Resources

Alta Instances

Resource	Description
GET /alta/alta_instances	*Action* Retrieves a list of Alta Virtual Encoder instances *Query Parameters* By default, this endpoint will return a list of instance IDs, each of which can be passed to other endpoints to get more info about an instance. However, if one or more of the following query parameters is provided and set to `true`, this endpoint will instead return a list of objects, each of which has additional info about an instance. **settings** — Set this query paremeter to `true` to include the currently configured settings for each instance in the response. **status** — Set this query paremeter to `true` to include the current status for each instance in the response.

Example Request:

GET http://username:password@localhost/alta/alta_instances

Example Response:

```
{
  "alta_instances": [
    "EEG_BR_3G46Z6VLSV8B01JZ",
    "EEG_BR_24RNNSDZ1J020IFJ"
  ]
}
```

Resource	Description

POST /alta/alta_instances

Action

Create and configure a new Alta Virtual Encoder instance

JSON Parameters

multicastinterface	IP address of the local interface to subscribe to media multicast through and send them out. Note device startup will fail if this server does not have a NIC using the specified IP address. In 2022-7 usage, set the primary NIC IP here, and use secondarymulticastinterface for the secondary NIC IP.	`ip address`
icapco	Company credential for logging into EEG iCap (omit if not connecting to iCap)	`string`
icapuser	Encoder username credential for logging into EEG iCap	`string`
icappass	Password credential for logging into EEG iCap	`string`
audiomix	Comma-separated list of channels to	`integer,integer`

	be included in the mono mix down sent to the captioner. Recommended to list one, two, or six (5.1) channels	
td1	Destination address for an external telnet caption encoder device that wants to receive cloned S1 caption data matching the iCap input to this channel (typically CTRL+A format)	`(ip address):(port number)`
tu1	Username for external S1 telnet clone device (if authentication is required)	`string`
tp1	Password for external S1 telnet clone device (if authentication is required)	`string`
td2	Destination address for a second external telnet device that wants to receieve cloned caption data matching the iCap input to the tlang2 channel	`(ip address):(port number)`
tu2	Username for second	`string`

	external telnet clone device (if authentication is required)	
tp2	Password for second externa telnet clone device (if authentication is required)	`string`
tlang2	Number 1-6 for the language/service number that second telnet clone should copy commands for (based on iCap access code service number)	`integer`
telnetinput	Local listen address to accept input caption data from non-iCap users or devices (insecure).	`(ip address):(port number)`
testcc	Write looping test closed caption messages into the output data. This feature will turn itself off for the remainder of the run if/when a live iCap captioner connects and begins writing.	`on`
warnlevel	Logging detail level	`(0|1|2|3|4|5)`

dvb_text_cfg	List of ISO 639 language codes and teletext magazine/page numbers, only needed for output of DVB Text. The string should be a comma separated list where each entry has a language code, a slash, and a page number, like "eng/801,spa/802".	`string`
audiosrc	Listen address for the 2110-30 reference audio stream	`(udp|rtp)://(ip address):(port number)`
audiosamplebits	Number of bits per audio sample - supports 16 or 24	`integer > 0`
audiosamplerate	Sampling rate of audio stream	`integer > 0`
audiochannels	Number of audio channels present in the stream (supports up to 64)	`integer`
ancillarysrc	Listen address for a 2110-40 upstream ancillary data stream. Captions will be merged together based on live control, while other ancillary	`(udp|rtp)://(ip address):(port number)`

	services will be preserved.	
output	Output destination for the 2110-40 ancillary data stream	`(ip address):(port number)`
video_standard	Determines the video standard that 2110-40 data will be output with reference to. Accepted values - 1080i60, 720p60, 1080p60, 1080i50, 1080p50. Strings with "60" are really 59.94 NTSC rate. Use "auto" to rely on system data collected from the SMPTE 2059-2 PTP TLV	`string`
ancvitc	Insert 6060 ANC VITC data with time-of-day time code	`on`
output_cc_coding	Options for coding the output captions in Alta, the default is atsc (6101), the alternative is OP47 Teletext (5201)	`(atsc|op47)`
nmosdevicelabel	Custom name for identifying this device. Also used as the label field	`string`

