SIEMENS

Industrial Remote Communication - Remote Networks
SINEMA RC API server V3.2 SP4

Getting Started

03/2025
C79000-G8976-C616-10

Legal information

Warning notice system

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

⚠ DANGER

indicates that death or severe personal injury **will** result if proper precautions are not taken.

⚠ WARNING

indicates that death or severe personal injury **may** result if proper precautions are not taken.

⚠ CAUTION

indicates that minor personal injury can result if proper precautions are not taken.

NOTICE

indicates that property damage can result if proper precautions are not taken.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

Qualified Personnel

The product/system described in this documentation may be operated only by **personnel qualified** for the specific task in accordance with the relevant documentation, in particular its warning notices and safety instructions. Qualified persons are those who, because of their training and experience, are familiar with the installation, assembly, commissioning, operation, decommissioning and disassembly of the product and can recognize risks and avoid possible hazards.

Proper use of Siemens products

Note the following:

⚠ WARNING

Siemens products may only be used for the application described in the catalog and the associated usage information. If products and components from other manufacturers are used, these must be recommended or approved by Siemens. Proper transport, storage, installation, assembly, commissioning, operation and maintenance are required to ensure that the products operate safely and without any problems. The permissible ambient conditions must be complied with. The information in the relevant documentation must be observed.

Trademarks

All names identified by ® are registered trademarks of Siemens Aktiengesellschaft. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Siemens Aktiengesellschaft
Digital Industries
Postfach 48 48
90026 NÜRNBERG
GERMANY

C79000-G8976-C616-10
Ⓟ 09/2025 Subject to change

Table of contents

Introduction

1

For SINEMA RC, you can use the HTTP-based Application Programming Interface (API) to configure the WBM of the SINEMA RC server. The SINEMA RC is the API server in this case that answers the API requests of the API client.

You have the following options to work with data via the API:

- Retrieve data
- Generate new datasets
- Change data
- Delete data

New in this release

- Server information text
  - showServerInformationOnLogin, showServerInformationOnServer
- Creating and managing roles
  - oidcRelationBetweenClaims
- Layer 2 network
  - endpointtype
- OAuth/OpenID CA certifiate

Cybersecurity information

2

Siemens provides products and solutions with industrial cybersecurity functions that support the secure operation of plants, systems, machines and networks.

In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial cybersecurity concept. Siemens' products and solutions constitute one element of such a concept.

Customers are responsible for preventing unauthorized access to their plants, systems, machines and networks. Such systems, machines and components should only be connected to an enterprise network or the internet if and to the extent such a connection is necessary and only when appropriate security measures (e.g. firewalls and/or network segmentation) are in place.

For additional information on industrial cybersecurity measures that may be implemented, please visit
https://www.siemens.com/cybersecurity-industry (https://www.siemens.com/cybersecurity-industry).

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends that product updates are applied as soon as they are available and that the latest product versions are used. Use of product versions that are no longer supported, and failure to apply the latest updates may increase customer's exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Cybersecurity RSS Feed under
https://new.siemens.com/cert (https://www.siemens.com/cert).

Using the SINEMA RC API server

3

3.1 Activating the SINEMA RC API server

Requirements

- SINEMA RC server V3.0 and higher
- User with administrator rights has been created

Activating the API server

1. In the navigation, select "Services > API".
2. Activate a license to use the API server.
   - Trial license
     With the trial license, you have unrestricted use of the API server for 14 days for test and evaluation purposes, but not for productive use. All liability claims are excluded. After the trial license has expired, you need to purchase a license.
   - SINEMA RC API (6GK1724-3VH03-0BV0)
     With an activated license, you can use the API server without restrictions in SINEMA RC. You activate the license under "System > Licenses > Online activation".
3. Click "Save".

Exporting and converting a CA certificate

The API client sends its requests via HTTPS. For the API client to send a request, the CA certificate of the SINEMA RC server must be loaded to the API client.

1. In the navigation panel, select "Security > Certificate management > CA certificate".
2. For "Actions", click on "Export CA certificate". The CA certificate is exported as crt file.
3. Load the certificate into your API client. It may be necessary to convert the crt file to PEM, for example, with OpenSSL:
   ```
   openssl x509 -in cert.crt -out cert.pem
   ```

3.2 API client

3.2.1 Introduction

The API client sends the request to a resource that is addressed via URI (Uniform Resource Identifier).

- `https://<IP address of the SINEMA RC server>/api/v1/<URI resource>`

Note

Some commands take some time to execute the internal operations. Allow the server enough time before sending the next command.

Requirement

- The SINEMA RC API license or the trial license is activated.
- The SINEMA RC API server is activated.
- The API client has downloaded the CA certificate from SINEMA RC.

HTTP methods for accessing the resource

POST	Create resource, append data
GET	Retrieve resource
DELETE	Delete resource

JSON

The standard data format for the SINEMA RC API server is JSON (ECMA-404 The JSON Data Interchange Standard). In the section "JSON and data types (Page 159)", you will learn how to format JSON-based requests to the API and how to process the replies.

3.2.2 Using the SINEMA RC API server with the API client

All users of the SINEMA RC server can send API requests. The number of API calls that can be executed depends on the user rights. Users with admin rights can execute all API requests.

In the following examples, `192.168.16.9` is used as the IP address of the SINEMA RC server.

Determining the authentication token

The authentication token is required as a parameter for executing the functions in the API.

1. Use the following function to determine the authentication token:
   POST `https://192.168.16.9/api/v1/api-auth`
2. In the body, transfer the data for the user login in the json format:
   ```
   {
   "username":"<User>",
   "password":"<PW APIUser>"
   }
   ```
3. Execute the function. An authentication token is returned as a reply.
   ```
   {       "token": "5d03bf9f350dc738dabcb393c28b6f2f3592ac13" }
   ```
4. This authentication token must be entered for almost all future calls to the SINEMA RC API server.
   To this end, you enter the authentication token in the header.
   `'Authorization': 'token 5d03bf9f350dc738dabcb393c28b6f2f3592ac13'`
   When you request a new token, you must also adapt the header.

Sending requests to the API server

In the following examples, you will see how to output, create and delete roles. You will find the description for it in the section "API requests (Page 13)".

Output all roles

1. Enter the following function:
   `GET https://192.168.16.9/api/v1/accounts/roles`
2. Execute the function. All roles are output as a reply. The output corresponds to the display in the WBM under "User accounts > Users & Roles > Roles".
   ```
   [
   {         "id": 1,
            "name": "vpn_user"
         },
   {         "id": 2,
            "name": "admin"
       },
       {
            "id": 3,
            "name": "Service"
       },
       {
            "id": 4,
            "name": "Monitoring"
       }]
   ```

Creating a new role

1. The "backup" role is created and assigned the right "Create backup copies".
2. Enter the following function:
   `Post https://192.168.16.9/api/v1/accounts/roles`

3. In the body, transfer the data for the user login in the json format:
```
{
"name":"backup",
"rights":[2]
}
```
4. Execute the function. The ID of the "backup" role is output as a reply.
```
{
    "id": 5
}
```

Deleting a role

1. To delete the backup role, enter the following function:
```
DELETE https://192.168.16.9/api/v1/accounts/roles/5
```
2. Execute the function. When the role is deleted, the status is 200 OK.

API requests

4

4.1 Retrieving the authentication token

To send API requests, you must first generate a valid token with the login information, see section "Using the SINEMA RC API server with the API client (Page 10)".

URL	/api-auth		
POST	Retrieves an authentication token		
Request	**Parameter**	**Data type**	**Values/Comments**
	username (required)	string	Enter the user name
	password (required)	string	Enter the valid password
	loginType (required)	integer	Selection of login method 1: password: Login with user name and password 2: umc: Login using a UMC server
	otpToken (optional)	string	Entering a valid one-time token Requirement: Two-factor authentication is enabled.
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	token	string	Authentication token
Failed call	400 - BAD REQUEST	error	Login is not possible with the entered login information.
	422 - UNPROCESS-ABLE ENTRY	error	The one-time token is invalid

4.2 API requests without license

4.2.1 Overview

The following API requests can be sent without a license:

URL resource	Description
/api-auth	Retrieves an authentication token
/system/license	Retrieves the license information from the license server
/system/license/activate	Activates the license after receiving the license information from the license server
/system/network/dns	Specifies a DNS.
/system/init	Initializes the server with the specified parameters.
/services/apisettings	Specifies the settings of the API configuration

4.2.2 Retrieving the authentication token

URL	/api-auth		
POST	Retrieves the authentication token.		
Request	**Parameter**	**Data type**	**Values/Comments**
	username (required)	string	User name
	password (required)	string	Password
	loginType (required)	integer	Login type 1: password 2: umc
	otpToken (optional)	string	
Successful call	200 - OK	-	-
Failed call	400 - BAD REQUEST	error	Invalid parameters specified
	422 - UNPROCESSABLE ENTRY	error	Invalid token

4.2.3 Configuring DNS

This setting is adopted in the VPN configuration of the clients and is required for licensing.

URL	/system/network/dns		
POST	Specifies the DNS.		
Request	**Parameter**	**Data type**	**Values/Comments**
	hostname (required)	string	Enter the host name under which the SINEMA RC can be reached, for example, sinemarc.example.org.
	externallyResolvable (optional)	boolean	Externally resolvable host name When activated, the host name is included in the VPN configuration and in the configuration of the VPN clients. Values: true/false
	dnsPrimary (optional)	string	Enter the IPv4 address of the primary DNS server.
	dnsSecondary (optional)	string	Enter the IPv4 address of the secondary DNS server that is used when the primary DNS server is not reachable.
	dohActive (required)	boolean	DNS over HTTPS • true = active • false = deactivate
	dnsResolver (optional)	String	DNS Resolver Required when dohActive is set to true.
	dnsStamp (optional)	String	DNS stamp for establishing a connection with the DNS Resolver Required when dohActive is set to true.

Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	The settings are invalid.

4.2.4 Configuring NTP

This setting is adopted in the VPN configuration of the clients and is required for licensing.

URL	/system/dateandtime/ntp		
POST	Specifies the settings of the NTP configuration. If set correctly, the NTP function is activated.		
Request	**Parameter**	**Data type**	**Values/Comments**
	primaryNtp (required)	string	IP address or FQDN of the primary NTP server
	secondNtp (optional)	string	IP address or FQDN of the secondary NTP server
	timezone (required)	string	Time zone of the SINEMA RC server in the format "+/- HH:MM" The time zone relates to UTC standard world time.
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid settings

4.2.5 Activating a license

Note

To activate a license, you must first retrieve the license information with "Retrieve license information". Only then can you activate the license with "Activate license".

Retrieving the license information

URL	/system/license		
POST	Returns the license information from the license server.		
Request	**Parameter**	**Data type**	**Values/Comments**
	ticketNo (required)	string	Retrieves the ticket information from the Wibu server.
Successful call	202 - Accepted	-	-

Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	A DNS server must be configured to activate the license, see section "Configuring DNS (Page 14)".
	404 - NOT FOUND	error	Ticket number not found.
	409 - CONFLICT	error	The license is already activated.

Activating a license

URL	/system/license/activate		
POST	Activates the license after receiving the license information from the license server		
Request	**Parameter**	**Data type**	**Values/Comments**
	ticketNo (required)	string	Activates the ticket information on the Wibu server.
	amount (optional)	integer	Activates the specified number of license nodes
Successful call	202 - Accepted	-	-
	Result	**Data type**	**Description**
	ID	integer	License ID
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	Ticket information is required to activate the license.
	404 - NOT FOUND	error	No entry found
	409 - CONFLICT	error	The license is already activated.
	422 - UNPROCESSABLE ENTRY	error	The license is invalid.

4.2.6 Configuring the password

Changing the password

URL	/api-auth/force-password		
POST	Changes the password for the defined user or on the first login.		
Request	**Parameter**	**Data type**	**Values/Comments**
	username	string	User name
	currentPassword	string	Current password
	newPassword	string	New password
Successful call	200 - OK	-	-
Failed call	400 - BAD REQUEST	error	Invalid parameters specified
	422 - UNPROCESSABLE ENTRY	error	Invalid token

4.2.7 Initializing the server

URL	/system/init		
POST	Initializes the server with the specified parameters. The function is blocked as soon as the server is initialized to protect the super administrator settings from being overwritten.		
Request	**Parameter**	**Data type**	**Values/Comments**
	superAdmin (required)	string	
	superAdminPw (required)	string	
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights.
	405 - METHODE NOT ALLOWED	error	The method is not allowed.
	422 - UNPROCESSABLE ENTRY	error	The action is invalid.

4.2.8 Configuring the API Server

Retrieving API settings

URL	/services/apisettings		
GET	Returns the API settings		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	tokenExpireTime	integer	Expiry time of the API authentication token
	active	boolean	The API server is activated/deactivated
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Setting up the API server

URL	/services/apisettings		
POST	Specifies the settings of the API configuration		
Request	**Parameter**	**Data type**	**Values/Comments**
	startTrial (optional)	boolean	Activates the 14-day trial version
	active (required)	boolean	Activates the API server
	tokenExpireTime (optional)	integer	Expiry time of the API authentication token
Successful call	200 - OK	-	-

Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	The API server cannot be activated due to insufficient licenses
	409 - CONFLICT	error	The API server trial version has already been used
	422 - UNPROCESSABLE ENTRY	error	Entry required

4.3 API requests with license

4.3.1 System

4.3.1.1 Retrieving the configuration overview

Retrieving the configuration overview

URL	/system/overview
GET	Retrieves the configuration overview of the SINEMA RC server

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	version	string	Version number of the current software
	orderId	string	Article number of the current software
	softwareDownloads	string	Link to download the current software version
	vmwareToolVersion	string	If installed, version number of the installed VMWare tool
	connectionLicenseActive	integer	Number of subscribers that are currently connected to the SINEMA RC server
	connectionLicenseUsage	integer	Number of total end devices that can be configured
	clientStandardLicenseActive	integer	Number of active SINEMA RC client connections
	clientStandardLicenseUsage	integer	Number of total client connections that are possible
	clientFloatingLicenseActive	integer	Number of active SINEMA RC client connections
	clientFloatingLicenseUsage	integer	Number of total client connections that are possible
	edgeClientLicenseActive	integer	Number of active Edge client connections
	edgeClientLicenseUsage	integer	Number of Edge client connections that are possible in total
	deviceLicenseActive	integer	Number of active device connections
	deviceLicenseUsage	integer	Total number of device connections that are possible
	users	integer	Number of users created in the project
	devices	integer	Number of devices created in the project
	vpnTotal	integer	Number of active VPN connections
	vpnUsers	integer	Number of active VPN connections to the users created in the project
	vpnDevices	integer	Number of active VPN connections to the devices created in the project
	serverTime	string	Current system time (UTC)
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Retrieving the system utilization

URL	/system/overview/load
GET	Retrieves the system utilization from the SINEMA RC server

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	diskSize	string	Total size of the hard disk in GB
	diskUsage	string	Utilized memory of the hard disk in GB
	ramTotal	string	Total work memory
	ramUsed	string	Work memory in use
	wanRx	string	Received bytes in the WAN
	wanTx	string	Sent bytes in the WAN
	lan1Rx	string	Received bytes in the LAN1
	lan1Tx	string	Sent bytes in the LAN1
	lan2Rx	string	Received bytes in the LAN2
	lan2Tx	string	Sent bytes in the LAN2
	lan3Rx	string	Received bytes in the LAN3
	lan3Tx	string	Sent bytes in the LAN3
	lan4Rx	string	Received bytes in the LAN4
	lan4Tx	string	Sent bytes in the LAN4
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

4.3.1.2 User log

Retrieving the user log

URL	/system/userlog		
GET	Returns the status of the user log		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	userLogTracing	boolean	Indicates whether or not the user activities are recorded in the user log
Failed to retrieve	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/system/userlog/entries
GET	Lists the entries of the user log

Successful call	200 - OK	-	-	
	Result	**Data type**	**Description**	
	userLogs	ListOf<id username, endpoint, destinationPort, startTime, endTime, duration, packageCounter>	Lists all entries with IDs and the following information:	
			username	User accessing the endpoint
			endpoint	Possible values for the endpoint are as follows: • Device name • Subnet name • End device name with or without IP address • Interface • IP address: When the IP address does not belong to a device or end device, the IP address is displayed. Appearance: • Device name.subnet name.end device name (IP address) • IP address
			destinationPort	Port used to access the end device
			startTime	Start of connection
			endTime	End of connection
			duration	Duration of the connection
			packageCounter	Number of packets and bytes sent
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights	
	404 - NOT FOUND	error	No entry found	

Enabling the user log

URL	/system/userlog/		
POST	Enables the "Connection tracking active" function. The user activities are recorded in the user log		
Request	**Parameter**	**Data type**	**Values/Comments**
	userLogTracing (required)	boolean	
Successful call	200 - ACCEPTED	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	The function is already active

