



SECUROS REST API

Version 10

Programming Guide

SecurOS REST API Programming Guide (PG - EN, build 93 on 30.03.2020).

© Copyright Intelligent Security Systems, 2020.

Printed in US.

Intelligent Security Systems reserves the right to make changes to both this Manual and to the products it describes. System specifications are subject to change without notice. Nothing contained within this Manual is intended as any offer, warranty, promise or contractual condition, and must not be taken as such.

No part of this publication may be reproduced, transmitted, transcribed, stored in a retrieval system or translated into any human or computer language in any form by any means without the express written permission of the copyright holder. Unauthorized copying of this publication may not only infringe copyright but also reduce the ability of Intelligent Security Systems to provide accurate and up-to-date information to both users and operators.

Contents

1 Preface	6
1.1 Scope	6
1.2 Target Audience	6
1.3 Using This Manual	6
1.4 Getting Technical Support	6
1.5 SecurOS Editions Naming Convention	8
1.6 Design Convention	8
1.7 Design Elements	9
2 General Description	10
3 Preparations for Use	11
4 Authorization In SecurOS	13
4.1 Authorization When Using HTTP/HTTPS	13
4.2 Authorization When Using WebSocket	13
5 Accessing Resources	15
6 Working With Cameras	16
6.1 Get Settings And States of All Cameras	16
6.2 Get Settings of Specified Camera	18
6.3 Get State of Specified Camera	19
6.4 Get Connection Settings of All Cameras	20
6.5 Get Connection Settings of Specified Camera	22
6.6 Change Connection Settings of Specified Camera/Rename Camera	23
6.7 Delete Camera	24
6.8 Working With PTZ Camera	25
6.8.1 Rotate/Scale Camera	25
6.8.2 Rotate/Scale Camera Using Absolute Coordinates	26
6.8.3 Get PTZ Camera State	27
6.8.4 Go To Preset	27
6.8.5 Start Patrol	28
6.8.6 Stop Patrol	29
6.9 Camera Recording Control	29
6.9.1 Start Record	30
6.9.2 Start Record and Add Pre-recorded Video from Pre-recording Buffer	30
6.9.3 Stop Record	31
6.10 Working with Cameras of RTSP Server	31
6.10.1 Get Information about RTSP Streams of Camera	32
6.10.2 Get Information about Live Video RTSP Stream of Camera	33
6.10.3 Get Information about Archive Video RTSP Stream of Camera	33
7 Video Archive Export	35
7.1 Create Archive Export Task	35

7.2	Get Information about All Current Archive Export Tasks	36
7.3	Get Information about Specified Archive Export Task.	37
8	Frame Export	39
8.1	Get Frame of Live Video from Specified Camera	39
8.2	Get Frame of Archive Video from Specified Camera	40
8.3	Get Frame of Specified Resolution from Specified Camera	40
9	Working with User Accounts	42
9.1	Get Information about All Departments.	42
9.2	Get Information about Specified Department.	43
9.3	Get Data of All User Accounts.	43
9.4	Get Data of Specified User Account.	44
9.5	Create User Account	45
9.6	Change User Account Settings.	46
9.7	Delete User Account	47
10	Working with Subscriptions	48
10.1	Working via HTTP.	48
10.1.1	Subscribe to event (HTTP)	48
10.1.2	Get Information about All Active Subscriptions (HTTP).	49
10.1.3	Get Information about Specified Subscription (HTTP).	50
10.1.4	Update Subscription (HTTP).	51
10.1.5	Delete Subscription (HTTP).	52
10.2	Working via WebSocket.	53
10.2.1	Common Request	53
10.2.2	Subscribe to Event (WebSocket)	55
10.2.3	Subscribe to State Change (WebSocket).	56
10.2.4	Delete All Subscriptions (WebSocket).	57
10.2.5	Delete Subscription (WebSocket).	57
10.2.6	Update Subscription (WebSocket).	58
10.2.7	WebSocket Server Messages.	58
10.2.7.1	Object State Change Message	58
10.2.7.2	Event Message	59
10.2.7.3	Object Deletion Message.	60
10.2.7.4	Error Message.	60
11	Working with Event Viewer	62
11.1	Event Filtering.	62
11.2	Get Event List for Specified Camera	62
12	Working with Map	64
12.1	Get Settings of All Maps	64
12.2	Get Settings of Specified Map.	66
13	Working with FaceX	68
13.1	Get FaceX Server Settings.	68
14	Working with Macros	70

14.1 Get Settings of All Macros.	70
14.2 Get Settings of Specified Macro.	71
14.3 Run Macro.	71
15 Working with Computers	73
15.1 Get Settings of All Computers	73
15.2 Get Settings of Specified Computer.	75
16 HTTP Status Codes	78
17 Technical Support Information	79

1 Preface

This section contains general information about this document, the means of its design and use, as well as how to get additional technical support for the product.

1.1 Scope

This guide describes SecurOS REST API resources used to obtain system information, configure the system, and manage system objects.

1.2 Target Audience

This guide is intended for developers who integrating external systems with SecurOS.

1.3 Using This Manual

This document is organized in such a way that the user can use both its printed and electronic versions. In the latter case one can use Adobe Reader's Bookmarks feature as well as cross-reference hyperlinks to navigate through content. In several topics this manual refers to other SecurOS manuals (for example, [SecurOS Quick User Guide](#)). One can find these manuals as separate files on the SecurOS installation CD or download them from our website (www.issivs.com).

To get online help (Microsoft HTML Help) just press the **F1** key when running SecurOS in administrative mode. You can get context help for a given object by pressing the **F1** key when its settings window is open.

1.4 Getting Technical Support

If you have any questions after reading this manual, please address them to your system administrator or supervisor.

For any further information you can contact the Intelligent Security Systems Technical Support Team:

Note. To get a quick response to a request use the Technical Support Portal, which [www](http://www.issivs.com) address is listed below.

- **in USA:**

phone: +1 732 855 1111 (Monday to Friday, 8:30am - 6pm EST);

e-mail: support@issivs.com

www: <https://support.issivs.com>

- **in Russia:**

phone: +7 (495) 645 21 21 (Monday to Thursday, 9am - 6pm MST; Friday 9am - 5pm MST);

www: <https://help.iss.ru>

Note. See the <https://help.iss.ru/user/manual> for the Portal User Guide.

- **in Brazil:**

phone: +55 11 2262 2894 (Monday to Friday, 9am - 6pm BRT);

e-mail: suporte@issivs.com

www: <https://support.issivs.com>

- **in Mexico:**

phone: +52 1 551330 0181 (Monday to Friday, 9am - 6pm CDT);

e-mail: supportlatam@issivs.com

www: <https://support.issivs.com>

- **in Colombia/Ecuador:**

phone: +57 300 442 2808 (Monday to Friday, 9am - 6pm COT/ECT);

e-mail: supportlatam@issivs.com

www: <https://support.issivs.com>

- **in Chile:**

phone: +56 9 6573 2993 (Monday to Friday, 9am - 6pm CLT);

e-mail: supportlatam@issivs.com

www: <https://support.issivs.com>

- **in Ukraine:**

phone: +380 (44) 299 08 10 (Monday to Friday, 9am - 6pm EET);

e-mail: supportua@issivs.com

www: <https://support.issivs.com>

- **in Peru/Bolivia:**

phone: +51 997 111 678 (Monday to Friday, 9am - 6pm PET/BOT);

e-mail: supportlatam@issivs.com

www: <https://support.issivs.com>

- **in Argentina:**

phone: +54 91152528779 (Monday to Friday, 9am - 6pm ART);

e-mail: supportlatam@issivs.com

www: <https://support.issivs.com>

To solve problems faster, we recommend preparing the service information described in the **Technical Support Information** Section before addressing the Technical Support Team.

1.5 SecurOS Editions Naming Convention

This document represents a common manual for several editions of the "SecurOS integrated video management platform" that differ in functional capabilities:

- *SecurOS Monitoring & Control Center*;
- *SecurOS Enterprise*;
- *SecurOS Premium*;
- *SecurOS Professional*;
- *SecurOS Xpress*;
- *SecurOS Lite*.

For product designation regardless of its edition the *SecurOS* general term is used in the framework of the given document.

Sections that describe the functionality available for some editions are marked by a special footnote as in the example below:

The functionality is available in the following editions: *SecurOS Monitoring & Control Center*, *SecurOS Enterprise*, *SecurOS Premium*, *SecurOS Professional*, *SecurOS Xpress*, *SecurOS Lite*.

1.6 Design Convention

For representation of various terms and titles the following fonts and formatting tools are used in this document.

Font	Description
bold type	Used in writing workstation names, utilities or screens, windows and dialog boxes as well as the names of their elements (GUI elements).
<i>italic type</i>	Used to mark out the SecurOS objects.
<i>bold italic type</i>	Used to mark out the elements of homogeneous lists.
<code>monospace</code>	Used to mark out macro text and programming code, file names and their paths. Also it is used to specify the necessary options, to mark out values specified by the user from the keyboard (manually).
green	Used to mark out the cross-references within the document and links to the external available ones.

1.7 Design Elements

Warning! Serves to alert the user to information which is necessary for the correct perception of the text set out below. Typically, this information has a warning character.

Note. Note text in topic body.

Additional Information

Used to display additional information. These type of elements contain, for example, the description of options for executing a task or reference to additional literature.



2 General Description

SecurOS REST API is a RESTful web-service that allows you to receive information about the system, configure it or manage system objects. All methods are implemented as web resources, which can be accessed using the HTTP / HTTPS or WebSocket protocols.

3 Preparations for Use

To use the REST API methods to interact with SecurOS perform the following preliminary settings:

- **Configuring SecurOS.**
- **Configuring HTTPS on the Client Side.**
- **Configuring Resource to Export Archive.**

Note. The last two procedures are optional and are necessary if you intend to use the HTTPS protocol and archive export feature.

Configuring SecurOS

To enable REST API capabilities it is required to create *REST API* object in the Object Tree under the *Computer* to which you are going to connect and specify the HTTP/HTTPS and WebSocket ports. For the details see **SecurOS Administration Guide**.

Configuring HTTPS on the Client Side

The following solution is based on a guide: <https://blogs.msdn.microsoft.com/jpsanders/2009/09/29/how-to-walkthrough-using-httpListener-or-http-server-unmanaged-code-c-as-an-ssl-simple-server/>.

To allow using HTTPS it is necessary to make a listener for a certain port with a certain SSL certificate.

To provide this it is necessary to do the following:

1. Get the thumbprint of the SSL certificate.
 2. Link the certificate to a port which is used for REST API.
 3. Create a GUID for the application (required to link certificate).
-
1. Depending on the source of the certificate its thumbprint can be obtained one way or another. If you are running Microsoft IIS then you can do the following:
 - 1.1. Open the Microsoft Management Console (MMC) and add the **Certificates (Local Computer)** snap-in.
 - 1.2. In the **Console Root** left pane click the **Certificates (Local Computer)**.
 - 1.3. Click the **Personal** folder to expand it.
 - 1.4. Click the **Certificates** folder to expand it.
 - 1.5. In the list of the certificates note the **Intended Purposes** heading. Find a certificate that lists **Client Authentication** as an intended purpose.
 - 1.6. Double-click the certificate.
 - 1.7. In the **Certificate** dialog box click the **Details** tab.
 - 1.8. Scroll through the list of fields and click the **Thumbprint** field.