	for this device in the NMOS Node Registry (if enabled).	
outputpayloadtype	RTP payload type - a 7-bit value in the output 2110-40 RTP packet header. Default is 100 per standard practice.	
use_nmos	Enable to register this device in NMOS IS-04 and control this device's multicast input and output settings at run-time using the NMOS IS-05 Connection Management API.	`on`
secondarymulticastinterface	IP address of local secondary interface for receiving and sending 2022-7 secondary multicasts. Note device startup will fail if this server but does not have a NIC using the specified IP address.	`ip address`
secondaryoutput	Secondary output destination for the 2110-40 ancillary data	`(ip address):(port number)`

	stream through the 2022-7 secondarymulticastinterface	
secondaryaudiosrc	Listen address for the secondary 2110-30 reference audio stream through the 2022-7 secondarymulticastinterface	`(udp|rtp)://(ip address):(port number)`
secondaryancillarysrc	Listen address for a secondary 2110-40 upstream ancillary data stream through the 2022-7 secondarymulticastinterface.	`(udp|rtp)://(ip address):(port number)`

Query Parameters

turn_on	For backwards compatibiity with previous versions of this API, the newly created instance will be immediately turned on by default. Set this query parameter to `false` to just create a new instance and save the settings without also turning the instance on.

Example Request:

POST http://username:password@localhost/alta/alta_instances?turn_on=false

JSON body:

```
{
  "output": "239.120.200.011:5000",
  "primary": "udp://239.120.100.015:5000",
  "source": "udp://239.120.100.014:5000",
```

```
  "warnlevel": 1
}
```

Example Response:

```
{
  "instanceID": "EEG_BR_3G46Z6VLSV8B01JZ"
}
```

Start Process

Resource	Description
POST /alta/alta_instances/**<instanceID>**/turn_on	*Action* Turns on a previously created Alta instance *URL parameters* **instanceID** — The instanceID for the Alta instance to turn on

Example Request:

POST

http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/turn_on

Example Response:

```
{}
```

Stop Process

Resource	Description
POST /alta/alta_instances/**<instanceID>**/turn_off	*Action* Turns off a previously created Alta instance *URL parameters* **instanceID** — The instanceID for the Alta instance to turn off

Example Request:

POST

http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/turn_off

Example Response:

```
{}
```

Toggle Process

Resource	Description
POST /alta/alta_instances/**<instanceID>**/toggle_on_off	*Action* Turns on a previously created Alta instance if it is currently off, or turns off a previously created Alta instance if it is currently on *URL parameters* **instanceID** — The instanceID for the Alta instance to toggle

Example Request:

POST

http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/toggle_on_off

Example Response:

```
{}
```

Settings

Resource	Description
GET /alta/alta_instances/**<instanceID>**/settings	*Action* Retrieves Alta instance configuration/settings *URL parameters* **instanceID** — The instanceID for the Alta instance to retrieve

Example Request:

GET

http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/settings

Example Response:

```
{
  "settings": {
```

```
    "primary": "udp://239.120.100.101:5000",
    "source": "rtp://224.0.101.140:5000",
    "output": "239.120.200.101:5000"
  }
}
```

Resource	Description
PUT /alta/alta_instances/**<instanceID>**/settings	*Action* Modifies Alta instance configuration/settings. Note that this operation is only supported for instances that are off. *JSON Parameters* The allowed JSON fields are the same as those for the above `POST /alta/alta_instances` endpoint for creating a new instance. Note that the settings provided to this endpoint will completely overwrite the previous settings (e.g., any settings that are absent from this request but were provided previously will be removed). *URL parameters* **instanceID** — The instanceID for the Alta instance to modify

Example Request:

PUT

http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/settings

JSON body:

```
{
  "primary": "udp://239.120.100.101:5000",
  "source": "rtp://224.0.101.140:5000",
  "output": "239.120.200.101:5000"
}
```

Example Response:

```
{
  "settings": {
    "primary": "udp://239.120.100.101:5000",
    "source": "rtp://224.0.101.140:5000",
    "output": "239.120.200.101:5000"
  }
}
```