4.3.1.3 Configuring the network

Retrieving the interface settings

URL	/system/network?interface={integer_value}
GET	Returns the settings of a selected interface

Request	**URL parameter**	**Data type**	**Values/Comments**
	interface (required)	integer	Entering the interface to be called: 0 = WAN 1 = LAN1 2 = LAN2 3 = LAN3 4 = LAN4
Successful call	200 - OK	-	
	Result	**Data type**	**Description**
	mtu	string	MTU size of the package
	dhcp	boolean	DHCP for IPv4
	ipAddress	string	IP address of the interface
	networkMask	string	Subnet mask
	gateway	string	Gateway of the interface Is displayed when the interface is the WAN.
	wanIp	string	Is displayed when the option "SINEMA Remote Connect is located behind a NAT device" is activated
	ipv6Enabled	boolean	Editing of IPv6 fields: • true: enabled • false: disabled
	ipv6Slaac	boolean	SLAAC for IPv6: • true: enabled • false: disabled
	ipv6Address	string	IPv6 address of the WAN interface
	ipv6PrefixLength	string	Prefix length of the WAN interface
	ipv6Gateway	string	Gateway address of the WAN interface
	masquerading	boolean	Masquerading for the LAN port: • true: enabled • false: disabled
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Configuring an interface

URL	/system/network
POST	Defines the network settings of the specified interface

Request	**Parameter**	**Data type**	**Values/Comments**
	interface (required)	integer	Input of the interface to be configured: 0 = WAN 1 = LAN1 2 = LAN2 3 = LAN3 4 = LAN4
	MTU (optional)	integer	MTU (Maximum Transmission Unit) specifies the maximum size of the packet. If packets are longer than the set MTU, they are fragmented. The maximum size is 1500 bytes. If no value is specified, the default value 1460 is used.
	dhcp (required)	boolean	Enables DHCP for IPv4 • true: enabled • false: disabled
	ipAddress (required)	string	IP address of the interface Entry only required when the interface dhcp = false.
	networkMask (required)	string	Subnet mask Entry only required when the interface dhcp = false.
	gateway (optional)	string	IP address for the gateway Entry only required when the interface 0 = WAN.
	wanIp (optional)	string	WAN IP address via which SINEMA RC can be reached. This can, for example, be the WAN IP address of a DSL router via which SINEMA RC is connected to the Internet This entry activates the option "SINEMA Remote Connect is located behind a NAT device"
	ipv6Enabled (optional)	boolean	Enables editing of the IPv6 fields • true: enabled • false: disabled
	ipv6Slaac (optional)	boolean	Enables SLAAC for IPv6 • true: enabled • false: disabled
	ipv6Address (optional)	string	IPv6 address of the WAN interface Entry only required when the interface ipv6Slaac = false.
	ipv6PrefixLength (optional)	string	Prefix length of the WAN interface Entry only required when the interface ipv6Slaac = false.
	ipv6Gateway (optional)	string	Gateway address of the WAN interface
	masquerading (optional)	boolean	Enables masquerading for the LAN interface: • true: enabled • false: disabled
Successful call	200 - OK	string	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	This field cannot be set to "false" on a WAN interface.
	422 - UNPROCESSABLE ENTRY	error	Invalid settings

4.3.1.4 Retrieving the DNS settings

Configuring DNS

No license is required for the DNS configuration. The procedure is described in the section "Configuring DNS (Page 14)".

Retrieving the DNS settings

URL	/system/network/dns		
GET	Returns the DNS settings		
Successful call	200 - OK	-	
	Result	**Data type**	**Description**
	hostname (required)	string	Host name of the DNS server
	externallyResolvable (optional)	boolean	Externally resolvable host name: • true • false
	dnsPrimary (optional)	string	Primary DNS server
	dnsSecondary (optional)	string	Secondary DNS server
	dohActive (required)	boolean	DNS over HTTPS • true = active • false = deactivate
	dnsResolver (optional)	String	DNS Resolver Required when dohActive is set to true.
	dnsStamp (optional)	String	DNS stamp for establishing a connection with the DNS Resolver Required when dohActive is set to true.
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

4.3.1.5 Configuring the Web server

Retrieving the Web server settings

URL	/system/network/webserver
GET	Returns the Web server settings

Successful call	200 - OK	-	
	Result	**Data type**	**Description**
	httpsPort	integer	Port for the HTTPS remote access
	httpsForwardingPort	integer	Port to which the HTTPS requests are forwarded
	fallBackPort	integer	HTTPS fallback port for automatic configuration
	blockWebserverAccess-FromWan	boolean	Web server access via the WAN interface blocked
	protocolTls13	integer	Version of TLS • 0 = Disabled: Only TLS1.2 is used. • 1 = Enabled: Only TLS1.3 is used. • 2 = Auto calculate: SINEMA RC checks whether the user equipment uses TLS1.3 or TLS1.2.
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Configuring the Web server

URL	/system/network/webserver		
POST	Defines the settings of the Web server		
Request	**Parameter**	**Data type**	**Values/Comments**
	httpsPort (required)	integer	Entry of the HTTPS port for remote access
	blockWebserverAccess-FromWan	boolean	Block Webserver access from WAN interface
	httpsForwardingPort (required)	integer	Entry of the port to which the HTTPS requests are forwarded
	protocolTls13 (optional)	integer	Version of TLS • 0 = Disabled: Only TLS1.2 is used. • 1 = Enabled: Only TLS1.3 is used. • 2 = Auto calculate: SINEMA RC checks whether the user equipment uses TLS1.3 or TLS1.2.
	fallBackPort (optional)	integer	Entry of the HTTPS fallback port for automatic configuration. If no value is specified, the default value 6220 is used. This port is used by OpenVPN devices that update the configurations using the auto enrollment mechanism (update interval). If these devices cannot be accessed via the HTTPS port, the update takes place via the fallback port.
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid settings

4.3.1.6 Configuring the Ping

URL	/system/network/ping		
POST	Starts the ping that sends ping requests to the IP address to be checked and receives responses from the target device, if it can be reached. After the timeout has elapsed, the ping reports the status.		
Request	**Parameter**	**Data type**	**Values/Comments**
	Address (required)	string	IP address of the device
	repeat (optional)	integer	Number of ping retries
	timeout (optional)	integer	Waiting time within which the ping checks the device
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	output	string	Status message on whether the device can be reached via the specified address:
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid settings

4.3.1.7 Managing static routes

Retrieving static routes

URL	/system/network/staticroutes?count={integer_value}		
GET	Returns all static routes with IDs and names as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	count (optional)	integer	Number of results that are to be returned
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	users	ListOf<id, destinationNetwork>	Lists the static routes with IDs and names. When the "count" parameter is specified, the first found static routes are listed in the specified number.
Failed call	400 - BAD REQUEST	error	Invalid parameters specified
	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Reading out a static route

URL	/system/network/staticroutes/<ID>
GET	Returns all information on a static route

Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the static route
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	destinationNetwork	string	Destination network of the static route
	networkMask	string	Network mask of the static route
	gateway	string	Gateway of the static route
	interface	integer	0 = WAN 1 = LAN1 2 = LAN2 3 = LAN3 4 = LAN4
	description	string	Description of the static route
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Reading out the status of a static route

URL	/system/network/staticroutes/<ID>/status		
GET	Returns all information on a static route		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the static route
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	status	string	Status of the static route added
	errorField	string	Shows the error field
	description	string	Description of the static route
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Adding a static route

URL	/system/network/staticRoutes
POST	Adds a static route

Request	**Parameter**	**Data type**	**Values/Comments**
	destinationNetwork (required)	string	Destination network of the static route
	networkMask (required)	string	Network mask of the static route
	gateway (required)	string	Gateway of the static route
	interface (required)	integer	0 = WAN 1 = LAN1 2 = LAN2 3 = LAN3 4 = LAN4
Successful call	202 - ACCEPTED	-	-
	Result	**Data type**	**Description**
	detail	string	Detail of static route added
	staticRouteRequestId	integer	ID of the static route
	staticRouteStatusCheck-Url	string	URL of the status of the static route added
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	The route already exists
	422 - UNPROCESSABLE ENTRY	error	Invalid settings

Deleting a static route

URL	/system/network/staticroutes/<ID>		
DELETE	Deletes the desired route with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the static route
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

4.3.1.8 Managing address space of the virtual subnet

Retrieving the address space settings

URL	/system/vpnaddressspace/virtualsubnetsettings
GET	Returns all information for the address space settings of the virtual subnet

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	networkPoolActive	boolean	Network address space activated
	startAddress	string	Start address of the address space
	networkMask	string	The network mask belonging to the address space
	endAddress	string	End address of the address space The address space is limited by the start address and the network mask. The end address must be within this range.
	totalAvailableNetworks	integer	Number of available networks determined from the start address and the end address
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Adding an address space

URL	/system/vpnaddressspace/virtualsubnetsettings		
POST	Specifies the address space of the virtual subnet network and activates it		
Request	**Parameter**	**Data type**	**Values/Comments**
	startAddress (required)	string	Network address of the destination that can be reached via this route
	networkMask (required)	string	Network mask of the destination
	endAddress (required)	string	IP address of the gateway via which this network address is reachable
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid settings

Edit address space

URL	/system/vpnaddressspace/virtualsubnetsettings		
PUT	Edits the address space of the virtual subnet network		
Request	**Parameter**	**Data type**	**Values/Comments**
	startAddress (optional)	string	Network address of the destination that can be reached via this route
	networkMask (optional)	string	Network mask of the destination
	endAddress (optional)	string	IP address of the gateway via which this network address is reachable
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid settings

Deactivating the address space

URL	/system/vpnaddressspace/virtualsubnetsettings		
DELETE	Deletes the address space of the virtual subnet network		
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	The network address space is already disabled

4.3.1.9 Managing VPN address spaces

Retrieving OpenVPN address space settings

URL	/system/vpnaddressspace/openvpn		
GET	Returns all information for the OpenVPN address space settings		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	tcpStartIp	string	Start address of the address space
	tcpNetmask	string	The network mask belonging to the address space
	tcpEndIp	string	End address of the address space The address space is limited by the start address and the network mask. The end address must be within this range.
	tcpIpUsed	integer	Number of assigned IP addresses
	tcpIpTotal	integer	Number of available IP addresses
	udpStartIp	string	Start address of the address space
	udpNetmask	string	The network mask belonging to the address space
	udpEndIp	string	End address of the address space The address space is limited by the start address and the network mask. The end address must be within this range.
	udpIpUsed	integer	Number of assigned IP addresses
	udpIpTotal	integer	Number of available IP addresses
	fixedIpActive	boolean	The device can be assigned a fixed IP address from the address space.
	fixedIpProtocol	integer	IP protocol: 1 = TCP 2 = UDP
	fixedIpLocation	integer	Location of the fixed IP address space: 1 = First 2 = Last
	fixedIpLength	integer	Number of fixed IP addresses
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Adding OpenVPN address space

URL	/system/vpnaddressspace/openvpn		
POST	Specifies the address space settings for OpenVPN. "Activate fixed IP address space" is only activated when all three optional values are specified.		
Request	**Parameter**	**Data type**	**Values/Comments**
	tcpStartIp (required)	string	Start address of the address space
	tcpNetmask (required)	string	The network mask belonging to the address space
	udpStartIp (required)	string	Start address of the address space
	udpNetmask (required)	string	The network mask belonging to the address space
	fixedIpProtocol (optional)	integer	IP protocol: 1 = TCP 2 = UDP
	fixedIpLocation (optional)	integer	Location of the fixed IP address space: • 1 = First The fixed IP addresses are from the start area of the address space. The first IP address is reserved for the SINEMA RC server. The first fixed IP address is always the second IP address after the start IP address. • 2 = Last The fixed IP addresses are from the end area of the address space. The last fixed IP address is always the end IP address.
	fixedIpLength (optional)	integer	Number of fixed IP addresses
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid settings

Editing OpenVPN address space

URL	/system/vpnaddressspace/openvpn
PUT	Changes the address space settings for OpenVPN.

Request	**Parameter**	**Data type**	**Values/Comments**
	tcpStartIp	string	Start address of the address space
	tcpNetmask	string	The network mask belonging to the address space
	udpStartIp	string	Start address of the address space
	udpNetmask	string	The network mask belonging to the address space
	fixedIpProtocol	integer	IP protocol: 1 = TCP 2 = UDP
	fixedIpLocation	integer	Location of the fixed IP address space: • 1 = First The fixed IP addresses are from the start area of the address space. The first IP address is reserved for the SINEMA RC server. The first fixed IP address is always the second IP address after the start IP address. • 2 = Last The fixed IP addresses are from the end area of the address space. The last fixed IP address is always the end IP address.
	fixedIpLength	integer	Number of fixed IP addresses
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid settings

Deactivating the fixed IP address space

URL	/system/vpnaddressspace/openvpn		
DELETE	Deactivates the option "Activate fixed IP address space"		
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

4.3.1.10 Date & Time

Configuring NTP

No license is required for NTP configuration. The procedure is described in the section "Configuring NTP (Page 15)".

Retrieving the NTP settings

URL	/system/dateandtime/ntp
GET	Returns all information for the NTP settings

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	primaryNtp	string	IP address or FQDN of the primary NTP server
	secondNtp	string	IP address or FQDN of the secondary NTP server
	timezone	string	Time zone of the SINEMA RC server
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Deleting NTP

URL	/system/dateandtime/ntp		
DELETE	Disables the NTP function		
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

4.3.1.11 SMS gateway provider

Retrieving the SMS gateway provider

URL	/system/smsandemail/sms?count={integer_value}&search={string}		
GET	Returns all SMS gateway providers with IDs and names as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	count (optional)	integer	Number of results that are to be returned
	search (optional)	string	Search text for group name
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	provider	ListOf<id, name>	Lists all SMS gateway providers with IDs and names. When the "count" parameter is specified, the first found SMS gateway providers are listed in the specified number. When "search" is specified, only the SMS gateway providers whose name includes the search text are returned.
Failed call	400 - BAD REQUEST	error	Invalid parameters specified
	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/system/smsandemail/sms/<ID>		
GET	Returns all information for an SMS gateway provider		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the SMS gateway provider

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	name	string	Name of the SMS gateway provider
	address	string	E-mail address of the recipient of the SMS message
	senderNumber	string	Sender number
	subject	string	Subject of the e-mail
	cc	string	E-mail address of another recipient
	text	string	Message text
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Adding an SMS gateway provider

URL	/system/smsandemail/sms		
POST	Adds an SMS gateway provider		
Request	**Parameter**	**Data type**	**Values/Comments**
	name (required)	string	Name of the SMS gateway provider
	address (required)	string	E-mail address of the recipient of the SMS message The e-mail address is generally made up of the call number of the SIM card and the SMS gateway name. The requirement is that the e-mail address is activated.
	senderNumber (optional)	string	Sender number Identification that is transferred in the e-mail
	subject (optional)	string	Subject of the e-mail
	cc (optional)	string	E-mail address of another recipient The recipient only receives an e-mail. This could, for example, be a service technician who always wants to be informed when a certain device is woken.
	text (required)	string	$MSG - The message text of the wake-up SMS message is entered automatically. Depending on the network provider either the text from the subject or the text box is sent as the SMS message. You can obtain more detailed information on this from your network provider.
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	ID	integer	ID of the SMS gateway provider
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	The SMS gateway provider already exists.
	422 - UNPROCESSABLE ENTRY	error	Invalid settings

Edit SMS gateway provider

URL	/system/smsandemail/sms/<ID>		
PUT	Edits an SMS gateway provider		
Request	**Parameter**	**Data type**	**Values/Comments**
	name (optional)	string	Name of the SMS gateway provider
	address (optional)	string	E-mail address of the recipient of the SMS message The e-mail address is generally made up of the call number of the SIM card and the SMS gateway name. The requirement is that the e-mail address is activated.
	senderNumber (optional)	string	Sender number Identification that is transferred in the e-mail
	subject (optional)	string	Subject of the e-mail
	cc (optional)	string	E-mail address of another recipient The recipient only receives an e-mail. This could, for example, be a service technician who always wants to be informed when a certain device is woken.
	text (optional)	string	$MSG - The message text of the wake-up SMS message is entered automatically. Depending on the network provider either the text from the subject or the text box is sent as the SMS message. You can obtain more detailed information on this from your network provider.
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid settings

Deleting an SMS gateway provider

URL	/system/smsandemail/sms		
DELETE	Deletes the SMS gateway provider with the specified ID		
Request	**Parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the SMS gateway provider
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

4.3.1.12 E-mail settings

Retrieving the e-mail settings

URL	/system/smsandemail/email
GET	Returns the e-mail settings

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	method	integer	Method of delivery: 1 = via relay server 2 = direct
	lifeInQueue	integer	Maximum life in the queue (s)
	sender	string	Sender Email Address
	smtpServer	string	IP address or FQDN of the SMPT relay server
	smtpPort	integer	SMTP relay port
	tls	integer	Transport Layer Security (TLS): 1 = opportunistic 2 = binding
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Adding e-mail settings

URL	/system/smsandemail/email
POST	Adds an e-mail configuration. When the user name and the password is specified, the function "Server requires authentication" is activated.