Copy the hexadecimal characters from the box. If this thumbprint is used in code for the **X509FindType**, remove the spaces between the hexadecimal numbers. For example, the thumbprint "a9 09 50 2d d8 2a e4 14 33 e6 f8 38 86 b0 0d 42 77 a3 2a 7b" should be specified as "a909502dd82ae41433e6f83886b00d4277a32a7b" in code.

Also if necessary see [https://msdn.microsoft.com/en-us/library/ms734695\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/ms734695(v=vs.110).aspx).

2. Certificate port should be the same as the port specified in the *REST API* object settings (by default is 8888).
3. To create a GUID it is possible to use any tool, for example, <https://www.guidgenerator.com/online-guid-generator.aspx>.

After finishing all preparations it's necessary to run Windows command line tool (**cmd**) as Administrator and execute the following string:

```
netsh http add sslcert iport=0.0.0.0:8888  
certhash=9526b167879d2050243b67842e7787fef83f9359  
appid={FCB0C645-E745-490F-9E7D-2171E03DC9B0}, where:
```

- **iport** — the 0.0.0.0 must be set as IP address. Value, specified in the SecurOS *REST API* object settings (**HTTP port**) must be set as port number;
- **certhash** — the previously saved thumbprint (without spaces);
- **appid** — previously prepared application GUID.

To check if everything is done right run the following command:

```
netsh http show sslcert
```

In the end of received list there should be newly added certificate. And now it is possible to access SecurOS via REST API using HTTPS protocol.

Configuring Resource to Export Archive

Files exported using REST API methods are saved in a directory specified in the *Archive Converter* object settings. By default, a file prepared in this way is not available for download. To provide an access to this file one must do one of the following actions:

- Open network access to the folder where exported files are located;
- Deploy an HTTP server that will allow to download files;
- Deploy an FTP server which will grant access to the files.

In each of the three cases the link to the file will look different, but it will contain its own constant part. After choosing the file access method, this constant part should be set as the value of the **SharedExportPath** parameter.

For example, exported file must be accessed by the `http://contoso.my.server:8080/converted/<filename>` link. In this case, the constant part will be the `http://contoso.my.server:8080/converted/`. This string must be set as the **SharedExportPath** parameter value.

The parameter should be created in the `HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\ISS\SecurOS\Niss400\Core\RestApi` key of the Windows registry.

4 Authorization In SecurOS

To protect against unauthorized access, the SecurOS REST API integrates the Basic authorization.

Authorization details are described in the following sections:

- **Authorization When Using HTTP/HTTPS.**
- **Authorization When Using WebSocket.**

Note. Access to the REST API resources without authorization is impossible. For authorization use the credentials of any of the existing SecurOS *User Accounts* or Windows Active Directory. Authorization with the *superuser* credentials is impossible.

Warning! When executing requests, the rights of a successfully authorized user to the *REST API* object are not taken into account. At the same time, if this user does not have rights to some SecurOS objects or the current access level to the objects is insufficient, the corresponding requests will fail.

4.1 Authorization When Using HTTP/HTTPS

When using the HTTP protocol, each resource request must include an *Authorization Header* with user data for Basic authorization. An example with JScript authorization is presented below.

Example

```
var url = "http://172.16.1.120:8888/event?id=12"; // Port number from the REST API object settings
var login = "user_login";
var password = "user_password";

function HTTPRequest(request, url, login, password){
    var HTTP = new ActiveXObject("Msxml2.ServerXMLHTTP.6.0");
    HTTP.open("POST", url, false, login, password);
    HTTP.send();
    var response = new HTTP.responseText();
    return response;}
```

4.2 Authorization When Using WebSocket

When using WebSocket protocol authorization in SecurOS is performed in two stages:

1. **Getting Authorization Token for WebSocket Session.**
2. **Establishing an Authorized WebSocket Connection to the Server.**

Getting Authorization Token for WebSocket Session

1. To get authorization token send GET HTTP request to the SecurOS server where the *REST API* object is located:

`http://<host>:<port>/api/v1/ws_auth`, where:

- <host> – DNS name or IP address of the server;
- <port> – port number specified in the *REST API* object settings for WebSocket connection.

2. The server will request authorization parameters. Specify credentials of the SecurOS *User Account* or Windows Active Directory user. If the parameters are specified correctly the server will return the session authorization token:

```
{
  "data": { "token": "83cd26be3d014d2dbe7e9cdba455e47e" },
  "status": "success"
}
```

The token remains valid:

- within 10 minutes after it has been received;
- throughout the session after its authorization;
- within 10 minutes after closing the authorized connection.

Establishing an Authorized WebSocket Connection to the Server

After receiving the token a connection with the specified server is established. To authorize the connection pass the token received in the first message to the server (`ws: // <host>: <websocket_port>`):

```
{
  "type": "auth",
  "token": "83cd26be3d014d2dbe7e9cdba455e47e"
}
```

If the request does not contain errors, then no response is provided. If the message is processed successfully, the client will be able to execute any resource requests. If an incorrect token is sent or it has expired (see above) the server will return the following error:

```
{
  "data": {
    "error": "INVALID_AUTH_TOKEN",
    "message": "Invalid authorization token"
  },
  "type": "error"
}
```

If the client sent the request to an unauthorized connection, the server will return the following error:

```
{
  "data": {
    "error": "UNAUTHORIZED",
    "message": "Unauthorized",
    "request_id": 1
  },
  "type": "error"
}
```

5 Accessing Resources

SecurOS REST API is a RESTful web-service that allows you to receive information about the system, configure it or manage system objects. All methods are implemented as a web resources. Each resource can be accessed by the URL, which is an HTTP link.

URL consists of two parts, in the `<Host:Port><URI>` format, where:

- `Host:Port` — SecurOS REST API server address and port number that the client is accessing (for example, `http://VIDEO_SERVER_I:8888` or `http://localhost:8888`). DNS name of the computer or IP address can be used;
- `URI` — resource identifier (for example, `/api/v1/cameras` or `/api/v1/persons`).

6 Working With Cameras

The following requests are provided for working with the *Cameras*:

- [Get Settings And States of All Cameras.](#)
- [Get Settings of Specified Camera.](#)
- [Get State of Specified Camera.](#)
- [Get Connection Settings of All Cameras.](#)
- [Get Connection Settings of Specified Camera.](#)
- [Change Connection Settings of Specified Camera/Rename Camera.](#)
- [Delete Camera.](#)

See also:

- [Working With PTZ Camera.](#)
- [Camera Recording Control.](#)
- [Working with Cameras of RTSP Server.](#)

6.1 Get Settings And States of All Cameras

Resource ID

`/api/v1/cameras`

Objects Access Rights

Camera — View (👁️) and higher.

Request Parameters

None.

Protocol/Method

HTTP/GET

Request Example

`http://172.16.1.170:8888/api/v1/cameras`

Response Example

```
{  
  "data": [  
    {
```



```

    "id": "3",
    "name": "Camera 3",
    "settings": {
        "PRESET.name.count": "0",
        "PRESET.preset.count": "0",
        "TOUR.name.count": "0",
        "TOUR.tour.count": "0",

... // Some parameters are excluded

        "send_forensic_data": "",
        "sharpness": "",
        "short_exposures": "",
        "telemetry_id": "native",
        "wiper": ""
    },
    "status": {
        "enabled": true,
        "focused": "true",
        "hold_ptz_SLAVE_ID": "",
        "hold_ptz_module_id": "",
        "hold_ptz_priority": "",
        "hold_ptz_user_id": "",
        "priority": "",
        "recorded": "0",
        "state": "DISARMED",
        "unblinded": "true",
        "valid": true
    },
    "type": "CAM"
},
{
    "id": "2",
    "name": "Camera 2",
    "settings": {
        "PRESET.name.count": "0",
        "PRESET.preset.count": "0",
        "TOUR.name.count": "0",
        "TOUR.tour.count": "0",
        "advanced": "{\"_stream2\":0,\"_stream3\":0,\"digest_auth_only\":0,\"edge
\":1,\"edge_skip_last_minutes\":10,\"fisheye_lens\":null,\"fisheye_position
\":0,\"over_rtsp1\":0,\"over_rtsp2\":0,\"over_rtsp3\":0,\"profile1\":null,\"profile2
\":null,\"profile3\":null,\"rtp_network_latency1\":500,\"rtp_network_latency2
\":500,\"rtp_network_latency3\":500,\"speaker\":null,\"speaker_ip
\": \"\", \"speaker_model\": \"Axis Network Speaker\", \"speaker_password
\": \"EJABDJPACKPDAFNH\", \"speaker_type\": \"Axis\", \"speaker_user\": \"\", \"strm_arch
\":0,\"strm_high\":0,\"strm_low\":0,\"strm_md\":0,\"strm_norm\":0,\"tour_type\": \"0
\", \"use_default_video1\":1,\"use_default_video2\":1,\"use_default_video3\":1,\"washing
\":null}",
        "alarm_rec": "1",

... // Some parameters are excluded

        "short_exposures": "",
        "telemetry_id": "",
        "wiper": ""
    },
    "status": {
        "enabled": true,
        "focused": "true",
        "hold_ptz_SLAVE_ID": "",
        "hold_ptz_module_id": "",
        "hold_ptz_priority": "",

```

```

        "hold_ptz_user_id": "",
        "priority": "-1",
        "recorded": "0",
        "state": "DISARMED",
        "unblinded": "true",
        "valid": true
    },
    "type": "CAM"
},
{
    "id": "1",
    "name": "Camera 1",
    "settings": {
        "PRESET.name.count": "0",
        "PRESET.preset.count": "0",

        ... // Some parameters are excluded

        "short_exposures": "",
        "telemetry_id": "",
        "wiper": ""
    },
    "status": {
        "enabled": true,
        "focused": "true",
        "hold_ptz_SLAVE_ID": "",
        "hold_ptz_module_id": "",
        "hold_ptz_priority": "",
        "hold_ptz_user_id": "",
        "priority": "",
        "recorded": "0",
        "state": "DISARMED",
        "unblinded": "true",
        "valid": true
    },
    "type": "CAM"
}
],
"status": "success"
}


```

6.2 Get Settings of Specified Camera

Resource ID

/api/v1/cameras/<cam_id>

Objects Access Rights

Camera — View () and higher.

Request Parameters

cam_id — identifier of the *Camera*.

Protocol/Method

HTTP/GET

Request Example

`http://172.16.1.170:8888/api/v1/cameras/1`

Response Example

```
{
  "data": {
    "id": "1",
    "name": "Camera 1",
    "settings": {
      "PRESET.name.count": "0",
      "PRESET.preset.count": "0",

      ... // Some parameters are excluded


      "short_exposures": "",
      "telemetry_id": "",
      "wiper": ""
    },
    "status": {
      "enabled": true,
      "focused": "true",
      "hold_ptz_SLAVE_ID": "",
      "hold_ptz_module_id": "",
      "hold_ptz_priority": "",
      "hold_ptz_user_id": "",
      "priority": "",
      "recorded": "0",
      "state": "DISARMED",
      "unblinded": "true",
      "valid": true
    },
    "type": "CAM"
  },
  "status": "success"
}
```

6.3 Get State of Specified Camera

Resource ID

`/api/v1/cameras/<cam_id>/status`

Objects Access Rights

Camera — View () and higher.

