.0.231 community john
Success
```

Related Commands

`delete snmp`
`show snmp`

add zoneDisplay

Adds a display or video-wall to an existing zone.

Care should be taken that individual displays found within walls are not added to a Zone. This would result in the same display being in a zone more than once.

Syntax

`add zoneDisplay name` *`id`*

Parameters

`name`

Type: **STRING**

The name of the zone. Names are case-sensitive. ("All" is an option to add selected id to every current zone)

`id`

Type: **STRING** or **MAC Address**

The name or MAC address of the device. String names are case-sensitive.

Example

```
add zoneDisplay Zone1 Decoder5
Success

add zoneDisplay All Decoder2
Success
```

Related Commands

```
create zone
delete zone
delete zoneDisplay
show zones
```

authenticate username

Used by browsers to authenticate users accessing ZMP.

Note: This command is not intended to be run directly from the API command line interface.

Syntax

```
authenticate username user password pwd token tkn newPasword npwd
```

Parameters

user

Type: **STRING**

The name of the user. Names are case-sensitive.

pwd

Type: **STRING**

Password. Passwords are case-sensitive

tkn

Type: **STRING**

Token.

npwd

Type: **STRING**

New Password.

Example

channel

Will cycle through all encoders (of the same type as the decoder) that have a number (channel) suffix, "_nnn", where nnn is an integer (channel).

If there are encoders with names: enc_1, enc_100, enc_50, then a decoder will cycle through them in the order: enc_1, enc_50, enc_100, then back to enc_1.
If there are no encoders (of the same type as the decoder) with the channel suffix, an error is returned.

Only fastSwitch connection types is supported. If there was already a connection of some other type, it is changed to fastSwitched.

If the decoder has no connection, the encoder with the lowest channel suffix will be connected using fast-switch.

If the decoder has a connection to an encoder that does not have the channel suffix, then it will connect to the encoder that has the lowest channel suffix.

Note: In fastSwitch mode the join videoSource <decoder> command must be used to set audio to follow video join. Otherwise audio will not follow the video during channel up/down command.

Syntax

```
channel direction <decoder-id>
```

Parameters

direction

Type: **STRING**

argument	Description
`up`	cycle to next higher numbered encoder
`down`	cycle to next lower numbered encoder

decoder-id

Type: **STRING** or **MAC Address**

The name or MAC address of the decoder. String names are case-sensitive.

Example

```
channel up MyDecoder
Channel changed to Channel_2
Success
```

Related Commands

`join videoSource`

clone Multiview

Used to create a copy of an exisitng multiview. (ZyPer4K only) Once created, the new multiview will be listed under the **Multiview** menu within the built-in ZMP.

Use the `set multiview` command to set a source encoder to a specified location and size within the multiview.

Refer to Creating a Multiview Screen (page 41) for information on managing multiview displays in the built-in ZMP.

Syntax

```
clone multiview name to newmvname
```

Parameters

name

Type: **STRING**

The name of the existing multiview to be cloned. Names are case-sensitive.

newmvname

Type: **STRING**

The name of the new multiview. The name of the multiview cannot exceed 255 characters in length. Names are case-sensitive.

Example

```
clone multiview mv2x2 to newmv2x2
Success
```

Related Commands

```
delete videoWallmultiview
delete multiviewWindow
set videoWall sizemultiview
set videoWall decodermultiview audioSource windowNumber
show multiviews config
show multiviews status
```

create account

Creates a new user account with assigned password.

Syntax

```
create account name passwordOption
```

Parameters

name

Type: **STRING**

The name of the account The name of the account cannot exceed 255 characters in length. Names are case-sensitive.

passwordOption

Type: **STRING**

Supply one of the following arguments before executing this command.

argument	Description
`password`	Enter the password to be used by this account.
`tempInitialPassword`	System will generate a temporary one time use password that will expire after the first use.

Examples

```
create account bartender tempInitialPassword
result: password=ovmOH;arZasuHS, expires=immediate
Success

create account bartender password 12345
Success
```

Related Commands

`delete account`

create Multiview

Creates an empty multiview display. (ZyPer4K only) Once created, the new multiview will be listed under the **Multiview** menu within the built-in ZMP.

Use the `set multiview` command to set a source encoder to a specified location and size within the multiview.

Refer to Creating a Multiview Screen (page 41) for information on managing multiview displays in the built-in ZMP.

Syntax

```
create multiview name
```

Parameters

name

Type: **STRING**

The name of the multiview. The name of the multiview cannot exceed 255 characters in length. Names are case-sensitive.

Example

```
create multiview myMultiview
Success
```

Related Commands

```
delete videoWallmultiview
delete multiviewWindow
set videoWall sizemultiview
set videoWall decodermultiview audioSource windowNumber
show multiviews config
show multiviews status
```

create presetNew

Creates a new preset. Once created, the new preset will be listed under the **Preset** menu within the built-in ZMP.

Syntax

```
create presetNew name commands connections
```

Parameters

name

Type: **STRING**

The name of the preset. The name of the preset cannot exceed 250 characters in length. Names are case-sensitive.

connections

Type: **STRING**

Supply one of the following arguments before executing this command.

argument	Description
`empty`	This preset has no commands or connections associated with it.
`existingConnections`	Use the current connections to generate the preset

Example

```
create presetNew EveningShutDown commands existingConnections
Success
```

Related Commands

```
create presetSchedule
delete preset
run preset
set preset
show preset
```

create presetSchedule

Inserts an existing preset into the schedule calendar Once created, the item must be assigned months/days/time to execute. By default without further setting, the preset will be schedule to occur every hour of every day.

Use the `set preset zoneDisplay` command to assign description, commands and schedule to the new schedule.

Syntax

```
create presetSchedule presetname schedule name
```

Parameters

presetname

Type: **STRING**

The name of an existing preset.

name

Type: **STRING**

The name of the schedule. The name of the schedule cannot exceed 250 characters in length. Names are case-sensitive.

Example

```
create presetSchedule EveningShutDown schedule GoHome
Success
```

Related Commands

```
create presetNew
delete preset
run preset
set preset
show preset
```

create role

Creates a account role with specified access level.

Syntax

```
create role name allSubsystems maxAccess accessLevel
```

Parameters

name

Type: **STRING**

The name of the role The name of the role cannot exceed 255 characters in length. Names are case-sensitive.

accessLevel

Type: **STRING**

Supply one of the following arguments before executing this command.

argument	Description
`admin`	Role has full administration privileges. No restrictions.
`config`	Role is able to configure existing items (endpoints, multiview, walls etc..) but cannot create/delete items.
`join`	Can only issue join commands
`none`	No access. Can only view the Help tab.
`view`	Role can only view. Cannot perform any actions.

Examples

```
create account bartender tempInitialPassword
result: password=ovmOH;arZasuHS, expires=immediate
Success

create account bartender password 12345
Success
```

Related Commands

`delete role`

create videoWall

Creates an empty 2x2 video wall. Once created, the new video wall will be listed under the **Display Config** menu within the built-in ZMP.

Use the `join videoWall`command to assign a source encoder to the wall. To modify the size of the video wall and/or control bezel parameters, use the `set videoWall` command.

Refer to Creating Video Walls (page 36) for information on managing video walls in the ZMP.

Syntax

```
create videoWall name
```

Parameters

name

Type: **STRING**

The name of the video wall. The name of the video wall cannot exceed 255 characters in length. Names are case-sensitive.

Example

```
create videoWall myWall
Success
```

Related Commands

`delete videoWall`
`set videoWall size`

create zone

Creates an empty zone Once created, the new zone will be listed under the **Zones** menu within the built-in ZMP.

Use the `add zoneDisplay` command to assign decoders or video walls to the zone.

Syntax

```
create zone name
```

Parameters

name

Type: **STRING**

The name of the zone. The name of the zone cannot exceed 255 characters in length. Names are case-sensitive.

Example

```
create zone Zone1
Success
```

Related Commands

```
add zoneDisplay
delete zone
delete zoneDisplay
show zones
```

dataConnect

Connects two devices for IR or RS232 communication over a specified TCP port. (**Note** TCP port only valid for connection between device and server. Not valid for connection between 2 devices)

The feature of dataConnect was added to allow a third party to connect to the ZMP server with a specific port and pass raw or telnet API commands (depending on the mode) to the server and port which is designated for a particular encoder or decoder.

Syntax

```
dataConnect id1 id2 mode tunnelPort port
```

Parameters

id1

Type: **STRING**

The name of the first device. String names are case-sensitive.

id2

Type: **STRING**

The name of the second device or server. String names are case-sensitive.

mode

Type: **STRING**

ir or rs232

port

Type: **INTEGER**

TCP-Port #. Integer range from 1,024 to 49,152

Example

```
dataConnect MediaPlayer server rs232 tunnelPort 2345
tunnel TCP port = 2345; telnet handshake mode
Success
```

Related Commands

```
show dataTunnels
set server dataTunnelMode
```

Notes on Tunnel Ports

There is a very convenient way to get RS232 data: TUNNELS.

```
Zyper$ dataConnect Decoder_1 server rs232
Dynamically assigned tunnel TCP port = 4100; telnet handshake mode
Success
Zyper$
Zyper$ show dataTunnels
data-sessions(d8:80:39:9b:9:a2);
  device: name=Decoder_1
  rs232Tunnel: port=4100
  rs232Tunnel-connections: none
Success
Zyper$
```

You can then connect to that tunnel port using TCP. Whatever is sent is forwarded to the device. Whatever the device returns is received on that TCP connection.

In the easiest case, you can just use telnet to connect to the tunnel.

```
You can specify the port number as well:
Zyper$ dataConnect Decoder_1 server rs232 tunnelPort 4101
tunnel TCP port = 4101; telnet handshake mode
Success
Zyper$
```

You can set the default TCP connection mode: raw|telnet (defaults to telnet).

```
Zyper$ set server dataTunnelMode raw|telnet
```

When in telnet mode the IAC commands are sent/received. Although most telnet clients will also work fine in raw mode.

delete account

Deletes the specified account from the Management Platform database.

Syntax

```
delete account id
```

Parameters

id

Type: **STRING**

The name of the account. String names are case-sensitive.

Example

```
delete account bartender1
Success
```

Related Commands

`create account`

delete allConfiguration

Deletes all device and server information from the Management Platform. The network configuration is preserved.

Syntax

```
delete allConfiguration action
```

Parameters

action

Type: **STRING**

Supply one of the following arguments before executing this command.

argument	Description
`reboot`	Unit is automatically rebooted
`restart`	The ZyPer server service is restarted
`shutdown`	Unit is shutdown

Example

```
delete allConfiguration restart
delete allConfiguration reboot
```

Related Commands

`factoryDefaults device`

delete device

Deletes the specified device from the Management Platform database.

Note that if the deleted device remains on the network, then it will be rediscovered by the Management Platform and reposted to the database. To permanently remove a device from the database, physically disconnected it and execute the `delete device` command.

Syntax

```
delete device id
```

Parameters

`id`

Type: **STRING** or **MAC Address**

The name or MAC address of the device. String names are case-sensitive.

Example

```
delete device myDevice
Success

delete device 0:1e:c0:f6:42:a1
Success
```

Related Commands

`factoryDefaults device`

delete multiview

Deletes the specified multiview from the database on the Management Platform.
(ZyPer4K family only)

Syntax

```
delete multiview name
```

Parameters

name

Type: **STRING**

The name of the multiview. Names are case-sensitive.

Example

```
delete multiview myMultiview
Success
```

Related Commands

```
create multiview
delete multiviewWindow
set videoWall sizemultiview
set videoWall decodermultiview audioSource windowNumber
show multiviews config
show multiviews status
```

delete multiviewWindow

Deletes the specified window from an existing multiview. (ZyPer4K family only)

Syntax

```
delete multiviewWindow name window arg
```

Parameters

name

Type: **STRING**

The name of the multiview. Names are case-sensitive.

arg

Type: **INTEGER**

Window number to remove. Integer range from 1 to 9

Example

```
delete multiviewWindow myMultiview window 5
Success
```

Related Commands

```
create multiview
delete multiview
set videoWall sizemultiview
set videoWall decoder multiview audioSource windowNumber
show multiviews config
show multiviews status
```

delete preset

Deletes the specified preset, preset runlog or preset schedule from the system.

Note: Runlog is history of when the preset has been executed. Deleting the runlog does not impact the preset itself or the schedule.

Syntax

```
delete preset name
delete preset name runLog
delete preset name schedule schname
```

Parameters

name

Type: **STRING**

The name of the preset. Names are case-sensitive.

schname

Type: **STRING**

The name of the preset schedule. Names are case-sensitive.

Examples

```
delete preset lunch runLog
Success

delete preset lunch schedule eat
Success
```

Related Commands

```
create preset
run preset
set preset
show preset
```

delete role

Deletes the specified role from the Management Platform database.

Syntax

```
delete role id
```

Parameters

id

Type: **STRING**

The name of the role. String names are case-sensitive.

Example

```
delete role bartender
Success
```

Related Commands

`create role`

delete snmp

Deletes an existing SNMP user or trap server. (Please see Section 5 of this manual for additional details on SNMP support)

Syntax

```
delete snmp arg name
```

Parameters

arg

Type: **STRING**

Supply one of the following arguments before executing this command.

argument	Description
`trapServer vc2Trap ipAddress <address> community <comm>`	Delete trap server at the specified IP Address
`user v2c`	Delete SNMP user
`user v3`	Delete SNMP user

name

Type: **STRING**

IP address and community of trapserver or name of new SNMP user

Example

```
delete snmp trapServer v2cTrap 192.168.0.231 community john
Success

delete snmp user v3 username john
Success
```

Related Commands

`add snmp`
`show snmp`

delete videoWall

Deletes the specified video wall from the database on the Management Platform.

Syntax

```
delete videoWall name
```

Parameters

name

Type: **STRING**

The name of the video wall. Names are case-sensitive.

Example

```
delete videoWall myWall
Success
```

Related Commands

```
create videoWall
set videoWall size
```

delete zone

Deletes the specified zone from the database on the Management Platform.

Syntax

```
delete zone name
```

Parameters

name

Type: **STRING**

The name of the zone. Names are case-sensitive.

Example

```
delete zone zone1
Success
```

Related Commands

`add zoneDisplay`
`create zone`
`delete zoneDisplay`
`show zones`

delete zoneDisplay

Deletes the specified display from an existing zone.

Syntax

```
delete zoneDisplay name id
```

Parameters

name

Type: **STRING**

The name of the zone. Names are case-sensitive.

id

Type: **STRING** or **MAC Address**

The name or MAC address of the decoder/display. String names are case-sensitive.

Example

```
delete zoneDisplay myzone mydisplay1
Success
```

Related Commands

```
add zoneDisplay
create zone
delete zone
show zones
```

diagnostics device

Runs a set to test diagnostics on the specified device

Syntax

```
diagnostics device id
```

Parameters

id

Type: **STRING** or **MAC Address**

The name or MAC address of the device. String names are case-sensitive.

Possible Results

Decoder/Encoder:

```
error,   Device is down
warning, Device has no HDMI link
warning, Device rebooted %d times in the last minute
warning, Device rebooted %d times in the last hour
warning, Device rebooted %d times in the last day
```

Decoder:

```
error,   HDMI Audio stream connection without video connection.
warning, HDCP is forced on, but may not be supported by display device (however it is unlikely)
error,   Decoder has never received a valid EDID
warning, Decoder resolution exceeds display EDID maximum -- very likely this will not work
warning, Decoder using encoder resolution, which may not be display's preferred based on its EDID
warning, Decoder using encoder resolution AND ignoring display EDID, which may allow resolution to
exceed display capability
warning, Decoder using user-defined resolution, which may allow resolution to exceed display
capability
error,   Encoder down
warning, Encoder hdmi down
error,   Encoder has multiview conflict with genlock
warning, Encoder stream disabled
warning, Video stream interrupted %d times in the last minute, indicating likely network problem
warning, Video stream interrupted %d times in the last hour, indicating likely network problem
warning, Video stream interrupted %d times in the last day, indicating possible network problem
warning, Encoder and decoder fps are not equal -- will result in very bad video
warning, Encoder and decoder fps are not equal -- will result in very bad video
warning, Encoder and decoder fps are not equal, but multiple of 2; this may still produce bad video
warning, Encoder and decoder HDCP versions are not the same
info,    Encoder HDCP is disabled; this will prevent copyrighted material from display
info,    Encoder HDCP is set to version 1.4; this may prevent copyrighted material from display
```

Encoder:

```
info,    HDCP is disabled; this will prevent copyrighted material from display
info,    HDCP is set to version 1.4; this may prevent copyrighted material from display
```

Examples

```
diagnostics device Top-Right
device(d8:80:39:9a:7f:ec);
  device.diags.summary; status=complete, error=0, warning=0, info=0
Success

diagnostics device ABC
device(d8:80:39:9a:96:7);
  device.diags.info.1; message=HDCP is disabled; this will prevent
copyrighted material from display
  device.diags.summary; status=complete, error=0, warning=0, info=1
Success

diagnostics device encoder1
device(34:1b:22:80:26:2a);
  device.diags.warning.1; message=Device has no HDMI link
  device.diags.summary; status=complete, error=0, warning=1, info=0
Success

diagnostics device MyEncoder
device(34:1b:22:80:63:9c);
  device.diags.error.1; message=Device is down
  device.diags.summary; status=complete, error=1, warning=0, info=0
Success


diagnostics device MeetingRoom6
device(34:1b:22:80:57:7d);
  device.diags.error.1; message=Device is down
  device.diags.info.1; message=No video connection
  device.diags.warning.1; message=HDCP is forced on, but may not be
supported by display device (however it is unlikely)
  device.diags.error.2; message=Decoder has never received a valid
EDID
  device.diags.summary; status=complete, error=2, warning=1, info=1
Success
```

dumpusb

Outputs details about USB devices found in ZyPerUHD, ZyPerUHD60 and/or ZyPer4K units. Information includes MAC address and ICRON IP_address if ICRON USB found in ZyPer4K unit.

Syntax

```
dumpusb
```

Example

```
dumpusb
Encoders/Decoders usb reported mac
  device UHDdec(UHDdec), usb mac 34:1b:22:80:57:df
  device UHDenc2(UHDenc2), usb mac 34:1b:22:80:7f:3d
  device Z4Kdec1(Z4Kdec1), usb mac 0:1b:13:1:1f:79
  device Arts_Encoder_1(Arts_Encoder_1), usb mac 0:1b:13:1:1e:90
Icrons reported info
  owner Arts_Encoder_1(80:1f:12:4d:9b:6b), deviceType local,
icronMac 0:1b:13:1:1e:90, ipAddr 169.254.4.123, fwRev
1.9.4, pairedInfoRcvd yes, numPairedMacs 1, 0:1b:13:1:1f:79
Z4Kdec1(Z4Kdec1)
  owner Z4Kdec1(80:1f:12:4d:2c:ff), deviceType remote, icronMac
0:1b:13:1:1f:79, ipAddr 169.254.4.125, fwRev 1.9.4, pairedInfoRcvd
yes, numPairedMacs 1, 0:1b:13:1:1e:90 Arts_Encoder_1(Arts_
Encoder_1)
Success
```

Note: This is a hidden command and will not appear in HELP

events

Causes the events mode to be entered.

Syntax

```
events
```

Server sends initial events and new events as they occur to the telnet session.   Any character entered to the server causes the mode to exit back to the API prompt.

See Section 4 of the ZyPer Management Platform User Guide for more information on Events

factoryDefaults device

Set the specified device to the factory-default settings.

Syntax

```
factoryDefaults device id
```

Parameters

id

Type: **STRING** or **MAC Address**

The name or MAC address of the device. String names are case-sensitive.

Example

```
factoryDefaults device Airshow
Success

factoryDefaults device 0:1e:c0:f6:a8:c3
Success
```

Related Commands

`delete allConfiguration`

flashLeds

Physically identifies the specified device on the network. When this command is executed, the LED indicators on the device will flash for 5 seconds.

Syntax

```
flashLeds id
```

Parameters

id

Type: **STRING** or **MAC Address**

The name or MAC address of the device. String names are case-sensitive.

Example

```
flashLeds myEncoder
Success

flashLeds 0:1e:c0:f6:59:13
Success
```

generate tls ca privKeyPass

Used to generate a local Transport Layer Security Certificate Authority private key.

Syntax

```
generate tls ca privKeyPass privKey country country state state
locality local organization org organizationUnit orgunit email
email

Enter passphrase: passphrase
```

Parameters

privKey

Type: **STRING** | *

Private key phrase. String. If * used; will be prompted for passphrase at the end.

country

Type: **STRING**

2 character string representing Country. Example “US”

state

Type: **STRING**

2 character string representing State. Example “MA”

local

Type: **STRING**

String representing local town/city. Example “Billerica”

org

Type: **STRING**

String representing organization. Example “ZeeVee”

orgunit

Type: **STRING**

String representing organization units. Example “money”

email

Type: **STRING**

String representing email address. Example “aweeks@zeevee.com” *passphrase*

Type: **STRING**

Private phrase used in generation of the tls key.   Prompted if * used earlier in command.

Example

```
generate tls ca privKeyPass * country US state MA locality
Billerica organization ZeeVee organizationUnit money email aweeks@
zeevee.com
Enter passphrase: ******
Success
```

Related Commands

```
generate tls server csr privKeyPass
show tls pem ca privKey

show tls pem ca privKey
pemData:
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-256-CBC,ADB163FA01562B533B617FA5792AB7F1

........

-----END RSA PRIVATE KEY-----
Success
```

generate tls server csr privKeyPass

Used to generate a local Transport Layer Security server Certificate Signing Request private key.

Syntax

```
generate tls server csr privKeyPass privKey fqdn domain
country country state state locality local organization org
organizationUnit orgunit email email

Enter passphrase: passphrase
```

Parameters

privKey

Type: **STRING | ***

Private key phrase. String. If * used; will be prompted for passphrase at the end.

domain

Type: **STRING**

String representing fully qualified domain name. Example “zeevee.com”

country

Type: **STRING**

2 character string representing Country. Example “US”

state

Type: **STRING**

2 character string representing State. Example “MA”

local

Type: **STRING**

String representing local town/city. Example “Billerica”

org

Type: **STRING**

String representing organization. Example “ZeeVee” *orgunit*

Type: **STRING**

String representing organization units. Example “money”

email

Type: **STRING**

String representing email address. Example “aweeks@zeevee.com”

passphrase

Type: **STRING**

Private phrase used in generation of the tls key.   Prompted if * used earlier in command.

Example

```
generate tls server csr privKeyPass * fqdn zeevee.com country US
state MA locality Billerica organization ZeeVee organizationUnit
money email aweeks@zeevee.com
Enter passphrase: ******
Success
```

Related Commands

```
generate tls ca privKeyPass
show tls pem server csr
show tls pem server privKey

show tls pem server csr
pemData:
-----BEGIN CERTIFICATE REQUEST-----
.................
-----END CERTIFICATE REQUEST-----
Success
```

help

Provides a listing of API commands grouped or sorted in various ways.

Syntax Options

```
help
    help all alphabetical
    help all byConcept
    help all bySubsystem
    help all byAccessLevel
    help concept <helpConcepts>
    help subsystem <helpSubsystems>
    help accessLevel <helpAccessLevels>
    help search string <keyWord:string>
```

Example

```
help

 Help commands:
    help
    help all alphabetical
    help all byConcept
    help all bySubsystem
    help all byAccessLevel
    help concept <helpConcepts>
    help subsystem <helpSubsystems>
    help accessLevel <helpAccessLevels>
    help search string <keyWord:string>
    <command> help
    <command> ?
    ?