Request	**Parameter**	**Data type**	**Values/Comments**
	method (required)	integer	Methods of delivery: 1 = via relay server Via relay server: The e-mail is forwarded via an SMTP relay server to the recipient. 2 = direct Direct: The e-mail is forwarded directly to the SMTP server.
	enabled (required)	boolean	Enables the service: • true • false
	lifeInQueue (required)	integer	Maximum life in the queue (s) Maximum time in seconds that the sender waits for a reply from the mail server. When the time elapses, the transfer of the e-mail is aborted.
	sender (required)	string	Sender The e-mail address specified as the sender when transferring to the mail server. With the transmission method relay host, the e-mail address of the user account of the SMTP relay server is specified.
	smtpServer (for method of delivery 1: required 2: optional)	string	IP address or FQDN of the SMTP relay server that is to forward the received e-mails
	smtpPort (for method of delivery 1: required 2: optional)	integer	Port on which the SMTP relay server accepts connections. Port 587 is set as default so that mail is received only from authenticated users.
	tls (for method of delivery 1: required; 2: optional)	integer	E-mail encryption that is transmitted via TLS: 1 = opportunistic Opportunistic: The transmission of the e-mail can be encrypted via TLS. If the receiving mail server does not support encrypted transfer, the e-mail is forwarded via an unencrypted connection. This setting is used automatically if you have selected "Direct" as the transmission method. 2 = binding Binding: The transmission of the e-mail is encrypted via TLS. When the receiving mail server does not support encrypted transmission, the e-mail is not forwarded.
	username (optional)	string	User name for access to the SMTP relay server
	password (optional)	string	Password for access to the SMTP relay server
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid settings

4.3.1.13 Manage licenses

Retrieving the license information

No license is required to retrieve license information from the license server. The procedure is described in the section "Activating a license (Page 15)".

Activating a license

No license is required to activate the license after receiving the license information. The procedure is described in the section "Activating a license (Page 15)".

Retrieving licenses

URL	/system/license?count={integer_value}		
GET	Returns all licenses with IDs and names as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	count (optional)	integer	Number of results that are to be returned
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	licenses	ListOf<id, licenseNr>	Lists all licenses with IDs and names
	count	integer	When the "count" parameter is specified, the first found licenses are listed in the specified number.
	previous	string	
	next	string	
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/system/license/<ID>		
GET	Returns all information for a license		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the license
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	licenseNr	string	License number used when the license was activated
	licensetype	string	License type
	date	string	Activation date (UTC)
	value	string	License value in "used/total" format
	status	string	Status
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/system/license/floating/client/		
GET	Returns all floating licenses as a list		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	standardTotal	integer	Number of client standard licenses that exist in total
	standardAvailable	integer	Number of client standard licenses currently not in use
	floatingTotal	integer	Number of floating licenses that exist in total
	FloatingAvailable	integer	Number of floating licenses currently not in use
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Client standard licenses

URL	/system/license/clients		
GET	Returns all client licenses with IDs and system IDs as a list		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	String	ID of the listed entry
	systemId	integer	Client system ID
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/system/license/clients/<ID>		
GET	Returns all information on a client license		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the client license
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	systemId	string	Client system ID
	deviceName	string	PC name from which the client logs into the server
	lastConnectionTime	string	Time stamp of the client login with date and time
	lastConnectedUserId	integer	Name of the user who last established the connection from the client to the server
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/system/license/clients/<ID>		
DELETE	Deletes the desired client license		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the client license
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Floating licenses

URL	/system/license/clients/floating		
GET	Returns all floating licenses with IDs and system IDs as a list		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	--	ListOf<id, sytemId>	Lists all floating licenses with IDs and system IDs.
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/system/license/clients/floating/<ID>		
PATCH	Assign floating license to the standard license		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the floating license
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/system/license/clients/floating/<ID>		
GET	Returns all information about a floating license		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the floating license
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	systemId	string	Client system ID
	deviceName	string	PC name from which the client logs into the server
	Status	boolean	Status of the connection
	lastConnectedUserId	integer	Name of the user who last established the connection from the client to the server
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

History

URL	/system/license/clients/floating/history		
GET	Returns all floating licenses with IDs and system IDs as a list		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	--	ListOf<id, sytemId>	Lists activities of the client floating licenses with IDs and system IDs.
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/system/license/clients/floating/history/<ID>
PATCH	Assign floating license to the standard license

Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the floating license
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/system/license/clients/floating/history/<ID>		
GET	Returns all activities for a floating license		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the floating license
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	systemId	string	Client system ID
	deviceName	string	PC name from which the client logs into the server
	lastActivityTime	string	Shows when the user was last active.
	lastConnectedUserId	integer	Name of the user who last established the connection from the client to the server
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Converting client standard licenses to floating licenses

URL	/system/license/floating/client/		
POST	Converts client standard licenses into floating licenses or vice-versa.		
Request	**Parameter**	**Data type**	**Values/Comments**
	direction (required)	integer	1 = Converts client standard licenses to floating licenses 2 = Converts floating licenses to client standard licenses
	count (required)	integer	Number of floating/standard licenses to be created
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	If there are not enough client standard licenses or only test licenses available on the system, no floating license can be restored.

Ticket number status

URL	/system/license/showticketnumbers		
GET	Indicates the status of the ticket number		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	showTicketNumbers	string	Ticket number status

URL	/system/license/showticketnumbers
POST	Updates the status of the ticket ID

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	showTicketNumbers	boolean	Status ticket ID
Failed call	409 - CONFLICT	error	The ticket number is already enabled (on) / The ticket ID is already disabled (off).

Licenses for Edge clients

URL	system/license/edgeclients		
GET	Returns all licenses with IDs and system IDs as a list		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	String	ID of the listed entry
	systemId	integer	Edge client system ID
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	system/license/edgeclients/<ID>		
GET	Returns all activities for a floating license		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the floating license
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	systemId	string	Client system ID
	deviceName	string	PC name from which the client logs into the server
	lastConnectionTime	string	Time stamp of the client login with date and time
	lastConnectedUserId	integer	Name of the user who last established the connection from the client to the server
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	system/license/edgeclients/<ID>		
DELETE	Deletes the desired floating license		
Request	**Parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the floating license
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Licenses for devices

URL	system/license/devices
GET	Returns all licenses with IDs and system IDs as a list

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	String	ID of the listed entry
	systemId	integer	Device system ID
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	system/license/devices/<ID>		
GET	Returns all information about a device license		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the device license
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	systemId	string	Client system ID
	deviceName	string	PC name from which the client logs into the server
	lastConnectionTime	string	Time stamp of the client login with date and time
	lastConnectedUserId	integer	Name of the user who last established the connection from the client to the server
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	system/license/devices/<ID>		
DELETE	Deletes the desired device license		
Request	**Parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the device license
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Releasing a license

URL	/system/license/<ID>		
DELETE	Releases a license with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the activated license
	amount (optional)	integer	Number of license end devices to be enabled
Successful call	202 - ACCEPTED	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	The license cannot be released because it contains active resources
	403 - FORBIDDEN	error	A demo license cannot be deleted.
	404 - NOT FOUND	error	License was not found

Manage license

URL	/system/license/manage		
POST	Moves used licenses from one ticket number to another ticket number.		
Request	**URL parameter**	**Data type**	**Values/Comments**
	firstLicenseId (required)	integer	First license ID
	secondLicenseId (required)	integer	Second license ID
	direction (required)	integer	Direction of transfer: • 1 = First to second license • 2 = Second to first license
Successful call	202 - ACCEPTED	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	If there are not enough licenses on the system

Reactivate license

URL	/system/license/retry_activate		
POST	Reactivates pending licenses		
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHOR-IZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	There is no license that can be activated after the restore.

Reset license container

URL	/system/license/reset_license_system		
POST	Resets license container. Only perform this if directed to do so by the hotline		
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHOR-IZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	The license cannot be released because it contains active resources

4.3.1.14 System Update

Calling the system update status

URL	/system/update/uploadstatus/<ID>
GET	Returns the status of the system update

Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the upload request
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	date	string	
	status	string	Status of the system update
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Applying the system update

URL	/system/update		
POST	Starts the update process of SINEMA RC with the specified update file		
Request	**Parameter**	**Data type**	**Values/Comments**
	file (required)	file upload	SINEMA RC update file
Successful call	202 - ACCEPTED	-	-
	Result	**Data type**	**Description**
	detail	string	Detail of the upload request
	uploadRequestId	integer	ID of the upload request
	uploadStatusCheckUrl	string	URL of the upload status
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid settings

4.3.1.15 Backing up & restoring

Retrieving a backup copy

URL	/system/backup/settings		
GET	Returns all settings for backup copies as a list		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	maxNrOfBackups	integer	Lists the maximum number of local backup copies
	interval	integer	Frequency of a data backup: 0 = Disabled 1 = Daily 2 = Every Sunday 3 = Every Saturday 4 = Every first day of the month
	time	string	Automatic backup time (UTC) hh: mm: ss
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/system/backup?count={integer_value}		
GET	Returns all backup copies with IDs and comments as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	count (optional)	integer	Number of results that are to be returned
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	backups	ListOf<id, comment>	Lists all backup copies with IDs and comments. When the "count" parameter is specified, the first found backup copies are listed in the specified number.
	count	integer	
	previous	string	
	next	string	
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/system/backup/<ID>		
GET	Returns all information for a backup copy		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the backup copy
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	date	string	Date on which the backup copy was created
	name	string	Name of the creator
	size	string	File size of the backup copy
	comment	string	Comment on the backup copy
	status	string	Status of the backup copy
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/system/backup/<ID>/status		
GET	Returns the status of the restore.		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	status	string	Status of the backup copy
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/system/backup/export/<ID>		
GET	Downloads the backup copy		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the backup copy
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Specifying the settings for backup copies

URL	/system/backup/settings		
POST	Specifies the settings for the backup and restoration configuration		
Request	**Parameter**	**Data type**	**Values/Comments**
	maxNrOfBackups (optional)	integer	Maximum number of local backup copies
	interval (optional)	integer	Frequency of a data backup: 0 = Disabled 1 = Daily 2 = Every Sunday 3 = Every Saturday 4 = Every first day of the month
	time (optional)	string	Automatic backup time (UTC) hh: mm: ss
	codingKey (optional)	string	Coding key for the backup file
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid settings

Creating a new backup copy

URL	/system/backup		
POST	Creates a new backup copy		
Request	**Parameter**	**Data type**	**Values/Comments**
	comment (optional)	string	Comment on the backup copy
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	ID of the backup copy
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Restore backup

URL	/system/backup/<ID>		
PATCH	Enables the function		
Successful call	202 - ACCEPTED	-	-
	Result	**Data type**	**Description**
	RestoreStatusCheckUrl	integer	Restores the backup file and returns a URL.
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Importing the backup copy

URL	/system/backup/import		
POST	Imports an existing backup copy and returns the new ID of this file in the system		
Request	**Parameter**	**Data type**	**Values/Comments**
	file (required)	file upload	SINEMA RC backup copy
	password (optional)	string	Encoding key of the selected backup copy After the import, the password is applied as new encryption key by SINEMA RC.
	comment (optional)	string	Comment on the backup copy
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	ID of the backup copy
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid settings

Deleting a backup copy

URL	/system/backup/<ID>		
DELETE	Deletes the desired backup copy with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the backup copy
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Restoring a backup copy

URL	/system/backup/<ID>		
PATCH	Restores the backup copy with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the backup copy that is to be restored
Successful call	200 - OK	-	-
	202 - Accepted		
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/system/backup/<ID>/status		
GET	Status of the restored backup file		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the backup copy that was restored

Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

4.3.1.16 Shutdown & Restart

Shutting down the SINEMA RC server

URL	/system/power/shutdown		
POST	Shuts down the SINEMA RC server		
Request	**Parameter**	**Data type**	**Values/Comments**
	comment (required)	string	Comment
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights.

Restarting the SINEMA RC server

URL	/system/power/restart		
POST	Restarts the SINEMA RC server		
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights.

4.3.1.17 Auto Logout

Retrieving the WBM session time

URL	/system/settings/autologout		
GET	Returns the time in minutes after which the server will end the session		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	wbmAutoLogoutTime	integer	WBM session time
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Defining the WBM session time

URL	/system/settings/autologout		
POST	Defines the WBM session time after which the server ends the session		
Request	**Parameter**	**Data type**	**Values/Comments**
	wbmAutoLogoutTime (required)	integer	WBM session time

Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

4.3.1.18 Restore factory defaults

Factory settings

URL	/system/timebasedsystemtoken		
GET	Token for restoring the factory settings		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	timeBasedSystemToken	UUID	Token for restoring the factory settings
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Restoring factory settings

System administrator

URL	/system/settings/factoryreset		
POST	Password to perform reset to factory settings		
Request	**URL parameter**	**Data type**	**Values/Comments**
	timeBasedSystemToken	UUID	Token for restoring the factory settings
	Parameter	**Data type**	**Values/Comments**
	password	string	System administrator password
	deleteInactiveBootPartition	integer	Deletes inactive partition
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Restoring factory settings. Time-based system token has expired. Time-based system token was not provided.

4.3.1.19 Server information text

Retrieving the server information text

URL	/system/settings/serverinformation
GET	Returns the server information text

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	showServerInformationOnLogin	boolean	Shows whether the server information text is shown on the login screen.
	showServerInformationOnServer	boolean	Shows whether the server information text is shown in the header.
	serverInformationText	string	Shows the server information text
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Adding the server information text

URL	/system/settings/serverinformation		
POST	Defines the server information text and activates it		
Request At least one show parameter must be enabled.	**Parameter**	**Data type**	**Values/Comments**
	showServerInformationOnLogin (required)	boolean	The server information text is shown on the login screen.
	showServerInformationOnServe(required)	boolean	The server information text is displayed in the header.
	serverInformationText (required)	string	Server information text
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid settings

Deleting the server information text

URL	/system/settings/serverinformation		
DELETE	Deletes the server information text		
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

4.3.2 Remote connections

4.3.2.1 Managing devices

Retrieving devices

URL	/connections/devices?count={integer_value}&search={string}		
GET	Returns all devices with IDs and names as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	count (optional)	integer	Number of results that are to be returned
	search (optional)	string	Search text for device name
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	devices	ListOf<id, name>	Lists all devices with IDs and names. When "search" is specified, only the devices whose name includes the search text are returned.
	count	integer	When the "count" parameter is specified, the first found devices are listed in the specified quantity.
	previous	string	
	next	string	
Failed call	400 - BAD REQUEST	error	Invalid parameters specified
	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/connections/devices/<ID>		
GET	Returns all information for a device		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Device ID

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	name	string	Device name
	vpnAddress	string	VPN address
	subnets: • subnetName • subnetAccessGroups • subnetIp • subnetMask • natMode • virtualSubnetIp • virtualSubnetMask	string	Remote subnet
	nodes: • nodeIp • virtualNodeIp • nodeAccessGroups	string	Virtual subnet
	state	string	Status
	lastChange	string	Date of the last login
	comment	string	Comment
	allAccessGroups	list of groups	All access The members of this subscriber group have access to the subnets and nodes reachable via the device.
	deviceAccessGroups	list of groups	Device access The members of this subscriber group only have access to the device. Access to the subnets or node connected to the device is not possible.
	groups	string	Subscriber groups
	connectionType	string	Type of connection
	smsGatewayProvider	string	SMS gateway provider
	location	string	Location
	deviceType	string	Device type
	gsmNumber	string	GSM number
	vendor	string	Vendor
	fingerprintSha1	string	Hash method for the certificate
	fingerprintSha256	string	Hash method for the certificate
	firmware	string	Last reported firmware version of the device
	lastAutoconfTime	string	Shows when the autoconfiguration file is from
	fallbackFingerprint	string	Fingerprint used by the fallback certificate
	fallbackPort	integer	When reachable, the used fallback port is shown
	fallbackConfirmationStatus	integer	• 0 No information: The device has not yet sent a request for configuration or has a different configuration. • 1 Not confirmed

			A response to automatic configuration was sent to the device but the device has not yet confirmed it. • 2 Confirmed The device confirms that the fallback information was updated and the fallback port can be reached
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/connections/devices/status/<ID>		
GET	Returns the active device status (activated / deactivated)		
Request	**URL parameter**	**Data type**	**Values/Comments**
	id (required)	integer	Device ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	status	boolean	Status • false Device is deactivated • true Device is activated
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Creating a device