Request Parameters

cam_id — identifier of the *Camera*.

Protocol/Method

HTTP/GET

Request Example

`http://172.16.1.170:8888/api/v1/cameras/1/status`

Response Example

```
{
  "data": {
    "id": "1",
    "name": "Camera 1",
    "status": {
      "enabled": true,
      "focused": "true",
      "hold_ptz_SLAVE_ID": "",
      "hold_ptz_module_id": "",
      "hold_ptz_priority": "",
      "hold_ptz_user_id": "",
      "priority": "",
      "recorded": "0",
      "state": "DISARMED",
      "unblinded": "true",
      "valid": true
    },
    "type": "CAM"
  },
  "status": "success"
}
```

6.4 Get Connection Settings of All Cameras

Resource ID

`/api/v2/cameras`

Objects Access Rights

Camera — View (👁️) and higher.

Request Parameters

None.

Protocol/Method

HTTP/GET

Request Example

`http://172.16.1.170:8888/api/v2/cameras`

Response Example

```
{
  "data": [
    {
      "id": "2",
      "integrator": "Axis",

```

```

    "login": "root",
    "model": "P3215-V",
    "name": "Camera 2",
    "server": "VIDEO_SERVER",
    "settings": {
      "PRESET.name.count": "0",
      "PRESET.preset.count": "0",
      "TOUR.name.count": "0",
      "TOUR.tour.count": "0",
      "telemetry_id": ""
    },
    "status": {
      "enabled": true,
      "focused": "true",
      "hold_ptz_SLAVE_ID": "",
      "hold_ptz_module_id": "",
      "hold_ptz_priority": "",
      "hold_ptz_user_id": "",
      "priority": "-1",
      "recorded": "0",
      "state": "DISARMED",
      "unblinded": "true",
      "valid": true
    },
    "url": "172.16.20.54"
  },
  {
    "id": "1",
    "integrator": "Virtual",
    "login": "",
    "model": "default",
    "name": "Camera 1",
    "server": "VIDEO_SERVER",
    "settings": {
      "PRESET.name.count": "0",
      "PRESET.preset.count": "0",
      "TOUR.name.count": "0",
      "TOUR.tour.count": "0",
      "telemetry_id": ""
    },
    "status": {
      "enabled": true,
      "focused": "true",
      "hold_ptz_SLAVE_ID": "",
      "hold_ptz_module_id": "",
      "hold_ptz_priority": "",
      "hold_ptz_user_id": "",
      "priority": "",
      "recorded": "0",
      "state": "DISARMED",
      "unblinded": "true",
      "valid": true
    },
    "url": ""
  }
],
"status": "success"
}


```

6.5 Get Connection Settings of Specified Camera

Resource ID

/api/v2/cameras/<cam_id>

Objects Access Rights

Camera — View () and higher.

Request Parameters

cam_id — identifier of the *Camera*.

Protocol/Method

HTTP/GET

Request Example

http://172.16.1.170:8888/api/v2/cameras/1

Response Example

```
{
  "data": {
    "id": "1",
    "integrator": "Virtual",
    "login": "",
    "model": "default",
    "name": "Camera 1",
    "server": "VIDEO_SERVER",
    "settings": {
      "PRESET.name.count": "0",
      "PRESET.preset.count": "0",
      "TOUR.name.count": "0",
      "TOUR.tour.count": "0",
      "telemetry_id": ""
    },
    "status": {
      "enabled": true,
      "focused": "true",
      "hold_ptz_SLAVE_ID": "",
      "hold_ptz_module_id": "",
      "hold_ptz_priority": "",
      "hold_ptz_user_id": "",
      "priority": "",
      "recorded": "0",
      "state": "DISARMED",
      "unblinded": "true",
      "valid": true
    },
    "url": ""
  },
  "status": "success"
}
```

6.6 Change Connection Settings of Specified Camera/Rename Camera

Resource ID

/api/v2/cameras/<cam_id>

Objects Access Rights

Camera — Configure (🔧) and higher.

Request Parameters

cam_id — identifier of the *Camera*.

Protocol/Method

HTTP/PUT

Required Header

Content-Type: application/json

Request Body

The new *Camera* connection setting or the new *Camera* name must be passed in the request. If parameter is not specified in the request it will keep its current value.

Request Body Example

```
{
  "integrator" : "Panasonic",
  "model" : "WV-SF342",
  "login" : "admin",
  "password": "admin",
  "name" : "New_camera_name",
  "url" : "172.16.1.170"
}
```

Request Example

http://172.16.1.170:8888/api/v2/cameras/7

Response Example

```
{
  "data": {
    "id": "7",
    "integrator": "Panasonic",
    "login": "admin",
    "model": "WV-SF342",
    "name": "New_camera_name",
    "server": "VIDEO_SERVER_II",
    "settings": {
      "PRESET.name.count": "0",

```


```
    "PRESET.preset.count": "0",
    "TOUR.name.count": "0",
    "TOUR.tour.count": "0",
    "telemetry_id": ""
  },
  "status": {
    "enabled": true,
    "focused": "true",
    "hold_ptz_SLAVE_ID": "",
    "hold_ptz_module_id": "",
    "hold_ptz_priority": "",
    "hold_ptz_user_id": "",
    "priority": "",
    "recorded": "0",
    "state": "DISARMED",
    "unblinded": "true",
    "valid": true
  },
  "url": "172.16.1.170"
},
"status": "success"
}
```

6.7 Delete Camera

Resource ID

/api/v2/cameras/<cam_id>

Objects Access Rights

Camera — Full access ()

Request Parameters

cam_id — identifier of the *Camera*.

Protocol/Method

HTTP/DELETE

Request Example

http://172.16.1.170:8888/api/v2/cameras/5

Response Example

```
{
  "status": "success"
}
```

Warning! If the *Camera* that is being deleted is the only one child of the *Video Capture Device*, then this *Video Capture Device* will also be deleted along with the *Camera*.

6.8 Working With PTZ Camera

The following requests are provided for working with the *Camera's* PTZ device:

- **Rotate/Scale Camera.**
- **Rotate/Scale Camera Using Absolute Coordinates.**
- **Get PTZ Camera State.**
- **Go To Preset.**
- **Start Patrol.**
- **Stop Patrol.**

6.8.1 Rotate/Scale Camera

Resource ID

```
/api/v1/cameras/<cam_id>/ptz/move?pan_speed=<value1>&tilt_speed=<value2>  
&zoom_speed=<value3>&duration=<value4>
```

or

```
/api/v2/cameras/<cam_id>/ptz/move?pan_speed=<value1>&tilt_speed=<value2>  
&zoom_speed=<value3>&duration=<value4>
```

Objects Access Rights

Camera — Control () and higher.

Request Parameters

- **cam_id** — identifier of the controlled *Camera*;
- **pan_speed** — horizontal rotation speed. Possible values: [-10; 10] (on the left; on the right);
- **tilt_speed** — vertical rotation speed. Possible values: [-10; 10] (up; down);
- **zoom_speed** — scaling speed. Possible values: [-10; 10] (zoom out; zoom in);
- **duration** — time after which the movement in the specified directions and scaling will be stopped (in milliseconds).

Protocol/Method

HTTP/POST

Request Example

```
http://172.16.1.170:8888/api/v1/cameras/8/ptz/move?pan_speed=4&tilt_speed=0  
&zoom_speed=0
```

Warning! If the **duration** is not set then rotation in specified direction and scaling is carried out until:

1. A command with zero parameter values is received (pan_speed=0, tilt_speed=0 and zoom_speed=0).
2. Thresholds of the camera tilt, pan or zoom are reached. In this case, the camera will be stopped automatically.

Response Example

```
{  
  "status": "success"  
}
```

6.8.2 Rotate/Scale Camera Using Absolute Coordinates

Note. The resource is only available for some camera models.

Resource ID

```
/api/v1/cameras/<cam_id>/ptz/move_abs?pan_position=<value1>  
&tilt_position=<value2>&zoom_position=<value3>
```

or

```
/api/v2/cameras/<cam_id>/ptz/move_abs?pan_position=<value1>  
&tilt_position=<value2>&zoom_position=<value3>
```

Objects Access Rights

Camera — Control () and higher.

Request Parameters

- **cam_id** — identifier of the controlled *Camera*;
- **pan_position** — horizontal coordinates. Possible values: [-10800; 10800] (on the left; on the right);
- **pan_position** — vertical coordinates. Possible values: [-5400; 5400] (up; down);
- **zoom_position** — scaling. Possible values: [0; 1000] (zoom out; zoom in).

Protocol/Method

HTTP/POST

Request Example

```
http://172.16.1.170:8888/api/v1/cameras/8/ptz/move_abs?pan_position=7250  
&tilt_position=5000&zoom_position=240
```

Response Example

```
{  
  "status": "success"
```

```
}
```

6.8.3 Get PTZ Camera State

Note. The resource is only available for some camera models.


Resource ID

```
/api/v1/cameras/<cam_id>/ptz/status?timeoutMs=<value>
```

or

```
/api/v2/cameras/<cam_id>/ptz/status?timeoutMs=<value>
```

Objects Access Rights

Camera — View () and higher.

Request Parameters

- **cam_id** — identifier of the controlled *Camera*;
- **timeoutMs** — server response timeout, in milliseconds. Optional parameter. If not specified, will be set to 1000. If specified value is less than 20 will be set to 20.

Protocol/Method

HTTP/GET

Request Example

```
http://172.16.1.170:8888/api/v1/cameras/8/ptz/status?timeoutMs=500
```

Response Example

```
{
  "data": {
    "pan_position": "-3242",
    "pantilt_status": "IDLE",
    "tilt_position": "-2272",
    "zoom_position": "1000",
    "zoom_status": "IDLE"
  },
  "status": "success"
}
```

6.8.4 Go To Preset


Resource ID

```
/api/v1/cameras/<cam_id>/ptz/presets/recall?preset=<preset_id>
&pt_speed=<value>
```

or

```
/api/v2/cameras/<cam_id>/ptz/presets/recall?preset=<preset_id>
&pt_speed=<value>
```

Objects Access Rights

Camera – Control () and higher.

Request Parameters

- **cam_id** – identifier of the controlled *Camera*;
- **preset** – identifier of the *Preset* on the *Camera*;
- **pt_speed** – speed of the *Camera* move. Possible values: [0 ; 10].

Note. The **pt_speed** parameter is supported only by some camera models.

Protocol/Method

HTTP/POST

Request Example

```
http://172.16.1.170:8888/api/v2/cameras/8/ptz/presets/recall?preset=3
&pt_speed=7
```

Response Example

```
{
  "status": "success"
}
```

6.8.5 Start Patrol


Resource ID

```
/api/v1/cameras/<cam_id>/ptz/patrols/play?patrol=<patrol_id>
```

or

```
/api/v2/cameras/<cam_id>/ptz/patrols/play?patrol=<patrol_id>
```

Objects Access Rights

Camera – Control () and higher.

Request Parameters

- **cam_id** – identifier of the controlled *Camera*;
- **patrol** – identifier of the *Tour* on the *Camera*.