** NOTE: Use <tab> to complete a command **

Success
```

join

Joins the specified decoder (display) with the specified encoder (source). The `mode` parameter must be specified and defines the type of join to execute.

- **analogAudio**
  Embeds analog audio stream from the encoder on the output of the decoder. The audio is from the (analog) Audio jack on the encoder. Will force UHD60 Dante enabled encoder into Dante Transmit mode.
  In order to control what type of audio is being output from the decoder, refer to the `set decoder AnalogAudioOut` command.

- **danteAudio**
  Will force UHD60 Dante enabled encoder into Dante Receive mode. This allows a received Dante audio stream to be transmitted as regular UHD60 audio to any UHD60 decoder.

- **fastSwitched**
  Allows the joining of an encoder and decoder with no video dropout. In order to make use of this feature, the resolution and frame rate of the "new" encoder must be the same as the previous encoder.

- **genlocked**
  This mode provides a very low-latency, all-purpose method of joining an encoder and decoder. (ZyPer4K family only)

- **genlockedScaled**
  This mode provides a very low-latency, all-purpose method of joining an encoder and decoder that includes scaling up or down at the decoder/display.

- **hdmiAudio**
  Embeds hdmi-downmix audio from an encoder to specified decoder.

- **multiview**
  Join the configured multiview to a display (decoder) (ZyPer4K family only)

- **video**
  Joins video only from encoder to decoder. No audio.

- **videoWall**
  Join the encoder to the named video-wall

- **window**
  Join any portion of a source to any portion of a display

- **usb**
  Creates USB connection between encoder and decoder. Note that multiple connections are valid.

- **none**
  Special command to disconnect existing connections (joins) Example: join none decoder fastSwitched

Syntax

```
join enc dec mode
join none dec fastSwitched
```

Parameters

enc

Type: **STRING** or **MAC Address**

The name or MAC address of the encoder. String names are case-sensitive.

dec

Type: **STRING** or **MAC Address**

The name or MAC address of the decoder. Can also be name of existing video-wall String names are case-sensitive.

zone

Type: **STRING**

The name of an existing zone. String names are case-sensitive.

mode

Type: **STRING**

Supply one of the following arguments before executing this command.

argument	Description
`analogAudio`	Embed audio from the specified encoder. Force Dante enabeld UHD60 encoder into Dante transmit mode.
`danteAudio`	Force UHD60 Dante enabled encoder into Dante receive mode.
`fastSwitched`	Join in “fast-switched” mode
`genlocked`	Low-latency join mode (ZyPer4K family only)
`genlockedScaled`	Low-latency with scale up/down (ZyPer4K family only)
`hdmiAudio`	Join hdmi-audio to either hdmi-out or analog-out. Note this command will cause hdmiAudioFollowVideo=False for specified decoder. See join videoSource command on next page.
`multiview`	Join a multiview to a display (ZyPer4K family only)
`videoWall`	Join a source to a video-wall
`video`	Join video only (audio not joined)
`window`	Join any portion of a source to any portion of a display (ZyPer4k family only)
`usb`	Establish USB connection
`“none”`	Disconnect existing joins

Notes:
Multiviews cannot be joined to a zone.
USB cannot be joined to a zone.

Examples

```
join myEncoder1 myDecoder2 fastSwitched
Success

join myEncoder1 myDecoder2 hdmiAudio
Success
(Note: If Dante Encoder currently set to Receive mode, this will set the
Encoder to Dante Transmit mode and force a reboot of the encoder)

join myMultiview2 Display4 multiview
Success

join myEncoder1 myWall videoWall
Success

join none myDecoder1 fastSwitched

join myEncoder1 myDecoder2 danteAudio
(Sets myEncoder1 to Dante Receive mode.  Will force reboot of UHD60 encoder
if not already in Dante Receive mode)
```

Window Example

```
join myEncoder1 myDecoder2 window viewportSource 0 0 1920 1080
viewportDest 500 500 500 500

(ViewportSource parameters are starting X/Y coordinates of the
source and desired X/Y size)

(ViewportDest parameters are starting X/Y coordinates in the
display and desired X/Y size)
```

join dante

Tells a Dante enabled UHD60 decoder what audio stream to output on HDMI and Analog audio ports.

Syntax

```
join dante|none dec directDanteAudio
```

Parameters

dec

Type: **STRING** or **MAC Address**

The name or MAC address of the decoder. String names are case-sensitive.

Example

```
join dante myDecoder1 directDanteAudio
Success

(Causes UHD60 decoder to output received Dante audio stream if the
decoder is in Dante receive mode)  Note if no Dante audio stream
routed to the UHD60 decoder, then decoder will have no audio
output.

join none myDecoder1 directDanteAudio
Success

(Causes UHD60 decoder to output the received UHD60 audio stream
from a UHD60 encoder if the decoder is in Dante receieve mode)
```

Related Commands

```
join hdmiAudio
set decoder danteAudioOut
set device utilityPort
```

join videoSource

Tells a decoder to automatically join corresponding audio from a source encoder whenever a join command is used to join video.

Syntax

```
join videoSource dec mode
```

Parameters

dec

Type: **STRING** or **MAC Address**

The name or MAC address of the decoder. String names are case-sensitive.

mode

argument	Description
`audio`	automatically join audio from connected encoder (ZyPerUHD only)
`hdmiAudio`	automatically join hdmi-audio from connected encoder (ZyPer4K family only)

Example

```
join videoSource MyDecoder hdmiAudio
Success
```

Related Commands

`join hdmiAudio`

load account

Uploads text and/or images to be displayed prior to and after the login screen. Can be used a warning or any other purpose.

Syntax

```
load account all PrePost Arg file
```

Parameters

PrePost

argument	Description
`preLoginBanner`	Specified text or image will appear before login
`postLoginBanner`	Specified text or image will appear after login

Arg

argument	Description
`terminal`	I dont' know
`webText`	Text that will appear before/after login
`webImage`	Image that will appear before/after login

file

Type: **STRING**

The name of the file to load. Text or .PNG

Examples

```
load account all preLoginBanner webImage DOD-Seal.png
Success

load account all preLoginBanner webText securePre.txt
Success

load account all postLoginBanner webImage mickey.png
Success
```

load encoderEdid

Uploads an EDID file to the specified encoder.

Important Note: Auto-EDID mode should be disabled when loading a specific EDID to an encoder. Otherwise the loaded EDID will immediately get replaced by the Auto-EDID option.

Syntax

```
load encoderEdid enc mode file
```

Parameters

enc

Type: **STRING** or **MAC Address**

The name or MAC address of the encoder. String names are case-sensitive.

mode

argument	Description
`auto`	use whatever EDID information is provided by connected decoder
`builtIn`	use one of the EDID files provided by ZeeVee. Many options available covering various 4k settings. See list below.
`default`	use default EDID with maximum capabilities of the encoder
`saved`	use a file that user has previously saved to the system with the save device-edid command

file

Type: **STRING**

The name of the file to load.

Build in EDID options

zyper-default
zyper4k25
zyper4k30

Build in EDID options continued

zyper4k50
zyper4k50-420
zyper4k50-420_hdmi14
zyper4k50-hbraudio
zyper4k50-hd-hdr
zyper4k50-hdr
zyper4k50-hdr-bt2020
zyper4k50-hdr-bt2020-hbraudio
zyper4k50-hdr-hbraudio
zyper4k60
zyper4k60-420
zyper4k60-420_hdmi14
zyper4k60-hbraudio
zyper4k60-hd-hdr
zyper4k60-hdr
zyper4k60-hdr-bt2020
zyper4k60-hdr-bt2020-hbraudio
zyper4k60-hdr-hbraudio
zyperHd50
zyperHd60
zyperPc
zyperUhd25
zyperUhd25-hbraudio
zyperUhd30
zyperUhd30-hbraudio
zyperUhd50
zyperUhd50-420
zyperUhd50-420_hdmi14
zyperUhd50-hbraudio
zyperUhd50-hd-hdr
zyperUhd50-hdr
zyperUhd50-hdr-bt2020
zyperUhd50-hdr-bt2020-hbraudio
zyperUhd50-hdr-hbraudio
zyperUhd60
zyperUhd60-420
zyperUhd60-420_hdmi14
zyperUhd60-hbraudio
zyperUhd60-hd-hdr
zyperUhd60-hdr
zyperUhd60-hdr-bt2020
zyperUhd60-hdr-bt2020-hbraudio
zyperUhd60-hdr-hbraudio

Examples

```
load encoderEdid myEncoder saved myEDID.bin
Success

load encoderEdid myEncoder builtIn zyper4k60
Success
```

Related Commands

```
save deviceEdid
set server autoEdidMode
```

load idleImage

Uploads an image to use at ZyPerUHD background when no video source streamed to the decoder.

Syntax

```
load idleImage dec filename file
```

Parameters

dec

Type: **STRING** or **MAC Address**

The name or MAC address of the decoder. String names are case-sensitive.

file

Type: **STRING**

The name of the file to load. (Must already exist on ZMP in Files directory)

Examples

```
load idleImage myDecoder filename background.jpg
Success
```

Notes:

```
Image must be in .JPG format
Image must be 1280 x 720 in size
(Will output from decoder at this resolution)

Image file must be previously copied onto ZMP into the Files
directory using FTP.  Alternately file can be loaded via the GUI.
See Display Grid “Config” tab.
```

Related Commands

save deviceEdidet decoder osdStatusMode

load tls ca cert

Options for loading Transport Layer Security Certifiate Authority certification

Syntax

```
load tls ca cert fromInput *

load tls ca cert fromFile filename
```

Parameters

input

Type: **STRING**

String representing the Certficiate Authority. The system will prompt for a string input. This should be the PEM data.

filename

Type: **STRING**

The name of the PEM data file to load. (Must already exist on ZMP in Files directory)

Example

```
load tls ca cert fromInput *
Enter PEM text (ctr-d to end):
-----BEGIN CERTIFICATE-----
MIIF1TCCA72gAwIBAgIBADANBgkqhkiG9w0BAQsFADB9MRswGQYJKoZIhvcNAQkB
..............RDz+0llBNWe2
-----END CERTIFICATE-----

Success
```

```
Notes:

File must be previously copied onto ZMP into the Files directory
using FTP.
```

Related Commands

```
load tls ca privateKey privKeyPass
show tls summary
```

load tls ca privateKey

Options for loading Transport Layer Security Certifiate Authority Private Key

Syntax

```
load tls ca privateKey privKeyPass * fromInput *

load tls ca privateKey privKeyPass * fromFile filename
```

Parameters

input

Type: **STRING**

String representing the Private Key. The system will promt for a string input. This should be the PEM data.

filename

Type: **STRING**

The name of the PEM data file to load. (Must already exist on ZMP in Files directory)

Example

```
load tls ca privateKey privKeyPass * fromInput *
Enter passphrase: ******
Enter PEM text (ctr-d to end):
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-256-CBC,16DD663CF1D9875E1B8102AD2C020A37

bQVUu4Bp9XrsudbAc2iYGl9cgSplbSD5mAsC3rsc/5XUi+Fe31nhXZKgIHfIui
2v.............................................................
f/NuPpeZ3KLUJGcpUGN4t393aaRXyoidSo4ekgUARJgnt/QND86zCyxJHyd7TmQS
-----END RSA PRIVATE KEY-----

Success
```

Notes:

File must be previously copied onto ZMP into the Files directory using FTP.

Related Commands

load tls ca cert
show cls summary

load tls server caIntermediates

Options for loading Transport Layer Security server Certifiate Authority Intermediates

Syntax

```
load tls server caIntermediates fromInput none|*

load tls server caIntermediates fromFile filename|none
```

Parameters

input

Type: **STRING**

String representing the Certficiate Authority Intermediates. The system will prompt for a string input. This should be the PEM data.

filename

Type: **STRING**

The name of the PEM data file to load. (Must already exist on ZMP in Files directory)

Example

```
load tls server caIntermediates fromInput *
Enter PEM text (ctr-d to end):
-----BEGIN CERTIFICATE-----
MIIF1TCCA72gAwIBAgIBADANBgkqhkiG9w0BAQsFADB9MRswGQYJKoZIhvcNAQkB
..............RDz+0llBNWe2
-----END CERTIFICATE-----

Success
```

Notes:

```
File must be previously copied onto ZMP into the Files directory
using FTP.
```

Related Commands

`show tls summary`

load tls server cert

Options for loading Transport Layer Security server certification

Syntax

```
load tls server cert fromInput *

load tls server cert fromFile filename
```

Parameters

input

Type: **STRING**

String representing the server certification. The system will prompt for a string input. This should be the PEM data.

filename

Type: **STRING**

The name of the PEM data file to load. (Must already exist on ZMP in Files directory)

Example

```
load tls server cert fromInput *
Enter PEM text (ctr-d to end):
-----BEGIN CERTIFICATE-----
MIIF1TCCA72gAwIBAgIBADANBgkqhkiG9w0BAQsFADB9MRswGQYJKoZIhvcNAQkB
..............RDz+0l1BNWe2
-----END CERTIFICATE-----

Success
```

Notes:

```
File must be previously copied onto ZMP into the Files directory
using FTP.
```

Related Commands

```
load tls server privateKey privKeyPass
show tls summary
```

load tls server privateKey

Options for loading Transport Layer Security server Private Key

Syntax

```
load tls server privateKey privKeyPass * fromInput *

load tls server privateKey privKeyPass * fromFile filename
```

Parameters

input

Type: **STRING**

String representing the Private Key. The system will promt for a string input. This should be the PEM data.

filename

Type: **STRING**

The name of the PEM data file to load. (Must already exist on ZMP in Files directory)

Example

```
load tls server privateKey privKeyPass * fromInput *
Enter passphrase: ******
Enter PEM text (ctr-d to end):
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-256-CBC,16DD663CF1D9875E1B8102AD2C020A37

bQVUu4Bp9XrsudbAc2iYGl9cgSplbSD5mAsC3rsc/5XUi+Fe31nhXZKgIHfIui
2v...........................................................
f/NuPpeZ3KLUJGcpUGN4t393aaRXyoidSo4ekgUARJgnt/QND86zCyxJHyd7TmQS
-----END RSA PRIVATE KEY-----

Success
```

Notes:

```
File must be previously copied onto ZMP into the Files directory
using FTP.
```

Related Commands

```
load tls server cert
show cls summary
```

logging

Used to set the level of detail captured by Trouble Reports and manually add text notes into log for Trouble report. To be used at direction of ZeeVee support team to aid in troubleshooting of issues.

Syntax

```
logging level arg
```

Parameters

arg

Type: **INTEGER**

Logging Level. Integer range from 1 to 4

Example

```
logging level 2
Success
```

Syntax

```
logging note string
```

Parameters

string

Type: text

String with length from 1 to 132 characters

Example

```
logging note "my inserted text"
Success
```

logout

Used to logout of the current session for force the logout of any other active session.

Syntax

```
logout force sessionId num
```

Parameters

num

Type: **INTEGER**

Session ID. Integer range from 1 to X, where X is the number is the session you wish to force a logout.

Examples

```
logout
Connection closed by foreign host.
```

```
logout force sessionId 2
Success
```

previewStream

Used to turn on/off a small thumbnail size preview stream that is viewable in the ZyPer Management Platform GUI. (ZyPer4K and ZyPerUHD only) **Note:** Preview streams are not supported by the ZyPer4K-XS and ZyPer4K-XR

Syntax

```
previewStream enc arg comp width size
```

Parameters

enc

Type: **STRING** or **MAC Address**

The name or MAC address of the encoder. String names are case-sensitive.

arg

argument	Description
`stop`	used to manually stop the preview stream. Note that it can turned back on from the GUI
`start`	used to manually start the preview stream.

comp

argument	Description
`hls`	set the format of the preview stream to HLS
`jpeg`	set the format of the preview stream images to JPEG

size

Type: **Integer**

Width of the preview stream in pixels. (180 to 400)

Example

```
previewStream MyEnc start hls width 300
BWG: After Start Preview Streams running = 3
Success


previewStream MyEnc stop
Success
```

HLS Notes

A maximum of 20 preview streams may be enabled at a single time.

ZyPer4K devices must be on firmware release 4.0.1.0 or newer for this feature to work.

The HLS stream can be viewed by any HLS capable viewer such as a browser. The path needed is shown below:

```
http://mp_ip_address/media/encoder_mac_address.m3u8
```

mp_ip_address is the IP address of the ZyPer Management Platform
encoder_mac_address is the MAC address of the Z4K encoder

Example

```
http://192.168.0.78/media/d8:80:39:eb:1c:ee.m3u8
```

JPEG Notes

JPEG images cannot be viewed in the ZyPer Management Platform GUI. This feature is intended for 3rd party control systems to grab individual JPEG images. (1 per second)

The JPEG images can be viewed by any JPEG capable viewer such as a browser. They can also be directly downloaded to a system. The path needed is shown below:

```
http://mp_ip_address/media/encoder_mac_address.jpeg
```

mp_ip_address is the IP address of the ZyPer Management Platform
encoder_mac_address is the MAC address of the Z4K encoder

Examples

```
http://192.168.0.78/media/d8:80:39:eb:1c:ee.jpeg

curl http://192.168.0.78/media/80:1f:12:4d:bb:11.jpeg > preview.jpg
```

redundancy switchover

If there is an active slave, this command causes the existing master to become the slave and the existing slave to become the master. The server does not restart or re-initialize any other state, including any existing video and audio connections.

The IP address that is always assigned to the master. If the active slave becomes the master, this IP address will then terminate at that system. Note that any existing TCP connection will terminate and have to be reopened (to the new master).

Syntax

```
redundancy switchover
```

Parameters

none

Example

```
redundancy switchover
Success
```

Related Commands

`set server redundancy`
`redundancy delete downServers`

redundancy delete downServers

Cleans up and removes any redundant servers from server list that are no longer available in the system.

Syntax

```
redundancy delete downServers
```

Parameters

none

Example

```
redundancy delete downServers
Success
```

Related Commands

```
set server redundancy
redundancy switchover
```

restart device

Restarts the specified device.

Syntax

```
restart device id
```

Parameters

id

Type: **STRING** or **MAC Address**

The name or MAC address of the device. String names are case-sensitive.

Example

```
restart device myEncoder2
Success

restart device 0:1e:c0:f6:cb:76
Success
```

Related Commands

```
shutdown server reboot
```

restore server database

Restores a stored server database from file. (Stored on the ZyPerMP hardware)

Important Note: Saved database to be restored MUST have been created using the exact same version of API that is currently running.

Syntax

```
restore server database name
```

Parameters

name

Type: **STRING**

The name of the stored database. Names are case-sensitive.

Example

```
restore server database jan16_2019
Loaded database jan16_2019; restarting server
Success
```

Related Commands

`save server database`

revert server

Returns to a previously installed version of the API and device database.

This feature can be used to go back to a previous software version and database version in case of a failed software upgrade.   Primarily used to recover previous state if something goes wrong.

Syntax

```
revert server
```

Note: The show sever info command will identify the Previous Version that will be restored to the system.

Example

```
revert server
Reverting from update_nuc_1.8.34605.zyper to update_nuc_2.0.34928.
zyper
Success
```

Related Commands

`show server info`

run preset

Manually executes an existing preset

Syntax

```
run preset name
```

Parameters

name

Type: **STRING**

The name of the existing preset. Names are case-sensitive.

Example

```
run preset lunch
Success

run preset closing
Success
```

Related Commands

```
create preset
delete preset
set preset
show preset
```

save deviceEdid

Saves the EDID of the downstream sink to the `srv/ftp/files` folder on the Management Server. Executing this command will generate two file types: `.bin` and `.txt`. The `.bin` file is the EDID is standard format. The `.txt` file is the decoded EDID data.
See Using Custom EDID Data (page 6) for more information on using this command.

Syntax

```
save deviceEdid id file
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the decoder that is connected to the sink device. String names are case-sensitive.

file

Type: **STRING**

The name of the EDID file. Two files will be created using the *file* name: `.txt` and a file with no extension.

Example

```
save deviceEdid 0:1e:c0:f6:a5:2f myEDID
Success
```

Related Commands

`load encoderEdid`
`set server autoEdidMode`

save server database

Saves the current MP database to a file. (Stored on the ZyPerMP hardware)

Syntax

```
save server database name
```

Parameters

name

Type: **STRING**

The name of the database. Names are case-sensitive.

Example

```
save server database jan16_2019
Saved database to jan16_2019
Success
```

Related Commands

`restore server database`

save system config

Saves the current system configuration to a file. (Stored on the ZyPerMP hardware)

Syntax

```
save system config name
```

Parameters

name

Type: **STRING**

The name of the file. Names are case-sensitive.

Example

```
save system config march24
Saved config to /srv/ftp/files/march24
Success
```

Related Commands

```
save server database
restore server database
```

script

Executes the specified script. The script must exist in the `/srv/ftp/files` folder. Use the optional `loop` argument to place the script in a loop. The script will continue running until a key is pressed on the keyboard.

Syntax

```
script file [loop]
```

Parameters

file

Type: **STRING**

The name of the script file.

Example

```
script myScript
Success
```

Related Commands

`sleep`

send

Sends an IR, RS232 or CEC string to the specified device. Use the `type` parameter to specify an IR, RS232 or CEC code.