URL	/connections/devices
POST	Creates a new device with the general device information. Group and network settings must be added with an additional command

Request	**Parameter**	**Data type**	**Values/Comments**
	name (required)	string	Device name
	password (required)	string	Password
	vendor (optional)	string	Vendor
	type (required)	integer	Device type: 0 = other 1 = SCALANCE 615 / ... 2 = SCALANCE M876 / ... 3 = SCALANCE SC-600 4 = CP 1243-1 / ... 5 = CP 1243-7 LTE 6 = RTU303xC 7 = RTU3010C 8 = SCALANCE MUM8XX 10 = SINEMA RC Edge Client 11 = SCALANCE MUB8XX
	smsGatewayProvider (optional)	integer	ID of the SMS gateway provider Only for M800 Mobile, RTU 303xC, RM1224
	gsmNumber (optional)	string	GSM number Only for M800 Mobile, RTU 303xC, RM1224 Enter the call number of the node to which the wake-up SMS is sent
	senderId (optional)	string	Only with RTU 303xC This ID identifies the SINEMA RC server to the RTU. The ID must also be configured in the RTU.
	location (optional)	string	Location
	comment (optional)	string	Comment
	layer2Network (optional)	integer	Layer 2 network
	connectionType (required)	integer	Type of connection 1 = permanent 2 = digital input 3 = wake up sms (SCALANCE M-800) 4 = wake up sms (RTU 3030) 5 = digital input / wake up sms (SCALANCE M-800)
	fixedVpnAddress (optional)	string	VPN protocol • OpenVPN • IPsec When this parameter is specified, the "Use fixed VPN address" option is activated.
	defaultGateway	boolean	Device is a network gateway

	(optional)		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	Device ID
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	Not enough licenses
	409 - CONFLICT	error	The device already exists.
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Edit device

URL	/connections/devices/<ID>		
PUT	Edits the settings of the device.		
Request	**URL parameter**	**Data type**	**Values/Comments**
	name	string	Device names
	password	string	Password
	vendor	string	Vendor
	type	integer	0 = other 1 = SCALANCE 615 / ... 2 = SCALANCE M876 / ... 3 = SCALANCE SC-600 4 = CP 1243-1 / .. 5 = CP 1243-7 LTE 6 = RTU303xC 7 = RTU3010C 8 = SCALANCE MUM8XX 10 = SINEMA RC Edge Client 11 = SCALANCE MUB8XX
	smsGatewayProvider	integer	Only for M800 Mobile, RTU 303xC, RM1224
	gsmNumber	string	
	senderId	string	Only with RTU 303xC
	location	string	Installation location
	comment	string	Comment
	layer2Network	integer	Layer 2 connection
	connectionType	integer	1 = permanent 2 = digital input 3 = wake up sms (SCALANCE M800) 4 = wake up sms (RTU 3030) 5 = digital input \ wake up sms (SCALANCE M800)
	fixedVpnAddress	string	VPN address
	defaultGateway	boolean	Gateway
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Set device status

URL	/connections/devices/status/<ID>		
PUT	Activates or deactivates the device		
Request	**URL parameter**	**Data type**	**Values/Comments**
	id (required)	integer	Device ID
	Parameter	**Data type**	**Values/Comments**
	status (required)	boolean	Status • false Deactivate device • true Activate device
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	409 - CONFLICT	error	The device is already activated

Deleting a device

URL	/connections/devices/<ID>		
DELETE	Deletes the desired device with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Device ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Retrieving the OpenVPN settings

URL	/connections/devices/connectionparameter?deviceId={integer_value}		
GET	Returns all OpenVPN settings for a device		
Request	**URL parameter**	**Data type**	**Values/Comments**
	deviceId (required)	integer	Device ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	connectionparameters	ListOf<id>	All connection parameters of the device
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/connections/devices/connectionparameter/<ID>
GET	Returns the specified OpenVPN settings for a device

Request	**URL parameter**	**Data type**	**Values/Comments**
	id (required)	integer	Connection parameter ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	ipAddress	string	IP address via which the SINEMA RC server can be reached
	port	integer	Port at which the SINEMA RC server receives the OpenVPN connection
	protocol	string	Protocol for the OpenVPN connection: • TCP • UDP
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Configuring OpenVPN settings

URL	/connections/devices/connectionparameter		
POST	Configures the OpenVPN settings for the specified device		
Request	**Parameter**	**Data type**	**Values/Comments**
	deviceId (required)	integer	Device ID
	ipAddress (required)	string	IP address via which the SINEMA RC server can be reached
	port (required)	integer	Port at which the SINEMA RC server receives the OpenVPN connection
	protocol (required)	string	Protocol for the OpenVPN connection: • TCP • UDP
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	Connection parameter ID
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	The device already exists.
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Deleting an OpenVPN connection

URL	/connections/devices/connectionparameter/<ID>		
DELETE	Deletes the OpenVPN connection parameters with the specified ID from the specified device		
Request	**URL parameter**	**Data type**	**Values/Comments**
	id (required)	integer	Connection parameter ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

4.3.2.2 Assigning the device to a participant group

The subnets and nodes accessible via the device are members of this participant group. You can assign one or more participant groups.

Retrieving participant groups

URL	/connections/devices/groups?deviceId={integer_value}		
GET	Returns all group IDs with the name for the specified device (all access groups)		
Request	**URL parameter**	**Data type**	**Values/Comments**
	deviceId (required)	integer	Device ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	groups	ListOf<id, name>	Lists all group IDs with the name for the device
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Assigning the device to a participant group

URL	/connections/devices/groups		
POST	Adds the group with the specified ID as a participant group to the specified device		
Request	**Parameter**	**Data type**	**Values/Comments**
	deviceId (required)	integer	Device ID
	groupId (required)	integer	Group ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/connections/devices/deviceaccess/groups		
POST	Add device access		
Request	**Parameter**	**Data type**	**Values/Comments**
	deviceId (required)	integer	Device ID
	groupId (required)	integer	Group ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Deleting a device from a participant group

URL	/connections/devices/groups		
DELETE	Removes the group with the specified ID from the specified device		
Request	**Parameter**	**Data type**	**Values/Comments**
	deviceId (required)	integer	Device ID
	groupId (required)	integer	Group ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/connections/devices/deviceaccess/groups		
DELETE	Removes device access		
Request	**Parameter**	**Data type**	**Values/Comments**
	deviceId (required)	integer	Device ID
	groupId (required)	integer	Group ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

4.3.2.3 Managing subnets

Retrieving subnets

URL	/connections/devices/subnets?deviceId={integer_value}		
GET	Returns all subnet IDs with the subnet names for the specified device as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	deviceId (required)	integer	Device ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	subnets	ListOf<id, name>	Lists all subnet IDs for the device
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/connections/devices/subnets/<ID>
GET	Returns all information on a subnet of a device

Request	**URL parameter**	**Data type**	**Values/Comments**
	id (required)	integer	
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	name	string	Name of the subnet
	ipAddress	string	Subnet IP address
	subnetMask	string	Subnet mask
	natMode	integer	NAT mode: • 0 = none • 1 = 1:1 NAT
	virtualIp	string	Virtual subnet IP Empty, when no NAT
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Adding a subnet

URL	/connections/devices/subnets		
POST	Creates a new subnet entry for the specified device. When the IP address overlaps with existing IP address ranges, the error code 422 is output.		
Request	**Parameter**	**Data type**	**Values/Comments**
	deviceId (required)	integer	Device ID
	name (required)	string	Name of the subnet
	ipAddress (required)	string	Subnet IP address
	subnetMask (required)	string	Subnet mask
	natMode (required)	integer	NAT mode: 0 = none 1 = 1:1 NAT
	virtualIp (optional)	string	Virtual subnet IP
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	Subnet ID
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	409 - CONFLICT	error	The subnet already exists.
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Edit subnet

URL	/connections/devices/subnets/<ID>		
PUT	Edits the subnet of the device		
Request	**Parameter**	**Data type**	**Values/Comments**
	name (optional)	string	Name of the subnet
	ipAddress (optional)	string	Subnet IP address
	subnetMask (optional)	string	Subnet mask
	natMode (optional)	integer	NAT mode: 0 = none 1 = 1:1 NAT
	virtualIp (optional)	string	Virtual subnet IP
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	Subnet ID
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

URL	/connections/devices/subnets/portgroups/<ID>		
PUT	Edits the name of the port group		
Request	**Parameter**	**Data type**	**Values/Comments**
	name (optional)	string	Name of the port group
	icmpEnabled (optional)	boolean	Allow or do not allow ICMP packets
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	Port group ID
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Deleting a subnet

URL	/connections/devices/subnets/<ID>
DELETE	Deletes the subnet with the specified ID from the specified device

Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Subnet ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Retrieving the groups of the subnet

URL	/connections/devices/subnets/groups?subnetId={integer_value}		
GET	Returns all group IDs with the names for the specified subnet		
Request	**URL parameter**	**Data type**	**Values/Comments**
	subnetId (required)	integer	Subnet ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	groups	ListOf<id, name>	Lists all group IDs for the device
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Adding a group to the subnet

URL	/connections/devices/subnets/groups		
POST	Adds a group to the specified subnet		
Request	**Parameter**	**Data type**	**Values/Comments**
	deviceId (required)	integer	Device ID
	subnetId (required)	integer	Subnet ID
	groupId (required)	integer	Group ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Deleting a group from the subnet

URL	/connections/devices/subnets/groups
DELETE	Deletes the group with the specified ID from the specified subnet of the specified device

Request	**Parameter**	**Data type**	**Values/Comments**
	deviceId (required)	integer	Device ID
	subnetId (required)	integer	Subnet ID
	groupId (required)	integer	Group ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Retrieve port groups of the subnet

URL	/connections/devices/subnets/portgroups?subnetId={integer_value}		
GET	Returns all port group IDs with the name of the specified subnet as list.		
Request	**URL parameter**	**Data type**	**Values/Comments**
	subnetId (required)	integer	Subnet ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	portgroups	ListOf<id, name>	Lists all port group IDs and names.
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/connections/devices/subnets/portgroups/<ID>		
GET	Returns the port group with the name and the ports		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the port group
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	name	string	Name of the port group
	ports	list	Port (port number, protocol)
	icmpEnabled	boolean	Information on ICMP packets
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Add port group to the subnet

URL	/connections/devices/subnets/portgroups
POST	Adds a port group to the specified device for the specified subnet

Request	**Parameter**	**Data type**	**Values/Comments**
	subnetId (required)	integer	Subnet ID
	name (required)	string	Name of the port group
	icmpEnabled (optional)	boolean	Allow or do not allow ICMP packets
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	ID of the port group
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	409 - CONFLICT	error	The port group already exists
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Edit port group

URL	/connections/devices/subnets/portgroups/<ID>		
PUT	Changes the name of the port group		
Request	**Parameter**	**Data type**	**Values/Comments**
	name	string	Name of the port group
	icmpEnabled (optional)	boolean	Allow or do not allow ICMP packets
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	409 - CONFLICT	error	The port group already exists
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Delete port group

URL	/connections/devices/subnets/portgroups/<ID>		
DELETE	Removes the specified port group		
Request	**Parameter**	**Data type**	**Values/Comments**
	ID	integer	ID of the port group
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Add port to port group

URL	/connections/devices/subnets/portgroups/ports		
POST	Adds a port to the specified port group		
Request	**Parameter**	**Data type**	**Values/Comments**
	portgroupId	integer	ID of the port group
	protocol	string	Protocol (TCP, UDP)
	port	integer	Port number Example: 77, 70-100
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	409 - CONFLICT	error	The port already exists+

Delete port from port group

URL	/connections/devices/subnets/portgroups/ports		
DELETE	Deletes the port from the port group		
Request	**Parameter**	**Data type**	**Values/Comments**
	portgroupId	integer	ID of the port group
	protocol	string	Protocol (TCP, UDP)
	port	integer	Port number Example: 77, 70-100
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Retrieve groups of the port group

URL	/connections/devices/subnets/portgroups/groups?portgroupId={integer_value}		
GET	Returns all group IDs with the names for the specified port group		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	portgroups	ListOf<id, name>	Lists all subscriber groups (ID and group name)
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Add group to port group

URL	/connections/devices/subnets/portgroups/groups
POST	Adds a subscriber group to the port group.

Request	**Parameter**	**Data type**	**Values/Comments**
	portgroupId	integer	ID of the port group
	groupId	integer	ID of the subscriber group
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	409 - CONFLICT	error	The subscriber group already exists
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Delete group from port group

URL	/connections/devices/subnets/portgroups/groups		
DELETE	Deletes the subscriber group from the port group.		
Request	**Parameter**	**Data type**	**Values/Comments**
	portgroupId	integer	ID of the port group
	groupId	integer	ID of the subscriber group
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

4.3.2.4 Managing nodes

Retrieving the nodes of a subnet

URL	/connections/devices/subnets/nodes?subnetId={integer_value}		
GET	Returns all node IDs with the node name of the specified subnet for the specified device		
Request	**URL parameter**	**Data type**	**Values/Comments**
	subnetId (required)	integer	Subnet ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	subnets	ListOf<id, name>	Lists all node IDs of the subnet for a device
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/connections/devices/subnets/nodes/<ID>
GET	Returns all information for a node

Request	**URL parameter**	**Data type**	**Values/Comments**
	id (required)	integer	Node ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	name	string	Node name
	ipAddress	string	IP address of the virtual subnet
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Adding a node to a subnet

URL	/connections/devices/subnets/nodes		
POST	Creates a new node entry for the specified subnet of a device. When the IP address of the node overlaps with existing IP address ranges or does not fit into the address space, the error code 422 is output.		
Request	**Parameter**	**Data type**	**Values/Comments**
	subnetId (required)	integer	Subnet ID
	name (required)	string	Node name
	nodeIp (required)	string	Node IP address
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	Node ID
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	Adding a node to NAT for a subnet of the local host type is not supported via API
	404 - NOT FOUND	error	No entry found
	409 - CONFLICT	error	The node already exists.

Editing a node of a subnet

URL	/connections/devices/subnets/nodes/<ID>		
PUT	Edits the parameters of the node of a subnet		
Request	**URL parameter**	**Data type**	**Values/Comments**
	id (required)	integer	Node ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	name	string	Node name
	ipAddress	string	IP address of the virtual subnet
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Deleting a node from the subnet

URL	/connections/devices/subnets/nodes/<ID>		
DELETE	Deletes the subnet with the specified ID from the specified device		
Request	**URL parameter**	**Data type**	**Values/Comments**
	nodeid (required)	integer	Node ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Retrieve subscriber group of the node

URL	/connections/devices/subnets/nodes/groups?nodeId={integer_value}		
GET	Returns all group IDs with the name of the specified node		
Request	**URL parameter**	**Data type**	**Values/Comments**
	nodeId (required)	integer	Node ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	groups	ListOf<id, name>	Lists all groups of the node
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Add subscriber group to the node

URL	/connections/devices/subnets/nodes/groups		
POST	Adds a subscriber group to the specified node within the subnet of the specified device		
Request	**Parameter**	**Data type**	**Values/Comments**
	nodeId (required)	integer	Node ID
	groupId (required)	integer	Group ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Delete subscriber group from the node

URL	/connections/devices/subnets/nodes/groups
DELETE	Deletes the group with the specified ID from the specified node within the subnet of the specified device

Request	**Parameter**	**Data type**	**Values/Comments**
	groupId (required)	integer	Group ID
	nodeId (required)	integer	Node ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Retrieve port groups of the node

URL	/connections/devices/subnets/nodes/portgroups?nodeId={integer_value}		
GET	Returns all port groups		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	portgroups	ListOf<id, name>	Lists all port groups of the node
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/connections/devices/subnets/nodes/portgroups/<ID>		
GET	Returns the port group with the name and the ports.		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	name	string	Name of the port group
	ports	list	Ports (protocol, port number)
	icmpEnabled (optional)	boolean	ICMP packets allowed or not
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Add port group to the node

URL	/connections/devices/subnets/nodes/portgroups		
POST	Adds a port group to the specified node		
Request	**Parameter**	**Data type**	**Values/Comments**
	nodeId	integer	Node ID
	name	string	Name of the port group
	icmpEnabled (optional)	boolean	Allow or do not allow ICMP packets
Successful call	200 - OK	-	
	Result	**Data type**	**Description**
	id	integer	ID of the port group

Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	409 - CONFLICT	error	The port group already exists
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Edit port group of the node

URL	/connections/devices/subnets/nodes/portgroups/<ID>		
PUT	Changes the name of the port group.		
Request	**Parameter**	**Data type**	**Values/Comments**
	name	string	Name of the port group
	icmpEnabled (optional)	boolean	Allow or do not allow ICMP packets
Successful call	200 - OK	-	
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	409 - CONFLICT	error	The port group already exists
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Delete port group from the node