Protocol/Method

HTTP/POST

Request Example

```
http://172.16.1.170:8888/api/v2/cameras/8/ptz/patrols/play?patrol=1
```

Response Example

```
{  
  "status": "success"  
}
```

6.8.6 Stop Patrol


Resource ID

```
/api/v1/cameras/<cam_id>/ptz/patrols/stop
```

or

```
/api/v2/cameras/<cam_id>/ptz/patrols/stop
```

Objects Access Rights

Camera — Control () and higher.

Request Parameters

cam_id — identifier of the controlled *Camera*.

Protocol/Method

HTTP/POST

Request Example

```
http://172.16.1.170:8888/api/v2/cameras/8/ptz/patrols/stop
```

Response Example

```
{  
  "status": "success"  
}
```

6.9 Camera Recording Control

The following requests are provided for *Camera* recording control:

- **Start Record.**
- **Start Record and Add Pre-recorded Video from Pre-recording Buffer.**
- **Stop Record.**

6.9.1 Start Record


Resource ID

/api/v1/cameras/<cam_id>/recording/rec

or

/api/v2/cameras/<cam_id>/recording/rec

Objects Access Rights

Camera – Control () and higher.

Request Parameters

cam_id – identifier of the *Camera*.

Protocol/Method

HTTP/POST

Request Example

http://172.16.1.170:8888/api/v2/cameras/1/recording/rec

Response Example

```
{
  "status": "success"
}
```

6.9.2 Start Record and Add Pre-recorded Video from Pre-recording Buffer


Resource ID

/api/v1/cameras/<cam_id>/recording/rec_rollback?rollback_time_abs=<time>

or

/api/v2/cameras/<cam_id>/recording/rec_rollback?rollback_time_abs=<time>

Objects Access Rights

Camera – Control () and higher.

Request Parameters

- **cam_id** – identifier of the *Camera*.
- **rollback_time_abs** – absolute time on the time line for the current date in the HH:MM:SS.FFF (milliseconds are optional). For more information about the parameter, see the [SecurOS Programming Guide](#).

Protocol/Method

HTTP/POST

Request Example

```
http://172.16.1.170:8888/api/v2/cameras/1/recording/rec_rollback  
?rollback_time_abs=10:08:14
```

Response Example

```
{  
  "status": "success"  
}
```

6.9.3 Stop Record


Resource ID

/api/v1/cameras/<cam_id>/recording/rec_stop

or

/api/v2/cameras/<cam_id>/recording/rec_stop

Objects Access Rights

Camera — Control () and higher.

Request Parameters

cam_id — identifier of the *Camera*.

Protocol/Method

HTTP/POST

Request Example

```
http://172.16.1.170:8888/api/v2/cameras/1/recording/rec_stop
```

Response Example

```
{  
  "status": "success"  
}
```

6.10 Working with Cameras of RTSP Server

The following requests are provided for working with the *Cameras* of the RTSP Server:

- **Get Information about RTSP Streams of Camera.**
- **Get Information about Live Video RTSP Stream of Camera.**

- **Get Information about Archive Video RTSP Stream of Camera.**

Note. For the details on the RTSP server, RTSP camera streams, and RTSP requests, see the [SecurOS Administration Guide](#).

6.10.1 Get Information about RTSP Streams of Camera


Resource ID

/api/v1/cameras/<cam_id>/rtsp

or

/api/v2/cameras/<cam_id>/rtsp

Objects Access Rights

Camera — View () and higher.

Request Parameters

cam_id — identifier of the *Camera*.

Protocol/Method

HTTP/GET

Request Example

http://172.16.1.170:8888/api/v2/cameras/4/rtsp

Response Example

```
{
  "data": {
    "id": "4",
    "rtsp": [
      {
        "type": "live",
        "url": "rtsp://172.16.1.170:554/live?id=4"
      },
      {
        "type": "archive",
        "url": "rtsp://172.16.1.170:554/archive?id=4"
      }
    ],
    "type": "CAM"
  },
  "status": "success"
}
```

The following parameters are returned in the response:

- **rtsp.type** — type of the *Camera*'s RTSP stream;

- **rtsp.url** — URL to receive the *Camera*'s RTSP stream.

6.10.2 Get Information about Live Video RTSP Stream of Camera


Resource ID

/api/v1/cameras/<cam_id>/rtsp/live

or

/api/v2/cameras/<cam_id>/rtsp/live

Objects Access Rights

Camera — View () and higher.

Request Parameters

cam_id — identifier of the *Camera*.

Protocol/Method

HTTP/GET

Request Example

http://172.16.1.170:8888/api/v2/cameras/4/rtsp/live

Response Example

```
{
  "data": {
    "id": "4",
    "rtsp": [
      {
        "type": "live",
        "url": "rtsp://172.16.1.170:554/live?id=4"
      }
    ],
    "type": "CAM"
  },
  "status": "success"
}
```

The following parameters are returned in the response:

- **rtsp.type** — type of the *Camera*'s RTSP stream;
- **rtsp.url** — URL to receive the live *Camera*'s RTSP stream.

6.10.3 Get Information about Archive Video RTSP Stream of Camera


Resource ID

/api/v1/cameras/<cam_id>/rtsp/archive

or

```
/api/v2/cameras/<cam_id>/rtsp/archive
```

Objects Access Rights

Camera — View () and higher.

Request Parameters

cam_id — identifier of the *Camera*.

Protocol/Method

HTTP/GET

Request Example

```
http://172.16.1.170:8888/api/v2/cameras/4/rtsp/archive
```

Response Example

```
{
  "data": {
    "id": "4",
    "rtsp": [
      {
        "type": "archive",
        "url": "rtsp://172.16.1.170:554/archive?id=4"
      }
    ],
    "type": "CAM"
  },
  "status": "success"
}
```

The following parameters are returned in the response:

- **rtsp.type** — type of the *Camera*'s RTSP stream;
- **rtsp.url** — URL to receive the archive *Camera*'s RTSP stream.

7 Video Archive Export

The following requests are provided for working with the video archive export tasks:

- **Create Archive Export Task.**
- **Get Information about All Current Archive Export Tasks.**
- **Get Information about Specified Archive Export Task.**

7.1 Create Archive Export Task

Note. The task will be performed by that *Archive Converter* of this *Video Server*, which has the lowest identifier.

Resource ID

`/api/v1/export/tasks`

Objects Access Rights

Not required.

Request Parameters

None.

Protocol/Method

HTTP/POST

Required Header

Content-Type: application/json

Request Body

The following parameters are passed in the request body:

- **cam_id** – identifier of the *Camera* whose archive fragment needs to be exported;
- **from, to** – left and right border of the requested time interval, in the YYYYMMDDTHHMMSS[.SSS] format (milliseconds are optional).

Request Body Example

```
{  
  "camera": "2",  
  "from": "20200219T090440.000",  
  "to": "20200219T092040.000"
```

```
}
```

Request Example

```
http://172.16.1.170:8888/api/v1/export/tasks
```

Response Example

```
{
  "data": {
    "camera": "2",
    "create_time": "2020-02-20T11:59:48.887",
    "from": "2020-02-19T09:04:40.000",
    "id": "412dc56f-73f7-4512-9a5a-c9d72a126a72",
    "state": "Registered",
    "to": "2020-02-19T09:20:40.000"
  },
  "status": "success"
}
```

7.2 Get Information about All Current Archive Export Tasks

Resource ID

```
/api/v1/export/tasks
```

Objects Access Rights

Not required.

Request Parameters

None.

Protocol/Method

HTTP/GET

Request Example

```
http://172.16.1.170:8888/api/v1/export/tasks
```

Response Example

```
{
  "data": [
    {
      "camera": "3",
      "create_time": "2020-02-20T12:01:04.110",
      "from": "2020-02-19T09:04:40.000",
      "id": "2f5a37a9-684f-4377-b43e-81ba13be8e25",
      "state": "Registered",
      "to": "2020-02-19T10:20:40.000"
    },
    {
      "actual_archive_start_time": "2020-02-19T09:04:40.009",
      "actual_archive_stop_time": "2020-02-19T09:20:39.971",

```

```

    "camera": "1",
    "create_time": "2020-02-20T11:59:48.887",
    "from": "2020-02-19T09:04:40.000",
    "id": "412dc56f-73f7-4512-9a5a-c9d72a126a72",
    "operation": "Converting archive",
    "percents": 100,
    "state": "Finished",
    "to": "2020-02-19T09:20:40.000",
    "total_progress": 100,
    "url": "\\412dc56f-73f7-4512-9a5a-c9d72a126a72.ASF"
  },
  {
    "camera": "2",
    "create_time": "2020-02-20T12:12:47 AM.581",
    "from": "2020-02-19T09:04:40.000",
    "id": "4db57466-f80a-4632-b693-3117542cb54f",
    "operation": "Converting archive",
    "percents": 39,
    "state": "Started",
    "to": "2020-02-19T10:00:40.000",
    "total_progress": 51
  },
  {
    "camera": "5",
    "create_time": "2020-02-20T12:12:22 PM.274",
    "from": "2020-02-19T09:04:40.000",
    "id": "ba26860f-03e4-4af6-8e09-99d5574dd372",
    "state": "Registered",
    "to": "2020-02-19T10:20:40.000"
  }
],
"status": "success"
}

```

Possible Export Tasks States

Code	Description
Registered	Task created
Enqueued	Task added to the <i>Archive Converter</i> task queue
Started	<i>Archive Converter</i> has started the task
Finished	Task finished successfully
Failed	Task failed
Cancelled	Task canceled by user

7.3 Get Information about Specified Archive Export Task

Resource ID

/api/v1/export/tasks/<export_task_id>

Objects Access Rights

Not required.

Request Parameters

export_task_id — identifier of the export task assigned when it has been created.

Protocol/Method

HTTP/GET

Request Example

```
http://172.16.1.170:8888/api/v1/export/tasks/  
ba26860f-03e4-4af6-8e09-99d5574dd372
```

Response Example

```
{  
  "data": {  
    "camera": "5",  
    "create_time": "2020-02-20T12:12:22 PM.274",  
    "error": "LoadingError",  
    "from": "2020-02-19T09:04:40.000",  
    "id": "ba26860f-03e4-4af6-8e09-99d5574dd372",  
    "problem": "It's impossible to receive archive (no fragments on servers), hosts:  
'VIDEO_SERVER_II, '",  
    "state": "Failed",  
    "to": "2020-02-19T10:20:40.000"  
  },  
  "status": "success"  
}
```

8 Frame Export

The following requests allow to export frames:

- **Get Frame of Live Video from Specified Camera.**
- **Get Frame of Archive Video from Specified Camera.**
- **Get Frame of Specified Resolution from Specified Camera.**

Warning! To export frames the *Image Processor* object must be created in SecurOS (see [SecurOS Administration Guide](#)). Otherwise, request will fail.

8.1 Get Frame of Live Video from Specified Camera

Resource ID

`/api/v2/cameras/<cam_id>/image`

Objects Access Rights

Camera — View (👁️) and higher.

Request Parameters

cam_id — identifier of the *Camera*.

Protocol/Method

HTTP/GET

Request Example

`http://172.16.1.170:8888/api/v2/cameras/1/image`

Image is sent to the REST API client in the response body and further can be saved to a file. The examples of the request headers and response body are presented below. Received image is conventionally designated as [DATA].