Status

Resource	Description
GET /alta/alta_instances/**<instanceID>**/status	*Action* Retrieves status of an Alta instance, including a list of log files that can be downloaded from the logging endpoints *URL parameters* **instanceID** — The instanceID for the Alta instance to retrieve

Example Request:

GET

http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/status

Example Response:


```
{
  "health_indicators": {
    "anc": {
      "ANC Rx IP": "239.106.40.1:50040",
      "ANC Rx BW": "36.2 kbps",
      "ANC Rx Pkt": "6101",
      "ANC Rx Timing": "Locked",
      "Output BW": "39.4 kbps"
    },
    "audio": {
      "Audio BW": "4606.2 kbps",
      "Audio IP": "226.0.30.9:1234",
      "Audio Timing": "Locked"
    },
    "video": {
      "ICap Status": "Connected",
      "Video Standard": "NTSC Interlace"
    }
   },
   "log_list": [
    "20180830_14.log",
    "20180828_13.log",
    "20180823_12.log",
    "20180821_11.log",
    "20180820_10.log",
    "20180820_9.log",
    "20180820_8.log",
    "20180818_7.log",
    "20180815_6.log",
```

```
    "20180815_5.log",
    "20180815_4.log",
    "20180815_3.log",
    "20180815_2.log",
    "20180815_1.log",
    "20180815_0.log"
  ],
  "state": "RUNNING",
  "status": [
    "18:09:10: Initiating primary input: RTP listen on port 5000\n",
    "18:09:10: Initiating source input: RTP listen on port 60000\n",
    "18:09:10: Receive: reset RTP counter to 13047\n",
    "18:09:10: Target pace in bytes per millisecond set to 0\n",
    "18:09:10: Initiating video send on port 8034\n",
    "18:09:10: Added destination 10.0.0.8:5000 to session\n",
    "18:09:10: Target pace in bytes per millisecond set to 2500\n",
    "18:09:11: Output GOP ready, preloading\n",
    "18:09:11: Output GOP ready, preloading\n",
    "18:09:11: Output GOP ready, preloading\n",
    "18:09:11: Output GOP ready, preloading\n",
    "18:09:12: Output GOP ready, preloading\n",
    "18:09:13: Receive: RTP received: 6956, lost: 0, recovered by FEC: 0\n",
    "18:09:19: Receive: RTP received: 16956, lost: 0, recovered by FEC: 0\n",
  ]
}
```


Logs

<table>
<tr><th>Resource</th><th>Description</th></tr>
<tr><td>GET /alta/alta_instances/<instanceID>/logs</td><td>Action
Retrieves the current or most recent log file associated with an Alta instance. The current log for a running instance will contain events logged only since midnight local time on the current day. For older events, see the status endpoint for a list of available files and specify a specific filename in the URI
URL parameters<table><tr><td>instanceID</td><td>The instanceID for the Alta channel that generated the log</td></tr></table></td></tr>
<tr><td>GET /alta/alta_instances/<instanceID>/logs/<logfilename></td><td>Action
Retrieves a specific log file related to this Alta channel. Use the Status endpoint first to retrieve a list of log filenames that exist for this Alta channel.</td></tr>
</table>

	URL parameters **instanceID** — The instanceID for the Alta channel that generated the logs **logfilename** — The filename for a specific log file. Typically this looks like YYYYMMDD_xyz.log, for a list of available files use the Status endpoint for this Alta channel first.

Example Request:

GET

http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/logs/20180820_9.log

Example Response iCap Alta 2110 Encoder:

```
14:04:48: <info> Startup: EEG iCap Alta 2110 Encoder Version 1.2.0
14:04:49: <error> Failed to register to local IS-04 Node: Internal server error
14:04:49: <warning> iCap: could not write server address list
14:08:29: <fatal> Interrupt - closing
```

Delete Instance

Resource	Description
DELETE /alta/alta_instances/**<instanceID>**	*Action* Deletes a previously created Alta instance. Note that this operation is only supported for instances that are off. **WARNING**: Use this endpoint with care. This will completely scrub the record of an Alta instance and cannot be undone. *URL parameters* **instanceID** — The instanceID for the Alta instance to delete