Syntax

```
send id type text
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the device.

type

Type: **STRING**

Specifies IR, CEC or RS232 command

argument	Description
`ir`	The string must be the hex representation of the binary data. (Pronto code) The maximum length for a string is 1024 characters. (Not supported on ZyPerUHD)
`cec`	on | off (Used to turn a device on or off)
`cec hexString`	hex-numerals-no-delimiters (ZyPer4K family only)
`rs232`	The string is ASCII and must not exceed 256 characters in length. Spaces and the following control characters are supported as a portion of the string: `\n` New line `\r` Carriage return `\t` Tab `\\` Slash `\xnn` Hex value, where `nn` is a two-digit hex value, including leading zeros

text

Type: **STRING**

The string to send. See the table, above, for restrictions.

Example

```
send myDecoder2 ir 0000006900000015005f0017003000170030001700300001
700300017001700170030001700170017001700170030001700170017003000170
030001700170017003000170017001700170017001700300017003000170030
00200
Success

send myDecoder2 rs232 ZeeVee_support_is_the_greatest\r\n
Success

send myDecoder2 cec on
Success

send myDecoder2 cec off
Success
```

Important Notes

CEC is not supported on ZyPerHD

CEC functionality on the ZyPer4K is only supported with hardware firmware version 3.5.2 and newer.

CEC hexString command is not supported on ZyPerUHD

Related Commands

`set device rs232`

set account all

Sets various security features for all accounts

Syntax

```
set account all option
```

Parameters

option

Type: **STRING**

The security feature to configure

argument	Description
`authMode`	Sets telnet or web authorization. (telnet oldAuth|fullAuth), (web backend|browser)
`concurrentSessionsMax`	Maximum number of sessions allowed. <int>|unlimited
`idleLogout minutes`	Number of idle minutes before a logout is forced. <int>|unlimited
`onThreeFailures`	What to do if login attempt fails 3 consecutive times. lockoutMinutes <int>|none disableAccount true|false
`password`	Set complexity or duration of passwords. complex enabled|disabled minLen <int> duration initialExpire enabled|disabled minDays <int> maxDays <int>|unlimited

Examples

```
set account all authMode telnet oldAuth
Success
set account all authMode web backend
Success
set account all concurrentSessionsMax 5
Success
set account all idleLogout minutes unlimited
Success
set account all onThreeFailures lockoutMinutes none disableAccount
false
Success
set account all onThreeFailures lockoutMinutes 1 disableAccount
true
Warning:(6) You set both actions for onThreeFails. Only setting
disableAccount true
Success
set account all password complex disabled minLen 8
Success
```

set account password

Used to change existing accounts password

Syntax

```
set account password existing currentpass |* newpass
```

Parameters

option

Type: **STRING**

The account feature to configure

argument	Description
currentpass	Current password (Case sensitive) Can also use wildcard * to be prompted for password
newpass	New password (Case sensitive)

Examples

```
set account password existing redsox new yankees
Success

set account password existing * new redsox
Existing password: *******
Success
```

set account username

Sets various features associated to a specific account

Syntax

```
set account username option
```

Parameters

option

Type: **STRING**

The account feature to configure

argument	Description
`2fa`	Enable or disable 2 factor authorization (enabled|disabled)
`expirePassword`	Set password to expire or not (enabled|disabled)
`lock`	Lock a specific account
`password new`	Set a new password for an account (<string>|*)
`role`	Assign an existing role to this account (<rolename>)
`unlock`	Unlock a specific account

Examples

```
set account username ArtW 2fa enabled
2fa-secret: GOCE5AMI6ZP7NNVZTWUK2375UQ
Success
set account username ArtW 2fa disabled
Success
set account username ArtW expirePassword enabled
Success
set account username ArtW lock
Success
set account username ArtW unlock
Success
set account username ArtW role admin
Success
```

set encoder analogAudioOut

Sets the analog audio output source type for the specified encoder. (ZyPer4K family and ZyPerUHD60 family only. Not including UHD60-0E or UHD60-0EA)

Also used to configure the port as an input.

Syntax

```
set encoder id mode source type
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the encoder. String names are case-sensitive.

mode

Type: **STRING**

The audio output to use.

argument	Description
`analogAudioOut`	Audio output from the Audio port on the Encoder.

type

Type: **STRING**

The audio mode (analog or HDMI).

argument	Description
`none`	No analog audio output from the encoder. Port is configured for analgo audio input in this case.
`hdmiAudioDownmix`	Uses downmixed audio from input HDMI stream.

Example

```
set encoder Myencoder1 analogAudioOut source hdmiAudioDownmix
Success
set encoder Myencoder1 analogAudioOut source none
Success
```

set encoder danteAudioOut

Sets the audio input source source type for the specified encoder. (ZyPerUHD60 Dante Enabled Encoders only)

Only used when UHD60 Encoder is configured as a Dante Tranmitter

Syntax

```
set encoder id danteAudioOut source mode
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the encoder. String names are case-sensitive.

mode

Type: **STRING**

The audio output to use.

argument	Description
`analogAudio`	Place the input Analog Audio from 5-pin phoenix connector onto Dante Network
`hdmiAudioDownmix`	Place the Downmix HDMI audio from HDMI connector onto Dante Network

Example

```
set encoder Myencoder1 danteAudioOut source hdmiAudioDownmix
Success
set encoder Myencoder1 danteAudioOut source analogAudio
Success
```

Related Commands

`set decoder danteAudioOut`

set encoder edid audio

Sets the allowable audio input formats at the encoder.

Detailed Background

ZeeVee added a feature that will allow compressed formats to be passed down in an encoder EDID file. This EDID will be then forwarded to the source device to determine the type of audio sent to the encoder.

This enhancement was to provide fast-switched connections the “compressed audio” options in the EDID file. Prior to this version with the fast-switched connection, ZeeVee modified the EDID passed from the decoder to the encoder and removed all compression formats. This left just LPCM as the only option under the “Audio data block” in the edid file.

```
>>> Audio data block <<<
   Linear PCM, max channels 8
     Supported sample rates (kHz): 192 176.4 96 88.2 48 44.1 32
     Supported sample sizes (bits): 24 20 16
```

The information provided to the Video Source device (such as BluRay Player or Media player) increases the possibility of compression being a chosen audio format. However it is still up to the device to choose uncompressed or compressed formats. It is important to know that some devices such as the Apple 4K TV requires the audio output type to be set (even if the audio format is available in the EDID). Compression will need to be set manually on these types of devices.

In addition any downmixed stream internal to ZyPer devices will not process compressed audio, so you will not hear compressed audio on these connections.

Syntax

`set encoder` *`id`* `edid audio` *`mode`*

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the encoder. String names are case-sensitive.

mode

Type: **STRING**

The supported input audio mode

argument	Description
`onlyPcm`	Force PCM audio format at encoder. Does not allow compressed formats such as AC3.
`allowCompressed`	Passes the decoders edid with unmodified audio information and thus allows compression options to be seen.
`serverDefault`	Follows the server setting

Example

```
set encoder Cuba edid audio allowCompressed
Success
```

Related Commands

`set server encoderDefaultAudioFormat`

Additional Information

In an attempt to properly Identify the Audio Streams used under the product the following changes were also made along with some modification to the API commands.

Product	Old Audio Stream Name	New Stream Name
`ZyPer4K`	hdmi (used in genlocked mode)	hdmiPassthroughAudio
`ZyPer4K`	hdmi-audio-downmix	hdmiAudio
`ZyPer4K`	analog-audio	analogAudio
`ZyPerUHD`	audio	hdmiAudio
`ZyPerUHD`	analog-audio	analogAudio
`ZyPerUHD60`	none	hdmiAudio
`ZyPerUHD60`	none	analogAudio

The ZyPer4k can have analog and digital audio streams going to the decoder at the same time and routing either way.

So:

```
join <enc> <dec> hdmiAudio
```

is simply used to route 'standard' HDMI audio from encoder to decoder.

Which port it goes out is based on defaults or the set command.

```
Zyper$ set decoder z4k_dec_desk_58 analogAudioOut source
    analogAudio
    hdmiAudioDownmix
```

or

```
Zyper$ set decoder z4k_dec_desk_58 hdmiAudioOut source
    analogAudio
    hdmiAudio (For HDMI out only)
    hdmiAudioDownmix
    hdmiPassthroughAudio (This is for genlockonly)
```

set encoder hdcpMode

Sets the hdcp mode for the specified encoder.

Syntax

```
set encoder id mode type
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the encoder. String names are case-sensitive.

mode

Type: **STRING**

The hdcp mode to use

argument	Description
`hdcpMode`	HDCP mode of the Encoder.

type

Type: **STRING**

Enable or Disable

argument	Description
`enabled`	encoder will accept HDCP 1.4/2.2 compatible streams. Also will accept unencrypted inputs.
`enabled1_4`	encoder will accept HDCP 1.4 compatible streams. Also will accept unencrypted inputs.
`disabled`	encoder will reject HDCP 1.4/2.2 compatible streams. Will only accept unencrypted inputs.

Example

```
set encoder Myencoder1 hdcpMode disabled
Success
```

Notes

```
Useful when user does not want Source such as Apple Macbook to
provide HDCP protected content to the Encoder.
```

set decoder

Sets the audio output type and video timing details for the specified decoder.

Syntax

```
set decoder id mode type
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the decoder. String names are case-sensitive.

mode

Type: **STRING**

argument	Description
`analogAudioOut`	Audio output from the Audio port on the decoder.(ZyPer4K family only)
`connectionMode`	Sets/changes current connection mode to decoder. (Options are fast-switched, genlocked and genlocked-scaled) (ZyPer4K family only)
`displayAdvancedTiming`	Set advanced features, Front porch, sync width, sync polarity and total size
`displayMode`	Set display to box, crop or stretch input stream within display resolution
`displayResolution`	Set display resolution manually (pixels) or automatically based on EDID.
`hdcpMode`	Allows user to force HDCP protection at level 1.4 or 2.2 on previously unprotected content. (ZyPerUHD and ZyPerUHD60 only) Will cause decoder to reboot.
`hdmiAudioOut`	Audio output from the HDMI port on the decoder. (ZyPer4K family only)

type

Type: **STRING**

HDCP options. (Note: Valid with ZyPerUHD and ZyPerUHD60 only) Used to minimize connection time.

argument	Description
`auto`	Maintain existing HDCP level. None if none
`forceVersion1.4`	Apply HDCP 1.4 protection to output stream
`forceVersion2.2`	Apply HDCP 2.2 protection to output stream

The audio mode (analog out or HDMI out).

argument	Description
`source analogAudio`	Uses the audio output created with the join command.
`source hdmiAudio`	Uses the HDMI stream (HDMI audio-out only) Use if video in Fast-Switch mode.
`source hdmiPassthroughAudio`	Used if video is in Genlock mode.
`source hdmiAudioDownmix`	Uses the HDMI-downmix stream.

Display timing, aspect ratio, mode, size.

argument	Description
`syncFrontPorch`	Synchronization mode.
`syncWidth`	Synchronization width
`hsyncPolarity`	Horizontal sync polarity (auto, negative, positive)
`vsyncPolarity`	Vertical sync polarity (auto, negative, positive)
`totalSize`	Horizontal and vertical size (Pixels or auto)
`box`	Box image within display. (Smaller source to larger display)
`crop`	Crop image within display (Larger source to smaller display)
`stretch`	Scale image to fill display. (Scale up or down) (Default Setting)
`pixelsHoriz`	Width in pixels or auto
`pixelsVert`	Height in pixels or auto
`fps`	Frames per second
`source`	Match decoder resolution to source input size
`auto`	automatically based on EDID

Command Description: Override output display size and fps

set decoder <Decoder_Name or MAC> displayResolution activeSize <int> pixelsHoriz <int> pixelsVert <int> fps <int>|source

This command allows an override of EDID parameters supplied by the display. Regardless of what the supplied EDID indicates, the decoder will generate a stream with specified overall size and frame rate parameters.

Note that in "genlock-scaled" mode, the frame rate parameter is ignored – it must be the same as the encoder frame rate. This does mean care must be taken when setting this parameter if the source stream is 60fps (e.g. 720p60fps) and scaled to 4K. That only works if the display supports 4K60.

If configured resolution specification in these parameters that exceed the displayed maximum resolution, the display will black out with no indication to the user.

Example command:
Zyper$ set decoder Dec1 displayResolution activeSize 3840 2160 fps 60

Command Description: Output display size determined by received EDID

Command Syntax
set decoder <Decoder_Name or MAC> displayResolution auto

The command causes the decoder to set output display size to the “preferred” value in the EDID received from the display.

Command Description: Override detailed video parameters

Command Syntax
set decoder <decoderMac|decoderName> displayAdvancedTiming activeSize <pixelsHoriz:int> <pixelsVert:int> fps <float> total-size <pixelsHoriz:int> <pixelsVert:int> syncFrontPorch <pixelsHoriz:int> <pixelsVert:int> syncWidth <pixelsHoriz:int> <pixelsVert:int> syncPolarity hPositive|hNegative vPositive|vNegative

This command allows an override of EDID parameters supplied by the display. Regardless of what the supplied EDID indicates, the decoder will generate a stream with specified detailed timing parameters.
If configured resolution specification in these parameters that exceed the displayed maximum resolution, the display will black out with no indication to the user.

Example command:
Zyper$ set decoder Dec1 displayAdvancedTiming activeSize 1920 1080 fps 60 totalSize 2200 1200 syncFrontPorch 88 4 syncWidth 44 5 syncPolarity hPositive vPositive

set decoder autoAudioConnections hdmiAudioFollowVideo

Tells the decoder to automatically join Audio associated with connected Video stream or not.

Syntax

```
set decoder id autoAudioConnections hdmiAudioFollowVideo arg
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the decoder. String names are case-sensitive.

arg

Type: **STRING**

Supply one of the following arguments.

argument	Description
`enabled`	Audio will follow video automatically
`disabled`	Audio will not follow video automatically

Example

```
set decoder myDecoder autoAudioConnections hdmiAudioFollowVideo
enabled
Success
```

set decoder danteAudioOut

Sets the ZyPerUHD60 Dante enabled decoder to either Transmit or Receive mode. (Note that unit will reboot after making a change)

Syntax

```
set decoder id danteAudioOut source arg
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the decoder. String names are case-sensitive.

arg

Type: **STRING**

Supply one of the following arguments.

argument	Description
`joinedAudio`	Makes the UHD60 a Dante Transmitter
`none`	Makes the UHD60 a Dante Receiver

Examples

```
set decoder MyDecoder1 danteAudioOut source none
Warning:(36) Device has been restarted
Success
```

```
set decoder myDecoder danteAudioOut source joinedAudio
Warning:(36) Device has been restarted
Success
```

Related Commands

```
join dante directDanteAudio
set decoder danteAudioOut
```

set decoder edidPreferMode

Sets the preferred resolution from the display EDID

Syntax

```
set decoder id mode type
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the decoder. String names are case-sensitive.

mode

Type: **STRING**

argument	Description
`edidPreferMode`	Select preferred EDID mode

type

Type: **STRING**

HDCP options. (Note: Valid with ZyPerUHD only) Used to minimize connection time.

argument	Description
`max`	Default mode. Selects the largest resolution defined in the EDID.
`strict`	Selects the Preferred resolution as stated in the display EDID

"**max**" – Default mode. Selects the largest resolution defined in the EDID. This has been the operating mode prior to this command. In almost all cases, this is the native resolution of the display. However, some displays can accept a resolution above the native (and scale down). In this case, it is better to use the "strict" mode.

"**strict**" – The Preferred Resolution is selected as defined in the EDID 1.3 specification. EDID 1.3 specifies that the first Detailed Timing Descriptor in the Standard Timing Information block is always the preferred resolution, although it is only the native resolution if the native-resolution flag is set. If the native-resolution flag is not set, then the maximum resolution will be chosen (falls back to "max" mode).

Note: All comparisons of "resolution" actually mean comparisons of the associated Pixel Clock. The Pixel Clock represents the entire resolution definition: horizontal and vertical size, fps, bit-depth and color decimation (RGB/4:4:4, 4:2:2, 4:2:0).

The command will immediately reanalyze the active EDID and if needed change the preferred resolution and reconnect to the encoder.

The reason for the "max" mode, and for it being the default, is that many displays do not follow the EDID 1.3 specification, claiming a native, Preferred Resolution below the display's actual native resolution. It is fairly common for a UHD display to have an HD resolution as the specified preferred resolution.

Note: ZyPer4K and ZyPerUHD, depending on mode, may support only a limited set of output resolutions, particularly when the scaler is enabled. ZMP will choose the active resolution based decoder capability, scaler mode and preferred resolution. However, the display's Preferred Resolution is displayed regardless of what the decoder ultimate actually uses. The active resolution is displayed in the decoder status as well.

Overriding Preferred Resolution Selection

It should rarely be required. But if the EDID supplied by the display is not correct, or for some reason ZMP chooses a Preferred Resolution that is not desired, the following command will force the decoder to a specific output resolution:

```
        set decoder <decoder> displayResolution activeSize <int>
<int> fps <float>
```

When set, the decoder output resolution will remain as specified without exception.

Note: When in this mode, it is very possible that no video will be displayed, and with no warning from ZMP. It is up to the user to ensure that the output settings are valid for the display.

Scaler Control

ZyPer4K, ZyPerUHD60 and ZyPerUHD decoders have output scaling. Besides the obvious benefit of supporting HD-only displays with a UHD source, the other major benefit is faster switching times. With ZyPer4K, there is virtually no delay. With ZyPerUHD it is less than a one second.

However, there are some cases where disabling the scaler produces a better image. Of course, if the scaler is disabled and the source provides a resolution greater than the display's ability, it will be black. To solve this problem, we have a new mode that disables the scaler, but only if the display can handle the source resolution.

The decoder display-resolution command now has an option called "source".

```
set decoder <decoder> display-resolution source
```

When in "source" mode the scaler is disabled if the display can handle the received resolution. Otherwise it is automatically enabled (e.g. if the source is 480 and the Preferred Resolution is 1080 then the scaler is disabled, but if source is UHD and the Preferred Resolution is 1080, then the scaler is enabled).

The downside to this mode: switching time between non-scaled resolutions is about 3 seconds. Switching time between scaled and non-scaled resolutions is closer to 4s.

Active Output Resolution Selection

Selecting the correct output resolution for a decoder is, unfortunately, a fairly complicated endeavor. Clearly depends on the display (Preferred Resolution), but also on the decoder capability and the source resolution.

Remember: All comparisons of "resolution" actually mean comparisons of the associated Pixel Clock. The Pixel Clock represents the entire resolution definition: horizontal and vertical size, fps, bit-depth and color decimation (RGB/4:4:4, 4:2:2, 4:2:0).

Also, setting "edidPreferMode" only affects which Preferred Resolution is chosen. It does not affect when that Preferred Resolution is used (or if it is used). Although the chosen Preferred Resolution is always reported in the decoder status output (as is the chosen active output resolution).

ZyPer4K Devices

Presently, the decoder active resolution is limited to a number of resolutions: 4096x2160, 3840x2160, 1080x1920 or 1280x720. The closest lower resolution is used.

There are a number of exceptions to the operation.
• Scaler always converts to 8bit 444/RGB. That means UHDp60 4:2:0 is converted to UHDp60 4:4:4. UHDp60 YUV 4:2:0 bit rate is lower than HDM 1.4. But UHDp60 4:4:4 is not. In this case, the output FPS is divided by 2.
• If in genlockScaled, videoWall or window mode, decoder FPS must equal encoder FPS
    o Means 1080p60 scaling to UHD must be UHDp60, which won't work if display is only UHDp30 capable.
    o If UHDp60 > decoder Preferred Resolution, then the output is left at 1080p60.
• If source is 1080i
    o Output must be input FPS * 2
    o If decoder resolution > 1080, it is set to 1080.

displayResolution = auto
When in this mode, the output resolution will always be the ***Preferred Resolution***. There really is no reason not to use this mode with the Z4K Charlie and will produce the lowest switching times.

displayResolution = source
When in this mode, the output resolution will always be the ***encoder resolution***, unless the source resolution greater than the encoder resolution (same case as displayResolution auto).

This mode may provide better video at or below the preferred resolution of the display. However, the switching time is somewhat slower (~3.3s).

displayResolution = sourceIgnoreEdid
Same operation as displayResolution = auto, but effectively using a manually entered Preferred Resolution. Generally only used if the EDID is incorrect.

ZyPerUHD

The ZyPerUHD scaler scales up fine (source resolution lower than display preferred). However, it can only scale down from UHD to 1080.

Even with this limitation, the vast majority of installations will be fine. The exception comes with PC-based resolutions. For example a case that will not work well:
• 1080-only display and source resolution of 1920x1200

For the cases where VESA/PC resolutions such as 1920x1200, 2560x1440 and 2560x1600 are needed, all displays must be at least that resolution or greater. For example, a 1920x1200 display can handle all resolutions up to 1920x1200 and it can also handle UHD, since the decoder will output UHD scaled down to 1080 (which is fine for a 1920x1200 display).

And, clearly, all of those resolutions will be fine if the displays are UHD capable (scaling up works, plus, the new mode "display-size source" can be used).

If a configuration that causes downscaling that is not handled well, likely generating poor video, a warning will be generated.

displayResolution = auto
When in this mode, the output resolution will always be the ***Preferred Resolution***, unless the source resolution greater than the preferred resolution.

This mode provides the fastest switching time (less than 1 second). However, there may be some cases where video quality is less than when using display-resolution = source.

If source is greater than decoder Preferred Resolution, then decoder output will be ***1920x1080*** (unless the display does not support it) with the preferred FPS. As noted, the only case this normally works for is when the source is 3840x2160.

displayResolution = source
When in this mode, the output resolution will always be the ***encoder resolution***, unless the source resolution greater than the encoder resolution (same case as displaySize auto).

This mode may provide better video at or below the preferred resolution of the display. However, the switching time is somewhat slower (~3.3s).

displayResolution = sourceIgnoreEdid
Same operation as displayResolution = auto, but effectively using a manually entered Preferred Resolution. Generally only used if the EDID is incorrect.

set decoder hdmi5vControl

Enables or disables 5V HDMI line of the decoder. (ZyPer4K-XS and ZyPer4K-XR only)

When decoder is not receiving a video stream the decoder will disable the 5V HDMI line.

Syntax

```
set decoder id hdmi5vControl arg
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the decoder. String names are case-sensitive.

arg

Type: **STRING**

Supply one of the following arguments.

argument	Description
`enabled`	5V HDMI line will disable when no video streamed ot the decoder.
`disabled`	5V HDMI line is never disabled. (Default)

Example

```
set decoder myDecoder hdmi5vControl enabled
Success
```

If you attempt to run this command on a decoder that is not XS/XR or not on correct firmware you get the following error.

```
Error:(29) Device myDecoder does not support or cannot change:
videoPort with value hdmi5vControl.
```

Notes

ZyPer4K-XS or ZyPer4K-XR must be updated to firmware version 1.3.2.4 or newer for this command to work.

The connection before disconnecting video from the decoder must be “genlocked” to fully disable video and cut the 5V line.

set decoder osdStatusMode

Enables or disables on-screen-display feature of the decoder. (ZyPerUHD and ZyPerUHD60 only)

When decoder is not receiving a stream the decoder will display a “No Source Found” screen. In the lower corner of this screen is displayed the following information:

Firmware version and date
IP address of the decoder
Remote IP: (Encoder it is attempting to get stream from if any)
MAC Address

The osdStatusMode command will make this information visible or not.
Note, changing status with the command will force the decoder to reboot.

Syntax

```
set decoder id osdStatusMode arg
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the decoder. String names are case-sensitive.

arg

Type: **STRING**

Supply one of the following arguments.

argument	Description
`enabled`	OSD feature enabled.
`disabled`	OSD feature disabled.

Example

```
set decoder myDecoder osdStatusMode enabled
Warning:(36) Device myDecoder has been restarted
Success
```

set decoder powerSave

Enables or disables power save feature of the decoder. (ZyPerUHD and ZyPerUHD60 only)

When decoder is not receiving a stream the decoder will enter a low power mode and the display will go black.