Response Example

```
Server: Microsoft-HTTPAPI/2.0
Cache-Control: no-store, must-revalidate
Content-Type: image/jpeg
Transfer-Encoding: chunked
Date: Fri, 21 Feb 2020 07:02:07 GMT
```

[DATA]

8.2 Get Frame of Archive Video from Specified Camera

Resource ID

/api/v2/cameras/<cam_id>/image/<frame_time>

Objects Access Rights

Camera – View (👁) and higher.

Request Parameters

- **cam_id** – identifier of the *Camera* whose archive frame needs to be exported;
- **frame_time** – frame timestamp in one of the following formats (milliseconds are optional):
 - YYYYMMDD^THHMMSS[.SSS];
 - YYYY-MM-DD^THH:MM:SS[.SSS];
 - YYYY-MM-DD%20HH:MM:SS[.SSS].

Protocol/Method

HTTP/GET

Request Example

http://172.16.1.170:8888/api/v2/cameras/2/image/20200219T091259

Image is sent to the REST API client in the response body and further can be saved to a file. The examples of the request headers and response body are presented below. Received image is conventionally designated as [DATA].

Response Example

```
Server: Microsoft-HTTPAPI/2.0
Cache-Control: no-store, must-revalidate
Content-Type: image/jpeg
Transfer-Encoding: chunked
Date: Fri, 21 Feb 2020 07:02:07 GMT
```

[DATA]

8.3 Get Frame of Specified Resolution from Specified Camera

Resource ID

/api/v2/cameras/<cam_id>/image?scale_y=<pixels>&scale_x=<pixels> - for the live video

or


```
/api/v2/cameras/<cam_id>/image/<frame_time>?scale_y=<pixels>
&scale_x=<pixels> - for the archive video
```

Objects Access Rights

Camera – View (👁) and higher.

Request Parameters

- **cam_id** – identifier of the *Camera* whose frame needs to be exported;
- **scale_y** – vertical image size (in pixels);
- **scale_x** – horizontal image size (in pixels);
- **frame_time** – frame timestamp in one of the following formats (milliseconds are optional):
 - YYYYMMDD**T**HHMMSS[.SSS];
 - YYYY-MM-DD**T**HH:MM:SS[.SSS];
 - YYYY-MM-DD%20HH:MM:SS[.SSS].

Protocol/Method

HTTP/GET

Request Example

```
http://172.16.1.170:8888/api/v2/cameras/2/image/20200219T091259?scale_y=400
&scale_x=300
```

Image is sent to the REST API client in the response body and further can be saved to a file. The examples of the request headers and response body are presented below. Received image is conventionally designated as [DATA].

Response Example

```
Server: Microsoft-HTTPAPI/2.0
Cache-Control: no-store, must-revalidate
Content-Type: image/jpeg
Transfer-Encoding: chunked
Date: Fri, 21 Feb 2020 07:02:07 GMT
```

[DATA]

9 Working with User Accounts

The following requests are provided for working with SecurOS *User Accounts*:

- **Get Information about All Departments.**
- **Get Information About Specified Department.**
- **Get Data of All User Accounts.**
- **Get Data of Specified User Account.**
- **Create User Account.**
- **Change User Account Settings.**
- **Delete User Account.**

9.1 Get Information about All Departments

Resource ID

/api/v1/departments

Objects Access Rights

Not required.

Request Parameters

None.

Protocol/Method

HTTP/GET

Request Example

http://172.16.1.170:8888/api/v1/departments

Response Example

```
{
  "data": [
    {
      "id": "1.1",
      "name": "Department 1"
    },
    {
      "id": "1.2",
      "name": "Department 2"
    }
  ]
}
```

```
  "status": "success"
}
```

9.2 Get Information about Specified Department

Resource ID

/api/v1/departments/<department_id>

Objects Access Rights

Not required.

Request Parameters

department_id — identifier of the *Department*.

Protocol/Method

HTTP/GET

Request Example

http://172.16.1.170:8888/api/v1/departments/1.2

Response Example

```
{
  "data": {
    "id": "1.2",
    "name": "Department 2"
  },
  "status": "success"
}
```

9.3 Get Data of All User Accounts

Resource ID

/api/v1/persons

Objects Access Rights

Not required.

Request Parameters

None.

Protocol/Method

HTTP/GET

Request Example

`http://172.16.1.170:8888/api/v1/persons`

Response Example

```
{
  "data": [
    {
      "comment": "",
      "department_id": "1.1",
      "email": "a@b.c",
      "id": "1.1",
      "name": "Operator_1",
      "phone": "55555555",
      "user_rights_id": "1.3"
    },
    {
      "comment": "Timetable: Wed-Fri from 9-00 AM till 4-00 PM",
      "department_id": "1.1",
      "email": "a@b.c",
      "id": "1.2",
      "name": "Operator_2",
      "phone": "55555555",
      "user_rights_id": "1.3"
    }
  ],
  "status": "success"
}
```

9.4 Get Data of Specified User Account

Resource ID

`/api/v1/persons/<person_id>`

Objects Access Rights

Not required.

Request Parameters

person_id — identifier of the *User Account*.

Protocol/Method

HTTP/GET

Request Example

`http://172.16.1.170:8888/api/v1/persons/1.1`

Response Example

```
{
  "data": {
```

```
    "comment": "",
    "department_id": "1.1",
    "email": "a@b.c",
    "id": "1.1",
    "name": "Operator_1",
    "phone": "5555555",
    "user_rights_id": "1.3"
  },
  "status": "success"
}
```

9.5 Create User Account

Resource ID

/api/v1/persons

Objects Access Rights

- **Department** — Full access (🔑);
- **User Rights** — Configure (⚙️) and higher.

Protocol/Method

HTTP/POST

Request Parameters

None.

Required Header

Content-Type: application/json

Request Body

In the request body one must pass the credentials of a new SecurOS *User Account*, identifier of the parent *Department* and identifier of the assigned *User Rights*. If parameter is not specified in the request it will be empty.

Request Body Example

```
{
  "department_id": "1.2",
  "name": "Operator 10",
  "password": "pwd",
  "phone": "88003222334",
  "email": "email@mail",
  "comment": "custom_comment",
  "user_rights_id": "1.3"
}
```

Warning! department_id, name and user_rights_id parameters are mandatory!

Request Example

`http://172.16.1.170:8888/api/v1/persons`

Response Example

```
{
  "data": {
    "comment": "custom_comment",
    "department_id": "1.2",
    "email": "email@mail",
    "id": "8",
    "name": "Operator 10",
    "phone": "88003222334",
    "user_rights_id": "1.3"
  },
  "status": "success"
}
```

9.6 Change User Account Settings

Resource ID

`/api/v1/persons/<person_id>`

Objects Access Rights

Department – Configure (⚙️) and higher.

Request Parameters

person_id – identifier of the *User Account*.

Protocol/Method

HTTP/PUT

Required Header

Content-Type: application/json

Request Body

In the request body one must pass new credentials of the SecurOS *User Account* and identifier of the new *User Rights*. If parameter is not specified in the request it will keep its current value.

Request Body Example

Content-Type: application/json

```
{
  "comment": "new_custom_comment",
  "email": "e@mail",
  "name": "Operator 5",
  "phone": "88003222334",
}
```

```
"password": "15",  
"user_rights_id": "1.2"  
}
```

Warning! department_id and user_rights_id parameters are mandatory!

Request Example

http://172.16.1.170:8888/api/v1/persons/1.5

Response Example

```
{  
  "data": {  
    "comment": "new_custom_comment",  
    "department_id": "1.2",  
    "email": "e@mail",  
    "id": "1.5",  
    "name": "Operator 5",  
    "phone": "88003222334",  
    "user_rights_id": "1.2"  
  },  
  "status": "success"  
}
```

9.7 Delete User Account

Resource ID

/api/v1/persons/<person_id>

Objects Access Rights

Department — Full control (🔑).

Request Parameters

person_id — identifier of the *User Account*.

Protocol/Method

HTTP/DELETE

Request Example

http://172.16.1.170:8888/api/v1/persons/1.4

Response Example

```
{  
  "status": "success"  
}
```

10 Working with Subscriptions

One can work with subscriptions via HTTP or WebSocket protocols:

- [Working via HTTP](#).
- [Working via WebSocket](#).

10.1 Working via HTTP

The following requests are provided for working with event subscriptions via HTTP:

- [Subscribe to event \(HTTP\)](#).
- [Get Information about All Active Subscriptions \(HTTP\)](#).
- [Get Information about Specified Subscription \(HTTP\)](#).
- [Update Subscription \(HTTP\)](#).
- [Delete Subscription \(HTTP\)](#).

10.1.1 Subscribe to event (HTTP)

Resource ID

`/api/v1/events/subscriptions`

Objects Access Rights

Not required.

Request Parameters

None.

Protocol/Method

HTTP/POST

Required Header

Content-Type: application/json

Request Body

To subscribe to the specified events of the specified objects the subscription filter must be passed in the request body. If the filter is created with no parameters then client will be subscribed to all events of all SecurOS objects.

Request Body Example

```
{
  "callback": "http://client_app_address",
  "filter": {
    "action": "ALARM",
    "id": "4",
    "type": "CAM"
  }
}
```

The following parameters are passed in the request body:

- **callback** — address of the service where notifications about events will be sent to. Mandatory parameter;
- **filter.action** — type of the event for which the subscription is created. Optional parameter. If not set the notifications about all events of the specified object(s) will be sent.
- **filter.id** — identifier of the object for which the subscription is created. Optional parameter. If not set the notifications about all events of all objects of the specified type will be sent.
- **filter.type** — type of the object for events of which the subscription is created. Optional parameter. If not set the notifications about all events of all SecurOS objects will be sent.

Request Example

```
http://172.16.1.170:8888/api/v1/events/subscriptions
```

Response Example

```
{
  "data": {
    "callback": "http://client_app_address",
    "filter": {
      "action": "ALARM",
      "id": "4",
      "type": "CAM"
    },
    "id": "1f2e91b7-9a97-4838-b39a-f424c1a04c57"
  },
  "status": "success"
}
```

Response returns the identifier of the subscription.

10.1.2 Get Information about All Active Subscriptions (HTTP)

Resource ID

```
/api/v1/events/subscriptions
```

Objects Access Rights

Not required.

Request Parameters

None.

Protocol/Method

HTTP/GET

Request Example

http://172.16.1.170:8888/api/v1/events/subscriptions

Response Example

```
{
  "data": {
    "callback": "http://client_app_address",
    "filter": {
      "action": "START_REC",
      "id": "3",
      "type": "CAM"
    },
    "id": "9d4b2852-36c4-4476-9992-af87107a3a02"
  },
  "status": "success"
} {
  "data": [
    {
      "callback": "http://client_app_address",
      "filter": {
        "action": "ALARM",
        "id": "4",
        "type": "CAM"
      },
      "id": "1f2e91b7-9a97-4838-b39a-f424c1a04c57"
    },
    {
      "callback": "http://client_app_address",
      "filter": {
        "action": "ARM",
        "id": "3",
        "type": "CAM"
      },
      "id": "d7c6c84d-c2d9-4cee-958b-9187335f36ea"
    }
  ],
  "status": "success"
}
```

The following parameters are returned in the response:

- **callback** — address of the service where notifications about events will be sent to;
- **filter.action** — type of the event for which there is a subscription;
- **filter.id** — identifier of the object for which there is a subscription;
- **filter.type** — type of the object for events of which there is a subscription;
- **subscription_id** — identifier of the subscription.