Example Request:

DELETE

http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ

Example Response:

```
{}
```

Network GPI's, Presets + Cue Point Triggering

<table>
<tr><th>Resource</th><th>Description</th></tr>
<tr><td>GET /system/gpi/settings</td><td>Action
Retrieves the current GPI settings including 'protocol', 'ip' and 'port'.</td></tr>
<tr><td>GET /system/gpi/triggers/<protocol></td><td>Action
Retrieves a list of the pre-saved GPI triggers for the given protocol.
URL parameters<table><tr><td>protocol</td><td>Network GPI protocol (i.e. 'JL Cooper eBox')</td></tr></table></td></tr>
<tr><td>POST /system/gpi/triggers/<protocol>/<gpi_num>/trigger</td><td>Action
Triggers a pre-saved GPI trigger preset with property 'gpi_num' for the given 'protocol'.
URL parameters<table><tr><td>protocol</td><td>Network GPI protocol (i.e. 'JL Cooper eBox')</td></tr><tr><td>gpi_num</td><td>GPI number of the preset to trigger</td></tr></table></td></tr>
<tr><td>POST /alta/instances/<instanceID>/cue-point</td><td>Action
Triggers a SCTE35 cue-in or cue-out point for the specified instanceID
JSON parameters<table><tr><td>action</td><td>'SCTE_35_SPLICE_INSERT' or 'SCTE_35_RETURN_TO_NETWORK'</td></tr><tr><td>spliceEventId</td><td>SCTE35 event ID (must be integer and > 0)</td></tr><tr><td>duration</td><td>Break duration in 90kHz clock cycles (i.e. 5400000 = 60secs)</td></tr></table></td></tr>
<tr><td>POST /alta/alta_instances/<instanceID>/trigger_preset</td><td>Action
Triggers a GPI preset for the specified instanceID</td></tr>
</table>

JSON parameters

type	**Trigger types:**' lexi', 'scte104'					
function	**Lexi functions:** 'speaker_change', 'turn_on', 'turn_off'	**SCTE104 functions:** 'start_normal', 'start_immediate', 'end_normal', 'end_immediate', 'cancel'				
parameters	**Lexi speaker change:** 'delay_compensation' (float in seconds)	**Lexi on/off:** 'instance_id' (Lexi instance ID)	**SCTE104 start normal:** 'event_id', 'program_id', 'break_duration' (1/10sec), 'preroll_	**SCTE104 start immediate:** 'event_id', 'program_id', 'break_duration' (1/10sec)	**SCTE104 end normal:** 'event_id', 'program_id', 'preroll_time' (1/1000sec)	**SCTE104 end immediate + cancel:** 'event_id', 'progra

				time ' (1/1000 sec)			m_id'	

Example Request:

POST http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/trigger_preset

JSON body:

```
{
   "type" : "scte104",
   "function" : "start_immediate",
    "parameters" : {
        "event_id" : "122345",
        "program_id" : "55",
        "break_duration" : "900"
    }
}
```

Example Response:

```
{
   "success": true
}
```

Example Request:

POST http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/trigger_preset

JSON body:

```
{
   "type" : "lexi",
   "function" : "speaker_change",
    "parameters" : {
        "delay_compensation" : "1.2"
    }
}
```

Example Response:

```
{
   "success": true
}
```

Example Request:

GET http://username:password@localhost/system/gpi/triggers/JL Cooper eBox

Example Response iCap Alta Encoder:


```
[
  {
    "alta_channel": "EEG_BR_P0BX1U3EL77MKLGJ",
    "gpi_num": 1,
    "gpi_pin": 2,
    "preset": {
      "function": "start_normal",
      "parameters": {
        "break_duration": "0",
        "preroll_time": "4000",
        "program_id": "4"
      },
      "type": "scte104"
    }
  },
  {
    "alta_channel": "EEG_BR_P0BX1U3EL77MKLGJ",
    "gpi_num": 2,
    "gpi_pin": 3,
    "preset": {
      "function": "speaker_change",
      "parameters": {
        "delay_compensation": "0.8"
      },
      "type": "lexi"
    }
  },
  {
    "alta_channel": "EEG_BR_WWO0T2H4Q2B9GSBC",
    "gpi_num": 3,
    "gpi_pin": 4,
    "preset": {
      "function": "start_immediate",
      "parameters": {
        "break_duration": "0",
        "program_id": "2"
      },
      "type": "scte104"
    }
  },
  {
    "alta_channel": "EEG_BR_WWO0T2H4Q2B9GSBC",
    "gpi_num": 4,
    "gpi_pin": 5,
    "preset": {
      "function": "end_normal",
      "parameters": {
        "preroll_time": "5",
        "program_id": "4"
      },
      "type": "scte104"
```

```
  }
},
{
  "alta_channel": "EEG_BR_WWO0T2H4Q2B9GSBC",
  "gpi_num": 5,
  "gpi_pin": 6,
  "preset": {
    "function": "end_immediate",
    "parameters": {
      "program_id": "6"
    },
    "type": "scte104"
  }
},
{
  "alta_channel": "EEG_BR_WWO0T2H4Q2B9GSBC",
  "gpi_num": 6,
  "gpi_pin": 7,
  "preset": {
    "function": "cancel",
    "parameters": {
      "program_id": "7"
    },
    "type": "scte104"
  }
},
{
  "alta_channel": "",
  "gpi_num": 7,
  "gpi_pin": 8,
  "preset": {
    "function": "",
    "parameters": {
    },
    "type": ""
  }
},
{
  "alta_channel": "",
  "gpi_num": 8,
  "gpi_pin": 9,
  "preset": {
    "function": "",
    "parameters": {
    },
    "type": ""
  }
},
{
  "alta_channel": "",
  "gpi_num": 9,
  "gpi_pin": 10,
  "preset": {
    "function": "",
```

```
    "parameters": {
    },
    "type": ""
  }
},
{
  "alta_channel": "",
  "gpi_num": 10,
  "gpi_pin": 11,
  "preset": {
    "function": "",
    "parameters": {
    },
    "type": ""
  }
},
{
  "alta_channel": "",
  "gpi_num": 11,
  "gpi_pin": 12,
  "preset": {
    "function": "",
    "parameters": {
    },
    "type": ""
  }
},
{
  "alta_channel": "",
  "gpi_num": 12,
  "gpi_pin": 13,
  "preset": {
    "function": "",
    "parameters": {
    },
    "type": ""
  }
},
{
  "alta_channel": "",
  "gpi_num": 13,
  "gpi_pin": 14,
  "preset": {
    "function": "",
    "parameters": {
    },
    "type": ""
  }
},
{
  "alta_channel": "",
  "gpi_num": 14,
  "gpi_pin": 15,
  "preset": {
```

```
    "function": "",
    "parameters": {
    },
    "type": ""
  }
},
{
  "alta_channel": "",
  "gpi_num": 15,
  "gpi_pin": 16,
  "preset": {
    "function": "",
    "parameters": {
    },
    "type": ""
  }
},
{
  "alta_channel": "",
  "gpi_num": 16,
  "gpi_pin": 17,
  "preset": {
    "function": "",
    "parameters": {
    },
    "type": ""
  }
},
{
  "alta_channel": "",
  "gpi_num": 17,
  "gpi_pin": 18,
  "preset": {
    "function": "",
    "parameters": {
    },
    "type": ""
  }
},
{
  "alta_channel": "",
  "gpi_num": 18,
  "gpi_pin": 19,
  "preset": {
    "function": "",
    "parameters": {
    },
    "type": ""
  }
},
{
  "alta_channel": "",
  "gpi_num": 19,
  "gpi_pin": 20,
```

```
  "preset": {
    "function": "",
    "parameters": {
    },
    "type": ""
  }
},
{
  "alta_channel": "",
  "gpi_num": 20,
  "gpi_pin": 21,
  "preset": {
    "function": "",
    "parameters": {
    },
    "type": ""
  }
},
{
  "alta_channel": "",
  "gpi_num": 21,
  "gpi_pin": 22,
  "preset": {
    "function": "",
    "parameters": {
    },
    "type": ""
  }
},
{
  "alta_channel": "",
  "gpi_num": 22,
  "gpi_pin": 23,
  "preset": {
    "function": "",
    "parameters": {
    },
    "type": ""
  }
},
{
  "alta_channel": "",
  "gpi_num": 23,
  "gpi_pin": 24,
  "preset": {
    "function": "",
    "parameters": {
    },
    "type": ""
  }
},
{
  "alta_channel": "",
  "gpi_num": 24,
```

```
    "gpi_pin": 25,
    "preset": {
      "function": "",
      "parameters": {
      },
      "type": ""
    }
  }
]
```