Syntax

```
set decoder id powerSave arg
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the decoder. String names are case-sensitive.

arg

Type: **STRING**

Supply one of the following arguments.

argument	Description
`enabled`	Power save feature enabled.
`disabled`	Power save feature disabled.

Example

```
set decoder myDecoder powerSave enabled
Success
```

set device general name

Sets the name for the specified encoder or decoder.

Syntax

```
set device id  general name str
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the device. String names are case-sensitive.

Important Note: The following characters are not valid for device names.

Colon :
Quotes “
Blank Spaces

str

Type: **STRING**

The name for the device.

Example

```
set device myDecoder5 general name Samsung-55
Success
```

Related Commands

```
set device ip
set device ip static
set device rs232
set device sourceDisplay iconImageName
set device sourceDisplay location
set device sourceDisplay manufacturer
set device sourceDisplay model
set device sourceDisplay serialNumber
```

set device ip

Sets DHCP mode for the specified device.

Syntax

```
set device id ip arg
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the decoder. String names are case-sensitive.

arg

Type: **STRING**

Supply one of the following arguments.

argument	Description
`dhcp`	IP address assigned by DHCP server
`linkLocal`	IP address self assigned Link-Local

Example

```
set device ABC ip dhcp
Success
```

Related Commands

```
set device general name
set device ip static
set device rs232
set device sourceDisplay iconImageName
set device sourceDisplay location
set device sourceDisplay manufacturer
set device sourceDisplay model
set device sourceDisplay serialNumber
```

set device ip static

Sets static mode for the specified device. The IP address, subnet mask, and gateway must be supplied.

Syntax

```
set device id  ip static addr mask gatew
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the decoder. String names are case-sensitive.

addr

Type: **IP Address**

The desired IP address for the device.

mask

Type: **IP Address**

The desired subnet mask for the device.

gatew

Type: **IP Address**

The desired gateway for the device.

Example

```
set device ABC ip static 10.5.68.121 255.255.255.0 10.5.64.1
Success
```

Related Commands

```
set device general name
set device ip
set device rs232
set device sourceDisplay iconImageName
set device sourceDisplay location
set device sourceDisplay manufacturer
set device sourceDisplay model
set device sourceDisplay serialNumber
```

set device irProcessing

Configures ZyPer4K endpoint to process input IR commands to issue *channel up* or *channel down* API command.

ZyPer Remote is and IR remote control. Part number: ZVREMOTE
Hitting Up or CH+ button will issue *channel up* API command.
Hitting Down or CH- button will issue *channel down* API command.

ZeeVee IR Receiver is required to be plugged into Decoder IR input port.
Part number: Z4KIRRX

ZyPer Trigger is a device to connect a "button" to the ZeeVee decoder IR ports.
Part number: Z4KIRTRIGTX

Syntax

```
set device id irProcessing arg
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the decoder. String names are case-sensitive.

arg

Type: **STRING**

Supply one of the following arguments.

argument	Description
`zyperTrigger`	Process button press from ZyPer Trigger device
`zyperRemote`	Process up/down button press from ZeeVee IR remote control
`none`	Do not process IR inputs

Example

```
set device Z4KDec irProcessing zyperRemote
Success
```

Related Commands

`set device general name`
`set device ip static`

set device rs232

Sets the RS232 settings for the specified device.

Syntax

```
set device id rs232 baud data stop parity
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the device. String names are case-sensitive.

baud

Type: **INTEGER**

The baud rate for the device. Supply one of the following values from the table below.

argument		
2400	9600	19200
38400	57600	115200

data

Type: **INTEGER**

The data bit setting for the device. Supply one of the following values from the table below.

argument
7-bits
8-bits

stop

Type: **INTEGER**

The stop bit setting for the device. Supply one of the following values from the table below.

argument
1-stop
2-stop

parity

Type: **STRING**

The parity setting for the device. Supply one of the following values from the table below.

argument
even
odd
none

Example

```
set device decoderNumber2 rs232 57600 8-bits 1-stop none
Success
```

Related Commands

```
send
set device general name
set device ip
set device ip static
set device sourceDisplay iconImageName
set device sourceDisplay location
set device sourceDisplay manufacturer
set device sourceDisplay model
set device sourceDisplay serialNumber
```

set device security

Mechanism to enable security over Semtech's server-device communication. First, there has to be an overall key associated with the server (deviceSecurityKey). Then, each device has to enable the security. Provides authentication and encryption. This only works with ZyPer4K-XS and ZyPer4K-XR devices. Once a device has been enabled for a specific server, it will not work with any server without the same key. Redundancy automatically sets the same key on both servers. If the key is lost, devices have to be hardware factory defaulted

Syntax

```
set device id security arg
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the device. String names are case-sensitive.

arg

Type: **STRING**

Supply one of the following arguments.

argument	Description
`enabled`	Feature enabled.
`disabled`	Feature disabled.

Example

```
set device Encoder_1 security enabled
Success
```

Related Commands

`set device ipserver security deviceSecurityKey`

set device sendIpMcastRange

Sets allowable range of multicast addresses for selected devices. (ZyPer4K family only)

Syntax

```
set device id sendIpMcastRange first:ip last:ip
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the encoder. String names are case-sensitive. Can all use “all” or “encoders” as an ID option.

first:ip / last:ip

Type: **Multicast Address**

Supply the starting and ending multicast addresses in the allowable range.

Note: Allowable range is from 224.1.1.1 to 239.255.255.255

Example

```
set device encoders sendIpMcastRange 224.1.1.25 224.1.2.125
```

Related Commands

```
set device general name
set device ip static
set device rs232
set device sourceDisplay iconImageName
set device sourceDisplay location
set device sourceDisplay manufacturer
set device sourceDisplay model
set device sourceDisplay serialNumber
```

set device sourceDisplay iconImageName

Assigns an icon to the desired device. The icon will be displayed within the ZMP to identify the device.

Syntax

```
set device id sourceDisplay iconImageName fname
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the encoder or decoder. String names are case-sensitive.

fname

Type: **FILENAME**

The full filename of the icon to be used. The filename is case-sensitive.

argument	Description
`abc`	ABC network icon
`cbs`	CBS network icon
`nbc`	NBC network icon
`fox`	Fox network icon
`xbox`	Xbox game console icon
`golf`	Golf channel icon
`espn`	ESPN network icon
`tennis`	Tennis channel icon
`cnn`	CNN network icon
`ps3`	PlayStation game console icon
`DVD`	DVD player icon
`BluRay`	BluRay icon
`VCR`	VCR icon
`CableBox`	Cable box icon
`Laptop`	Laptop icon
`BroadcastCamera`	Broadcast camera icon
`SecurityCamera`	Security camera icon

Example

```
set device Encoder1 sourceDisplay iconImageName cbs
Success
```

Related Commands

```
set device general name
set device ip
set device ip static
set device rs232
set device sourceDisplay location
set device sourceDisplay manufacturer
set device sourceDisplay model
set device sourceDisplay serialNumber
```

set device sourceDisplay location

Assigns a location description for the specified device.

Syntax

```
set device id sourceDisplay location loc
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the device. String names are case-sensitive.

loc

Type: **STRING**

The location description of the device (e.g. “Conference_Rm”, “Den”, etc.). Do not use quotes when specifying this string value.

Example

```
set device myDecoder3 sourceDisplay location VideoWall-1
Success
```

Related Commands

```
set device general name
set device ip
set device ip static
set device rs232
set device sourceDisplay iconImageName
set device sourceDisplay manufacturer
set device sourceDisplay model
set device sourceDisplay serialNumber
```

set device sourceDisplay manufacturer

Assigns a manufacturer description for the specified device.

Syntax

```
set device id sourceDisplay manufacturer mfg
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the device. String names are case-sensitive.

mfg

Type: **STRING**

The manufacturer description of the device (e.g. “Sony”, “Panasonic”, etc.). Do not use quotes when specifying this string value.

Example

```
set device myDecoder3 sourceDisplay manufacturer Sony
Success
```

Related Commands

```
set device general name
set device ip
set device ip static
set device rs232
set device sourceDisplay iconImageName
set device sourceDisplay location
set device sourceDisplay model
set device sourceDisplay serialNumber
```

set device sourceDisplay model

Assigns a model description for the specified device.

Syntax

```
set device id sourceDisplay model model
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the device. String names are case-sensitive.

model

Type: **STRING**

The manufacturer's model number of the device.
Do not use quotes when specifying this string value.

Example

```
set device myDecoder3 sourceDisplay model DVPSR210P
Success
```

Related Commands

```
set device general name
set device ip
set device ip static
set device rs232
set device sourceDisplay iconImageName
set device sourceDisplay location
set device sourceDisplay manufacturer
set device sourceDisplay serialNumber
```

set device sourceDisplay serialNumber

Assigns the manufacturer serial number for the specified device.

Syntax

```
set device id sourceDisplay serialNumber serial
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the device. String names are case-sensitive.

serial

Type: **STRING**

The manufacturer serial number of the device.

Example

```
set device myDecoder3 sourceDisplay serialNumber 123456789
Success
```

Related Commands

```
set device general name
set device ip
set device ip static
set device rs232
set device sourceDisplay iconImageName
set device sourceDisplay location
set device sourceDisplay manufacturer
set device sourceDisplay model
```

set device usbFilter

Allows restrictions to USB use on selected device. (ZyPer4K only. Not supported on ZyPer4K-XS or ZyPer4K-XR units)

Syntax

```
set device id usbFilter arg
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the encoder or decoder. String names are case sensitive

arg

Type: **STRING**

Supply one of the following arguments.

argument	Description
none	No restrictions on USB port
exceptHid	Allows any USB device except HID devices
storage	Allows any USB device except Storage devices

Example

```
set device myDecoder2 usbFilter none
Success
```

Related Commands

```
set device general name
set device ip
set device rs232
set device sourceDisplay iconImageName
set device sourceDisplay location
set device sourceDisplay manufacturer
set device sourceDisplay model
set device sourceDisplay serialNumber
```

set device utilityPort

Enables or disables the 1Gb Utility Ethernet port on the specified encoder or decoder. (ZyPer4K and ZyPerUHD60 only)

Syntax

```
set device id utilityPort arg
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the device. String names are case-sensitive.

arg

Type: **STRING**

Supply one of the following arguments.

argument	Description
`enabled`	Ethernet port is enabled.
`disabled`	Ethernet port is disabled
`dante`	Ethernet port is enabled as a sepate LAN with Dante TX/RX. (ZyPerUHD60-2EA only)

Example

```
set device myDecoder5 utilityPort disabled
Success
```

Related Commands

```
set device general name
set device ip
set device rs232
set device sourceDisplay iconImageName
set device sourceDisplay location
set device sourceDisplay manufacturer
set device sourceDisplay model
set device sourceDisplay serialNumber
```

set device videoPort

Selects active input port for ZyPer4K units with multiple inputs. (ZyPer4K and ZyPerUHD60 only)

Syntax

```
set device id videoPort arg
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the encoder. String names are case sensitive.

arg

Type: **STRING**

Supply one of the following arguments.

argument	Description
`hdmi`	Use the HDMI input (Z4K Located to the right) “HDMI IN1” on UHD60
`hdmiOptionalIn`	Use the HDMI input (Located to the left). “Primary Input” Z4K Dual input HDMI only. “HDMI IN2” on UHD60
`usbc`	Use the USB-C input.
`auto`	Use whatever port has an active input if only one source is connected. Note this is only valid for DisplayPort and Dual-HDMI options. Does not work with SDI or Analog inputs. Please see ZyPer4K User Guide for details on what port is used if both ports have an active input.
`displayPort`	Use the Display-Port input
`hdsdi`	Use the SDI input port
`12gsdi`	Use the 12G SDI input port
`component`	Use component input. (Requires ZeeVee Hydra cable)
`composite`	Use composite input (Requires ZeeVee Hydra cable)
`s-video`	Use s-video input (Audio not supported)
`vga`	Use vga input. (Requires ZeeVee VGA cable)

Example

```
set device myEncoder1 videoPort displayPort
Success
```

set multiview

Assigns source to a position and size within a multiview display. (ZyPer4K family only)

Syntax

```
set multiview id windowNumber wn encoderName enc position
percentPositionX posx percentPositionY posy percentSizeX sx
percentSizeY sy layer ly

set multiview id windowNumber wn encoderName enc position
pixelPositionX posx pixelPositionY posy pixelSizeX sx pixelSizeY sy
layer ly
```

Parameters

id

Type: **STRING**

Name of previously created multiview. String names are case-sensitive.

wn

Type: **Integer**

Window number within the multiview (1-19)

enc

Type: **STRING or MAC Address**

The name or MAC address of the encoder. String names are case-sensitive.

percentPositionX

Type: **Integer**

X coordinate in percentage of multiview canvas. Upper left corner of window. (0-99)

percentPositionY

Type: **Integer**

Y coordinate in percentage of multiview canvas. Upper left corner of window. (0-99)

pixelPositionX

Type: **Integer**

X coordinate of multiview in multiview canvas. Upper left corner of window.

pixelPositionY

Type: **Integer**

Y coordinate of multiview in multiview canvas. Upper left corner of window.

`pixelSizeX`

Type: Integer

Size/Length of multiview window. Number of pixels in multiview canvas.

`pixelSizeY`

Type: Integer

Size/Height of multiview window. Number of Pixels in multiview canvas

`percentSizeX`

Type: Integer

Size/Length of multiview window. As a percentage of X dimension of multiview canvas. (0-99)

`percentSizeY`

Type: Integer

Size/Height of multiview window. As a percentage of Y dimension of multiview canvas. (0-99)

`ly`

Type: Integer

Window Layer. Value from 1-9 with layer 1 being the bottom layer and 9 being the top.

Examples

Using Percentages

```
set multiview myMview1 windowNumber 1 encoderName myEnc1
percentPositionX 50 percentPositionY 50 percentSizeX 25
percentSizeY 25 layer 3
```

Using Pixel Values

```
set multiview myMview1 windowNumber 1 encoderName myEnc1
pixelPositionX 1920 pixelPositionY 1080 pixelSizeX 800 pixelSizeY
600 layer 3
```

Related Commands

```
create multiview
delete videoWalll
delete multiviewWindow
set multiview audioSource windowNumber
show multiviews config
show multiviews status
```

set multiview (layer, position, size)

Allows user to change a multiview window layer, position or size without specifying other parameters. (ZyPer4K family only)

Syntax

```
set multiview id windowNumber wn positionX posx positionY posy
sizeX sx sizeY sy layer ly
```

Parameters

`id`

Type: STRING

Name of previously created multiview. String names are case-sensitive.

`wn`

Type: Integer

Window number within the multiview (1-19)

`percentPositionX`

Type: Integer

X coordinate in percentage of multiview canvas. Upper left corner of window. (0-99)

`percentPositionY`

Type: Integer

Y coordinate in percentage of multiview canvas. Upper left corner of window. (0-99)

`pixelPositionX`

Type: Integer

X coordinate of multiview in multiview canvas. Upper left corner of window.

`pixelPositionY`

Type: Integer

Y coordinate of multiview in multiview canvas. Upper left corner of window.

`pixelSizeX`

Type: Integer

Size/Length of multiview window. Number of pixels in multiview canvas.

pixelSizeY

Type: **Integer**

Size/Height of multiview window. Number of Pixels in multiview canvas

percentSizeX

Type: **Integer**

Size/Length of multiview window. As a percentage of X dimension of multiview canvas. (0-99)

percentSizeY

Type: **Integer**

Size/Height of multiview window. As a percentage of Y dimension of multiview canvas. (0-99)

ly

Type: **Integer**

Window Layer. Value from 1-9 with layer 1 being the bottom layer and 9 being the top.

Examples

```
set multiview myMview1 windowNumber 2 layer 4
Success

set multiview myMview1 windowNumber 2 size percentSizeX 50 percentSizeY
50
Success

set multiview mv1 windowNumber 1 size pixelSizeX 500 pixelSizeY 400
Success
```

Related Commands

`create multiview`
`delete multiview`
`delete multiviewWindow`
`set multiview audioSource windowNumber`
`show multiviews config`
`show multiviews status`

set multiview allowMainStream

Controls if the main unscaled video stream from an encoder can be used in a multiview. (ZyPer4K family only)

Syntax

```
set multiview id allowMainStream arg
```

Parameters

id

Type: **STRING**

Name of previously created multiview. String names are case-sensitive.

arg

Type: **STRING**

Supply one of the following arguments.

argument	Description
`enabled`	Unscaled stream is allowed
`disabled`	Unscaled stream is not allowed

Example

```
set multiview myMview1 allowMainStream enabled
Success
```

Related Commands

```
create multiview
delete multiview
delete multiviewWindow
show multiviews config
show multiviews status
```

set multiview audioSource windowNumber

Selects the input source to provide Audio for multiview display. (ZyPer4K family only)

Syntax

```
set multiview id audioSource windowNumber arg
```

Parameters

id

Type: **STRING**

Name of previously created multiview. String names are case-sensitive.

arg

Type: **STRING / Integer**

Supply one of the following arguments.

argument	Description
`Integer`	Integer from 1-19 identifying source to use for audio
`none`	Set no audio for the multiview window

Example

```
set multiview myMview1 audioSource window number 4
Success
```

Related Commands

```
create multiview
delete multiview
delete multiviewWindow
show multiviews config
show multiviews status
```

set multiview windowNumber channel up/down

Cycles the encoder source up/down for a specified multiview window. (ZyPer4K family only)

Syntax

```
set multiview id windowNumber channel arg
```

Parameters

id

Type: **STRING**

Name of previously created multiview. String names are case-sensitive.

arg

Type: **STRING**

Supply one of the following arguments.

argument	Description
up	Will cycle the encoder source in the speficied multiview window to next higher numbered encoder. Will cycle around to lowest encoder number when maximum value is reached.
down	Will cycle the encoder source in the speficied multiview window to next lower numbered encoder. Will cycle around to highest encoder number when minimum value is reached.

Examples

```
set multiview mv3x3 windowNumber 3 channel up
Channel changed to Z4Kenc_2
Success

set multiview mv3x3 windowNumber 3 channel up
Channel changed to Arts_Encoder_1
Success
```

Related Commands

```
create multiview
delete multiview
delete multiviewWindow
show multiviews config
show multiviews status
```

set multiview canvasSize

Selects the canvas size for creating multiview windows. (ZyPer4K family only)

Helpful feature to control bandwidth of scaled streams for a multiview. Default canvas size is 3840x2160. This can create case where datarate from encoder is greater than 9.5Gb limit.(Full size stream plus scaled stream.) Reducing the canvas size will reduce required size and datarate of scaled stream used for multiview.

Syntax

```
set multiview id canvasSize pixelsHoriz pixelVert
```

Parameters

id

Type: **STRING**

Name of previously created multiview. String names are case-sensitive.

pixelsHorz

Type: **Integer**

Horizontal width of the multiview canvas. (640 to 8192)

pixelsVert

Type: **Integer**

Vertical height of multiview window. (480 to 8192)

Note: Maximum canvas pixels is 8,847,360

Example

```
set multiview MyView1 canvasSize 1920 1080
Success
```

Related Commands

```
create multiview
delete multiview
delete multiviewWindow
show multiviews config
show multiviews status
```

set multiview newEncoderName

Assigns a new encoder to an existing multiview window. (ZyPer4K family only)

Syntax

```
set multiview id windowNumber wn newEncoderName encName|none
```

Parameters

id

Type: **STRING**

Name of previously created multiview. String names are case-sensitive.

wn

Type: **Integer**

Window number within existing multiview. (1 to 19)

encName

Type: **STRING / STRING**

The name or MAC address of the encoder. String names are case sensitive. None is also an option to remove existing encoder and replace with nothing.

Example

```
set multiview mv2x2-Art windowNumber 3 newEncoderName ABC
Success

set multiview mv2x2-Art windowNumber 3 newEncoderName none
Success
```

Related Commands

```
create multiview
delete multiview
delete multiviewWindow
show multiviews config
show multiviews status
```

set multiview title

Used to create a text overlay in a multiview window. (ZyPer4K family only)

Create a string of text to be overlayed somewhere in a multiview window. Color of text and color of background can be specified. Size of text can be specified. Transparency of text and background can be specified. Note that 100% transparent setting is not fully transparent.

Syntax

```
set multiview id windowNumber wn title textString title

set multiview id windowNumber wn title text-size ts

set multiview id windowNumber wn title transparency text tt
background bt

set multiview id windowNumber wn title color text tc background bc
```

Parameters

id

Type: **STRING**

Name of previously created multiview. String names are case-sensitive.

wn

Type: **Integer**

Window number within the multiview (1-19)

ts

Type: **Integer**

Size of text (1-10)

tt

Type: **Integer**

Text Transparency. Percentage (0-100)

bt

Type: **Integer**

Background Transparency. Percentage (0-100) *tc*

Type: **STRING**

Text color. Can be any of the following options: black, blue, brown, cyan, darkBlue, gray, green, lightBlue, lightGray, lime, magenta, maroon, olive, orange, purple, red, silver, white, yellow.

bc

Type: **STRING**

Background color. Can be any of the following options: black, blue, brown, cyan, darkBlue, gray, green, lightBlue, lightGray, lime, magenta, maroon, olive, orange, purple, red, silver, white, yellow.

title

Type: **STRING**

Any text string to be associted and displayed in the selected multiview window. Strings contains spaces must be enclosed in quotations.

Examples

```
set multiview MyView1 windowNumber 1 title textString “Window #1”
Success

set multiview MyView1 windowNumber 1 title textSize 10
Success

set multiview MyView1 windowNumber 1 title transparency text 0
background 100
Success

set multiview MyView1 windowNumber 1 title color black background-
color green
```

Related Commands

```
create multiview
delete multiview
delete multiviewWindow
show multiviews config
show multiviews status
```

set preset commands auto

Used to update an existing preset commands

Syntax

```
set preset id commands auto connections
```

Parameters

id

Type: **STRING**

The name of the preset. String names are case-sensitive.

connections

Supply one of the following arguments.

argument	Description
`existingConnections`	Uses the existing set of connections to create the command list
`empty`	Creates an empty set of commands. No connections

Example

```
set preset morning commands auto existingConnections
Success

set preset morning commands auto empty
Success
```

Related Commands

`create preset`
`delete preset`
`run preset`
`show preset`

set preset commands blob

Used to update an existing preset commands

Syntax

```
set preset id commands blob connections
```

Parameters

id

Type: **STRING**

The name of the preset. String names are case-sensitive.

connections

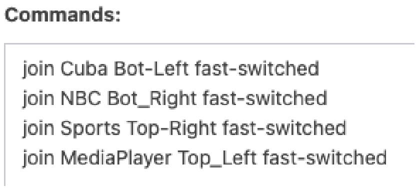
Type: **STRING**

Manually enter a list of commands contained within quotations. Insert a semi-colon between commands. Maximum character limit is 4096.