10.1.3 Get Information about Specified Subscription (HTTP)

Resource ID

/api/v1/events/subscriptions/<subscription_id>

Objects Access Rights

Not required.

Request Parameters

subscription_id — identifier of the subscription.

Protocol/Method

HTTP/GET

Request Example

```
http://172.16.1.170:8888/api/v1/events/subscriptions/  
8382fe09-7312-438b-8350-1884b0dc68bc
```

Response Example

```
{  
  "data": {  
    "callback": "http://localhost",  
    "filter": {  
      "action": "ALARM",  
      "id": "4",  
      "type": "CAM"  
    },  
    "id": "8382fe09-7312-438b-8350-1884b0dc68bc"  
  },  
  "status": "success"  
}
```

The following parameters are returned in the response:

- **callback** — address of the service where notifications about events will be sent to;
- **filter.action** — type of the event for which there is a subscription;
- **filter.id** — identifier of the object for which there is a subscription;
- **filter.type** — type of the object for events of which there is a subscription;
- **subscription_id** — identifier of the subscription.

10.1.4 Update Subscription (HTTP)

Resource ID

/api/v1/events/subscriptions/<subscription_id>

Request Parameters

subscription_id — identifier of the subscription.

Protocol/Method

HTTP/PUT

Required Header

Content-Type: application/json

Request Body

The new parameters of the subscription filter must be passed in the request body.

Request Body Example

Content-Type: application/json

```
{
  "callback": "http://client_app_address",
  "filter": {
    "action": "ALARM",
    "id": "4",
    "type": "CAM"
  }
}
```

Request Example

http://172.16.1.170:8888/api/v1/events/subscriptions/
d7c6c84d-c2d9-4cee-958b-9187335f36ea

Response Example

```
{
  "data": {
    "callback": "http://client_app_address",
    "filter": {
      "action": "ALARM",
      "id": "4",
      "type": "CAM"
    },
    "id": "1f2e91b7-9a97-4838-b39a-f424c1a04c57"
  },
  "status": "success"
}
```

10.1.5 Delete Subscription (HTTP)

Resource ID

/api/v1/events/subscriptions/<subscription_id>

Objects Access Rights

Not required.

Request Parameters

subscription_id — identifier of the subscription.

Protocol/Method

HTTP/DELETE

Request Example

```
http://172.16.1.170:8888//api/v1/events/subscriptions/  
1f2e91b7-9a97-4838-b39a-f424c1a04c57
```

Response Example

```
{  
  "data": {  
    "callback": "http://client_app_address",  
    "filter": {  
      "action": "ALARM",  
      "id": "4",  
      "type": "CAM"  
    },  
    "id": "1f2e91b7-9a97-4838-b39a-f424c1a04c57"  
  },  
  "status": "success"  
}
```

10.2 Working via WebSocket

The following requests are provided for working with subscriptions via WebSocket:

- **Common Request.**
- **Subscribe to Event (WebSocket).**
- **Subscribe to State Change (WebSocket).**
- **Delete All Subscriptions (WebSocket).**
- **Delete Subscription (WebSocket).**
- **Update Subscription (WebSocket).**

Warning! The authorization is required before executing requests (see [Authorization When Using WebSocket](#)).

10.2.1 Common Request

When using the WebSocket protocol, all operations with subscriptions (subscribe to an event / change of state and delete a subscription) can be performed in one request (see the example below). Examples of separate operations with the subscriptions are discussed in the relevant sections.

Request Example

```
{  
  "type": "subscribe",  
  "id": 12345,  
  "data": {  
    "clear": false,  
    "add_rules": [  
      {  
        "type": "CAM",  
        "id": "1",  
        "states": [  

```

```

        "attached",
        "armed",
        "alarmed"
    ],
    "action": "STATE_CHANGED"
  },
  {
    "type": "LPR_CAM",
    "id": "1",
    "events": [
      "CAR_LP_RECOGNIZED"
    ],
    "action": "EVENT"
  }
],
"delete_rules": [
  {
    "type": "CAM",
    "id": "2",
    "action": "STATE_CHANGED"
  }
]
}

```

Request Parameters

- **type** — type of the request (subscription control). Mandatory parameter.
- **id** — numeric message identifier. An optional parameter that is sent by the server in response to an incorrect request.
- **data.clear** — optional parameter. Delete all subscriptions. Possible values:
 - `false` — do not delete current subscriptions (default value);
 - `true` — delete all current subscriptions.
- **data.add_rules** — list of the rules that will be added to the subscription:
 - **data.add_rules[n].type** — type of the object(-s) to subscribe to;
 - **data.add_rules[n].id** — id of the object to subscribe to. If the parameter is not specified, then subscribe to all objects of a given type;
 - **data.add_rules[n].states** — list of the state type names for subscribing to states (`STATE_CHANGED`), the changes of which will be sent to the client (see [Object State Change Message](#)). If the parameter is omitted then the client will receive all types. One can subscribe to change of the following states only:
 - **data.states.alarmed** — alarm is not detected/alarm detected;
 - **data.states.armed** — object is armed/disarmed;
 - **data.states.attached** — object is enabled/disabled.
 - **data.add_rules[n].events** — list of the event names for the event subscription (`EVENT`). If the parameter is omitted then the client will receive all events;
 - **data.add_rules[n].action** — type of the subscription. Two values are valid in the current version of the REST API:
 - `STATE_CHANGED` — notification about changing state;
 - `EVENT` — receiving events.
- **data.delete_rules** — list of the rules to be deleted from the subscription:

- **data.add_rules[n].type** — type of the object(-s) that will be deleted from the subscription;
- **data.add_rules[n].id** — `id` of the object that will be deleted from the subscription. If there is no such object, an error is returned. If this parameter is not specified then the subscription to all objects of the given type is canceled regardless of how the subscription has been made (for example, subscription to the object or to the objects type). If the `id` of an existing object is specified, but there was a subscription by type before that, then conversion to a subscription by `id` is performed for all objects except the excluded one;
- **data.delete_rules[n].action** — textual identifier of the event.

Response

If the request does not contain errors, then there is no response to the command. For all `STATE_CHANGED` rules server sends states of the objects.

10.2.2 Subscribe to Event (WebSocket)

To subscribe to an event you need to specify parameters of the action having `EVENT` type in the `add_rules` section.

Request Example

```
{
  "type": "subscribe",
  "id": 12345,
  "data": {
    "add_rules": [
      {
        "type": "LPR_CAM",
        "id": "1",
        "events": [
          "CAR_LP_RECOGNIZED"
        ],
        "action": "EVENT"
      }
    ]
  }
}
```

Request Parameters

- **type** — type of the request (subscription control). Mandatory parameter.
- **id** — numeric message identifier. An optional parameter that is sent by the server in response to an incorrect request.
- **data.add_rules** — list of the rules that will be added to the subscription:
 - **data.add_rules[n].type** — type of the object(-s) to subscribe to;
 - **data.add_rules[n].id** — `id` of the object to subscribe to. If the parameter is not specified, then subscribe to all objects of a given type;
 - **data.add_rules[n].events** — list of the event names for the event subscription (`EVENT`). If the parameter is omitted then the client will receive all events;
 - **data.add_rules[n].action** — type of the subscription (`EVENT`).

Response

If the request does not contain errors, then there is no response to the command.

10.2.3 Subscribe to State Change (WebSocket)

To subscribe to a state change you need to specify parameters of the action having `STATE_CHANGED` type in the `add_rules` section.

Request Example

```
{
  "type": "subscribe",
  "id": 12345,
  "data": {
    "add_rules": [
      {
        "type": "CAM",
        "id": "1",
        "states": [
          "attached",
          "armed",
          "alarmed"
        ],
        "action": "STATE_CHANGED"
      }
    ]
  }
}
```

Request Parameters

- **type** — type of the request (subscription control). Mandatory parameter.
- **id** — numeric message identifier. An optional parameter that is sent by the server in response to an incorrect request.
- **data.add_rules** — list of the rules that will be added to the subscription:
 - **data.add_rules[n].type** — type of the object(-s) to subscribe to;
 - **data.add_rules[n].id** — id of the object to subscribe to. If the parameter is not specified, then subscribe to all objects of a given type;
 - **data.add_rules[n].states** — list of the state types names for subscribing to states (`STATE_CHANGED`), the change of which will be sent to the client. If the parameter is omitted set client will receive all types. Names of all state types are represented below (see [Object State Change Message](#)). One can subscribe to change of the following states only:
 - **data.states.alarmed** — alarm is not detected/ alarm detected;
 - **data.states.armed** — object is armed/disarmed;
 - **data.states.attached** — object is enabled/disabled.
 - **data.add_rules[n].events** — list of the event names for the event subscription (`EVENT`). If the parameter is omitted then the client will receive all events;
 - **data.add_rules[n].action** — type of the subscription (`STATE_CHANGED`).

Response

If the request does not contain errors, then there is no response to the command. For all `STATE_CHANGED` rules server sends states of the objects.

10.2.4 Delete All Subscriptions (WebSocket)

To delete all subscription it enough to set the **data.clear** parameter to `true` (see below).

Request Example

```
{
  "type": "subscribe",
  "id": 12345,
  "data": {
    "clear": true
  }
}
```

10.2.5 Delete Subscription (WebSocket)

To delete a subscription, you need to specify action parameters with the `EVENT` type (delete event subscription) or `STATE_CHANGED` type (delete state change subscription) in the `delete_rules` section.

Request Example

```
{
  "type": "subscribe",
  "id": 12345,
  "data": {
    "delete_rules": [
      {
        "type": "CAM",
        "id": "2",
        "action": "STATE_CHANGED"
      }
    ]
  }
}
```

Request Parameters

- **type** — type of the request (subscription control). Mandatory parameter.
- **id** — numeric message identifier. An optional parameter that is sent by the server in response to an incorrect request.
- **data.delete_rules** — list of the rules to be deleted from the subscription:
 - **data.add_rules[n].type** — type of the object(-s) that will be deleted from the subscription;
 - **data.add_rules[n].id** — `id` of the object that will be deleted from the subscription. If there is no such object, an error is returned. If this parameter is not specified then the subscription to all objects of the given type is canceled regardless of how the subscription has been made (for example, subscription to the object or to the objects type). If the `id` of an existing object is specified, but there was a subscription by type before that, then conversion to a subscription by `id` is performed for all objects except the excluded one;
 - **data.delete_rules[n].action** — textual identifier of the event (`EVENT` or `STATE_CHANGED`).

Note. The states of the object for which a subscription was previously created is not required to be listed. Subscriptions to all possible states of the object will be deleted.

Response

If the request does not contain errors, then there is no response to the command.

10.2.6 Update Subscription (WebSocket)

Subscription update is carried out in two stages:

1. Deletion of the current subscription (see [Delete Subscription \(WebSocket\)](#));
2. Creation of the new subscription for the same object with new parameters (see [Subscribe to Event \(WebSocket\)](#) or [Subscribe to State Change \(WebSocket\)](#)).

10.2.7 WebSocket Server Messages

When using WebSocket the following server notifications are possible:

- [Object State Change Message](#).
- [Event Message](#).
- [Object Deletion Message](#).
- [Error Message](#).