URL	/connections/devices/subnets/nodes/portgroups/<ID>		
DELETE	Removes the port group from the node.		
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/connections/devices/subnets/nodes/portgroups/ports		
DELETE	Removes the port from the port group of the node.		
Request	**Parameter**	**Data type**	**Values/Comments**
	portgroupId	integer	ID of the port group
	protocol	string	Protocol (TCP, UDP)
	port	integer	Port Example: 77, 70-100
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Retrieve subscriber group of the port group

URL	/connections/devices/subnets/nodes/portgroups/groups?portgroupId={integer_value}		
GET	Returns all subscriber groups of the port group		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	portgroups	ListOf<id, name>	Lists all subscriber groups of the port group
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Add subscriber group to the port group

URL	/connections/devices/subnets/nodes/portgroups/groups		
POST	Adds a subscriber group to the port group of the node.		
Request	**Parameter**	**Data type**	**Values/Comments**
	portgroupId	integer	ID of the port group
	groupid	integer	ID of the subscriber group
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	409 - CONFLICT	error	The subscriber group already exists
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Delete subscriber group from the port group

URL	/connections/devices/subnets/nodes/portgroups/groups		
DELETE	Removes a subscriber group from the port group of the node.		
Request	**Parameter**	**Data type**	**Values/Comments**
	portgroupId	integer	ID of the port group
	groupid	integer	ID of the subscriber group
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

4.3.2.5 Sending a Wake-up SMS

Retrieving a wake-up SMS

URL	/connections/devices/wakeup/<deviceID>		
GET	Returns information on the wake-up SMS status on the specified device		
Request	**URL parameter**	**Data type**	**Values/Comments**
	deviceID (required)	integer	Device ID
Successful call	200 - OK	-	-
	202 - ACCEPTED	-	-
	Result	**Data type**	**Description**
	detail	string	Detail
	wakeUpSmsStatusCheck-Url	string	URL of the wake-up SMS status
	status	boolean	Status of the sent wake-up SMS
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Sending a wake-up SMS

URL	/connections/devices/wakeup		
POST	If the device is not connected, the SINEMA RC server sends a wake-up SMS to the device. Only available with the type of connection "Wake-up SMS" or "Wake-up SMS & Digital input".		
Request	**Parameter**	**Data type**	**Values/Comments**
	deviceId (required)	integer	Device ID
	wakeupTime (optional)	string	Only with RTU 303xC: Schedule for wake-up SMS
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	details	string	If the device is online, it is set to idle state. If the device is offline, it is set to idle state.
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	422 - UNPROCESSABLE ENTRY	error	The device does not support wake-up SMS

4.3.2.6 Managing Firmware

Retrieving a firmware file

URL	/connections/devices/fw		
GET	Returns all information on the currently uploaded firmware file. Empty if no file has been uploaded.		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	firmwareFile	file upload	Name of the uploaded firmware file
	version	integer	Version of the uploaded firmware file
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/connections/devices/activatefw?count={integer_value}&search={string}		
GET	Returns all devices with IDs, device names, the last known firmware version, the last known request of the firmware and the status as list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	count (optional)	integer	Number of results that are to be returned
	search (optional)	string	Search text for device name
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	devices	ListOf<id, deviceName, lastKnownFw, lastKnownRequest, deviceLocation, deviceComment, status>	Lists all devices with IDs, device names, the last known firmware version, the last known request of the firmware and the status (online / offline). When "search" is specified, only the devices whose name includes the search text are returned.
	count	integer	When the "count" parameter is specified, the first found devices are listed in the specified quantity.
	previous	string	
	next	string	
Failed call	400 - BAD REQUEST	error	Invalid parameters specified
	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Uploading a firmware file

URL	/connections/devices/fw
POST	Uploads the firmware file of the device or overwrites the last loaded version

Request	**Parameter**	**Data type**	**Values/Comments**
	firmwareFile (required)	file upload	Firmware file of the device
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Deleting a firmware file

URL	/connections/devices/activatefw/<ID>		
DELETE	Removes the firmware file with the specified ID from the specified device		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Device ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	409 - CONFLICT	error	The firmware was already deactivated on this device

Activating a firmware file

URL	/connections/devices/activatefw/<ID>		
PUT	Activates the firmware download with the specified ID on the specified device		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Device ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	409 - CONFLICT	error	The firmware was already activated on this device
	422 - UNPROCESSABLE ENTRY	error	Device is not supported for the firmware upload

Activate device

URL	/connections/devices/activate/<ID>		
PUT	Activates devices after completion of the trial phase or after backup restore		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Device ID
Successful call	200 - OK	-	-

Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	Resource forbidden
	404 - NOT FOUND	error	No entry found
	409 - CONFLICT	error	The device is already active
	422 - UNPROCESSABLE ENTRY	error	Device is not supported for the firmware upload

Retrieve device status

URL	/connections/devices/activate/<ID>		
GET	Returns all information on the device after completion of the trial phase or after backup restore		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Device ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	name	string	Name of the device
	licenseStatus	integer	License status, passive or active
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

4.3.2.7 Managing communication relations between participant groups

Retrieving communication relations

URL	/connections/groups/destination/<ID>		
GET	Returns all communication relations of a source group		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Group ID (source group)
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	groups	ListOf<id, name>	Lists all communication relations of a source group with IDs and names of the destination groups
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights.
	404 - NOT FOUND	error	No entry found

Creating communication relations

URL	/connections/groups/destination/<ID>		
POST	Creates a new communication relation to a destination group		
Request	**URL parameter**	**Data type**	**Values/Comments**
	destinationId (required)	integer	Destination group ID
Successful call	200 - OK	-	-

Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights.
	404 - NOT FOUND	error	Source group not found
	404 - NOT FOUND	error	Destination group not found
	409 - CONFLICT	error	The specified destination group ID is already in the destination list.
	422 - UNPROCESSABLE ENTRY	error	A valid entry is required.

Deleting a communication relation to a destination group

URL	/connections/groups/destination/<ID>		
DELETE	Deletes the desired destination group with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	destinationId (required)	integer	Destination group ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights.
	404 - NOT FOUND	error	Source group not found
	404 - NOT FOUND	error	The specified destination group ID is not included in the destination list.
	422 - UNPROCESSABLE ENTRY	error	A valid entry is required.

4.3.3 Local connections

Retrieve the local connection settings

URL	/localconnections/connections?interface={integer_value}		
GET	Returns all local connection settings		
Request	**URL parameter**	**Data type**	**Values/Comments**
	interface (required)	integer	LAN interface that is called
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	groups	ListOf<id, name>	Lists all subscriber groups with IDs and names
	subnets: • subnetName • subnetIp • netmask • groups	 string string string ListOf<id, name>	Provides information about subnets that can be reached via the local interface

Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Managing a group of the local connection

Retrieving a group of the local connection

URL	/localconnections/connections/groups?interface={integer_value}		
GET	Returns all subscriber group information as a list for the specified local connection		
Request	**URL parameter**	**Data type**	**Values/Comments**
	interface (required)	integer	LAN interface that is called
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	groups	ListOf<id, name>	Lists all subscriber groups with IDs and names for the LAN interface
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Adding a group to the local connection

URL	/localconnections/connections/groups/		
POST	Adds a group with the specified ID as a subscriber group to the specified local connection		
Request	**Parameter**	**Data type**	**Values/Comments**
	interface (required)	integer	LAN interface
	groupId (required)	integer	ID of the subscriber group
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Removing a group from the local connection

URL	/localconnections/connections/groups/		
DELETE	Removes the subscriber group with the specified ID from the specified local connection		
Request	**Parameter**	**Data type**	**Values/Comments**
	interface (required)	integer	LAN interface
	groupId (required)	integer	ID of the subscriber group
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Managing local subnets

Retrieving subnets of the local connection

URL	/localconnections/connections/subnets?interface={integer_value}		
GET	Returns all information about subnets as a list, which are reachable via the specified local connection		
Request	**URL parameter**	**Data type**	**Values/Comments**
	interface (required)	integer	LAN interface
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	subnets	ListOf<id, name>	Lists all subnet IDs of the local connection
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/localconnections/connections/subnets/<ID>		
GET	Returns all information about a subnet of the local connection		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	name	string	Name of the local subnet
	subnetIp	string	IP address of the local subnet
	netmask	string	Netmask of the subnet
	gateway	string	Gateway via which the subnet can be reached
	groups	ListOf<id, name>	Lists all subscriber group IDs for the subnet
	nodes	ListOf<id, name>	Lists all end device IDs for the subnet
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Adding subnets to the local connection

URL	/localconnections/connections/groups/		
POST	Adds the destination network to the specified local connection		
Request	**Parameter**	**Data type**	**Values/Comments**
	interface (required)	integer	LAN interface
	destinationNetwork (required)	integer	Destination network (subnet)
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Removing subnets from the local connection

URL	/localconnections/connections/subnets/<ID>		
DELETE	Removes the destination network with the specified subnet ID from the local connection		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the subnet
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Managing a subscriber group of the local subnet

Retrieving a group of the local subnet

URL	/localconnections/connections/subnets/groups?subnetId={integer_value}		
GET	Returns all subscriber group information for the specified subnet of the local connection as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	subnetId (required)	integer	ID of the calling subnet
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	groups	ListOf<id, name>	Lists all subscriber groups of the local subnet with IDs
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Adding a group to the local subnet

URL	/localconnections/connections/subnets/groups		
POST	Adds a subscriber group to the specified local subnet		
Request	**Parameter**	**Data type**	**Values/Comments**
	subnetId (required)	integer	ID of the subnet
	groupId (required)	integer	ID of the subscriber group
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Removing a group from the local subnet

URL	/localconnections/connections/subnets/groups
DELETE	Removes the subscriber group from the specified local subnet

Request	**Parameter**	**Data type**	**Values/Comments**
	subnetId (required)	integer	ID of the subnet
	groupId (required)	integer	ID of the subscriber group
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Managing end devices of the local subnet

Retrieving an end device on the local subnet

URL	/localconnections/connections/subnets/nodes?subnetId={integer_value}		
GET	Returns all local end device IDs of the local connection as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	subnetId (required)	integer	ID of the calling subnet
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	nodes	ListOf<id, name>	Lists all end device IDs in the retrieved local subnet
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/localconnections/connections/subnets/nodes/<ID>		
GET	Returns all information about the local end device with the given ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the end device
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	name	string	Name of the end device
	groups	ListOf<id, name>	ID of the subscriber group
	nodeIp	string	End device IP address
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Adding an end device to local subnet

URL	/localconnections/connections/subnets/nodes
POST	Adds an end device to the specified local subnet

Request	**Parameter**	**Data type**	**Values/Comments**
	subnetId (required)	integer	ID of the subnet
	name (required)	string	Name of the end device
	groupId (required)	integer	ID of the subscriber group
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Edit end device

URL	/localconnections/connections/subnets/nodes/<ID>		
POST	Edits the parameters		
Request	**Parameter**	**Data type**	**Values/Comments**
	name	string	Name of the end device
	nodeIp	string	End device IP address
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Retrieving an end device from local subnet

URL	/localconnections/connections/subnets/nodes/<ID>		
DELETE	Removes the end device from the local subnet		
Request	**Parameter**	**Data type**	**Values/Comments**
	nodeId (required)	integer	ID of the end device
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Managing a subscriber group of the end device

Retrieving an end device group of the local subnet

URL	/localconnections/connections/subnets/nodes/groups?nodeId={integer_value}		
GET	Returns all subscriber groups of the local end device with the end device IDs as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	nodeId (required)	integer	ID of the end device to be retrieved

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	groups	ListOf<id, name>	Lists all subscriber end device groups of the local subnet
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Adding an end device group to the local subnet

URL	/localconnections/connections/subnets/nodes/groups		
POST	Adds a subscriber group to the specified end device of the local subnet		
Request	**Parameter**	**Data type**	**Values/Comments**
	nodeId (required)	integer	ID of the end device
	groupId (required)	integer	ID of the subscriber group
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Removing an end device group from the local subnet

URL	/localconnections/connections/subnets/nodes/groups		
DELETE	Removes the subscriber group of the end device from the specified local subnet		
Request	**Parameter**	**Data type**	**Values/Comments**
	nodeId (required)	integer	ID of the end device in the local subnet
	groupId (required)	integer	ID of the subscriber group
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

4.3.3.1 Managing nodes

Retrieve end devices of a local connection

URL	/localconnections/connections/subnets/nodes?subnetId={integer_value}		
GET	Returns all end device IDs with the end device name of the local connection		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	nodes	ListOf<id, name>	Lists all end devices
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Add end device to a local connection

URL	/localconnections/connections/subnets/nodes		
POST	Adds an end device to the local connection		
Request	**Parameter**	**Data type**	**Values/Comments**
	subnetId (required)	integer	Subnet ID
	name (required)	string	Node name
	nodeIp (required)	integer	Node ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Edit end device of a local connection

URL	/localconnections/connections/subnets/nodes/<ID>		
PUT	Edits the parameters of the node from the local connection		
Request	**Parameter**	**Data type**	**Values/Comments**
	name	string	Node name
	nodeIp	integer	Node ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Delete end device of a local connection

URL	/localconnections/connections/subnets/nodes/<ID>		
DELETE	Removes the end device from the local connection		
Request	**URL parameter**	**Data type**	**Values/Comments**
	name	string	Node name
	nodeip	integer	Node ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	Deleting a node from NAT for a subnet of the local host type is not supported via API
	404 - NOT FOUND	error	No entry found
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Retrieving groups of the node

URL	/connections/devices/subnets/nodes/groups?nodeId={integer_value}		
GET	Returns all group IDs with the name of the specified node		
Request	**URL parameter**	**Data type**	**Values/Comments**
	nodeId (required)	integer	Node ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	groups	ListOf<id, name>	Lists all group IDs of the node
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Adding a group to the node

URL	/connections/devices/subnets/nodes/groups		
POST	Adds a group to the specified node within the subnet of the specified device		
Request	**Parameter**	**Data type**	**Values/Comments**
	nodeId (required)	integer	Node ID
	groupId (required)	integer	Group ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Deleting a group from the node

URL	/connections/devices/subnets/nodes/groups		
DELETE	Deletes the group with the specified ID from the specified node within the subnet of the specified device		
Request	**Parameter**	**Data type**	**Values/Comments**
	groupId (required)	integer	Group ID
	nodeId (required)	integer	Node ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

4.3.4 Connection Management

4.3.4.1 Managing participant groups

Retrieving participant groups

URL	/connections/groups?count={integer_value}&search={string}		
GET	Returns all participant groups with IDs and names as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	count (optional)	integer	Number of results that are to be returned
	search (optional)	string	Search text for group name
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	groups	ListOf<id, name>	Lists all participant groups with IDs and names. When "search" is specified, only the participant groups whose name includes the search text are returned.
	count	integer	When the "count" parameter is specified, the first found participant groups are listed in the specified number.
	previous	string	
	next	string	
Failed call	400 - BAD REQUEST	error	Invalid parameters specified
	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/connections/groups/<ID>		
GET	Returns all information for a participant group		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Group ID

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	name	string	Group name
	description	string	Description
	communicationAllowed	boolean	false = Members may not communicate (default) true = Members may communicate with each other
	lan1	boolean	LAN interface via which the VPN tunnel can be reached: true = Communication to LAN 1
	lan2	boolean	true = Communication to LAN 2
	lan3	boolean	true = Communication to LAN 3
	lan4	boolean	true = Communication to LAN 4
	roles	Lis- tOf<id ,na me>	Roles assigned to the group
	users	ListOf<id, name>	Users assigned to the group
	devices	ListOf<id, name>	Devices assigned to the group
	subnets	ListOf<id, name>	Subnets assigned to the group
	nodes	ListOf<id, name>	Nodes assigned to the group
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Creating participant groups

URL	/connections/groups		
POST	Creates a new participant group		
Request	**Parameter**	**Data type**	**Values/Comments**
	name (required)	string	Group name
	description (optional)	string	Description
	communicationAllowed (optional)	boolean	false = Members may not communicate (default) true = Members may communicate with each other
	lan1 (optional)	boolean	LAN interface via which the VPN tunnel can be reached: true = activates the communication to LAN 1
	lan2 (optional)	boolean	true = activates the communication to LAN 2
	lan3 (optional)	boolean	true = activates the communication to LAN 3
	lan4 (optional)	boolean	true = activates the communication to LAN 4
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	Group ID

Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	The group already exists
	422 - UNPROCESSABLE ENTRY	error	Entry required

Edit subscriber groups

URL	/connections/groups/<ID>		
PUT	Edits the desired subscriber group with the specified ID		
Request	**Parameter**	**Data type**	**Values/Comments**
	name	string	Group name
	description	string	Description
	communicationAllowed	boolean	false = Members may not communicate (default) true = Members may communicate with each other
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	Group ID
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Deleting a participant group

URL	/connections/groups/<ID>		
DELETE	Deletes the desired participant group with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Group ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

4.3.5 Layer 2

4.3.5.1 Settings

Retrieve Layer 2 configuration setting

URL	/layer2/settings
GET	Returns the settings of the Layer 2 configuration.