Example

```
set preset morning commands blob "join Cuba Bot-Left fast-
switched;join NBC Bot_Right fast-switched;join Sports Top-Right
fast-switched;join Media Player Top_Left fast-switched"
```

Below is image from ZMP GUI showing these commands in the Preset window:



Commands:

join Cuba Bot-Left fast-switched
join NBC Bot_Right fast-switched
join Sports Top-Right fast-switched
join MediaPlayer Top_Left fast-switched

Related Commands

create preset
delete preset
run preset
show preset

set preset description

Used to update an existing preset description

Syntax

```
set preset id description description
```

Parameters

id

Type: **STRING**

The name of the preset. String names are case-sensitive.

description

Type: **STRING**

Updated description of the preset

Example

```
set preset morning description “Open for business”
Success
```

Related Commands

```
create preset
delete preset
run preset
show preset
```

set preset schedule eventColor

Used to update an existing preset schedule color in the calendar.

Syntax

```
set preset id schedule scname eventColor color
```

Parameters

id

Type: **STRING**

The name of the preset. String names are case-sensitive.

scname

Type: **STRING**

Name of the schedule.

color

Type: **STRING**

Name of the new color. Options include the following: aqua, aquamarine, black, blue, brown, coral, cyan, darkBlue, darkSlateGray, deepPink, deepSkyBlue, fuchsia, gray, green, hotPink, khaki, lightBlue, lightGray, lightSeaGreen, lightSlateGray, lime, magenta, maroon, mistyRose, olive, orange, pink, purple, red, silver, teal, web-hex-color starting with # (e.g. #22ffee), white, yellow, zvGreen, zvPurple

Example

```
set preset morning schedule opentime zvGreen
Success
```

Related Commands

create preset
delete preset
run preset
show preset

set preset schedule month

Used to update an existing preset schedule month/day/time to run

Syntax

```
set preset id schedule scname month month dayOfMonth day dayOfWeek
day hour hour minute minute
```

Parameters

id

Type: **STRING**

The name of the preset. String names are case-sensitive.

scname

Type: **STRING**

Name of the schedule.

month

Type: **STRING**

Months to run this preset: Options are all, jan, feb, mar, apr, may, jun, jul, aug, oct, nov, dec

dayOfMonth

Type: **Integer**

Days of the month to run this preset. Enter an integer date or “all”

dayOfWeek

Type: **STRING**

Days of week to run this preset: Options are all, sunday, monday, tuesday, wednesday, thursday, friday, saturday, weekday, weekend. (Note: Weekday = M-F, Weekend = Sat+Sun)

hour

Type: **String**

Hour to run this preset. Enter an integer time (24 hour format) or “all”

```
-----------------------------------------------------------------------
```

minute

Type: **Integer**

Enter the minute (0-59) for this preset to run.

Example

```
set preset test1 schedule LateLunch month all dayOfMonth all
dayOfWeek weekday hour 14 minute 30
Success
```

Related Commands

```
create preset
delete preset
run preset
show preset
```

set responses rs232TermChars

Specifies the termination character for an RS232 string. The default string is "`\n\r`". Any character in the termination string causes the response-string to terminate and be placed into the response-string ring buffer.

This string is optional. If it is not specfied, then the string is empty and each low-level response is handled as a separate response.

Syntax

```
set responses id chr
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the decoder. String names are case-sensitive.

chr

Type: **STRING**

The specified string.

Example

```
set responses decoder2 rs232TermChars "\r"
Success
```

Related Commands

`set device rs232`

set role

Sets permisison levels for a specific role.   Note that the role must have been previously created.

Syntax

```
set role rolename rolename subsystem subinfo maxAccess accessLevel
```

Parameters

rolename

Type: **STRING**

String names are case-sensitive.

subinfo

Type: **STRING**

Supply one of the following arguments

argument	Description
`account`	Abiltiy of this role to modify accounts
`all`	Set all priority fields with a single setting
`device`	Ability of this role to modify devices
`log`	Abiltiy of this role to access logs
`multiview`	Ability of this role to modify/create multiviews
`netmap`	Abiltiy of this role to modify netmaps
`preset`	Ability of this role to modify/create presets
`role`	Ability of this role to modify/create other roles
`server`	Ability of this role to modify server settings
`snmpagent`	Ability of this role to modify/view snmp
`tls`	Ability of this role to modify/view tls settings
`videowall`	Ability of this role to modify/create videowalls
`zone`	Ability of this role to modify/create zones

```
-----------------------------------------------------------------
```

accessLevel

Type: **STRING**

Supply one of the following arguments

argument	Description
`admin`	Full unlimited access/control
`config`	Ability to configure
`join`	Ability to join (only applies to certain items such as multiviews)
`none`	No permissions
`view`	Ability to view only

Examples

```
set role rolename junior subsystem all maxAccess admin
Success

set role rolename junior subsystem account maxAccess config
Success

set role rolename junior subsystem role maxAccess view
Success

set role rolename junior subsystem videowall maxAccess none
Success
```

Related Commands

```
create role
delete role
show role

show role junior maxAccess
role(junior);
  role.account; maxAccess=admin
  role.device; maxAccess=admin
  role.log; maxAccess=admin
  role.multiview; maxAccess=admin
  role.netmap; maxAccess=admin
  role.preset; maxAccess=admin
  role.role; maxAccess=admin
  role.server; maxAccess=admin
  role.snmpagent; maxAccess=admin
  role.tls; maxAccess=admin
  role.videowall; maxAccess=admin
  role.zone; maxAccess=admin
lastChangeIdMax(37);
Success
```

Examples

```
set server redundancy allServers virtualIp address 192.168.0.25
networkInterface video
Success

set server redundancy thisServer preferredMaster true
preferredSlave false
Success

set server redundancy 192.168.1.202 preferredMaster false
preferredSlave true
Success
```

Related Commands

`create role`

set server api lineWrap

Sets the number of characters the API will display in the Command Line Interface before wrapping to a new line.

Syntax

```
set server api lineWrap wrap
```

Parameters

wrap

Type: **INTEGER**

Integer value from 100 to 512

Example

```
set server api lineWrap 200
Success
```

set server autoEdidMode

Sets the EDID mode for the Management Platform. By default, Auto-EDID mode is enabled.

Syntax

```
set server autoEdidMode mode
```

Parameters

mode

Type: **STRING**

Supply one of the following arguments.

argument	Description
`disabled`	Disables auto-EDID mode.
`enabled`	Enables auto-EDID mode.

Example

```
set server autoEdidMode disabled
Success
```

Related Commands

`set server timezone`

set server dataTunnelMode

Sets the transfer mode for the Management Platform.

Syntax

```
set server dataTunnelMode mode
```

Parameters

mode

Type: **STRING**

Supply one of the following arguments.

argument	Description
`raw`	Sets raw communication mode.
`telnet`	Sets telnet communication mode.

Notes

Telnet is a way of passing control information about the communication channel. It defines line-buffering, character echo, etc, and is done through a series of will/wont/do/dont messages when the connection starts.

Raw is a TCP stream with no telnet escape sequences.

Telnet is an application layer protocol while TCP is a transport layer protocol. Telnet uses TCP in order to transmit data. That is a big fundamental difference between Telnet and TCP.

Example

```
set server dataTunnelMode telnet
Success
```

Related Commands

`set server timezone`

set server date

Used to set server date manually or via ntp server. Note: NTP Server must be IPV4

Syntax

```
set server date mode

set server date ntpServer address <domainName>

set server date manual month <int> day <int> year <int> hour <int>
minute <int>
```

Parameters

mode

Type: **STRING**

Supply one of the following arguments.

argument	Description
`manual`	Sets date/time manually
`ntpServer`	Sets date/time via ntp server. Must provide valid IP address for an ntp Server.

Example

```
set server date manual month 4 day 1 year 2021 hour 15 minute 1
Success

set server date ntpServer address 129.6.15.28
Success
```

Link to NTP Servers:

https://tf.nist.gov/tf-cgi/servers.cgi

Related Commands

set server timezone
show server config
show server info

set server discoverMode

Sets how ZyPerUHD endpoints are discovered by the Management Server on the network

Syntax

```
set server discoverMode mode
```

Parameters

mode

Type: **STRING**

Supply one of the following arguments.

argument	Description
`broadcast`	Sets discovery mode to broadcast (Default)
`multicast`	Sets discovery mode to multicast

Notes

Allows the server to discover ZyPerUHD endpoints using multicast across subnets when multicast routing is enabled.   When in multicast mode there must be a igmp querier running – usually that would be the multicast router querier.

Example

```
set server discoverMode multicast
Success
```

Related Commands

`set server timezone`

set server encoderDefault audio

Sets the default encoder audio format for HDMI audio input.

Detailed Background

ZeeVee added a feature that will allow compressed formats to be passed down in an encoder EDID file. This EDID will be then forwarded to the source device to determine the type of audio sent to the encoder.

This enhancement was to provide fastSwitched connections the “compressed audio” options in the EDID file. Prior to this version with the fast-switched connection, ZeeVee modified the EDID passed from the decoder to the encoder and removed all compression formats. This left just LPCM as the only option under the “Audio data block” in the edid file.

```
>>> Audio data block <<<
  Linear PCM, max channels 8
    Supported sample rates (kHz): 192 176.4 96 88.2 48 44.1 32
    Supported sample sizes (bits): 24 20 16
```

The information provided to the Video Source device (such as BluRay Player or Media player) increases the possibility of compression being a chosen audio format.
However it is still up to the device to choose uncompressed or compressed formats.
It is important to know that some devices such as the Apple 4K TV requires the audio output type to be set (even if the audio format is available in the EDID). Compression will need to be set manually on these types of devices.

In addition any downmixed stream internal to ZyPer devices will not process compressed audio, so you will not hear compressed audio on these connections.

Syntax

```
set server encoderDefault edid audio mode
```

Parameters

mode

Type: **STRING**

Supply one of the following arguments.

argument	Description
`allowCompressed`	Passes the decoders edid with unmodified audio information and thus allows compression options to be seen.
`onlyPcm`	Forces the EDID modification described above

Example

```
set server encoderDefault edid audio allowCompressed
Success
```

Related Commands

`set encoder edid audio`

Additional Information

In an attempt to properly Identify the Audio Streams used under the product the following changes were also made along with some modification to the API commands.

Product	Old Audio Stream Name	New Stream Name
ZyPer4K	hdmi (used in genlocked mode)	hdmiPassthroughAudio
ZyPer4K	hdmi-audio-downmix	hdmiAudio
ZyPer4K	analog-audio	analogAudio
ZyPerUHD	audio	hdmiAudio
ZyPerUHD	analog-audio	analogAudio
ZyPerUHD60	none	hdmiAudio
ZyPerUHD60	none	analogAudio

set server ftp mode

Used to enable to disable FTP access to the Management Server.

Syntax

```
set server ftp mode arg
```

Parameters

arg

Type: **STRING**

Supply one of the following arguments.

argument	Description
`enabled`	Enable FTP access to the Management Server
`disabled`	Disable FTP access to the Management Server

Examples

```
set server ftp mode enabled
Success

set server ftp mode disabled
Success
```

Related Commands

`set server timezone`

set server ip

Sets the IP Address of the Management Platform. For MP hardware with multiple Network Interfaces this command is used to set the IP Address of each interface independently.

Syntax

```
set server ip id mode address IP Address mask Mask gateway Gateway
dns DNS-Server reboot
```

Parameters

id

Type: **STRING**

Supply one of the following arguments.

argument	Description
`videoPort`	Select the "Video" network. (ZyPer Network)
`managementPort`	Select the "Management" network. (Non-ZyPer Network)

mode

Type: **STRING**

Supply one of the following arguments.

argument	Description
`static`	Manually select/assign IP Address
`dhcp`	Allow DHCP server to automatically assign IP Address

Example

```
set server ip videoPo dhcp reboot
Success

set server ip videoPort static address 172.16.16.111 mask
255.255.255.0 gateway 172.16.6.1 dns none reboot
Success

set server ip managementPort static address 192.160.20.2 mask
255.255.255.0 gateway 192.168.20.1 dns none reboot
Success
```

Related Commands

`set server timezone`

set server isaac address

Sets the domain name of the isaac server.

Syntax

```
set server isaac address domainname
```

Parameters

domainname

Type: **STRING**

domainname of the Isaac server

Example

```
set server isaac address
Success
```

Related Commands

`set server isaac subsystemId`

set server isaac subsystemId

Sets the subsystemID on isaac server.

Syntax

```
set server isaac address subsystemId
```

Parameters

subsystemID

Type: **STRING**

Subsystem ID of the Isaac server

Example

```
set server isaac subsystemId Wallyworld
Success
```

Related Commands

set server isaac address

set server license

Sets the license for the Management Platform. This controls the maximum number of endpoints supported by the Management Platform.

Syntax

```
set server license key
```

Parameters

key

Type: **STRING**

License key obtained from ZeeVee that sets maximum number of endpoints

Example

```
set server license QDZV-AYYA-0048-303D-5C0E-BD5D-56AA-154D-976C-
BCE3-BAC4
Success
```

Related Commands

`set server autoEdidMode`

set server redundancy

Sets a virtual IP address and Mask for the Master and Slave Management Platforms in the system. (See Appendix for additional Redundancy Configuration Instructions)

Syntax

```
set server redundancy serv_id virtualIp address IP_Address
networkInterface video|management
```

Parameters

serv_id

Type: **STRING**

The servers to apply Virtual-ID to.

argument	Description
`allServers`	All Management Platforms on the Network. (Master and Slave)
`thisServer`	The specific server (Master or Slave) currently logged into.
`server IP Address`	Manually enter IP address of a specific Management Platform. (Master or Slave)

IP_Address and Mask

Type: **STRING**

Virtual IP address with Subnet Mask

argument	Description
`IP Address`	Virtual IP address to use for designated servers: Example: 192.168.0.25
`networkInterface`	Selects either the Video or Management interface for MP units with Dual Network Interfaces

Note: The virtual address has to be accessible within the subnet already defined for the interface. So, if the "video network", aka the original interface has 172.6.2.22/24, then the virtual address has to be 172.16.2.xxx.

Examples

```
set server redundancy allServers virtualIp address 192.168.0.25
networkInterface video
Success

set server redundancy thisServer preferredMaster true
preferredSlave false
Success

set server redundancy 192.168.1.202 preferredMaster false
preferredSlave true
Success
```

set server security deviceSecurityKey

Part of the mechanism to enable security over Semtech's server-device communication. First, there has to be an overall key associated with the server (deviceSecurityKey). Then, each device has to enable the security. It's authentication and encryption. This only works with ZyPer4K-XS and ZyPer4K-XR devices. Once a device has been enabled for a specific server, it will not work with any server without the same key. Although redundancy automatically sets the same key on both servers. If the key is lost, devices have to be hardware factory defaulted.

Syntax

```
set server security deviceSecurityKey key
```

Parameters

key

Type: **STRING**

Server security key. Text from 8 to 64 characters in length

Example

```
set server security deviceSecurityKey patriotsrule
Success
```

Notes

To change the server key; all devices (encoder and decoders) must have the security feature disabled first. Then change the key and re-enable the security feature on the devices.

Related Commands

`set server device security`

set server telnet password

Sets the password for Telnet. If a password is not provided, then the current password will be deleted. In this case, no password prompt will be displayed.

By default Telnet has no password.

Syntax

```
set server telnet pass
```

Parameters

pass

Type: **STRING**

The desired password.

Example

```
set server telnet password biGB055
Success
```

Notes

To reset system to no telnet password:
FTP the empty file named “defaultPasswords” to the /files directory of the MP (no file extension)
Power cycle the MP within **1 minute**, when it comes back the passwords will be defaulted.

This provides the very secure requirement of having physical access to the MP in order to reset the password.

Related Commands

```
set server autoEdidMode
set server telnet mode
set server timezone
```

set server telnet mode

Used to enable or disable telnet access to the server.

Syntax

```
set server telnet mode mode
```

Parameters

mode
Type: **STRING**

Supply one of the following arguments.

argument	Description
`enabled`	Telnet access is enabled
`disabled`	Telnet access is disabled

Example

```
set server telnet mode disabled
Success
```

Example trying to access via Telnet once disabled

```
telnet 192.168.0.78
Trying 192.168.0.78...
telnet: connect to address 192.168.0.78: Connection refused
telnet: Unable to connect to remote host
```

Related Commands

```
set server autoEdidMode
set server telnet password
set server timezone
```

set server timezone

Sets the time zone for the Management Platform. The time zone must be specified in POSIX format.

Syntax

```
set server timezone zone
```

Parameters

zone

Type: **STRING**

The time zone in POSIX format.

Example

```
set server timezone America/New_York
Success
```

Link to list of POSIX format timezones:

https://en.wikipedia.org/wiki/List_of_tz_database_time_zones

Related Commands

```
set server autoEdidMode
set server date ntpServer address <domainName>
set server date manual month <int> day <int> year <int> hour <int>
minute <int>
show server info
show server config
```

set terminal output

Set terminal output options between normal and JSON format.

The web interface has always been “JSON encoded responses” (computer friendly). The major benefit for this is for a web app to easily process the response. The downside is that it’s not at all “human friendly”.

There are two output format options from the API. One over telnet and ssh that is human friendly, and one over http that’s computer friendly.

This new command allows users to select the format of responses from the API.

Syntax

```
set terminal output normal|json echo yes|no prompt yes|no
```

Parameters

normal | json - allows user to select between these two options

echo - allows characters/commands to be seen while typing in telnet/ssh session

prompt - provides “ZyPer$” prompt as que for entering commands in telnet/ssh session

Examples

```
set terminal output normal echo yes prompt yes
Success

set terminal output json echo yes prompt yes
[   114]{“status”:”Success”,”text”:[],”errors”:[],”warnings”:[],”co
mmand”:”set terminal output json echo yes prompt yes “}

set terminal output json echo no prompt yes
[   112]{“status”:”Success”,”text”:[],”errors”:[],”warnings”:[],”co
mmand”:”set terminal output json echo no prompt yes”}
```

Warning

Removing echo feature from a normal Telnet or SSH session can be challenging as the user would no longer be able to see the commands being typed into the Telnet or SSH window. Copy/paste the first example above to return to normal operation.

set tls server mode

Used to enable web server TLS mode.

Syntax

```
set tls server mode mode
```

Parameters

mode
Type: **STRING**

Supply one of the following arguments.

argument	Description
`enabled`	Telnet access is enabled
`disabled`	Telnet access is disabled

Example

```
set tls server mode enabled
Success

set tls server mode disabled
Success
```

Related Commands

```
show tls summary
show tls pem ca signedCert
```

set tls server fqdn

Used to set server Fully Qualified Domain Name.   Either manually entered by user or contained in Certification file.

Syntax

```
set tls server fqdn domain | fromCert
```

Parameters

domain

Type: **STRING**

Full domain name

Examples

```
set tls server fqdn www.zeevee.com
Success

set tls server fqdn fromCert
Success
```

Related Commands

```
show tls summary
show tls pem ca signedCert
```

set videoWall size

Changes the size of the specified video wall and bezel parameters. Bezel values are measured in pixels.

Setting bezel values will affect a resolution change to the display. If the resolution is not supported by the display, then the display will have no picture. If this is the case, try assigning a different bezel pixel value.

Syntax

```
set videoWall id size rows rows columns cols topBezel bezt
bottomBezel bezb leftBezel bezl rightBezel bezr
```

Parameters

id

Type: **STRING**

The name of the video wall. String names are case-sensitive.

rows

Type: **INTEGER**

The number of rows. (Maximum 15 for ZyPer4K, Maximum 15 for ZyPerUHD, Maximum 4 for ZyPerHD)

cols

Type: **INTEGER**

The number of columns. (Max 15 for ZyPer4K, Max 15 for ZyPerUHD, Max 4 for ZyPerHD)

bezt

Type: **INTEGER**

The top bezel pixel value.

bezb

Type: **INTEGER**

The bottom bezel pixel value.

bezl

Type: **INTEGER**

The left bezel pixel value.

bezr

Type: **INTEGER**

The right bezel pixel value.

Note: Bezel adjustment only supported on ZyPer4K family

Example

```
set videoWall Mywall1 size rows 5 columns 5 topBezel 0 bottomBezel
0 leftBezel 0 rightBezel 0
Success
```

Related Commands

```
create videoWall
set videoWall decoder
show videoWalls
join videoWall
set videoWall newName
```

set videoWall decoder

Assigns the specified decoder, to the desired row and column, on the specified video wall.

Syntax

```
set videoWall wallid decoder id row col
```

Parameters

id

Type: **STRING or MAC Address**

The name or MAC address of the decoder. String names are case-sensitive. If `none` is passed as the argument, then any existing display is disconnected from that position in the video wall.

wallid

Type: **STRING**

The name of the video wall. String names are case-sensitive.

row

Type: **INTEGER**

The row of the specified video wall.

col

Type: **INTEGER**

The column of the specified video wall.

Example

```
set videoWall myVideoWall decoder myDecoder row 2 column 3
Success
```

Related Commands

`create videoWall`
`set videoWall size`
`show videoWalls`
`join videoWall`

set videoWall newName

Changes the name of an existing video wall

Syntax

```
set videoWall id newName name
```

Parameters

id

Type: **STRING or MAC Address**

The name of the video wall. String names are case-sensitive.

name

Type: **STRING**

The updated name of the video wall.

Example

```
set videoWall myWall2 newName yourWall2
Success
```

Related Commands

`create videoWall`
`set videoWall size`
`show videoWalls`
`join videoWall`

show account

Displays information about accounts

Syntax

```
show account select [since]
```

Parameters

select

Type: **STRING**

Supply one of the following arguments.

argument	Description
`active users`	Used to show information about currently active users logged into the system
`allConfig`	Shows account settings that apply to all accounts
`list`	Shows information about accounts including security settings and status
`login banner filenames`	Shows currently used banner image filenames
`login banner text webPreLogin`	Shows currently used login banner Pre login text
`login banner text webPostLogin`	Shows currently used login banner Post login text.

since

This parameter is optional and can be specified to display units based on the number of changes, using the `lastChangeId` value on each device. However, if used, a lastChangeId value must follow. Supply the `since` argument before the providing the `lastChangeId` value.

argument	Description
`since`	Required when using this parameter.

Examples