10.2.7.1 Object State Change Message

The message is sent to all clients who have a subscription to change the state of the object (STATE_CHANGED). Acknowledgment of a state change message is not required.

```
{
  "type": "state",
  "data": {
    "type": "CAM",
    "id": "1",
    "ticks": 501586,
    "time": "2017-02-02T16:10:07.241",
    "states": {
      "alarmed": false,
      "armed": false,
      "attached": true
    }
  }
}
```

Message parameters:

- **type.state** — type of the message state (state changing);
- **data.type** — type of the object whose state has been changed;
- **data.type** — identifier of the object whose state has been changed;
- **data.ticks** — millisecond counter (from the WebSocket server start). It can be used for additional control in case of delays when transmitting messages over the network;
- **data.time** — date and time of sending the message to the client (according to the server system time);
- **data.states** — state flags (true/false). The associative array contains only those flags that have been modified:
 - **data.states.alarmed** — alarm was registered;
 - **data.states.armed** — object is armed;

- **data.states.attached** – object is connected to the network.

10.2.7.2 Event Message

The message is sent to all clients who have a subscription to event of the object (EVENT). Acknowledgment of a event message is not required.

```
{
  "type": "event",
  "data": {
    "type": "LPR_CAM",
    "id": "1",
    "action": "CAR_LP_RECOGNIZED",
    "ticks": 501586,
    "time": "2018-03-26T21:23:16.122",
    "parameters": {
      "best_view_date_time": "26-03-2018 21:23:15.898",
      "best_view_mask_id": "6.0",
      "camera_id": "6",
      "direction_id": "1",
      "direction_name": "approaching",
      "number": "T345 LYW",
      "number_utf8": "T345 LYW",
      "plate_bottom_i": "0.88958333333333328",
      "plate_left_i": "0.096093750000000006",
      "plate_right_i": "0.22421875000000002",
      "plate_top_i": "0.85624999999999996",
      "recognizer_id": "1",
      "recognizer_name": "License plate recognizer 1",
      "recognizer_type": "LPR_CAM",
      "speed": "0.94696974754333496",
      "template_country_id": "7",
      "template_country_iso_code": "GB",
      "template_country_name": "UNITED_KINGDOM",
      "template_name": "GB_7S_R",
      "time_enter": "26-03-2018 21:23:14.998",
      "time_leave": "26-03-2018 21:23:15.965",
      "track_id": "18",
      "tracked_out": "1",
      "units": "0",
      "velocity": "0.94696974754333496",
      "weight": "1684.4208984375"
    }
  }
}
```

Message parameters:

- **type** – type of the message event (event);
- **data.type** – type of the object that generated an event;
- **data.type** – identifier of the object that generated an event;
- **data.ticks** – millisecond counter (from the WebSocket server start). It can be used for additional control in case of delays when transmitting messages over the network;
- **data.time** – event generation timestamp;
- **data.parameters** – list of the event parameters.

10.2.7.3 Object Deletion Message

Object deletion message is sent to all clients who have a subscription to an object removed from the SecurOS *Object tree*.

```
{
  "type": "event",
  "data": {
    "type": "CAM",
    "id": "1.8",
    "action": "DELETED",
    "time": "2017-04-30T14:10:09.315"
  }
}
```

Message parameters:

- **type** — type of the message event (event);
- **data.type** — type of the deleted object;
- **data.type** — identifier of the deleted object;
- **data.time** — date and time of sending the message to the client (according to the server system time);
- **data.action** — event code DELETED (object has been deleted).

10.2.7.4 Error Message

An error message is sent in response to an incorrect client request:

```
{
  "type": "error",
  "data": {
    "request_id": "12345",
    "message": "Object not found",
    "error": "OBJECT_NOT_FOUND"
  }
}
```

Message parameters:

- **type** — type of the message ("error"): error message;
- **data.request_id** — request identifier passed in the request;
- **data.message** — error text message;
- **data.error** — text error code.

Error codes are represented in the Table 1.

Table 1. Error Codes

Error code	Description
BAD_JSON	Syntax error when parsing JSON message
UNKNOWN_MESSAGE_TYPE	Unknown request type
INCORRECT_MESSAGE_STRUCTURE	Incorrect message structure (mandatory parameter is omitted, invalid field type, etc.)

Error code	Description
UNKNOWN_OBJECT_TYPE	Unknown object type
UNKNOWN_SUBSCRIBE_ACTION	Unknown subscription code
OBJECT_NOT_FOUND	Object not found
UNKNOWN_STATE_IDENTIFIER	Invalid state name in state change subscription (STATE_CHANGED)

11 Working with Event Viewer

The following requests are provided for working with the *Event Viewer*:

- **Get Event List for Specified Camera.**

When retrieving events the additional **Event Filtering** is available.

11.1 Event Filtering

When using REST API it is possible to filter events returned by request response. The filter is set in the *REST API* object settings (see **SecurOS Administration Guide**).

Selected filter will be applied when requesting *Camera* events list (see below) from the *Event Viewer*.

11.2 Get Event List for Specified Camera

Resource ID

```
/api/v1/protocol/<cam_id>?start_time=<value>&stop_time=<value>  
&max_count=<value>
```

Objects Access Rights

Camera — View (👁) and higher.

Request Parameters

- **cam_id** — identifier of the *Camera*.
- **start_time** — beginning of the sampling time period. Mandatory parameter. Is set in the `YYYYMMDDTHHMMSS[.SSS]` format (milliseconds are optional).
- **start_time** — end of the sampling time period. Optional parameter. Is set in the `YYYYMMDDTHHMMSS[.SSS]` format (milliseconds are optional).
- **max_count** — maximum number of returned records starting with the newest one. Optional parameter. If specified value is less than total records number within the specified sampling time period, then the oldest records will be ignored.

Protocol/Method

HTTP/GET

Request Example

```
http://172.16.1.170:8888/  
api/v1/cameras/4/protocol?start_time=20200218T101000  
&stop_time=20200218T101400&max_count=10
```

Response Example

```
{  
  "data": {  
    "actual_count": 3,  
    "events": [  
      {  
        "action": "ARMED",  
        "comment": "",  
        "id": "e9f8da05-aacd-49f8-9a15-595a8602321b",  
        "time": "18-02-20 10:12:12.310",  
        "time_write": "18-02-20 10:12:12.963"  
      },  
      {  
        "action": "MD_START",  
        "comment": "",  
        "id": "a210caa1-7829-4e1b-9486-8d5cf3d6d0b3",  
        "time": "18-02-20 10:12:12.000",  
        "time_write": "18-02-20 10:12:12.963"  
      },  
      {  
        "action": "REC",  
        "comment": "",  
        "id": "1280b772-723d-4988-84f6-5558ced9eb75",  
        "time": "18-02-20 10:12:12.548",  
        "time_write": "18-02-20 10:12:12.964"  
      }  
    ]  
  },  
  "status": "success"  
}
```

12 Working with Map

The following requests are provided for working with the *Map*:

- **Get Settings of All Maps.**
- **Get Settings of Specified Map.**

12.1 Get Settings of All Maps

Resource ID

/api/v1/maps

Objects Access Rights

Not required.

Request Parameters

None.

Protocol/Method

HTTP/GET

Request Example

http://172.16.1.170:8888/api/v1/maps

Response Example

```
{
  "data": [
    {
      "id": "1",
      "map": {
        "1": {
          "color": "#ffffff",
          "image": "/api/v1/maps/1/image/1",
          "name": "Main office",
          "objects": [
            {
              "id": "1",
              "rotation": -0.66322511575784526,
              "text_elided": false,
              "text_x": 0,
              "text_y": 1,
              "type": "CAM",
              "x": 37.396694214876035,
              "y": 57.665681782497614
            }
          ]
        }
      }
    }
  ]
}
```



```

    },
    {
      "id": "2",
      "rotation": 2.3212879051524582,
      "text_elided": false,
      "text_x": 0,
      "text_y": 0.88774596494757085,
      "type": "CAM",
      "x": 55.785123966942152,
      "y": 37.682524729156853
    },
    {
      "id": "Dining room",
      "rotation": 0,
      "text_elided": false,
      "text_x": 0,
      "text_y": 1,
      "type": "LEVEL",
      "x": 22.933884297520663,
      "y": 82.787364935268855
    }
  ]
},
"2": {
  "color": "#ffffff",
  "image": "/api/v1/maps/1/image/1",
  "name": "Dining room",
  "objects": [ ]
}
},
"name": "Map 1"
},
{
  "id": "2",
  "map": {
    "1": {
      "color": "#ffffff",
      "image": "/api/v1/maps/2/image/1",
      "name": "Test Lab",
      "objects": [
        {
          "id": "4",
          "rotation": 0,
          "text_elided": false,
          "text_x": 0,
          "text_y": 1,
          "type": "CAM",
          "x": 30.785123966942145,
          "y": 30.545682924392302
        },
        {
          "id": "5",
          "rotation": 0,
          "text_elided": false,
          "text_x": 0,
          "text_y": 1,
          "type": "CAM",
          "x": 64.256198347107443,
          "y": 60.234944832212854
        }
      ]
    }
  }
},

```

```
    "name": "Map 2"
  }
],
"status": "success"
}
```

12.2 Get Settings of Specified Map

Resource ID

/api/v1/maps/<map_id>

Objects Access Rights

Not required.

Request Parameters

map_id — identifier of the *Map*.

Protocol/Method

HTTP/GET

Request Example

http://172.16.1.170:8888/api/v1/maps/2

Response Example

```
{
  "data": {
    "id": "2",
    "map": {
      "1": {
        "color": "#ffffff",
        "image": "/api/v1/maps/2/image/1",
        "name": "Test Lab",
        "objects": [
          {
            "id": "4",
            "rotation": 0,
            "text_elided": false,
            "text_x": 0,
            "text_y": 1,
            "type": "CAM",
            "x": 30.785123966942145,
            "y": 30.545682924392302
          },
          {
            "id": "5",
            "rotation": 0,
            "text_elided": false,
            "text_x": 0,
            "text_y": 1,
            "type": "CAM",
            "x": 64.256198347107443,
            "y": 60.234944832212854
          }
        ]
      }
    }
  }
}
```

```
    }  
  ]  
}  
,  
  "name": "Map 2"  
,  
  "status": "success"  
}
```

13 Working with FaceX

The following requests are provided for working with the SecurOS FaceX:

- **Get FaceX Server Settings.**

13.1 Get FaceX Server Settings

Resource ID

/api/v1/facexconfig

Objects Access Rights

Not required.

Request Parameters

None.

Protocol/Method

HTTP/GET

Request Example

http://172.16.1.170:8888/api/v1/facexconfig

▼ Response Example

```
{
  "data": [
    {
      "acs_verification_modes": {
        "multi_factor_mode": "disabled",
        "single_factor_mode": "disabled"
      },
      "facex_feed_ids": [ "1", "2" ],
      "facex_server_port": "21093",
      "server_dns": "VIDEO_SERVER_II",
      "server_ip": ""
    }
  ],
  "status": "success"
}
```

The following parameters are returned in the response:

- **server_dns** — DNS name of the *Video Server* where FaceX server is deployed;
- **server_ip** — IP address of the *Video Server* where FaceX server is deployed. This parameter is displayed if it is specified in the correspondent *Computer* object settings;

- **facex_server_port** — FaceX server port number for interacting with the client application, specified in the *FaceX: Server* object settings;
- **facex_feed_ids** — identifiers of the *Cameras* selected to work with the FaceX server;
- **multi_factor_mode, single_factor_mode** — selected mode of the authentication:
 - enabled — mode is on;
 - disabled — mode is off.