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	vxlanEnabled	boolean	Status Layer-2 • true = activate • false = deactivate
	vniStartId	integer	VNID
	vniPort	integer	Current port number that is used for Layer 2 connections.
	clientStartMac	string	Client start MAC address
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Define Layer 2 settings

URL	/layer2/settings		
POST	Specifies the settings of the Layer 2 configuration.		
Request	**Parameter**	**Data type**	**Values/Comments**
	startTrial (optional)	boolean	• true = Start of demo version
	vxlanEnabled (required)	boolean	Activate Layer 2 • true = activate • false = deactivate
	vniStartId (optional)	integer	VNID Range: 1000000 ... 16777216
	vniPort (optional)	integer	Current port number that is used for Layer 2 connections. Range: 4789 ... 65535
	clientStartMac (optional)	string	Client start MAC address
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	Resource forbidden
	409 - CONFLICT	error	The user is already deactivated
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Change Layer 2 settings

URL	/layer2/settings
PUT	Adds settings

Request	**Parameter**	**Data type**	**Values/Comments**
	vxlanEnabled	boolean	Status Layer-2 • true = activate • false = deactivate
	vniStartId	integer	VNID
	vniPort	integer	Current port number that is used for Layer 2 connections.
	clientStartMac	string	Client start MAC address
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	Resource forbidden
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

4.3.5.2 Network

Retrieve settings of the Layer 2 network

URL	/layer2/networks		
GET	Retrieve settings of the Layer 2 network		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	ID of the Layer 2 network
	name	string	Name of the Layer 2 network
	endpointType	integer	• 0 Device: Layer 2 communication takes place on the SCALANCE devices • 1 SRC Server: Layer 2 communication via the SRC server
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/layer2/networks/<ID>		
GET	Returns all information on a Layer 2 network		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Layer 2 device ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	ID of the Layer 2 network
	name	string	Name of the Layer 2 network
	endpointType	integer	Type of end point (device or SINEMA RC)
	assignedDeviceIds	list of Integers	List of the assigned devices
	assignedUsersIds	list of Integers	List of the assigned user IDs
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Create Layer 2 network

URL	/layer2/networks		
POST	Creates a Layer 2 network		
Request	**Parameter**	**Data type**	**Values/Comments**
	name	string	Name of the Layer 2 network, max. 100 characters
	endpointType	integer	• 0 SRC server disabled: Layer 2 communication takes place on the SCALANCE device. • 1 SRC server enabled: Layer 2 communication takes place via the SINEMA RC server.
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	ID of the Layer 2 network
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	The group already exists
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Delete Layer 2 network

URL	/layer2/networks/<ID>		
DELETE	Deletes the desired Layer 2 network		
Request	**URL parameter**	**Data type**	**Values/Comments**
	id (required)	integer	ID of the Layer 2 network
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

4.3.5.3 Devices

Retrieve Layer 2 device settings

URL	/layer2/devices		
GET	Returns all devices with Layer 2 as a list		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	ID of the Layer 2 connection
	status	integer	0: Layer 2 not connected 1: Layer 2 connection is established
	deviceId	integer	Device ID
	name	string	Name of the device
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/layer2/devices/<ID>		
GET	Returns all information on a Layer 2 device		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Layer 2 device ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	ID of the Layer 2 connection
	status	integer	0: Layer 2 not connected 1: Layer 2 connection is established
	deviceId	integer	Device ID
	name	string	Name of the device
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/layer2/devices/communicationnodes?deviceId={integer}		
GET	Lists the nodes that can be reached via the Layer 2 device.		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	Device ID
	name	string	Device name
	ipAddress	ipv4 string	Node IP address
	macAddress	string	Node MAC address
	profinetName	string	PROFINET name
	profinetType	string	PROFINET device type of the node
	groupId	string	ID of the subscriber group
	comment	string	--
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/layer2/devices/foundnodes?deviceId={integer}		
GET	Lists the nodes that were found via the DCP Discovery Scan.		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	Device ID
	name	string	Device name
	ipAddress	ipv4 string	Node IP address
	macAddress	string	Node MAC address
	profinetName	string	PROFINET name
	profinetType	string	PROFINET device type of the node

Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Start DCP Discovery Scan

URL	/layer2/devices		
POST	Starts a DCP Discovery Scan.		
Request	**URL parameter**	**Data type**	**Values/Comments**
	id (required)	integer	Layer 2 device ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Add a node to the table of allowed communication

URL	/layer2/devices/foundnodes		
POST	Adds a node to the table for permissible communication.		
Request	**Parameter**	**Data type**	**Values/Comments**
	deviceId (required)	integer	Device ID
	nodeId (required)	integer	ID of the found node
	nodeName (required)	string	Name of the found node
	ipAddress (required)	ipv4 string	IP address of the found node
	macAddress	string	MAC address
	profinetName	string	PROFINET name
	profinetType	string	PROFINET device type of the node
	comment	string	Comment
	groupIds	integer	ID of the subscriber group
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Delete nodes from the Layer 2 device

URL	/layer2/devices/communicationnodes
DELETE	Deletes nodes from permitted Layer2 devices

Request	**URL parameter**	**Data type**	**Values/Comments**
	deviceId (required)	integer	Device ID
	nodeId (required)	integer	ID of the found node
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

4.3.6 User accounts

4.3.6.1 Managing users

Retrieving users

URL	/accounts/users?count={integer_value}&search={string}		
GET	Returns all users with IDs and names as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	count (optional)	integer	Number of results that are to be returned
	search (optional)	string	Search text for user name
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	users	ListOf<id, name>	Lists all users with IDs and names. When "search" is specified, only users whose name includes the search text are returned.
	count	integer	When the "count" parameter is specified, the first found users are listed in the specified number.
Failed call	400 - BAD REQUEST	error	Invalid parameters specified
	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/accounts/users/<ID>		
GET	Returns all information for a user		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	User ID

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	username	string	User name
	loginMethod	integer	1 = Password 2 = PKI
	firstname	string	First name of the user
	lastname	string	Last name of the user
	phone	string	Phone number of the user
	pkiFilterRule	string	PKI DN filter rule for the authentication Only permissible when loginMethod = 2
	email	string	Email address of the user
	roles	ListOf<id, name>	List with all roles of the user
	rights	ListOf<id, name>	A comma-separated list with all user rights Example: "global_privileges": [1, 4, 5], "group_privileges": [16] } Rights: global_privileges: • 1 = manage users and roles • 2 = create backup copies • 3 = restore the system • 4 = edit system parameters • 5 = manage devices • 6 = manage address spaces • 7 = manage remote connections • 8 = certificate management • 9 = manage firmware updates • 10 = force comment • 11 = download client software • 12 = See Logfile message • 13 = Read-only full access • 14 = Manage Layer-2 group_privileges • 15 = Manage own devices • 16 = Manage own users • 17 = Manage own remote connections • 18 = Manage own Layer-2

	groups	ListOf<id, name>	List with all groups of the user
	state	boolean	Status: • false = offline • true = online
	lastLogin	string	UTC time stamp of the last login
	createdSince	string	UTC time stamp of the account creation
	vpnAddress	string	The IP address of the device used during communication via VPN. The address is automatically assigned by SINEMA RC. If communication via VPN is not active, "none" is displayed; if it is active, the current VPN address is displayed.
	layer2Configuration	string	Layer 2 configuration status of the user
	macAddress	string	Layer 2 MAC address for the new user
	layer2Activated	boolean	Layer 2 status for the new user
	password_policy	string	Password policy of the user
	2FA	boolean	2-factor authentication status
	sessionPolicy	string	Information on the user's session policies
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/accounts/users/status/<ID>		
GET	Returns the active user status (activated / deactivated)		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	User ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	status	boolean	Status • false: User is deactivated • true: User is activated
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Add user

URL	/accounts/users
POST	Creates a new user

Request	**Parameter**	**Data type**	**Values/Comments**
	username (required)	string	User name
	firstName (optional)	string	First name of the user
	lastName (optional)	string	Last name of the user
	phone (optional)	string	Phone number of the user
	email (optional)	string	Email address of the user
	loginMethod (required)	integer	1 = password 2 = PKI
	pkiFilterRule (optional)	string	PKI DN filter rule for the authentication Only permissible when loginMethod = 2
	rights (optional)	list of <integer>	A comma-separated list with all user rights Example: "rights":[1,5] Rights: • 1 = manage users and roles • 2 = create backup copies • 3 = restore the system • 4 = edit system parameters • 5 = manage devices • 6 = manage address spaces • 7 = manage remote connections • 8 = certificate management • 9 = manage firmware updates • 10 = force comment • 11 = download client software • 12 = See Logfile message • 13 = Read-only full access • 14 = Manage Layer-2
	macAddress (optional)	string	Layer 2 MAC address for the new user
	layer2Activated (optional)	boolean	Layer 2 status for the new user
	fixedVpnAddress (optional)	string	The IP address always assigned to the user. When this parameter is set, it is used as fixed IP address and the "Use fixed VPN address" option is activated.
	password (required)	string	Password for the new user
Successful call	200 - OK	-	-

Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	Not enough licenses
	404 - NOT FOUND	error	No entry found
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Edit user

URL	/accounts/users/users/<ID>
PUT	Edits the user with the specified ID

Request	**Parameter**	**Data type**	**Values/Comments**
	username (optional)	string	User name
	firstName (optional)	string	First name of the user
	lastName (optional)	string	Last name of the user
	phone (optional)	string	Phone number of the user
	email (optional)	string	Email address of the user
	loginMethod (optional)	string	PKI DN filter rule for the authentication
	pkiFilterRule (optional)	string	PKI DN filter rule for the authentication Only permissible when loginMethod = 2
	rights (optional)	list of <integer>	A comma-separated list with all user rights Example: "rights":[1,5] Rights: • 1 = manage users and roles • 2 = create backup copies • 3 = restore the system • 4 = edit system parameters • 5 = manage devices • 6 = manage address spaces • 7 = manage remote connections • 8 = certificate management • 9 = manage firmware updates • 10 = force comment • 11 = download client software • 12 = See Logfile message • 13 = Read-only full access • 14 = Manage Layer-2
	macAddress (optional)	string	Layer 2 MAC address for the new user
	layer2Activated (optional)	boolean	Layer 2 status for the new user
	fixedVpnAddress (optional)	string	The IP address always assigned to the user. When this parameter is set, it is used as fixed IP address and the "Use fixed VPN address" option is activated.
	password (optional)	string	Password for the new user
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	User ID

Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Setting the user status

URL	/accounts/users/status/<ID>		
PUT	Activates and deactivates the user with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	User ID
	Parameter	**Data type**	**Values/Comments**
	status (required)	boolean	Status • false: User is deactivated • true: User is activated
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	Resource forbidden
	404 - NOT FOUND	error	No entry found
	409 - CONFLICT	error	The user is already deactivated

Deleting users

URL	/accounts/users/<ID>		
DELETE	Deletes the desired user with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	User ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

4.3.6.2 Creating and managing roles

Retrieving roles

URL	/accounts/roles?count={integer_value}&search={string}		
GET	Returns all roles with IDs and names as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	count (optional)	integer	Number of results that are to be returned
	search (optional)	string	Search text for role name

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	roles	ListOf<id, name>	Lists all roles with IDs and names. When "search" is specified, only the roles whose name includes the search text are returned.
	count	integer	When the "count" parameter is specified, the first found roles are listed in the specified number.
Failed call	400 - BAD REQUEST	error	Invalid parameters specified
	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/accounts/roles/<ID>		
GET	Returns all information for a role		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Role ID

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	name	string	Role name
	rights	ListOf<integer>	List with all rights of the user • 1 = manage users and roles • 2 = create backup copies • 3 = Restore the system • 4 = Edit system paremters • 5= Manage devices • 6 = Manage address spaces • 7 = Manage remote connections • 8 = Certificate management • 9 = Manage firmware updates • 10 = Force commente • 11 = Download client software • 12 = See Logfile messages • 13 = Read-only full access • 14 = Manage Layer-2 • 15 = Manage own devices • 16 = Manage own users • 17 = Manage own remote connections • 18 = Manage own Layer-2
	groups	ListOf<id, name>	List with all groups of the user
	passwordExpire	integer	Password expires (in days)
	passwordExpirationInfiniteTolerance	boolean	When this is enabled, users can log in after password expiry and are forwarded to the "Change Password" page.
	passwordExpirationToleranceTime	integer	How long the password can be used after expiry
	enablePasswordReset	boolean	The "Reset password" link is shown on the login page
	passwordResetLinkValidTime	integer	Password reset link validity
	passwordReusing	integer	Reusing the same password
	passwordChangeFirstLogin	boolean	The password needs to be changed on first login or not.
	pkiDnFilterRule	string	PKI DN filter rule for this role
	enable2FA	boolean	Activates 2-factor authentication
	enableRememberToken	boolean	When enabled, the one-time token is saved.
	rememberTokenTime	integer	Duration of validity (in days)
	accessTokenTimeout	integer	Specifies the validity of the new access token in minutes.
	sessionNeverExpires	boolean	Specifies whether the session expires or not.
	refreshTokenTimeout	integer	Specifies the duration of the session in hours.
	pkiDeleteTmpUser	integer	Delete temporary users (in hours)
	umcUserGroup	string	UMC user group

			When this value is empty, the UMC policy is deactivated.
	umcDeleteTmpUser	integer	Delete temporary UMC user (in hours)

	oidcPolicies	List of *oidc entries	"oidcPolicies":[{
			"oidcKeyInToken":"roles",
			"oidcValueInToken":"SINEMARC_role3"
			},
			{
			"oidcKeyInToken":"roles1",
			"oidcValueInToken":"SINEMARC_role4"
			}
			]

	*oidcRoleKeyInToken	string	Key of the OAuth/OpenId application role
	*oidcRoleValueInToken	string	Value of the OAuth/OpenId application role
	oidcRelationBetween-Claims	integer	Relationship between claims • AND: The specified claims must be fulfilled. • OR: At least one of the specified clients must be fulfilled.
	oidcDeleteTmpUser	integer	Delete temporary user (in days).
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Creating a new role

URL	/accounts/roles
POST	Creates a new role

Request	Parameter	Data type	Values/Comments
	name (required)	string	Name of the new role
	rights (optional)	ListOf<integer> Example: "global_privileges": [1, 4, 5], "group_privileges": [16] }	A list separated by comma of all rights for the new user • 1 = manage users and roles • 2 = create backup copies • 3 = Restore the system • 4 = Edit system paremters • 5= Manage devices • 6 = Manage address spaces • 7 = Manage remote connections • 8 = Certificate management • 9 =Manage firmware updates • 10 = Force commente • 11 = Download client software • 12 = See Logfile messages • 13 = Read-only full access • 14 = Manage Layer-2 • 15 = Manage own devices • 16 = Manage own users • 17 = Manage own remote connections • 18 = Manage own Layer-2
	passwordExpire (optional)	integer	Password expires (in days): • 0 = Never (set as default) • 30 days • 90 days • 360 days When no value is defined, the default value (0 = Never) is used. 14 days before expiry the user receives an e-mail. Requirement: • An e-mail address is configured for the user • The SMTP client is configured
	passwordExpiratio-nInfiniteTolerance (optional)	boolean	• True: Users can still log in after password expiry and are forwarded to the "Change password" page • False: Setting disabled
	passwordExpira-tionToleranceTime (optional)	integer	Defines how long the password can be used after expiry.
	enablePasswordRe-set (optional)	boolean	• True: The "Reset password" link is shown on the login page • False:

		Setting disabled
passwordReset-LinkValidTime (optional)	integer	Password reset link validity
passwordReusing (optional)	integer	Reusing the same password: • 0: The setting is disabled • 1 - 5: If, for example, you enter 3, the current password can be reused only after 3 different passwords. As default, 3 is set. If no value is specified, the default value 3 is used.
passwordChange-FirstLogin (optional)	boolean	The password needs to be changed on first login or not.
pkiDnFilterRule (optional)	string	PKI DN filter rule for this role that is checked for on login. The attributes of the names (Distinguished Name acc. to the X.509 standard) are used as filter criteria. This requires that the attributes are included in the PKI certificate of the user.
pkiDeleteTmpUser (optional)	integer	Delete temporary users (in hours) • 0: The setting is disabled. The temporary user must be deleted manually. • 1 - 72 hours: When the time expires, the temporary user is deleted. As default, 24 is set. If no value is specified, the default value 24 is used.
enable2FA (optional)	boolean	Activates 2-factor authentication

oidcPolicies	List of *oidc entries	"oidcPolicies":[{
		"oidcKeyInToken":"roles",
		"oidcValueInToken":"SINEMARC_role3"
		},
		{
		"oidcKeyInToken":"roles1",
		"oidcValueInToken":"SINEMARC_role4"
		}
		]

*oidcRoleKeyInToken	string	Key of the OAuth/OpenId application role
*oidcRoleValueInToken	string	Value of the OAuth/OpenId application role
oidcRelationBetweenClaims	integer	Defines the relationship between claims. • AND: The specified claims must be fulfilled. • OR: At least one of the specified clients must be fulfilled.
oidcDeleteTmpUser	integer	Delete temporary user (in days).
enableRememberToken (optional)	boolean	When enabled, the one-time token is saved.

	rememberToken-Time (optional)	integer	Specifies the duration of validity (in days). When the "enableRememberToken" function is enabled via the API without time value, the default value (1) is used.
	accessTokenTime-out (optional)	integer	Specifies the validity of the new access token in minutes.
	sessionNeverEx-pires (optional)	boolean	Specifies whether the session expires or not.
	refreshTokenTime-out (optional)	integer	Specifies the duration of the session in hours.
	umcUserGroup (optional)	string	UMC user group The entered name should match the name on the UMC server. When this value is specified, login via UMC is activated.
	umcDeleteTmpUs-er (optional)	integer	Delete temporary UMC user (in hours) • 1 - 9999 hours When login via UMC is activated but this value is not specified, the default value 30 is used.
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	Role ID
Failed call	401 - UNAUTHOR-IZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	The role already exists.