```
show account active users all
session(1);
  session.status; user=admin, type=Telnet, extId=none,
start=12/21/22T13:55:10-0500, lastActive=12/21/22T14:07:04-0500
session(2);
  session.status; user=admin, type=Web, extId=Qf,lebMNNAn,
start=12/21/22T14:00:10-0500, lastActive=12/21/22T14:04:30-0500
lastChangeIdMax(19478);
Success

show account allConfig
allAccounts(192.168.0.22);
  allAccounts.gen; idleLogoutMins=unlimited, concurrentSesionsMax=5
  allAccounts.password; complexity=disabled, minLen=NA,
initialExpire=disabled, minDays=0, maxDays=unlimited
  allAccounts.onThreeFailures; lockoutMins=none, disable=false
  allAccounts.authMode; telnetFullAuth=oldAuth, webFullAuth=noAuth
lastChangeIdMax(9);
Success

show account list all
account(admin);
  account.gen; role=admin, lastLogin=12/21/22T14:00:17-0500,
twoFactor=disabled
  account.status; locked=disabled, passwordExpires=never
account(zyper);
  account.gen; role=admin, lastLogin=none, twoFactor=disabled
  account.status; locked=disabled, passwordExpires=never
account(sftp);
  account.gen; role=none, lastLogin=none, twoFactor=disabled
  account.status; locked=disabled, passwordExpires=never
lastChangeIdMax(9);
Success

show account login banner filenames
allAccounts(192.168.0.22);
  allAccounts.webBanners; preLoginText=none,
postLoginText=securePre.txt, preLoginImage=none,
postLoginImage=mickey.png
  allAccounts.terminalBanners; preLoginText=securePre.txt,
postLoginText=securePost.txt
lastChangeIdMax(9);
Success

show account login banner text webPostLogin
allAccounts(192.168.0.22);
  allAccounts.bannerText; webPostLogin="You are about enter a
secure site.\nIf you do not have authorization, do not proceed."
lastChangeIdMax(17);
Success
```

show dataTunnels

Shows what rs232 or IR data relay ports are opened on the server.

The feature of data-relays was added to allow a third party to connect to the ZMP server with a specific port and pass raw or telnet API commands (depending on the mode) to the server and port which is designated for a particular encoder or decoder.

Syntax

```
show dataTunnels
```

Parameters

none

Example

```
show dataTunnels
dataSessions(d8:80:39:9a:96:7);
  device: name=Cuba
  irTunnel: port=1234
  irTunnel-connections: none
Success
```

Related Commands

dataConnect
set server dataTunnelMode

show device capabilities

Displays device capabilities for the specified device(s).

Syntax

```
show device capabilities id select [since]
```

Parameters

`id`

Type: **STRING or MAC Address**

The identifier of the device. Either the full or portion of a string name or MAC address can be supplied.

`select`

Type: **STRING**

Supply one of the following arguments.

argument	Description
`all`	Displays configuration information for all available devices.
`encoders`	Only encoders are displayed.
`decoders`	Only decoders are displayed.

`since`

This parameter is optional and can be specified to display units based on the number of changes, using the `lastChangeId` value on each device. However, if used, a lastChangeId value must follow. Supply the `since` argument before the providing the `lastChangeId` value.

argument	Description
`since`	Required when using this parameter.

Example

```
show device capabilities MyEnc1 since 20
device(d8:80:39:eb:1:cb);
  device.gen; lastChangeId=28
  device.CapabilitiesVersion; values=1
  device.analogAudioPort; values=none:hdmiAudioDownmix
  device.colorDepth; values=fastSwitchDeepColor:multiviewDeepColor
  device.colorEncoding; values=fastSwitchSubsample:multiviewSubsam
ple
  device.edid; values=save:load
  device.edidAudioFormat; values=onlyPcm:allowCompressed:serverDef
ault
  device.ethernetManagementPortMode; values=enabled:disabled
  device.factoryDefaults; values=supported
  device.firmwareUpdate; values=...apz
  device.flashLeds; values=supported
  device.hdcpMode; values=enabled:enabled1.4:disabled
  device.hdmiStatus; values=link:hdcp:resolution:fps
  device.ipMode; values=dhcp:static
  device.ipStaticGateway; values=supported
  device.ir; values=device:server:none
  device.joinAudio; values=analogAudio:hdmiAudio
  device.joinUsb; values=false
  device.joinVideo; values=fastSwitched:genlocked:genlockedScaled:m
ultiview:window
  device.multiview; values=title
  device.previewStream; values=enabled:disabled
  device.rs232; values=device:server:none
  device.sendMulticasts; values=settable
  device.streamMcastSettable; values=video:analogAudio:hdmiAudio
  device.streamModeSettable; values=video:hdmiAudio:videoScaled:ana
logAudio
  device.streamsSupported; values=video:hdmiAudio:videoScaled:analo
gAudio
  device.temperature; values=main
  device.usbFilter; values=none
  device.videoPort; values=hdmi:auto
  device.videoWall; values=maxSize(15):bezelsSupported
lastChangeIdMax(29);
Success
```

Related Commands

show device status
show device config

show device config

Displays device information for the specified device(s).

Syntax

```
show device config id [since]
```

Parameters

id

Type: **STRING or MAC Address**

The identifier of the device. Either the full or portion of a string name or MAC address can be supplied. Can also enter in one of the arguments below.show

argument	Description
`all`	Displays configuration information for all available devices.
`commands`	Shows all the commands used to configure every device, multiview, video wall in the system. (Can be a lot of output)
`encoders`	Only encoders are displayed.
`decoders`	Only decoders are displayed.

since

This parameter is optional and can be specified to display units based on the number of changes, using the `lastChangeId` value on each device. However, if used, a lastChangeId value must follow. Supply the `since` argument before the providing the `lastChangeId` value.

argument	Description
`since`	Required when using this parameter.

Example

```
show device config XSdec
device(0:16:c0:4d:e3:12);
  device.gen; model=Zyper4KXS, type=decoder, virtualType=none,
name=XSdec, state=Up, lastChangeId=173
  device.gen; productCode=Z4KDECFXS, productDescription=Fiber
Decoder - HDMI 2.0, pid=0xd
  device.gen; controlAuthenticationMode=disabled
  device.gen; firmware=1.3.2.4
  device.gen; ethernetManagementPortMode=disabled
  device.optionalPorts; video=none, usb=hid, analogAudio=yes,
rs232=no, ir=no
  device.hdmi; hdcpMode=auto, 5vControl=disabled
  device.ports; videoPort=auto
  device.ip; mode=dhcp, address=169.254.19.227, mask=255.255.0.0,
gateway=NA
  device.display; iconImageName=GenericDisplay, manufacturer=none,
model=none, location=none, serialNumber=none
  device.edid; preferMode=strict
  device.display; mode=stretch
  device.displayResolution; allParameters=auto
  device.displayTiming; allParameters=auto
  device.connectedEncoder; macAddr=0:16:c0:4d:e3:67, name=XSenc_1,
connectionMode=fastSwitched
  device.audioConnections; analogSourceMac=none,
analogSourceName=none, hdmiAudioSourceMac=0:16:c0:4d:e3:67,
hdmiAudioSourceName=XSenc_1
  device.autoAudioConnections; hdmiAudioFollowVideo=false
  device.audioOutSourceType; analogOutSourceType=hdmiAudioDownmix,
hdmiOutSourceType=hdmiAudio
  device.usb; filter=none, internalIpAddress=none
  device.usbUplink; macAddr=none, name=none
lastChangeIdMax(176);
lastDeleteIdMax(3);
Success
```

Related Commands

```
show device status
show device capabilities
show device connections
```

show device connections

Shows encoder connections to decoders

Syntax

```
show device connections
```

Parameters

none

Example

```
show device connections
encoder.GalapogosHD; BotLeftHD
encoder.RaptorsHD; SamsungHD
encoder.MuralsHD; BotRightHD
encoder.Soccer4K; TopRight, BotLeft
Success
```

Related Commands

```
show device status
show device capabilities
show device config
```

show device status

Displays status information for the specified device(s). This command functions the same as the `show device config` command.

Syntax

```
show device status id [since]
```

Parameters

id

Type: **STRING or MAC Address**

The identifier of the device. Either the full or portion of a string name or MAC address can be supplied.

Supply one of the following arguments.

argument	Description
`all`	Displays configuration information for all available devices.
`encoders`	Only encoders are displayed.
`decoders`	Only decoders are displayed.

since

This parameter is optional and can be specified to display units based on the number of changes. Supply this argument followed by the desired value to query.

argument	Description
`since`	Required when using this parameter.

Example

```
show device status Cuba
device(d8:80:39:9a:96:7);
  device.gen; model=Zyper4K, type=encoder, name=Cuba, state=Up,
uptime=4d:1h:57m:24s, lastChangeId=78
  device.temperature; main=59C
  device.firmwareUpdate; status=idle, loadingFile=none, percentComplete=0
  device.hdmiInput; cableConnected=connected, hdcp=inactive,
hdcpVersion=none, hdmi2.0=yes, horizontalSize=1280, verticalSize=720,
fps=60.000, interlaced=no
  device.hdmiInput; hTot=1650, hBlank=370, hFront=110, hSync=40,
hSyncPol=positive
  device.hdmiInput; vTot=750, vBlank=30, vFront=5, vSync=5, vSyncPol=positive
  device.hdmiInput; pixelClock=74.250, colorEncoding=YCBCR_444, colorDepth=8,
colorSpace=BT709, colorQuantRange=limited, timingStandard=CEA-861-F VIC-4
  device.edid; sourceType=file, sourceFilename=George.edid
  device.edid; edidStatus=valid, edidMonitorName=SyncMaster
  device.edid; firstDescriptorPreferredResolution=yes
  device.edid; maxFps=75.00, maxPixelClockMhz=170.00,
maxDeepColorPixelClockMhz=0.00, rgbColorDepth=8, yuv420ColorDepth=0
  device.edid; only420=none, also420=none, yuvQuantRange=default,
rgbQuantRange=default
  device.edid.audio.PCM; channels=2, sampleRates=48Khz-44.1Khz-32Khz,
sampleBits=16-20
  device.edid.preferredResolution; pixelClockMhz=148.50, sizeX=1920,
sizeY=1080, fps=60.00
  device.edid.maxResolution; pixelClockMhz=148.50, sizeX=1920, sizeY=1080,
fps=60.00
  device.videoStream; inputFps=60.00, inputDatarate=1451Mbps,
compressionFactor= 1.00, streamFps=60.00, streamDatarate=1451Mbps
  device.videoScaledStream; inputFps=60.00, inputDatarate=1451Mbps,
streamFps=30.00, streamDatarate=0Mbps
  device.previewStream; status=down, recvData=false
lastChangeIdMax(78);
Success
```

Related Commands

show device config

show device userAdded

Shows devices that have been manually added to the Management Platform using the add device command.

Syntax

```
show device userAdded
```

Parameters

none

Example

```
show device userAdded
device(d8:80:39:eb:1c:ee);
device.gen; model=Zyper4K, type=encoder, name=London, state=Up,
uptime=0d:18h:32m:36s, lastChangeId=55
device.ip; address=192.168.10.79
device(d8:80:39:59:f1:ff);
device.gen; model=Zyper4K, type=decoder, name=Right, state=Up,
uptime=0d:18h:32m:36s, lastChangeId=52
device.ip; address=192.168.10.81
device(d8:80:39:59:af:be);
device.gen; model=Zyper4K, type=decoder, name=Left, state=Up,
uptime=0d:18h:30m:5s, lastChangeId=56
device.ip; address=192.168.10.82
Success
```

Related Commands

```
add device
show device status
show device capabilities
show device config
```

show files

Shows files currently stored on the Managment Server. (EDID, Firmware, Icons and Idle Images)

Syntax

```
show files type
```

Parameters

type

Type: **STRING**

Supply one of the following arguments.

argument	Description
`all`	Show all files. (EDID, Firmware, Icons and Idle Images)
`edid`	Show EDID files.
`firmware`	Show Firmware files.
`icon`	Show Icon files.
`idleImage`	Show Idle Image files (ZyPerUHD and ZyPerUHD60 use)

Examples

```
show files icon
server(192.168.0.22);
  files.encoderIcon; names=SatelliteReceiver.png:BluRay.png:ps3.
png:SecurityCamera.png:FlatPanelDisplay.png:Laptop.png:cbs.
png:BroadcastCamera.png:fox.png:abc.jpg:DVD.png:xbox.png:DesktopPC.
png:MediaPlayer.png:tennis.png:espn.png:VCR.png:cnn.jpg:nbc.
png:foxSports.png:CableBox.png:golf.png:nflNetwork.jpg
  files.decoderIcon; names=FlatPanelDisplay.png:Projector.png:vw.png
  files.savedIcon; names=none
lastChangeIdMax(1);
Success

show files idleImage
server(192.168.0.22);
  files.idleImage; names=001Rupdated.jpg:IPD5000-B70_idle_image_
v1.0.jpg:SLupdated.jpg:test720.jpg
lastChangeIdMax(1);
Success
```

show logs authentications

Shows a listing of server login/logout events ordered from newest to oldest.

Syntax

```
show logs authentications max quantity
```

Parameters

quantity

Type: **INTEGER**

Number of past authentications to display

Example

```
show logs authentications max 5
log(192.168.0.22);
  log.msg.1; dt=Dec-16-22-12:47:02, user=system, sid=0, msg="EVENT
for server; Login -- account=admin, sessionId=1"
  log.msg.2; dt=Dec-16-22-12:47:00, user=system, sid=0,
msg="EVENT for server; Logout -- account=admin, sessionId=1,
reason=remoteClose"
  log.msg.3; dt=Dec-16-22-12:46:43, user=system, sid=0, msg="EVENT
for server; Login -- account=admin, sessionId=1"
  log.msg.4; dt=Dec-16-22-12:46:40, user=system, sid=0,
msg="EVENT for server; Logout -- account=admin, sessionId=1,
reason=remoteClose"
  log.msg.5; dt=Dec-16-22-12:46:40, user=system, sid=0, msg="EVENT
for server; Logout -- account=admin, sessionId=1, reason=User"
Success
```

Related Commands

`set server autoEdidModehow logs commands`

show logs commands

Shows a listing of last commands send to the Management Server.

Syntax

```
show logs commands max quantity
```

Parameters

quantity

Type: **INTEGER**

Number of past commands to display

Example

```
show logs commands max 5
log(192.168.0.22);
  log.msg.1; dt=Dec-16-22-12:43:38, user=admin, sid=1,
msg="CommandLine: show logs commands max 5"
  log.msg.2; dt=Dec-16-22-12:43:36, user=admin, sid=2,
msg="Error:(29) Device Z4KDante does not support or cannot change:
joinUsb with value true."
  log.msg.3; dt=Dec-16-22-12:43:36, user=admin, sid=2,
msg="CommandLine: join Enc1 Z4KDante usb"
  log.msg.4; dt=Dec-16-22-12:43:36, user=admin, sid=2,
msg="Error:(29) Device Z4KDante does not support or cannot change:
joinUsb with value true."
  log.msg.5; dt=Dec-16-22-12:43:36, user=admin, sid=2,
msg="CommandLine: join none Z4KDante usb"
Success
```

Related Commands

`set server autoEdidModehow logs authentications`

show multiviews config

Shows configuration information on all multiview displays. (ZyPer4K family only)

Syntax

```
show multiviews config
```

Parameters

none

Example

```
show multiviews config
multiview(Ltest1);
  multiview.audio; sourceWindow=none;
  multiview.window1; encoder-name=Airshow4K, percentPosX=40,
percentPosY=5, percentSizeX=55, percentSizeY=55, layer=1;
  multiview.window2; encoder-name=Soccer4K, percentPosX=5,
percentPosY=5, percentSizeX=30, percentSizeY=30, layer=1;
  multiview.window3; encoder-name=Wildlife4K, percentPosX=5,
percentPosY=65, percentSizeX=30, percentSizeY=30, layer=1;
  multiview.window4; encoder-name=Soccer4K, percentPosX=65,
percentPosY=65, percentSizeX=30, percentSizeY=30, layer=1;
  multiview.window5; encoder-name=USA4K, percentPosX=5,
percentPosY=35, percentSizeX=30, percentSizeY=30, layer=1;
  multiview.window6; encoder-name=USA4K, percentPosX=35,
percentPosY=65, percentSizeX=30, percentSizeY=30, layer=1;
multiview(MView4k);
  multiview.audio; sourceWindow=1;
  multiview.window1; encoder-name=Airshow4K, percentPosX=0,
percentPosY=0, percentSizeX=50, percentSizeY=50, layer=1;
  multiview.window2; encoder-name=USA4K, percentPosX=0,
percentPosY=50, percentSizeX=50, percentSizeY=50, layer=1;
  multiview.window3; encoder-name=Soccer4K, percentPosX=50,
percentPosY=0, percentSizeX=50, percentSizeY=50, layer=1;
  multiview.window4; encoder-name=Wildlife4K, percentPosX=50,
percentPosY=50, percentSizeX=50, percentSizeY=50, layer=1;
multiview(LBar);
  multiview.audio; sourceWindow=none;
  multiview.window1; encoder-name=Soccer4K, percentPosX=5,
percentPosY=5, percentSizeX=30, percentSizeY=30, layer=1;
  multiview.window2; encoder-name=Wildlife4K, percentPosX=5,
percentPosY=65, percentSizeX=30, percentSizeY=30, layer=1;
  multiview.window3; encoder-name=USA4K, percentPosX=35,
percentPosY=65, percentSizeX=30, percentSizeY=30, layer=1;
  multiview.window4; encoder-name=Soccer4K, percentPosX=65,
percentPosY=65, percentSizeX=30, percentSizeY=30, layer=1;
  multiview.window5; encoder-name=USA4K, percentPosX=5,
percentPosY=35, percentSizeX=30, percentSizeY=30, layer=1;
  multiview.window6; encoder-name=Airshow4K, percentPosX=35,
percentPosY=5, percentSizeX=60, percentSizeY=60, layer=1;
Success
```

Related Commands

```
create multiview
delete multiview
delete multiviewWindow
show multiviews status
```

show multiviews status

Shows status information for all multiview displays. (ZyPer4K family only)

Syntax

```
show multiviews status
```

Parameters

none

Example

```
show multiviews status
multiview(mv1);
  multiview.gen; totalDatarate=0Mbps
  multiview.window1; encoderName=MediaPlayer,
encoderMac=d8:80:39:eb:1:cb, streamType=none, datarate=0Mbps,
multicast=0.0.0.0, titleStatus=none, status=inactive, reason=no
decoder joined
  multiview.window2; encoderName=Curved,
encoderMac=d8:80:39:9a:e6:d, streamType=none, datarate=0Mbps,
multicast=0.0.0.0, titleStatus=none, status=inactive, reason=no
decoder joined
  multiview.window3; encoderName=Cuba,
encoderMac=d8:80:39:9a:96:7, streamType=none, datarate=0Mbps,
multicast=0.0.0.0, titleStatus=none, status=inactive, reason=no
decoder joined
  multiview.window4; encoderName=Camera2,
encoderMac=d8:80:39:9a:af:a3, streamType=none, datarate=0Mbps,
multicast=111.117.114.99, titleStatus=none, status=inactive,
reason=no decoder joined
Success
```

Related Commands

`create multiview`
`delete multiview`
`delete multiviewWindow`
`show multiviews config`

show multiviews titles

Shows title information for all multiview displays. (ZyPer4K family only)

Syntax

```
show multiviews titles arg
```

Parameters

arg

Type: **STRING**

Supply one of the following arguments.

argument	Description
config	Displays title configuration information for multiview.
text	Displays text configuration information for multiview.

Examples

```
show multiviews titles text
multiview(mv1);
  multiview.gen; audioSourceWindow=none, canvasWidth=3840,
canvasHeight=2160
  multiview.window1; title=Window1
  multiview.window2; title=Window2
  multiview.window3; title=none
  multiview.window4; title=none
Success

show multiviews titles config
multiview(mv1);
  multiview.gen; audioSourceWindow=none, canvasWidth=3840,
canvasHeight=2160
  multiview.window1; position=bottomCenter, textSize=8,
textColor=lightGray, backgroundColor=black, textTransparency=0,
backgroundTransparency=80
  multiview.window2; position=bottomCenter, textSize=8,
textColor=lightGray, backgroundColor=black, textTransparency=0,
backgroundTransparency=80
  multiview.window3; position=bottomCenter, textSize=8,
textColor=lightGray, backgroundColor=black, textTransparency=0,
backgroundTransparency=80
  multiview.window4; position=bottomCenter, textSize=8,
textColor=lightGray, backgroundColor=black, textTransparency=0,
backgroundTransparency=80
Success
```

show preset

Shows information about a preset

Syntax

```
show preset name arg since
```

Parameters

name

Type: **STRING**

The name of the preset

arg

Type: **STRING**

Supply one of the following arguments.

argument	Description
`commandBlob`	Displays command list in “blob” format. Commands separated by semi-colons.
`commands`	Shows current list of preset commands
`config`	Shows preset description
`runLog`	Shows information about last time preset was run
`schedule`	Shows schedule details for the preset
`status`	Displays text configuration information for multiview.

Examples

```
show preset test1 schedule all
preset(test1);
  preset.schedule.today; mode=enabled, color=#652d90, month=all,
dayOfMonth=all, dayOfWeek=weekday, hour=14, minute=30
lastChangeIdMax(92);
Success

show preset test1 runLog since 0
preset(test1);
lastChangeIdMax(92);
Success
```

Examples

```
show preset test1 config since 0
preset(test1);
  preset.gen; description=Playing with preset
lastDeleteIdMax(3);
lastChangeIdMax(90);
Success

show preset test1 commands since 0
preset(test1);
  preset.line1; cmd=join MediaPlayer Bot_Right fastSwitched
  preset.line2; cmd=join none Bot_Right analogAudio
  preset.line3; cmd=join videoSource Bot_Right hdmiAudio
  preset.line4; cmd=set decoder Bot_Right hdmiAudioOut source
hdmiAudio
  preset.line5; cmd=join mv2x2-Art Top-Right multiview
  preset.line6; cmd=join none Top-Right analogAudio
  preset.line7; cmd=join videoSource Top-Right hdmiAudio
  preset.line8; cmd=join mv3x3-Art Top_Left multiview
  preset.line9; cmd=join none Top_Left analogAudio
  preset.line10; cmd=join videoSource Top_Left hdmiAudio
  preset.line11; cmd=join mv4x4-Art Bot-Left multiview
  preset.line12; cmd=join none Bot-Left analogAudio
  preset.line13; cmd=join videoSource Bot-Left hdmiAudio
lastChangeIdMax(94);
Success

show preset test1 commandBlob since 0
preset(test1);
  preset.cmdBlob; cmdBlob=join MediaPlayer Bot_Right
fastSwitched;join none Bot_Right analogAudio;join videoSource
Bot_Right hdmiAudio;set decoder Bot_Right hdmiAudioOut source
hdmiAudio;join mv2x2-Art Top-Right multiview;join none Top-Right
analogAudio;join videoSource Top-Right hdmiAudio;join mv3x3-Art
Top_Left multiview;join none Top_Left analogAudio;join videoSource
Top_Left hdmiAudio;join mv4x4-Art Bot-Left multiview;join none Bot-
Left analogAudio;join videoSource Bot-Left hdmiAudio
lastChangeIdMax(94);
Success
```

Related Commands

create preset
delete preset
run preset
set preset

show responses

Displays response strings from the specified device.

Syntax