Example Request:
POST http://username:password@localhost/system/gpi/triggers/JL Cooper eBox/3/trigger

Example Response iCap Alta Encoder:

```
{
  "msg": "GPI 3 preset executed successfully",
  "success": true
}
```

Utils/Helpers

Resource	Description
POST /alta/alta_instances/get_id/**<name>**	*Action* Gets the bridgeID of an instance from the instance's name property. Returns a 404 error if name is not found. *URL parameters* **name** — The name of the Alta instance

Example Request:
POST http://username:password@localhost/alta/alta_instances/get_id/AltaInstance1

Example Response:
```
{ "instanceID": "EEG_BR_3G46Z6VLSV8B01JZ" }
```

Deprecated Endpoints

Resource	Description

POST /alta/alta_instances/**<instanceID>**/terminate	*Action* Legacy endpoint that will both turn off and delete an Alta instance. It is not recommended that new applications use this endpoint. *URL parameters* **instanceID** — The instanceID for the Alta instance to terminate

Example Request:

POST

http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/terminate

Example Response:

```
{}
```

Instance State Conditions

State	Description
RUNNING	The instance is on
STOPPED with code: n	The instance has encountered an unrecoverable error
TERMINATED	The instance is off

HTTP Error Codes

Error Code	Possible Cause
400 Bad Request	Missing or malformed JSON
404 Not Found	Referenced instanceID no longer exists
422 Unprocessable Entity	The request is well-formed but semantically incorrect (e.g., trying to turn off an instance that is already off)
500 Internal Server Error	Server logic issues
503 Service Unavailable	Server connectivity issues

PTP

Resource	Description
GET /ptp/status	*Action* Retrieves the current PTP status.

<table>
<tr>
<td>POST/GET /ptp/settings</td>
<td>
<h3>Action</h3>
Settings for PTP.
<h3>URL parameters</h3>
<table>
<tr><td>ptp_domain</td><td>Domain to use for PTP</td><td>string</td></tr>
<tr><td>ptp_interface</td><td>Network interface to use for PTP operations</td><td>string</td></tr>
<tr><td>ptp_ip_mode</td><td>IP mode for PTP</td><td>string ("hybrid" or "multicast")</td></tr>
<tr><td>ptp_hardware_timestamping</td><td>Enable/disable hardware timestamping for PTP</td><td>string ("y" or "n")</td></tr>
</table>
</td>
</tr>
</table>

Example Request:

GET http://username:password@localhost/ptp/settings

Example Response:

```
{
   "ptp_domain": "127",
   "ptp_hardware_timestamping": "n",
   "ptp_interface": "enp0s8",
   "ptp_ip_mode": "hybrid"
 }
```

NMOS

<table>
<tr><th>Resource</th><th>Description</th></tr>
<tr>
<td>GET /nmos/status</td>
<td>
<h3>Action</h3>
Retrieves the current NMOS status.
</td>
</tr>
<tr>
<td>POST/GET /nmos/settings</td>
<td>
<h3>Action</h3>
Settings for NMOS.
<h3>URL parameters</h3>
<table>
<tr><td>node_description</td><td>Description string used by IS-04</td><td>string</td></tr>
<tr><td>node_enable</td><td>Turn NMOS on/off</td><td>string ("on" or "off")</td></tr>
</table>
</td>
</tr>
</table>

	registration_ip	Static registration server IP:Port	string (format "IP:Port" or leave blank "" if using discovery)

Example Request:

GET http://username:password@localhost/nmos/settings

Example Response:

```
{
   "node_description": "EEG Alta Node",
   "node_enable": "on",
   "registration_ip": ""
 }
```