Change role

URL	/accounts/roles/<ID>
PUT	Changes the properties of the role

Request	Parameter	Data type	Values/Comments
	name	string	Name of the new role
	rights	ListOf<integer>	A list separated by comma of all rights for the new user • 1 = manage users and roles • 2 = create backup copies • 3 = Restore the system • 4 = Edit system paremters • 5= Manage devices • 6 = Manage address spaces • 7 = Manage remote connections • 8 = Certificate management • 9 =Manage firmware updates • 10 = Force commente • 11 = Download client software • 12 = See Logfile messages • 13 = Read-only full access • 14 = Manage Layer-2 • 15 = Manage own devices • 16 = Manage own users • 17 = Manage own remote connections • 18 = Manage own Layer-2
	passwordExpire	integer	Password expires (in days): • 0 = Never (set as default) • 30 days • 90 days • 360 days When no value is defined, the default value (0 = Never) is used. 14 days before expiry the user receives an e-mail. Requirement: • An e-mail address is configured for the user • The SMTP client is configured
	passwordExpirationInfiniteTolerance	boolean	• True: Users can still log in after password expiry and are forwarded to the "Change password" page • False: Setting disabled
	passwordExpirationToleranceTime	integer	Defines how long the password can be used after expiry.
	enablePasswordReset	boolean	• True: The "Reset password" link is shown on the login page • False: Setting disabled
	passwordResetLinkValidTime	integer	Password reset link validity

passwordReusing	integer	Reusing the same password: • 0: The setting is disabled • 1 - 5: If, for example, you enter 3, the current password can be reused only after 3 different passwords. As default, 3 is set. If no value is specified, the default value 3 is used.
passwordChangeFirstLogin	boolean	The password needs to be changed on first login or not.
pkiDnFilterRule	string	PKI DN filter rule for this role that is checked for on login. The attributes of the names (Distinguished Name acc. to the X.509 standard) are used as filter criteria. This requires that the attributes are included in the PKI certificate of the user.
pkiDeleteTmpUser	integer	Delete temporary users (in hours) • 0: The setting is disabled. The temporary user must be deleted manually. • 1 - 72 hours: When the time expires, the temporary user is deleted. As default, 24 is set. If no value is specified, the default value 24 is used.
enable2FA	boolean	Activates 2-factor authentication
enableRememberToken	boolean	When enabled, the one-time token is saved.
rememberTokenTime	integer	Specifies the duration of validity (in days). When the "enableRememberToken" function is enabled via the API without time value, the default value (1) is used.
accessTokenTimeout	integer	Specifies the validity of the new access token in minutes.
sessionNeverExpires	boolean	Specifies whether the session expires or not.
refreshTokenTimeout	integer	Specifies the duration of the session in hours.
umcUserGroup	string	UMC user group The entered name should match the name on the UMC server. When this value is specified, login via UMC is activated.
umcDeleteTmpUser	integer	Delete temporary UMC user (in hours) • 1 - 9999 hours When login via UMC is activated but this value is not specified, the default value 30 is used.

oidcPolicies	List of *oidc entries	"oidcPolicies":[{
		"oidcKeyInToken":"roles",
		"oidcValueInToken":"SINEMARC_role3"
		},
		{
		"oidcKeyInToken":"roles1",
		"oidcValueInToken":"SINEMARC_role4"
		}
		]

*oidcRoleKeyInToken	string	Key of the OAuth/OpenId application role
*oidcRoleValueInToken	string	Value of the OAuth/OpenId application role
oidcRelationBetweenClaims	integer	Relationship between claims • AND: The specified claims must be fulfilled.

			• OR: At least one of the specified clients must be fulfilled.
	oidcDeleteTmpUser	integer	Delete temporary user (in days).
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	Role ID
Failed call	422 - UNPROCESSABLE ENTRY	error	Group authorizations and read-only access cannot be used at the same time

Deleting a role

URL	/accounts/roles/<ID>		
DELETE	Deletes the desired role with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Role ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

4.3.6.3 Creating and managing user groups

Retrieving user groups

URL	/accounts/users/groups?userId={integer_value}		
GET	Returns all groups with the group name of the specified user as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	userId (required)	integer	User ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	groups	ListOf<id, name>	Lists all groups for the user
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Adding a user group

URL	/accounts/users/groups		
POST	Assigns a user group to the specified user		
Request	**Parameter**	**Data type**	**Values/Comments**
	userId (required)	integer	User ID
	groupId (required)	integer	Group ID

Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	"detail": "Resource forbidden"
	404 - NOT FOUND	error	No entry found

Deleting a user group

URL	/accounts/users/groups		
DELETE	Deletes the group with the specified ID from the specified user		
Request	**Parameter**	**Data type**	**Values/Comments**
	userId (required)	integer	User ID
	groupId (required)	integer	Group ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	"detail": "Resource forbidden"
	404 - NOT FOUND	error	No entry found

4.3.6.4 Assigning roles to the groups

Retrieving the groups of a role

URL	/accounts/roles/groups?roleId={integer_value}		
GET	Returns all groups from a role with IDs and names as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	roleId (required)	integer	Role ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	groups	ListOf<id, name>	Lists all groups of a role with IDs and names.
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights.
	404 - NOT FOUND	error	No entry found

Adding a new group for the role

URL	/accounts/roles/groups
POST	Adds a new group for a role

Request	**Parameter**	**Data type**	**Values/Comments**
	roleId (required)	integer	Role ID
	groupId (required)	integer	Group ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights.
	404 - NOT FOUND	error	No entry found

Deleting a group from the role

URL	/accounts/roles/groups		
DELETE	Deletes the desired group with the specified ID from the specified role		
Request	**Parameter**	**Data type**	**Values/Comments**
	roleId (required)	integer	Role ID
	groupId (required)	integer	Group ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights.
	404 - NOT FOUND	error	No entry found

4.3.6.5 Assigning roles to a user

Retrieving roles of a user

URL	/accounts/users/roles?userId={integer_value}		
GET	Returns all roles with the roles name of the specified user as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	userId (required)	integer	User ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	roles	ListOf<id, name>	Lists all roles for the user
Failed call	401 - UNAUTHOR-IZED	error	The user does not have the necessary access rights.
	404 - NOT FOUND	error	No entry found

Assigning roles to the user

URL	/accounts/users/roles
POST	Assigns a role to the specified user

Request	**Parameter**	**Data type**	**Values/Comments**
	userId (required)	integer	User ID
	roleId (required)	integer	Role ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights.
	404 - NOT FOUND	error	No entry found
	422 - UNPROCESSABLE ENTRY	error	Entry required

Deleting the roles of a user

URL	/accounts/users/roles		
DELETE	Deletes the role with the specified ID for the specified user		
Request	**Parameter**	**Data type**	**Values/Comments**
	userId (required)	integer	User ID
	roleId (required)	integer	Role ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights.
	404 - NOT FOUND	error	No entry found
	422 - UNPROCESSABLE ENTRY	error	Entry required

4.3.6.6 Managing the OpenVPN connection parameters of a user

Retrieving the OpenVPN connection parameters of a user

URL	/accounts/users/connectionparameter?userId={integer_value}		
GET	Returns all OpenVPN connection parameters of the specified user as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	userId (required)	integer	User ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	connectionparameters	ListOf<integer>	Lists all OpenVPN connection parameters for the user
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/accounts/users/connectionparameter/<ID>		
GET	Returns all information for an OpenVPN connection parameter		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the connection parameter
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	ipAddress	string	IP address of the OpenVPN connection parameter
	port	integer	Port of the OpenVPN connection parameter
	protocol	integer	1 = TCP 2 = UDP
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Adding OpenVPN connection parameters to a user

URL	/accounts/users/connectionparameter		
POST	Creates a new OpenVPN connection parameter for the specified user		
Request	**Parameter**	**Data type**	**Values/Comments**
	userId (required)	integer	User ID
	ipAddress (required)	string	IP address of the new OpenVPN connection parameter
	port (required)	integer	Port of the new OpenVPN connection parameter
	protocol (required)	integer	1 = TCP 2 = UDP
Successful call	200 - OK	-	-
	Result	**Data type**	**Values/Comments**
	id	integer	ID of the connection parameter
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	The connection parameter already exists.
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Deleting OpenVPN connection parameters from a user

URL	/accounts/users/connectionparameter/<ID>		
DELETE	Deletes the OpenVPN connection parameters with the specified ID for the specified user		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the connection parameter
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

4.3.6.7 Managing the Layer 2 connection parameters of a user

Retrieving the Layer 2 connection parameters of a user

URL	/accounts/users/layer2connectionparameter?userId={integer_value}		
GET	Returns all Layer 2 connection parameter IDs of the specified user as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	userId (required)	integer	User ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	connectionparameters	ListOf<integer>	Lists all Layer 2 connection parameters for the user
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/accounts/users/layer2connectionparameter/<ID>		
GET	Returns all information on a Layer 2 connection parameter of the specified user		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the connection parameter
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	userId	integer	User ID
	networkId	integer	ID of the Layer 2 network
	ipAddress	string	IP address of the new Layer 2 connection parameter
	netmask	string	Subnet mask
	groupId	integer	Group ID
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Adding Layer 2 connection parameters to a user

URL	/accounts/users/layer2connectionparameter/
POST	Creates a new Layer 2 connection parameter for the specified user

Request	**Parameter**	**Data type**	**Values/Comments**
	userId (required)	integer	User ID
	networkId (required)	integer	ID of the Layer 2 network
	ipAddress (required)	string	IP address of the new Layer 2 connection parameter
	netmask (required)	string	Subnet mask
	groupId (required)	integer	Group ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Values/Comments**
	id	integer	ID of the connection parameter
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	The connection parameter already exists.
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Deleting Layer 2 connection parameters from a user

URL	/accounts/users/layer2connectionparameter/<ID>		
DELETE	Deletes the Layer 2 connection parameters with the specified ID for the specified user		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the connection parameter
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

4.3.6.8 User agreement

Retrieving the user agreement

URL	/accounts/useragreement
GET	Returns the current user agreement

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	message	string	Text of the user agreement
	displayOption	integer	Display option: • 0 = Never The user agreement is not displayed. • 1 = First login When the user logs in the first time, the user agreement is displayed. After accepting the user agreement, the user can access the SINEMA RC server API. • 2 = Every login Every time the user logs in, the user agreement is displayed. After accepting the user agreement, the user can access the SINEMA RC server API.
	version	string	Current version of the user agreement
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Creating the user agreement

URL	/accounts/useragreement		
POST	Defines the current user agreement		
Request	**Parameter**	**Data type**	**Values/Comments**
	displayOption (required)	integer	Display option: • 0 = Never The user agreement is not displayed. • 1 = First login When the user logs in the first time, the user agreement is displayed. After accepting the user agreement, the user can access the SINEMA RC server API. • 2 = Every login Every time the user logs in, the user agreement is displayed. After accepting the user agreement, the user can access the SINEMA RC server API.
	message (optional)	string	Text of the user agreement
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

4.3.6.9 Management of client licenses

Retrieving client licenses

URL	/accounts/clients		
GET	Returns all client licenses as list		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	ID of the listed client entry
	systemId	string	Client system ID on the server
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/accounts/clients/<ID>		
GET	Returns all information on a client license		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the client entry
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	systemId	string	Client system ID on the server
	deviceName	string	PC name from which the client logged into the server
	lastConnectionTime	string	Time stamp of the client login with date and time
	lastConnectedUserId	integer	ID of the user who last established the connection from the client to the server
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/accounts/clients/floating		
GET	Returns all client floating licenses as a list		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	floating	ListOf<id, sytemId>	Lists all client floating licenses with IDs of the client entry and system IDs of the client on the server
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/accounts/clients/floating/<ID>		
GET	Returns all information about a client floating license		
Query	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the client entry

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	systemId	string	Client system ID on the server
	deviceName	string	PC name from which the client logged into the server
	status	boolean	Connection status, whether the connection is currently established
	lastConnectedUserId	integer	ID of the user who last established the connection from the client to the server
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

URL	/accounts/clients/floating/history		
GET	Returns the history of all client floating licenses as a list		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	history	ListOf<id, sytemId>	Lists the history of client floating licenses with IDs of the client entry and system IDs of the client at the server
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/accounts/clients/floating/history/<ID>		
GET	Returns details about the history of a client floating license		
Query	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the client entry
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	systemId	string	Client system ID on the server
	deviceName	string	PC name from which the client logged into the server
	lastConnectionTime	string	Time stamp of the client login with date and time
	lastConnectedUserId	integer	ID of the user who last established the connection from the client to the server
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Assigning licenses

URL	/accounts/clients/floating/<ID>		
PATCH	Assigns a floating license to the standard license		
Query	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the client entry
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/accounts/clients/floating/history/<ID>		
PATCH	Assigns a floating license from the history to the standard license		
Query	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the client entry
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Deleting a client entry

URL	/accounts/clients/<ID>		
DELETE	Deletes the desired client license with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the client entry
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

4.3.6.10 Managing two-factor authentication

Retrieving two-factor authentication

URL	/accounts/twofactor		
GET	Returns the status of two-factor authentication		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	alphanumericCode	string	Status of two-factor authentication, whether or not it is enabled. Must be used to generate the OTP token for enabling two-factor authentication.
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	The user does not have permission to perform this action

URL	/accounts/users/activate/<ID>		
GET	Returns the status to the user after completion of the trial phase or after backup restore		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Device ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	status	boolean	User status
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Enabling two-factor authentication

URL	/accounts/twofactor		
POST	Enables two-factor authentication.		
Request	**Parameter**	**Data type**	**Values/Comments**
	otpToken (required)	string	Alphanumeric one-time token
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	The user does not have permission to perform this action

Disabling two-factor authentication

URL	/accounts/twofactor		
DELETE	Disables two-factor authentication		
Successful call	200 - OK	-	-
Failed call	403 - FORBIDDEN	error	The user does not have permission to perform this action
	409 - CONFLICT	error	Two-factor authentication is already disabled

Activating users

URL	/accounts/users/activate/<ID>		
PUT	Activates users after completion of the trial phase or after backup restore		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	User ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	Resource forbidden
	404 - NOT FOUND	error	No entry found
	409 - CONFLICT	error	The user is already active

4.3.7 Services

4.3.7.1 Configuring the connection to the UMC server

Retrieving the UMC settings

URL	/services/umc
GET	Returns the settings for the connection to the UMC server

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	active	boolean	true = UMC is activated false = UMC is deactivated
	serverIp	string	IP address of the UMC server Empty when the value of the "active" parameter is false
	port	integer	UMC server port Empty when the value of the "active" parameter is false
	activeTrialLicense	boolean	Displays the status when the trial license is active
	trialLicenseDaysRemaining	integer	Shows the number of days remaining of the trial license. The value is 0 when the trial license is not active or has expired.
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights.