14 Working with Macros

The following requests are provided for working with the *Macros*:

- **Get Settings of All Macros.**
- **Get Settings of Specified Macro.**
- **Run Macro.**

14.1 Get Settings of All Macros

Resource ID

/api/v1/macros

Objects Access Rights

Not required.

Request Parameters

None.

Protocol/Method

HTTP/GET

Request Example

http://172.16.1.170:8888/api/v1/macros

Response Example

```
{
  "data": [
    {
      "id": "1.1",
      "local": "1",
      "name": "Macro 1"
    },
    {
      "id": "1.2",
      "local": "0",
      "name": "Macro 2"
    }
  ],
  "status": "success"
}
```

The list of returned parameters also contains the *Macro* locality flag:

- **local** — is the *Macro* local or global:
 - 1 — macro is local (may be executed only on the specified *Computer*);
 - 0 — macro is global (may be executed on all *Computer* within the SecurOS network).

14.2 Get Settings of Specified Macro

Resource ID

/api/v1/macros/<macro_id>

Objects Access Rights

Not required.

Request Parameters

macro_id — identifier of the *Macro*.

Protocol/Method

HTTP/GET

Request Example

http://172.16.1.170:8888/api/v1/macros/1.2

Response Example

```
{
  "data": {
    "id": "1.2",
    "local": "0",
    "name": "Macro 2"
  },
  "status": "success"
}
```

The list of returned parameters also contains the *Macro* locality flag:

- **local** — is the *Macro* local or global:
 - 1 — macro is local (executed only on the specified *Computer*);
 - 0 — macro is global (executed on all *Computer* within the SecurOS network).

14.3 Run Macro

Resource ID

/api/v1/macros/<macro_id>/run

Objects Access Rights

Macro — Control (🔑) and higher.

Request Parameters

macro_id — identifier of the *Macro*.

Protocol/Method

HTTP/POST

Request Example

```
http://172.16.1.170:8888/api/v1/macros/1.2/run
```

Response Example

```
{  
  "status": "success"  
}
```


15 Working with Computers

The following requests are provided for working with the *Computers*:

- **Get Settings of All Computers.**
- **Get Settings of Specified Computer.**

15.1 Get Settings of All Computers

Resource ID

/api/v1/servers

Objects Access Rights

Computer — View (👁) and higher.

Request Parameters

None.

Protocol/Method

HTTP/GET

Request Example

http://172.16.1.170:8888/api/v1/servers

Response Example

```
{
  "data": [
    {
      "FOLDER.audio_mode.count": "0",
      "FOLDER.login.count": "0",
      "FOLDER.network_path.count": "0",
      "FOLDER.pass.count": "0",
      "FOLDER.video_mode.count": "0",
      "SUBSCRIBE.mode.count": "0",
      "SUBSCRIBE.object.count": "0",
      "acs_database": "",
      "bookmarks_alarms_lifetime": "",
      "disable_protocol": "0",
      "display_id": "",
      "drives": "",
      "drives_a": "",
      "drives_a_r": "",
      "drives_and_folders": "",
      "drives_r": ""
    }
  ]
}
```

```

    "event_filter_id": "",
    "has_own_subs": "",
    "id": "OPERATOR",
    "init_user_name": "",
    "ip_address": "",
    "is_cs": false,
    "is_profile": "1",
    "max_proto_db_size": "",
    "name": "Operator Workstation",
    "parent_id": "1",
    "record_priority": "0",
    "start_display_mode": "1",
    "state": "DISCONNECTED",
    "type": "2",
    "use_virtual_ip": "",
    "user_id": "",
    "video_database": "",
    "write_protocol_params": ""
  },
  {
    "FOLDER.audio_mode.count": "0",
    "FOLDER.login.count": "0",
    "FOLDER.network_path.count": "0",
    "FOLDER.pass.count": "0",
    "FOLDER.video_mode.count": "0",
    "SUBSCRIBE.mode.count": "0",
    "SUBSCRIBE.object.count": "0",
    "acs_database": "1",
    "bookmarks_alarms_lifetime": "0",
    "connected": true,
    "disable_protocol": "0",
    "display_id": "",
    "drives": "",
    "drives_a": "",
    "drives_a_r": "",
    "drives_and_folders": "{\\"drives\\":[{\\"min_free_space\\":0.1,\\"path\\":\\"D:\\\\\\\",
    \\",\\"permissions\\":{\\"audio\\":\\"rw\\",\\"video\\":\\"rw\\"}}],\\"iscsi_drives\\":
    [],\\"network_folders\\":[{\\"login\\":\\"\\",\\"min_free_space\\":0.1,\\"password
    \":\\"EJABDJPACKPDAFNH\\",\\"path\\":\\"\\\\\\\\\\\\\\\\storage\\\\\\\\archive\\\\\\\\\\",\\"permissions\\":
    {\\"audio\\":\\"unused\\",\\"video\\":\\"rw\\"}}]}",
    "drives_r": "",
    "event_filter_id": "-1",
    "has_own_subs": "",
    "id": "VIDEO_SERVER_I",
    "init_user_name": "",
    "ip_address": "",
    "is_cs": true,
    "is_profile": "0",
    "max_proto_db_size": "0",
    "name": "Computer VIDEO_SERVER_I",
    "parent_id": "1",
    "record_priority": "0",
    "start_display_mode": "1",
    "state": "CONNECTED",
    "type": "16",
    "use_virtual_ip": "",
    "user_id": "",
    "video_database": "",
    "write_protocol_params": "0"
  },
  {
    "FOLDER.audio_mode.count": "0",
    "FOLDER.login.count": "0",

```

```

    "FOLDER.network_path.count": "0",
    "FOLDER.pass.count": "0",
    "FOLDER.video_mode.count": "0",
    "SUBSCRIBE.mode.count": "0",
    "SUBSCRIBE.object.count": "0",
    "acs_database": "",
    "bookmarks_alarms_lifetime": "0",
    "disable_protocol": "0",
    "display_id": "",
    "drives": "",
    "drives_a": "",
    "drives_a_r": "",
    "drives_and_folders": "{\"drives\":[],\"iscsi_drives\":[],\"network_folders\":
[]}",
    "drives_r": "",
    "event_filter_id": "-1",
    "has_own_subs": "",
    "id": "OPERATOR_II",
    "init_user_name": "",
    "ip_address": "",
    "is_cs": false,
    "is_profile": "0",
    "max_proto_db_size": "0",
    "name": "Computer OPERATOR_II",
    "parent_id": "1",
    "record_priority": "0",
    "start_display_mode": "1",
    "state": "DISCONNECTED",
    "type": "2",
    "use_virtual_ip": "",
    "user_id": "",
    "video_database": "",
    "write_protocol_params": "0"
  }
],
"status": "success"
}

```

List of the returned parameters of the *Computer* also contains its role and status:

- **is_cs** — is this *Computer* a *Configuration Server* or not:
 - true — yes;
 - false — no.
- **connected** — current state of the computer:
 - true — connected;
 - false — disconnected.

15.2 Get Settings of Specified Computer

Resource ID

/api/v1/servers/<computer_DNS_name>

Objects Access Rights

Computer — View (👁) and higher.

Request Parameters

computer_DNS_name — DNS name of the *Computer*.

Protocol/Method

HTTP/GET

Request Example

http://172.16.1.170:8888/api/v1/servers/VIDEO_SERVER_II

Response Example

```
{
  "data": {
    "FOLDER.audio_mode.count": "0",
    "FOLDER.login.count": "0",
    "FOLDER.network_path.count": "0",
    "FOLDER.pass.count": "0",
    "FOLDER.video_mode.count": "0",
    "SUBSCRIBE.mode.count": "0",
    "SUBSCRIBE.object.count": "0",
    "acs_database": "",
    "bookmarks_alarms_lifetime": "0",
    "connected": false,
    "disable_protocol": "0",
    "display_id": "",
    "drives": "",
    "drives_a": "",
    "drives_a_r": "",
    "drives_and_folders": "{\"drives\":[],\"iscsi_drives\":[],\"network_folders\":[]}",
    "drives_r": "",
    "event_filter_id": "-1",
    "has_own_subs": "",
    "id": "VIDEO_SERVER_II",
    "init_user_name": "",
    "ip_address": "172.16.12.174",
    "is_cs": false,
    "is_profile": "0",
    "max_proto_db_size": "0",
    "name": "Computer VIDEO_SERVER_II",
    "parent_id": "1",
    "record_priority": "0",
    "start_display_mode": "1",
    "state": "DISCONNECTED",
    "type": "16",
    "use_virtual_ip": "",
    "user_id": "",
    "video_database": "",
    "write_protocol_params": "0"
  },
  "status": "success"
}
```

List of the returned parameters of the *Computer* also contains its role and status:

- **is_cs** — is this *Computer* a *Configuration Server* or not:
 - true — yes;

- false - no.
- **connected** - current state of the computer:
 - true - connected;
 - false - disconnected.

16 HTTP Status Codes

To inform user about request result the following standard HTTP status codes are used:

Table 2. State Codes

Code	Description
200 (OK)	Request successful
400 (Bad Request)	Invalid request parameters
401 (Unauthorized)	Authorization error
403 (Forbidden)	User not having the necessary permissions to the resource
404 (Not Found)	SecurOS object with specified ID not found
408 (Request Timeout)	Timeout expired
503 (Service Unavailable)	Server cannot handle the request

17 Technical Support Information

Current section contains service information that is necessary on addressing to Intelligent Security Systems Technical Support.

Note. Collected data have to be send to the Intelligent Security Systems Technical Support Team (see [Getting Technical Support](#)).

To ensure quick technical support, prepare the following technical information:

Warning! Data in items marked by "*" are necessary to report.

1. (*) User (customer) name to address to.
2. (*) Organization name.
3. (*) User (or organization) contacts: phone, e-mail.
4. Name of a personal Intelligent Security Systems manager (on Intelligent Security Systems authorized partner case). Otherwise, give the following data:
 - Company where the hardware and software components were purchased.
 - Actions proposed to solve the problems announced by a partner from whom the product was purchased.
5. (*) Problem description.
6. (*) Actions results in the problem.
7. List of changes which result to the problem in case of applying after some changes in system settings/configuration.
8. System and diagnostic information on computer and SecurOS system configuration obtained from the **SystemInfo** utility (see [SecurOS Administration Guide](#) for detailed information about utility).

If it is impossible to run the utility provide the following information:

- (*) Guardant keys identifier and Dallas code;

Note. Equipment Dallas code can be found by the **ISS Hardware Report** utility (see [SecurOS Administration Guide](#) for detailed information about utility).

- (*) name and version of the installed Intelligent Security Systems company software.
 - total number of video servers and monitoring (operator) workstations in the system;
 - operating system (name and service pack version).
9. Another useful information, if possible. For example:
 - computer equipment configuration.
 - central processors load.
 - main and virtual memory used volumes.

- network load.
- network and network neighborhood configuration.