```
show responses id type param3
```

Parameters

`id`

Type: **STRING or MAC Address**

The name or MAC address of the device. String names are case-sensitive.

`type`

Type: **STRING**

Supply one of the following arguments.

argument	Description
`ir`	Displays IR response strings.
`rs232`	Displays RS232 response strings.

`param3`

Supply one of the following arguments.

argument	Description
`last`	Displays the last received response, based on the argument supplied for the *`type`* parameter.
`lastChangeId`	Displays the **`lastChangeId`** of the most recently received response.
`since`	Displays only new response data. Follow this argument with desired value to query.

Example

```
show responses 0:1e:c0:f6:b0:8a rs232 since 10
lastChangeId(0);
Success

show responses 0:1e:c0:f6:b0:8a ir lastChangeId
lastChangeId(0);
Success

show responses 0:1e:c0:f6:b0:8a ir last
lastChangeId(0);
Success

show responses UHDdec1 rs232 last
device(34:1b:22:80:64:68);
  device.rs232Response.19; string=”Yes ZeeVee Support is the 
Greatest\x0D”
lastChangeId(20);
Success

Zyper$ show responses UHDdec1 rs232 since 19
device(34:1b:22:80:64:68);
  device.rs232Response.19; string=”Yes ZeeVee Support is the 
Greatest\x0D”
  device.rs232Response.20; string=”Really, still the greatest!\x0D”
lastChangeId(21);
Success
```

show role

Shows information about a specific role or all roles.

Syntax

```
show role rolename|all maxAccess[since]
```

Parameters

role

Type: **STRING**

The name of the role

since

This parameter is optional and can be specified to display units based on the number of changes, using the `lastChangeId` value on each device. However, if used, a lastChangeId value must follow. Supply the `since` argument before the providing the `lastChangeId` value.

argument	Description
`since`	Required when using this parameter.

Examples

```
show role admin maxAccess since 0
role(admin);
  role.account; maxAccess=admin
  role.device; maxAccess=admin
  role.log; maxAccess=admin
  role.multiview; maxAccess=admin
  role.netmap; maxAccess=admin
  role.preset; maxAccess=admin
  role.role; maxAccess=admin
  role.server; maxAccess=admin
  role.snmpagent; maxAccess=admin
  role.tls; maxAccess=admin
  role.videowall; maxAccess=admin
  role.zone; maxAccess=admin
lastChangeIdMax(12);
Success
```

```
--------------------------------------------------------------------

show role all maxAccess
role(admin);
  role.account; maxAccess=admin
  role.device; maxAccess=admin
  role.log; maxAccess=admin
  role.multiview; maxAccess=admin
  role.netmap; maxAccess=admin
  role.preset; maxAccess=admin
  role.role; maxAccess=admin
  role.server; maxAccess=admin
  role.snmpagent; maxAccess=admin
  role.tls; maxAccess=admin
  role.videowall; maxAccess=admin
  role.zone; maxAccess=admin
role(junior);
  role.account; maxAccess=admin
  role.device; maxAccess=admin
  role.log; maxAccess=admin
  role.multiview; maxAccess=admin
  role.netmap; maxAccess=admin
  role.preset; maxAccess=admin
  role.role; maxAccess=admin
  role.server; maxAccess=admin
  role.snmpagent; maxAccess=admin
  role.tls; maxAccess=admin
  role.videowall; maxAccess=admin
  role.zone; maxAccess=admin
lastChangeIdMax(12);
Success
```

Related Commands

```
create role
delete role
set role rolename
set account username role
```

show server config

Displays configuration information for the Management Platform.

Syntax

```
show server config [since]
```

Parameters

since

This parameter is optional and can be specified to display units based on the number of changes, using the `lastChangeId` value on each device. However, if used, a lastChangeId value must follow. Supply the `since` argument before the providing the `lastChangeId` value.

argument	Description
`since`	Required when using this parameter.

Example

```
show server config
server(192.168.0.22);
  server.gen; autoEdidMode=enabled, redundancy=enabled
  server.ipServerAddress; mode=static, address=192.168.0.22,
mask=255.255.255.0, gateway=none, dns=none
  server.ipManagementAddress; mode=none, address=NA
  server.ntpServer; address=ntp.ubuntu.com
  server.telnetAccess; mode=enabled
  server.encoderDefault.edid; audio=onlyPcm
  server.dataTunnelMode; telnet=telnetHandshakeMode
  server.logging; level=1
  server.isaac; address=none, subsystemId=none
Success
```

Related Commands

`show server info`

show server info

Displays information for the Management Platform, including IP settings, uptime, and license level.

Syntax

```
show server info [since]
```

Parameters

since

This parameter is optional and can be specified to display units based on the number of changes, using the `lastChangeId` value on each device. However, if used, a lastChangeId value must follow. Supply the `since` argument before the providing the `lastChangeId` value.

argument	Description
`since`	Required when using this parameter.

Example

```
show server info
server(192.168.0.22);
  server.gen; hostname=zyper.local, version=2.3.36999,
 previousVersion=2.2.36870,
 macAddress=94:c6:91:a0:47:fc, serialNumber=ZZM1K400011D
  server.gen; uptime=3d:21h:25m:24s, freeMem= 6.71GB, bootCount=173
  server.gen; runningInVm=false
  server.ipActive; ipServerAddr=192.168.0.22, ipManagementAddr=NA,
 gatewayAddr=none, dnsAddr=none
  server.time; time="Tue Aug 31 08:43:59 2021",
 timezone=America/New_York
  server.pollStats; count=0, interval: 0-minutes, monListSize=0
  server.license; productID=F9188182-AF72-C6C8-92C6-94C691A047FC,
 license=none
  server.license; Zyper4KLimit=24, Zyper4KDevices=6, allDevices=12,
 allDevicesUp=6, Zyper4KDevicesExceeded=0
  server.deviceUpdates; active=0
  server.activeDeviceVersions; num_0.0.0.0=1, num_2.0.4.0=2,
 num_4.1.2.0=3
Success
```

Related Commands

`show server config`
`revert server`

show server ip duplicates

Shows if there any duplicate IP addresses in the system. Can include encoders, decoders, ICRON or Dante units

Syntax

```
show server ip duplicates [since]
```

Parameters

`since`

This parameter is optional and can be specified to display units based on the number of changes, using the `lastChangeId` value on each device. However, if used, a lastChangeId value must follow. Supply the `since` argument before the providing the `lastChangeId` value.

argument	Description
`since`	Required when using this parameter.

Example

```
show server ip duplicates
server(192.168.0.22);
lastChangeIdMax(88);
Success
```

Related Commands

`show server config`

show server redundancy

Displays information about master and slave Management Platforms

Syntax

```
show server redundancy
```

Parameters

since

This parameter is optional and can be specified to display units based on the number of changes, using the `lastChangeId` value on each device. However, if used, a lastChangeId value must follow. Supply the `since` argument before the providing the `lastChangeId` value.

argument	Description
`since`	Required when using this parameter.

Example

```
show server redundancy
server(172.16.6.111);
  server.status; state=master, version=2.1.1.36527, wasMaster=true,
wasSlave=true
  server.config; preferredMaster=true, preferredSlave=true
  server.virtualIp; address=0.0.0.0, networkInterface=video
Success
```

Related Commands

`set server redundancy`
`redundancy switchover`

show snmp

Displays information related to SNMP. (Please see Section 5 of the ZyPer Management Platform User Guide for additional details on SNMP support)

Syntax

```
show snmp arg
```

Parameters

type

Type: **STRING**

Supply one of the following arguments.

argument	Description
`trapServers`	Displays snmp trap servers.
`users`	Displays snmp users.

Example

```
show snmp trapServers
snmp(172.16.6.111);
Success

show snmp users
snmp(172.16.6.111);
Success
```

Related Commands

`add snmp`
`delete snmp`

show tls pem ca

Shows Transport Layer Security information for the Certificate Authority

Syntax

```
show tls pem ca arg
```

Parameters

type

Type: **STRING**

Supply one of the following arguments.

argument	Description
`cert`	Displays the certificate
`privKey`	Displays the RSA Private Key
`signedCert`	Display the signed certificate

Examples

```
show tls pem ca cert
pemData:
-----BEGIN CERTIFICATE-----
..................
-----END CERTIFICATE-----
Success

show tls pem ca privKey
pemData:
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-256-CBC,10E7EB7C47A3B07D64608BC1D4A63F5F
.................
-----END RSA PRIVATE KEY-----
Success

show tls pem ca signedCert
pemData:
-----BEGIN CERTIFICATE-----
..................
-----END CERTIFICATE-----
Success
```

Related Commands

`show tls summary`

show tls pem server

Shows Transport Layer Security information for the Server

Syntax

```
show tls pem server arg
```

Parameters

type

Type: **STRING**

Supply one of the following arguments.

argument	Description
`csr`	Displays certificate signing request
`cert`	Displays the certificate
`privKey`	Displays RSA private key
`caIntermediates`	Displays chain of certificates between root cert and your cert. Used when cert received from trusted certificate authority such as Verisign.

Examples

```
show tls pem server csr
pemData:
-----BEGIN CERTIFICATE REQUEST-----
.............
-----END CERTIFICATE REQUEST-----
Success

show tls pem server cert
pemData:
-----BEGIN CERTIFICATE-----
..................
-----END CERTIFICATE-----
Success
```

Related Commands

`show tls summary`

show tls summary

Shows a summary of Transport Layer Security settings.

Syntax

```
show tls summary
```

Parameters

none

Examples

```
show tls summary
server(192.168.0.22);
  server.tls.server; tlsMode=disabled, fqdnMode=fromCert, fqdn=NA
  server.tls.csr; status=invalid
  server.tls.serverCert; status=invalid
  server.tls.caChainCert; status=invalid
  server.tls.caCert; status=invalid
  server.tls.signed; status=invalid
Success

show tls summary
server(192.168.0.22);
  server.tls.server; tlsMode=disabled, fqdnMode=fromCert, fqdn=NA
  server.tls.csr; status=invalid
  server.tls.serverCert; status=invalid
  server.tls.caChainCert; status=invalid
  server.tls.caCert; status=valid, C=US, CN=caCert, L=Billerica,
 O=awCerts, OU=money, ST=MA, emailAddress=aweeks@zeevee.com
  server.tls.caCert; issuer=caCert
  server.tls.caCert;
 fingerprint=1CB41C0DA0FCE58E8F5601A976AB1C49FD4DC4EA
  server.tls.caCert; expires=12/20/32T10:43:23-0500
  server.tls.signed; status=invalid
Success
```

Related Commands

```
show tls pem server privKey
show tls pem ca privKey
load tls ca cert
load tls ca privateKey
load tls server
generate tls ca privKeyPass
generate tls server csr privKeyPass
```

show values

Shows all possible information/values associated with encoders, decoders, servers or multiviews.

Syntax

```
show values arg
```

Parameters

arg

Type: **STRING**

Supply one of the following arguments.

argument	Description
`all`	Displays all values encoders, decoders, servers and multiviews. (Status, Config, Info, Redundancy)
`encoder status`	Shows values associated with encoder status
`encoder config`	Shows values associated with encoder config
`decoder status`	Shows values associated with decoder status
`decoder config`	Shows values associated with decoder config
`server info`	Shows values associated with server info
`server config`	Shows values associated with server config
`server redundancy`	Shows values associated with server redundancy
`multiview status`	Shows values associated with multiview status
`multiview config`	Shows values associated with multiview config

Examples

```
show values server config
values(serverConfig);
  server.ipServerAddress.mode; values=dhcp|static
  server.ipManagementAddress.mode; values=none|dhcp|static
  server.telnetAccess; values=enabled|disabled
  server.encoderDefault.edid.audio; values=onlyPcm|allowCompressed
  server.dataTunnelMode; values=telnet|raw
  server.logging; values=<integer 1-5>
Success
```

Examples

```
show values encoder config
values(encoderConfig);
  device.gen.ethernetManagementPortMode; values=enabled|disabled
  device.gen.name; values=<string 1-256>
  device.ip.mode; values=dhcp|static|linkLocal
  device.ip.address; values=<IPv4Address>
  device.ip.mask; values=<IPv4Mask>
  device.ip.gateway; values=<IPv4Address>|NA
  device.rs232.baud; values=2400|9600|19200|38400|57600|115200
  device.rs232.parity; values=none|even|odd
  device.ports.videoPort; values=auto|initializing|unknown|
hdmi|displayPort|hdmiOptionalIn|vga|component|composite|s-
video|analogNone|hdsdi
  device.analogAudioStream.mode; values=enabled|disabeld
  device.audioOutSourceType.analogOutSourceType; values=analogAudio
|hdmiAudioDownmix
  device.edid.loadMode; values=auto|file
  device.edid.audio; values=onlyPcm|allowCompressed|serverDefault
  device.hdmi.hdcpMode; values=enabled|disabled|enabled1.4
  device.hdmiAudioStream.mode; values=enabled|disabeld
  device.previewStream.mode; values=enabled|disabeld
  device.previewStream.type; values=hls|jpeg
  device.previewStream.width; values=auto|<integer 180-400>
  device.usb.downlinks; values=[none] | [mac=<decMac1>|link_1, name
=<decName1>|existsButUnknown], [mac=<decMacN>|link_N, name=<decName
N>|existsButUnknown]
  device.usb.filter; values=none|exceptHid|storage
  device.videoStream.mode; values=enabled|disabeld
  device.videoScaledStream.mode; values=enabled|disabeld
Success
```

show videoWalls

Displays all video walls that have been created and all associated information.

Syntax

```
show videoWalls
```

Parameters

```
none
```

Example

```
show videoWalls
videoWall(wall1);
  videoWall.gen; videoSourceMac=none, numDisplayRows=2,
numDisplayCols=2
  videoWall.bezel; top=0, bottom=0, left=0, right=0
  videoWall.decodersRow1; col1=Top_Left, col2=Top-Right
  videoWall.decodersRow2; col1=Bot-Left, col2=Bot_Right
Success
```

Related Commands

```
create videoWall
set videoWall size
```

show zones

Displays all zones that have been created and all associated information.

Syntax

```
show zones
```

Parameters

none

Example

```
show zones
  1stfloor; Top-Right, Top_Left
  1stfloor.1stfloorroom2; empty
Success
```

Related Commands

```
add zoneDisplay
create zone
delete zone
delete zoneDisplay
```

shutdown server

Performs a shutdown of the Management Platform.

Syntax

```
shutdown server
```

Parameters

none

Example

```
shutdown server
Success
Connection closed by foreign host.
```

sign tls csr caPrivateKeyPass

Use the CSR to create signed TLS certificate

Syntax

```
sign tls csr caPrivateKeyPass * fromInput *

sign tls csr PrivateKeyPass * fromFile filename
```

Parameters

input

Type: **STRING**

String representing the Private Key Password. The system will promt for a string input. This should be the PEM data.

filename

Type: **STRING**

The name of the PEM data file to load. (Must already exist on ZMP in Files directory)

Example

```
sign tls csr caPrivateKeyPass * fromInput *
Enter passphrase: ******
Enter PEM text (ctr-d to end):
-----BEGIN CERTIFICATE REQUEST-----
............
-----END CERTIFICATE REQUEST-----
Success
```

Notes:

File must be previously copied onto ZMP into the Files directory using FTP.

Related Commands

show cls summary
show tls pem ca signedCert

sleep

Specifies a sleep duration in milliseconds. This command is sometime required when executing a series of commands within a web page, using AJAX. Often times, a pause must occur in order for a device or the Management Platform to change states before another command is executed.

Syntax

```
sleep ms
```

Parameters

ms

Type: **INTEGER**

The duration in milliseconds.

Example

```
sleep 500
Success
```

Related Commands

`script`

start encoder

Used to start a specific encoder multicast stream. This command only has affect if at least one decoder has been "joined" to the encoder and the "encoder stop" command has been used to override the enabling of the encoder stream. In effect, this command removes a previously entered "encoder stop" command – it returns stream control to normal operation based on existing "join" configuration. The command will immediately restore stream operation based on existing join configuration. No further join commands are required. (ZyPer4K family only)

Syntax

```
start encoder id stream arg
```

Parameters

id

Type: **STRING or MAC Address**

The identifier of the device. Either the full or portion of a string name or MAC address can be supplied.

arg

Supply one of the following arguments.

argument	Description
`analogAudio`	analog audio multicast stream.
`hdmiAudio`	downmix audio multicast stream
`video`	full scale video stream
`videoScaled`	downscaled video stream (for multiview)

Example

```
start encoder Myencoder1 stream video
Success
```

Related Commands

`stop encoder`

stop encoder

Used to stop a specific encoder multicast stream. This command only has affect if at least one decoder has been "joined" to the encoder. In effect, this command overrides any existing "join" command – either present or future. (ZyPer4K family only)

When stopping a "scaled-video" stream, any multiview window receiving that stream will go black. The rest of the multiview will be unaffected.

Syntax

```
stop encoder id stream arg
```

Parameters

id

Type: **STRING or MAC Address**

The identifier of the device. Either the full or portion of a string name or MAC address can be supplied.

arg

Supply one of the following arguments.

argument	Description
`analogAudio`	analog audio multicast stream.
`hdmiAudio`	downmix audio multicast stream
`video`	full scale video stream
`videoScaled`	downscaled video stream (for multiview)

Example

```
stop encoder Myencoder1 stream videoScaled
Success
```

Related Commands

`start encoder`

switch

This command is used in conjunction with the IR and RS232 switching commands. Both the `rs232` and the `ir` argument specify unidirectional connection between two devices. When switching data to the server, use the `show responses` command to retrieve the data.

Syntax

```
switch txid rxid type
```

Parameters

txid

Type: **STRING or MAC Address**

The name or MAC address of the encoder. String names are case-sensitive.

rxid

Type: **STRING or MAC Address**

The name or MAC address of the decoder. String names are case-sensitive.

type

Type: **STRING**

Supply one of the following arguments.

argument	Description
`ir`	Specifies a IR connection. (ZyPer4K family and ZyPerUHD only)
`rs232`	Connection to another device or the server. Set *rxid* = `none` to pass data to an arbitrary IP host.

Example

```
switch Wildlife SonyXBR4 rs232
Success
```

Related Commands

`send`

troubleReport

Generates capture logs and system state information and is used by the ZeeVee support team for troubleshooting purposes. This unencrypted file is in `.tgz` format and is written to the `/srv/ftp/files` folder on the Management Platform.

If using password option; the encrypted file is in `.gpg` format and written to the same location.

Syntax

```
troubleReport
troubleReport password pw
```

Parameters

pw

Type: **STRING**

Password to open the encrypted Trouble Report file.

Note the password is optional feature and will create an encrypted trouble report file.

Examples

```
troubleReport password 1234
Clean up files
Creating Trouble Report
Saving device status and configuration...
Saving SQL database...
Saving system files...
Saving device EDIDs...
Saving device specific information; this may take a few seconds...
Success

troubleReport
Clean up files
Creating Trouble Report
Saving device status and configuration...
Saving SQL database...
Saving system files...
Saving device EDIDs...
Saving device specific information; this may take a few seconds...
Success
```

update device

Updates the firmware on the encoder and/or decoder units. The firmware update file uses the `.apz or .zip` extension.

Syntax

```
update device arg file
```

Parameters

arg

Supply one of the following arguments.

argument	Description
`id`	Encoder or Decoder name. Names are case-sensitive
`all`	All encoders and decoders in the system
`encoders`	All encoders in the system
`decoders`	All decoders in the system

file

Type: **STRING**

The full filename of the software file.

Example

```
update device all Z4K_Firmware_HDMI2.0_v4_1_2_9.apz
Warning:(18) Firmware updating started, use 'show device status' to
monitor progress
Success
```

update server

Updates the Management Platform software. The server software file uses the `.zyper` extension. Refer to Updating the Software (page 227) for more information on using this command.

Syntax

```
update server file
```

Parameters

file

Type: **STRING**

The full filename of the software file.

Example

```
update server new-sofware-file.zyper
Success

Server rebooting; connection will end
```

Important Note:

The ZyPer MP update file will be available in three, platform-specific versions. Please use the correct version for the hardware platform being updated.

ZyPerMP NUC computer: update_nuc_3.2.xxxxx.zyper
ZyPerMP Proserver: update_proserver_3.2.xxxxx.zyper
ZyPerMP VMware: update_vm_3.2.xxxxx.zyper

2 Modules / Plug-Ins

Disclaimers

ZeeVee control offerings

ZeeVee provides modules / plug-ins for third-party control systems as a courtesy to our channel partners to speed development time. In addition to the modules/drivers, our fully documented ZyPer Management Platform API allows our customers to connect to nearly any control system.

ZeeVee support for control products

The ZeeVee support team is available to assist you in getting your new ZyPer encoders, decoders, and management system working correctly and providing direction with API calls for desired functionality.

ZeeVee limitation

Given the variety of capabilities across various control products, ZeeVee cannot commit to deep-rooted knowledge that would allow us to address all support requests. Further, the drivers provided may not include all commands needed for a particular solution that your organization has devised.

How you can prepare

While ZeeVee provides modules/drivers and the API, your organization will need expertise on the chosen control system and may require technical support from the control system vendor.

Where to find documentation

All ZeeVee and ZyPer documentation can be found on our website here: https://www.zeevee.com/documentation/

ZeeVee Control Modules / Plug-Ins

ZeeVee Modules / Plug-Ins

ZeeVee provides many different control options for the ZyPer family of products. (ZyPer4K, ZyPerUHD60, ZyPerUHD). This includes the availability of modules or built in support directly for the following 3rd party control systems.

Support

Modules / Plug-Ins for 3rd party control systems are provided as a courtesy to developers in an effort to reduce development time. They are provided “as-is” from ZeeVee with no additional support provided other than the accompanying documentation. See Disclaimers section at the beginning of this document.

Availability

ZeeVee produced modules can be accessed on the ZeeVee ConnectivityXchange

https://github.com/ConnectivityXchange

Module / Plug-In Basics

A module or driver is an interface between the primary controller (Crestron, RTI, Control4 etc..) and the ZeeVee Management Platform (ZMP). Think of the module or driver as a translator. It will translate commands from the primary control system into the appropriate commands that can be understood by the ZyPerMP.

Note: Not all ZyPerMP API commands are supported by the module/driver. Our goal has been to support the most common and basic commands to allow routing of video from any source to any destination as well as some special features such as multiview and video walls.

Configuration Tips

Documentation

You will need to know the ZyPer Management Platform API commands to perform actions that are not provided by the module / plug-in. (The first part of this document).

Basic Setup

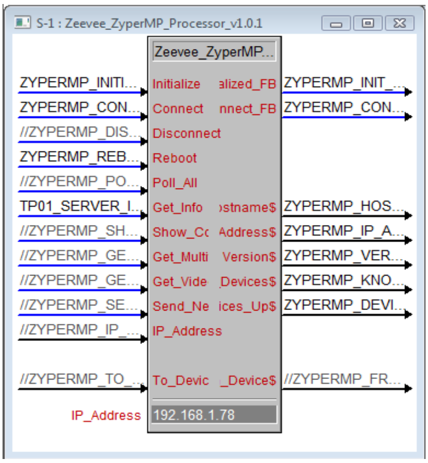
Regardless of the control system used to control the ZyPer equipment, it is a requirement that the ZyPer Management Platform (ZMP) must be present.

1. The ZMP IP address should be configured either via DHCP or given a static address.
2. The ZMP IP address MUST be on the same network as the primary hardware controller. (Crestron 4-Series controller, Control4 EA1, EA3, EA5, DTVGameControl iPad)
3. All ZyPer endpoints should be configured using either the ZMP Graphical User Interface (GUI) or via the ZMP API (Telnet or SSH).
   a. Assign all endpoints a logical name (Encoders and Decoders)
   b. Design/create any video walls and give them logical names
   c. Design/create any multiview displays and give them logical names (ZyPer4K only)

Module / Plug-In Setup

Every control system is different but will have some common features when it comes to basic setup and configuration.

1. The IP address of the ZMP must be provided so that the primary controller can communicate with the ZMP.
   a. Where the IP address is provided will vary based on the control system. Below are some examples
      i. Crestron – SIMPL Windows under IP_Address field for the ZeeVee_ZyPerMP_Processor logic module. See image below:



IP_Address 192.168.1.78

2. The names of the endpoints must be provided. Similar to the IP address detailed above.

3. The names of Multiview's (ZyPer4K only) need to be provided.

   a. Crestron – SIMPL Windows in the appropriate Multiview logic module.

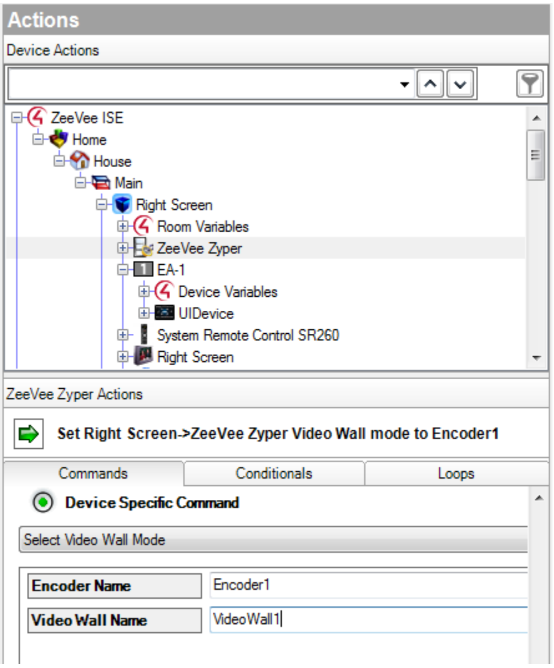
   b. Control4 – Create a "virtual" encoder and assign it appropriate multiview name and set the "Input Join Type" to multiview.

Video Walls

Video walls are configured differently depending on the control system.

a. Crestron – Similar to IP address and multiview. SIMPL Windows in the appropriate Video Wall logic module.

b. Control4 – Video walls need to be assigned to a special programmed command. For example, a Red Button press on the SR260 remote control. In Composer highlight the room, select Programming and Commands. Select the "Red Button". Then in the Actions window select ZeeVee ZyPer, scroll down to Device Specific Command. Enter the Encoder and Video Wall Names. (See image below)



Set Right Screen->ZeeVee Zyper Video Wall mode to Encoder1

Device Specific Command

Select Video Wall Mode

Encoder Name	Encoder1
Video Wall Name	VideoWall1

Serial / RS-232

The ZyPer Management Platform (ZMP) must first be linked to the specific endpoint to send RS232 information. This can be done with either the dataConnect or switch command.

Examples:

dataConnect Dec1 server rs232 tunnelPort 1234

switch Dec1 server rs232

Note: The feature of dataConnect was added to allow a third party to connect to the ZMP server with a specific port and pass raw or telnet API commands (depending on the mode) to the server and port which is designated for a particular encoder or decoder.

Important Note: Issuing the dataConnect or switch command can cause the ZyPer endpoint to reboot to enable the link. Disconnecting the link can also cause the endpoint to reboot. The link should only be established once and then left alone to prevent undesired endpoint reboots.

When using any control system; that system is communicating with our ZMP and not to any specific endpoint.

When sending RS232 commands to an endpoint via the ZMP you must follow very specific syntax.

The ZeeVee command is: send <decoder_name> rs232 text

Here are examples on this. (Assume decoder name is Dec1)

Input command: send Dec1 rs232 Hello
Received at Dec1: Hello (Note, no line feed or carriage return)

Input command: send Dec1 rs232 Hello\r\n
Received at Dec1: Hello (with carriage return and line feed)

Input command: send Dec1 rs232 Hello World
Received at Dec1: Nothing. You get an error. Bad syntax. You cannot have a space between Hello and World.

Input command send Dec1 rs232 Hello_World
Received at Dec1: Hello_World (Note, no line feed or carriage return)

Input command send Dec1 rs232 “Hello World”
Received at Dec1: Hello World (Note, no line feed or carriage return)

Input command send Dec1 rs232 “Hello World”\r\n
Received at Dec1: Nothing. You get an error. Bad syntax. Token \r\n is invalid.

You need to contain the line feed and carriage return symbols inside the quotes in this case.

Input command send Dec1 rs232 “Hello World\r\n”
Received at Dec1: Hello World (with carriage return and line feed)

Text can also be Hexadecimal Code as shown below:

Input command:

```
send Dec1 rs232 \x48\x65\x6c\x6c\x6f\x20\x57\x6f\x72\x6c\x64\x0A\x0D
```

Received at Dec1: Hello World (with carriage return and line feed)

The ZyPer Management Platform also has the ability to receive RS232 communications that were input into a ZyPer endpoint. To view any such RS232 string, you use the “show responses” command.

Example:

```
Zyper$ show responses DEC1 rs232 since 0
device(d8:80:39:59:bf:57);
  device.rs232Response.0; string=”Have a great day!\x0D”
  device.rs232Response.1; string=”\x0A”
lastChangeId(2);
Success
```

Important Crestron Note

When using Crestron as the control system, you may need to append an extra \ symbol before the Carriage return symbol. Otherwise carriage return may not work.

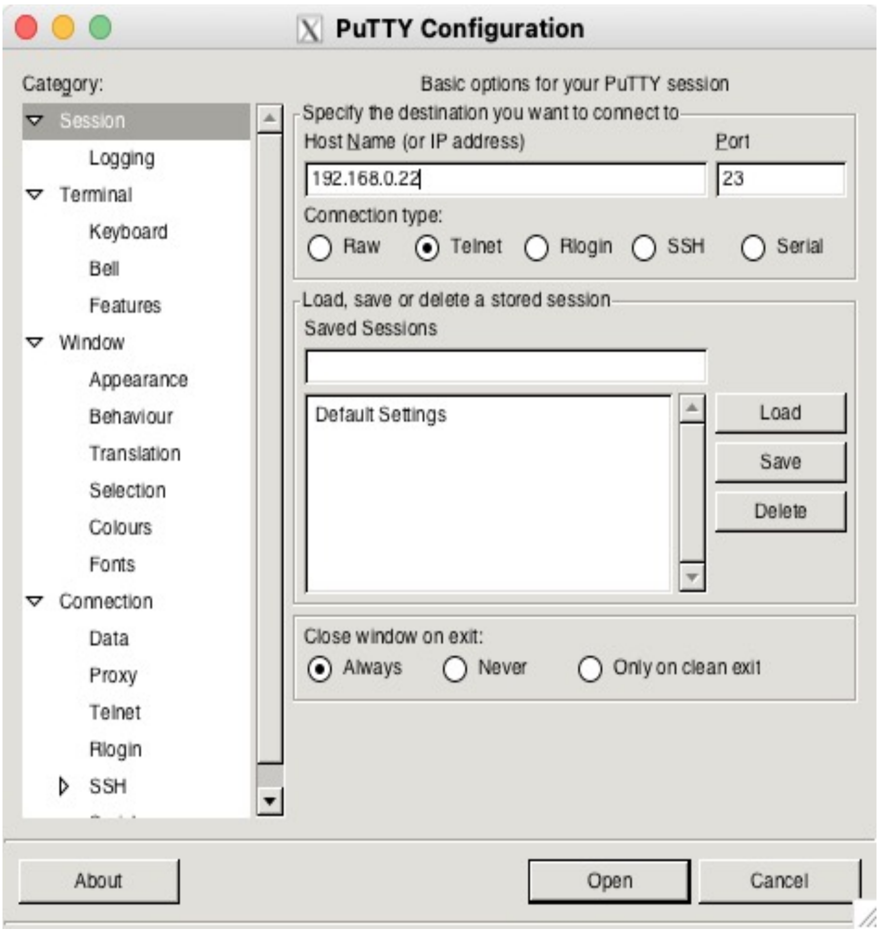
Example using Crestron to turn on/off LG display.

LG TV
ON
```
send DecoderName rs232 \x6B\x61\x20\x30\x30\x20\x30\x31\\x0D
```
OFF
```
send DecoderName rs232 \x6B\x61\x20\x30\x30\x20\x30\x30\\x0D
```

Troubleshooting Tips

If a command issued from a 3rd party control system is not working, the first thing to do is confirm the command works when issued directly in the API. To do this you should Telnet or SSH into the API using a tool such as Putty.

Once the Telnet or SSH session is open, you can issue any API command to the ZyPer Management Platform manually. You can also use the HELP feature of the API to learn the correct API commands and syntax.



Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): 192.168.0.22

Port: 23

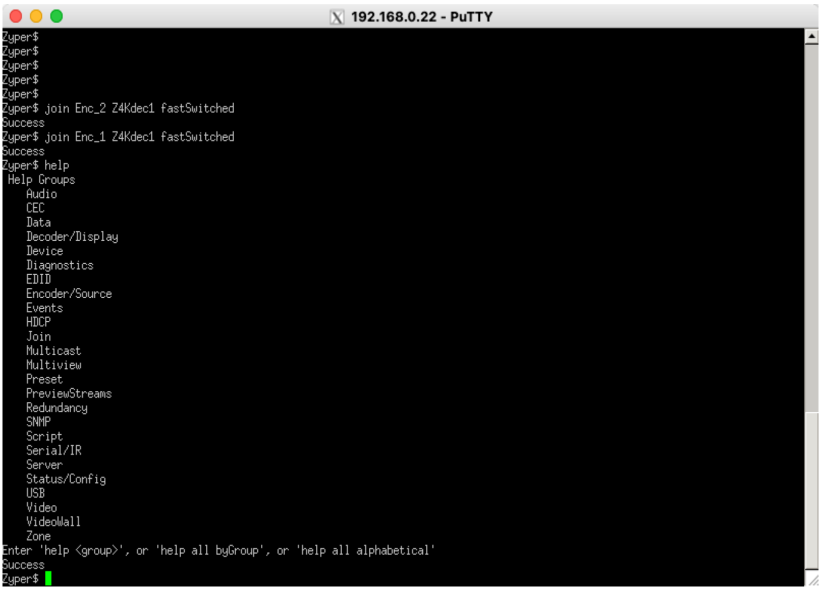
Connection type: Raw [✓ Telnet] Rlogin SSH Serial

Load, save or delete a stored session

Saved Sessions: Default Settings

Close window on exit: [✓ Always] Never Only on clean exit

Once the command is working via Telnet directly to the ZMP, copy the exact syntax as needed into the 3rd party control system.



```
Zyper$
Zyper$
Zyper$
Zyper$
Zyper$
Zyper$ join Enc_2 Z4Kdec1 fastSwitched
Success
Zyper$ join Enc_1 Z4Kdec1 fastSwitched
Success
Zyper$ help
 Help Groups
    Audio
    CEC
    Data
    Decoder/Display
    Device
    Diagnostics
    EDID
    Encoder/Source
    Events
    HDCP
    Join
    Multicast
    Multiview
    Preset
    PreviewStreams
    Redundancy
    SNMP
    Script
    Serial/IR
    Server
    Status/Config
    USB
    Video
    VideoWall
    Zone
Enter 'help <group>', or 'help all byGroup', or 'help all alphabetical'
Success
Zyper$
```

3 Appendix

Updating the Software

Using Mac OS X

1. Make sure the Management Platform is powered and is working correctly.

2. Download the latest software from the ZeeVee website. Make note of the location of where the software was downloaded.

3. Launch the Terminal app, found under the Applications > Utilities folder. By default, the current directory will be the Home directory.

```
Last login: Tue Mar 22 14:24:08 on console
Andrews-MacBook-Pro:~ Andrew$
```

4. Change the directory to the location of the downloaded software file. For example, if the software was downloaded to the Desktop, then change to the Desktop folder, as shown:

```
Last login: Tue Mar 22 14:24:08 on console
Andrews-MacBook-Pro:~ Andrew$ cd desktop
Andrews-MacBook-Pro:desktop Andrew$
```

5. Use the FTP protocol to login to the Management Platform. At the terminal prompt, type the following and press the [ENTER] key.

```
Andrews-MacBook-Pro:desktop Andrew$ ftp 192.168.1.6
```

6. Enter the user name and password. Use anonymous for the user name and use guest for the password. The password will not be echoed to the screen.

```
Andrews-MacBook-Pro:desktop Andrew$ ftp 192.168.1.6
Connected to 192.168.1.6
220 (vsFTPd 3.0.2)
Name (192.168.1.6:Andrew): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

7. Type `cd files` at the ftp prompt to change to the /files directory.

```
ftp> cd files
250 Directory successfully changed.
ftp>
```

8. Enter and run the `put` command, followed by the full name of the software file, as shown. Make sure to replace [version] with the version of the filename you are using. For example:

```
ftp> put update_nuc_3.0.38847.zyper
```

9. Press the [ENTER] key. Information similar to the following will be displayed.

```
local: update_nuc_3.0.38847.zyper remote: update_nuc_3.0.38847.
zyper
229 Entering Extended Passive Mode (|||35257|).
150 Ok to send data.
100% |**********************************|  6830 KiB   94.30
MiB/s    00:00 ETA
226 Transfer complete.
6994519 bytes sent in 00:00 (92.30 MiB/s)
```

10. Type the `exit` command to exit FTP.

```
ftp> exit
Andrews-MacBook-Pro:desktop Andrew$
```

11. Telnet to the Management Platform, as shown.

```
$ telnet 192.168.1.6
Trying 192.168.1.6...
Connected to 192.168.1.6
Escape character is '^]'.
zyper$
```

12. Use the `update` command to update the Management Platform. Once entered, the Management Platform will reboot and the software will be updated. Note that the connection will be lost, temporarily, during the update process.

```
zyper$ update server update_nuc_3.0.38847.zyper
Success

Server rebooting; connection will end
```

Using Windows

1. Make sure the Management Platform is powered and is working correctly.

2. Download the latest software from the ZeeVee website. Make note of the location of where the software was downloaded.

3. Open Chrome and enter the IP address of the Management Platform using the FTP protocol. For example:

   ```
   ftp://169.254.185.207
   ```

4. The /files folder will be displayed.

5. Drag-and-drop the latest software file to the /files folder.

6. Use the Telnet protocol to access the Management Platform API.

7. Use the `update` command to update the Management Platform. Once entered, the Management Platform will reboot and the software will be updated. Note that the connection will be lost, temporarily, during the update process.

   ```
   zyper$ update server update_nuc_3.0.38847.zyper
   Success

   Server rebooting; connection will end
   ```

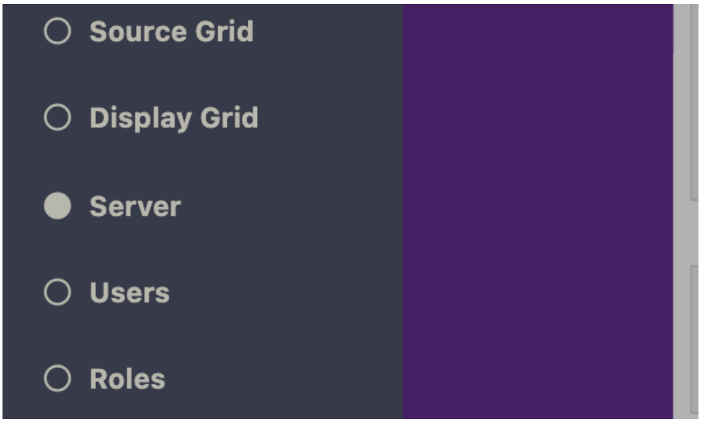
Using ZyPer Management Platform

1. Make sure the Management Platform is powered and is working correctly.

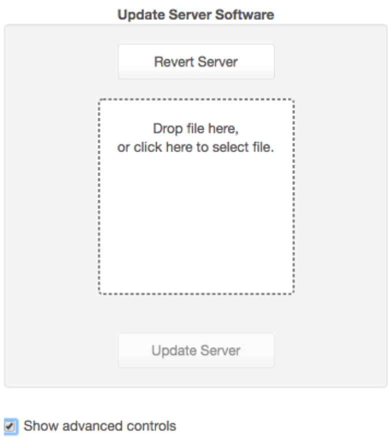
2. Download the latest software from the ZeeVee website. Make note of the location of where the software was downloaded.

3. Login to the ZyPer Management Platform. Refer to Accessing ZyPer Management Platform (page 11) for more information.

4. Click the **Server** option at the left of the page.



5. Scroll down within the Server pain until you see the option to Update Server Software. Drag the latest software into the box and press **Update Server** to begin process. (**Note:** You can also revert the server to the previously installed version of software by clicking the **Revert Server** button) "Show advanced controls" must be enabled to use this option.



Important Notes:

The ZyPer MP update file will be available in four, platform-specific versions. Please use the correct version for the hardware platform being updated.

ZyPerMP NUC computer (Single Ethernet Port): update_nuc_3.0.xxxxx.zyper
ZyPerMP NUC computer (Two Ethernet Ports): update_nuc2004_3.0.xxxxx.zyper
ZyPerMP Proserver: update_proserver_3.0.xxxxx.zyper
ZyPerMP VMware: update_vm_3.0.xxxxx.zyper

- First generation ZMP NUC devices are not supported with the 3.0 release of ZMP API. These devices are running an incompatible version of the Linux Operating System and were last shipped by ZeeVee back in 2017. These units can be easily identified as they have the brand name "GigaByte" written on the underside of the unit.
- Customers using this older NUC that wish to upgrade to the 3.0 ZMP API release should contact the ZeeVee sales team (sales@zeevee.com) to purchase an updated ZMP Hardware device.

Redundancy Configuration Instructions

To configure redundancy, follow the steps below. The secondary server must be running for the redundancy fields to be visible in ZMP or the API.

Configuring redundancy through the API

Configuring the IP Address

1) Login to the main ZMP, or Master through telnet.

2) Issue the "**set server redundancy all-servers**" command to configure redundancy

IE: set server redundancy all-servers virtual-ip address 172.16.5.239 network-interface video

3) Use the "**show server redundancy**" command to review the redundancy configuration and confirm the changes

4) Login to the Secondary server, or Slave,through telnet.

5) Use the "**show server redundancy**" command to review the redundancy configuration and confirm the changes

Configure the preferred roles

1) Login to the Master ZMP through telnet.

2) Issue the "**set server redundancy this-server**" command to set the preferred master and slave states on the server.

IE: set server redundancy this-server preferred-master true preferred-slave false

3) Use the "**show server redundancy**" command to review the redundancy configuration and confirm the changes

4) Login to the Slave ZMP through telnet.

5) Use the "**show server redundancy**" command to review the redundancy configuration and confirm the changes

Configuring redundancy through ZMP

1) Login through you Master ZMP GUI with Chrome.

2) Open the Server Panel

3) Scroll down to the Redundancy fields

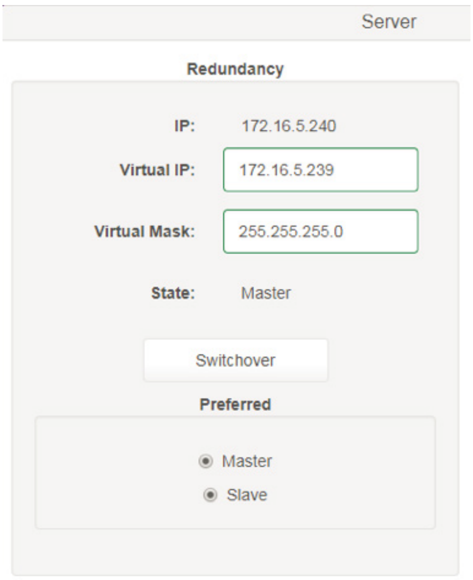
4) Set the fields listed below.

Virtual IP: The IP address that the Master and Slave servers will use. This IP address must be unique and available on the network as it will be used for telnet access for the API as well as ZMP.

Virtual Mask: The subnet mask for the virtual interface, must be correct for the IP address listed above and not it should not conflict with the main eth0 interface.

Preferred Roles Radio Button: The preferred roles for the server. This field is used to decide the Master or Slave upon both servers initializing at the same time. Although rare, this can occur.

State: The current role of the current Server connected to.



IP: 172.16.5.240
Virtual IP: 172.16.5.239
Virtual Mask: 255.255.255.0
State: Master

After configuration is complete on the Master, the information should populate to the Slave server. The preferred roles for the Slave server will still need to be configured. This can be done by logging into ZMP using the Slave server IP address and modifying the Preferred roles.

The "**State**" field will reflect the servers current state.

5) After the configuration changes are made, login into ZMP with the Virtual IP address configured above.

The server panel should show the correct redundancy information.

Note: The "switchover" button above will allow the servers to swap roles as needed.

Virtual interface on the ZMP.

Below is an example of the output of the “ifconfig” from the ZMP showing the virtual IP configured on the current master server.

```
eth0      Link encap:Ethernet  HWaddr 40:8d:5c:32:46:0e
          inet addr:172.16.5.240  Bcast:172.16.5.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX bytes:36015816 (36.0 MB)  TX bytes:31515642 (31.5 MB)

eth0:ZMP  Link encap:Ethernet  HWaddr 40:8d:5c:32:46:0e
          inet addr:172.16.5.239  Bcast:0.0.0.0  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1

eth0:avahi Link encap:Ethernet  HWaddr 40:8d:5c:32:46:0e
          inet addr:169.254.4.58  Bcast:169.254.255.255  Mask:255.255.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX bytes:4873342 (4.8 MB)  TX bytes:4873342 (4.8 MB)
```

Important Note Regarding the Virtual Interface on the ZMP

If the Master ZMP goes offline the Slave ZMP will take over and become the new Master. There is a brief period of between 10 and 20 seconds where the Virtial IP address may not be availalbe on the network. This is due to the ARP Age Time parameter setting of the associated network.

Please consult your network switch documentation regarding this parameter and set it to the lowest setting possible.

zeevee™
Intelligent AV Distribution

North America: +1 (347) 851-7364 • sales@zeevee.com
International: +1 (347) 851-7364 • sales@zeevee.com
EMEA: +44 1494 956677 • EMEAsales@zeevee.com