Setting up the UMC server

URL	/services/umc		
POST	Specifies the settings of the UMC configuration. When this command is called, the "UMC server" function is activated.		
Request	**Parameter**	**Data type**	**Values/Comments**
	serverIp (required)	string	IP address of the UMC server
	port (required)	integer	UMC server port
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights.
	403 - FORBIDDEN	error	To be able to use this function, you need the SINEMA RC UMC (MLFB 6GK1724-2VH03-0BV0) license.
	422 - UNPROCESSABLE ENTRY	error	Entry required

Deactivating the UMC server

URL	/services/umc		
DELETE	Deactivates the UMC server		
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights.
	403 - FORBIDDEN	error	To be able to use this function, you need the SINEMA RC UMC (MLFB 6GK1724-2VH03-0BV0) license.

4.3.7.2 Configuring the upload server

Retrieving the settings of the upload server

URL	/services/uploadsettings		
GET	Returns the settings for the upload server		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	files	integer	Files for upload • Configuration • Log files • Configuration and log files
	sftpServerName	string	IP address or FQDN of the SFTP server
	fingerprint	string	Fingerprint SFTP server
	directory	string	Upload directory path
	active	boolean	The upload server is enabled/disabled
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Setting up the upload server

URL	/services/uploadsettings
POST	Specifies the settings for the configuration of the upload server. When the SFTP server name is specified, the "Automatic upload of files" function is activated.

Request	**Parameter**	**Data type**	**Values/Comments**
	files (required)	integer	Files for upload Specify which file types are to be uploaded: • 1 = Configuration • 2 = Log files • 3 = Configuration and log files
	sftpServerName (required)	string	IP address or FQDN of the SFTP server If you use a port other than the standard port 22, enter the port number along with the IP address. A colon ":" is entered as separator between the IP address and the port number, e.g.: 192.168.234.1:622.
	directory (optional)	string	Upload directory The user is assigned a storage directory, the so-called home directory. If you do not enter anything, the file is uploaded directly to the home directory. To upload the file to a subdirectory, specify the subdirectory. Provided that the subdirectory is created in the home directory.
	username (required)	string	User name for access to the SFTP server
	password (required)	string	Password for access to the SFTP server
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Changing the settings of the upload server

URL	/services/uploadsettings		
PUT	Changes the settings of the upload server		
Request	**Parameter**	**Data type**	**Values/Comments**
	files	integer	Files for upload Specify which file types are to be uploaded: • 1 = Configuration • 2 = Log files • 3 = Configuration and log files
	sftpServerName	string	IP address or FQDN of the SFTP server
	directory	string	Upload directory
	username	string	User name for access to the SFTP server
	password	string	Password for access to the SFTP server
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Deactivating the settings of the upload server

URL	/services/uploadsettings		
DELETE	Disables the function "Automatic upload of files"		
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	The upload is already disabled

4.3.7.3 Configuring the connection to the Syslog server

Retrieving the client ID

URL	/services/syslog/hostname		
GET	Returns the client ID		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	hostname	string	Client ID (host name)
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Setting up the client ID

URL	/services/syslog/hostname		
POST	Specifies the client ID		
Request	**Parameter**	**Data type**	**Values/Comments**
	hostname (required)	string	Client ID (host name)
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Retrieving the Syslog server

URL	/services/syslog?count={integer_value}		
GET	Returns all Syslog servers with IDs and IP addresses as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	count (optional)	integer	Number of results that are to be returned

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	server	ListOf<id, ipAddress>	Returns all Syslog server IDs with the IP address as a list
	count	integer	When the "count" parameter is specified, the first found users are listed in the specified number.
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/services/syslog/<ID>		
GET	Returns all information on the Syslog server with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Syslog server ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	serverIp	string	Syslog server IP Address
	port	string	Syslog server port number
	protocol	integer	IP protocol: • 0 = UDP • 1 = TCP
	status	string	Syslog server status
	clientAuthentication	boolean	Client authentication is enabled or not
	certificate	integer	ID of the Syslog client certificate that is used for the authentication
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Establishing a connection to the Syslog server

URL	/services/syslog		
POST	Sets up a new Syslog client on the SINEMA RC server to establish the connection to the Syslog server and check its connection status. When the TCP protocol is used, the certificate must be imported by the Syslog server beforehand; otherwise, the connection will fail. The function that the user can accept the Syslog server certificate while "saving and checking the connection" is not possible within API.		
Request	**Parameter**	**Data type**	**Values/Comments**
	serverIp (required)	string	Syslog server IP Address
	port (required)	string	Syslog server port number
	protocol (required)	integer	IP protocol: • 0 = UDP • 1 = TCP
	certificate (optional)	integer	ID of the Syslog client certificate that is used for the authentication. When this value is not specified, a certificate of the self-signed SINEMA RC certificate authority is used.

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	Syslog server ID
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	The "serverIp" value must be unique
	409 - CONFLICT	error	The provider already exists.
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Deleting the Syslog server

URL	/services/syslog/<ID>		
DELETE	Deletes the Syslog server with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Syslog server ID
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

4.3.7.4 Configure debug login

Retrieve debug login settings

URL	/services/debuglogin		
GET	Returns the settings		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	enabled	boolean	Debug login enabled or not
	timeout	integer	Debug login timeout
	port	integer	Port number
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Setting up the debug login

URL	/services/debuglogin		
POST	Defines the settings for debug login.		
Request	**URL parameter**	**Data type**	**Values/Comments**
	timeout	integer	Duration of the access
	port	integer	TCP port via which the system of the SINEMA RC server is accessed.
	password	string	Password The new password must be at least 8 characters long and contain special characters, upper and lowercase characters as well as numbers.

Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Remove the debug login

URL	/services/debuglogin		
DELETE	Removes settings for debug login		
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	The entry already exists

4.3.7.5 Manage tools

Retrieve status of the VMware tool

URL	/services/tools/vmware		
GET	Returns the status of the VMWare installation		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	vmwareInstallationStatus	string	Status of the VMWare installation
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Install Vmware tools

URL	/services/tools/vmware		
POST	Install Vmware tools		
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	The entry already exists

4.3.7.6 Configure SMPT client

Retrieve SNMP client settings

URL	/services/snmpsettings
GET	Returns the SNMP client settings

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	enabled	boolean	SMTP client enabled or not
	supportSNMPv1AndSNMPv2	boolean	SNMPv1 & SNMPv2c support
	interface	integer	Interface used • "WAN": 0 • "LAN 1-n": 1 • "WAN + LAN 1-n": 2
	authenticationMethod	integer	Authentication algorithm • "MD5": 0 • "SHA": 1
	securityLevel	integer	Security level • "noAuthNoPriv":0 • "authNoPriv":1 • "authPriv":2
	port	integer	Port at which the SNMP agent waits for the SNMP queries
	encryptionMethod	integer	Encryption algorithm • "DES":0 • "AES":1
	username	string	User name for SNMP access
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Set up SMTP client

URL	/services/snmpsettings
POST	Set up SMTP client

Request	**Parameter**	**Data type**	**Values/Comments**
	enabled	boolean	Enable SMTP client
	supportSNMPv1AndSNMPv2	boolean	Enable SNMPv1 & SNMPv2c support
	interface	integer	Define interface used • "WAN": 0 • "LAN 1-n": 1 • "WAN + LAN 1-n": 2
	authenticationMethod	integer	Authentication algorithm • "MD5": 0 • "SHA": 1
	securityLevel	integer	Security level • "noAuthNoPriv":0 • "authNoPriv":1 • "authPriv":2
	port	integer	Port at which the SNMP agent waits for the SNMP queries
	encryptionMethod	integer	Encryption algorithm • "DES":0 • "AES":1
	username	string	User name for SNMP access
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	enabled	boolean	SMTP client enabled or not
	supportSNMPv1AndSNMPv2	boolean	SNMPv1 & SNMPv2c support
	interface	integer	Interface used • "WAN": 0 • "LAN 1-n": 1 • "WAN + LAN 1-n": 2
	authenticationMethod	integer	Authentication algorithm • "MD5": 0 • "SHA": 1
	securityLevel	integer	Security level • "noAuthNoPriv":0 • "authNoPriv":1 • "authPriv":2
	port	integer	Port at which the SNMP agent waits for the SNMP queries
	encryptionMethod	integer	Encryption algorithm • "DES":0 • "AES":1
	username	string	User name for SNMP access
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Delete SMTP client

URL	/services/snmpsettings		
DELETE	Deletes the settings of the SMTP client		
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	The entry already exists

4.3.7.7 Configuring OAuth/OpenID

Retrieving the OAuth/OpenID settings

URL	/services/oidcsettings		
GET	Returns the settings of OAuth/OpenID		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	active	boolean	OAuth/OpenID is enabled/disabled
	clientId	string	The client ID is assigned by the identity provider.
	clientSecret	string	The client secret is assigned by the identity provider.
	metadataUrl	string	Path on the authorization server to the OpenID configuration document The path must end with `/.well-known/openid-configuration`.
	providerDescription	string	The description text is shown on the login page as button on the "OAuth/OpenID" tab.
	redirectUrl	string	The authorization server can return to the SINEMA RC Server with the authorization feedback via the redirect URL.
	clientRedirectUrl	string	The request to the application registration server is sent by the SINEMA RC client to the application registration server.
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Setting up OAuth/OpenID

URL	services/oidcsettings
POST	Defines the settings for OAuth/OpenID

Request	**Parameter**	**Data type**	**Values/Comments**
	clientId (required)	string	The client ID is assigned by the identity provider.
	clientSecret (required)	string	The client secret is assigned by the identity provider.
	metadataUrl (required)	string	Path on the authorization server to the OpenID configuration document The path must end with `/.well-known/openid-configuration`.
	providerDescription (required)	string	The description text is shown on the login page as button on the "OAuth/OpenID" tab.
	clientRedirectUrl (required)	string	The authorization server can return to the SINEMA RC Server with the authorization feedback via the redirect URL.
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	OAuth/OpenID cannot be enabled due to insufficient licenses
	422 - UNPROCESSABLE ENTRY	error	Entry required or OpenID configuration could not be retrieved.

Removing OAuth/OpenID

URL	services/oidcsettings.		
DELETE	Removes the settings for OAuth/OpenID.		
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	OAuth/OpenID cannot be removed due to insufficient licenses

4.3.8 Safety

4.3.8.1 General

Configuring global keys

Retrieving global key settings

URL	/security/general/ciphers
GET	Returns information about the setting for the keys

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	level	integer	Setting for the encryption (ciphers): 0 = Low 1 = High
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Specifying the setting for the global keys

URL	/security/general/ciphers		
POST	Specifies the settings for the keys		
Request	**Parameter**	**Data type**	**Values/Comments**
	level (required)	integer	Setting for the encryption (ciphers): 0 = Low 1 = High
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	The encryption (ciphers) is already high/low.

Configure access

Retrieve password policy

URL	/security/general/passwordpolicy		
GET	Returns information on the password policy		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	policy	integer	Password rule
	minPasswordLength	integer	Length of the password
	minNumericCharacter	integer	Number of numeric characters
	minSpecialCharacter	integer	Number of special characters
	minLowercaseLetter	integer	Number of lowercase letters
	minUppercaseLetter	integer	Number of uppercase letters
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	The entry already exists

Change password policy

URL	/security/general/passwordpolicy
POST	Specifies the settings for the keys

Request	**Parameter**	**Data type**	**Values/Comments**
	policy	integer	Password rule • 0 = Default: Pre-configured settings • 1 = User-defined The desired requirements for passwords are configured.
	minPasswordLength	integer	Minimum length that the password must have.
	minNumericCharacter	integer	Minimum number of digits that a password must contain.
	minSpecialCharacter	integer	Minimum number of special characters that a password must contain.
	minLowercaseLetter	integer	Minimum number of lowercase letters that a password must contain.
	minUppercaseLetter	integer	Minimum number of uppercase letters that a password must contain.
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	The entry already exists
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Retrieve settings for Brute Force Prevention

URL	/security/general/bruteforceprevention		
GET	Returns the settings.		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	policy	integer	Rule for protection against brute force attacks
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Specify settings for Brute Force Prevention

URL	/security/general/bruteforceprevention
POST	Specifies the directive for Brute Force Prevention.

Request	**Parameter**	**Data type**	**Values/Comments**
	policy (required)	integer	Rule for protection against brute force attacks • 0 = Default: Pre-configured settings • 1 = User-defined The desired requirements are configured.
	failedLoginsBefore-Blocked (optional)	integer	The maximum number of invalid login attempts that will be accepted. Further login attempts will be blocked for a specified time.
	ipAddressBlockPeriod (optional)	integer	Time for which login is blocked because the maximum number of invalid login attempts was exceeded.
	disableInsteadOfBlock (optional)	integer	If enabled, login will not be blocked, but the user or device will be disabled.
	failedTokenRequestsBe-foreBlocked (optional)	integer	If two-factor authentication is enabled, this is the number of failed one-off tokens that a user can enter before their user account is locked.
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	The entry already exists
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

4.3.8.2 Managing certificates

CA certificate

Retrieving a CA certificate

URL	/security/certificates/ca		
GET	Returns all CA certificates		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	certificates	ListOf<id, commonName>	Returns all CA certificates with IDs and names as a list
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/security/certificates/ca/<ID>		
GET	Returns all information for a CA certificate		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the CA certificate

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	commonName	string	CA certificate name that is generated automatically by the system
	serialNr	string	Serial number
	validFrom	string	Valid from
	validTo	string	Valid to
	key	integer	Key length (bits)
	signature	string	Signature method
	keyId	string	Key ID
	status	string	Status of the CA certificate
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Exporting a CA certificate

URL	/security/certificates/ca/download/<ID>		
GET	Exports the selected CA certificate as a file		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the CA certificate
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	CA file	file down-load	Export of the CA certificate
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Adding a CA certificate

URL	/security/certificates/ca		
POST	Creates a new CA certificate		
Successful call	200 - OK	-	-
	Result	**Data type**	**Values/Comments**
	id	integer	ID of the CA certificate
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Renewing a CA certificate

URL	/security/certificates/ca		
PATCH	Renews the CA certificate		
Successful call	200 - OK	-	-
	Result	**Data type**	**Values/Comments**
	id	integer	ID of the CA certificate
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Deleting a CA certificate

URL	/security/certificates/ca/<ID>		
DELETE	Deletes the CA certificate with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the CA certificate
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	This CA certificate cannot be deleted
	404 - NOT FOUND	error	No entry found

Retrieve fallback certificate

URL	/security/certificates/fallback		
GET	Returns all information on a fallback certificate		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	serialNr	string	Serial number
	commonName	string	Fallback certificate name
	validFrom	string	Valid from
	validTo	string	Valid to
	key	integer	Key length (bits)
	signature	string	Signature method
	sha1	string	Fingerprint with SHA1 as hash algorithm
	sha256	string	Fingerprint with SHA256 as hash algorithm
	alternateIp1	string	The IP address of the WAN interface.
	alternateIp2	string	The WAN IP address when you have activated the function "SINEMA Remote Connect is located behind a NAT device" and have entered an IP address.
	alternateDns	string	The DNS name when you have activated the option "" and have entered a value.
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Renew fallback certificate

URL	/security/certificates/fallback		
PATCH	Renews the fallback certificate		
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Importing the Web server certificate

Retrieving the Web server certificate

URL	/security/certificates/webserver		
GET	Returns all information for a Web server certificate		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	serialNr	string	Serial number
	commonName	string	Certificate name
	issuer	string	Issuer
	validFrom	string	Valid from
	validTo	string	Valid to
	key	integer	Key length (bits)
	signature	string	Signature method
	sha1	string	SHA-1 Fingerprint
	sha256	string	SHA-256 Fingerprint
	alternateIp1	string	Alternative name IP 1
	alternateIp2	string	Alternative name IP 2
	alternateDns	string	Alternative name DNS
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/security/certificates/webserver/uploadstatus/<ID>		
GET	Returns the status of the Web server certificate with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the Web server certificate
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	date	string	Date from which the certificate is valid
	status	string	Status of the Web server certificate
	description	string	Description
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Importing the Web server certificate

URL	/security/certificates/webserver
POST	Imports and activates a new Web server certificate

Request	**Parameter**	**Data type**	**Values/Comments**
	format (required)	integer	Certificate format 1 = Unencrypted PEM/DER 2 = Encrypted PEM/DER 3 = PKCS12
	cert (required)	file upload	Certificate file if PKCS12, unencrypted PEM/DER or encrypted PEM/DER is selected
	key (required)	file upload	Key file if unencrypted PEM/DER or encrypted PEM/DER is selected
	ca (required)	file upload	CA file if unencrypted PEM/DER or encrypted PEM/DER is selected
	password (optional)	string	Password for the p12 certificate if PKCS12 is selected
Successful call	202 - ACCEPTED	-	-
	Result	**Data type**	**Description**
	detail	string	Detail of the upload request
	uploadRequestId	integer	ID of the upload request
	uploadStatusCheckUrl	string	URL of the upload status
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

Renewing a Web server certificate

URL	/security/certificates/webserver		
PATCH	Renews the Web server certificate		
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Retrieving a VPN certificate

Retrieving a VPN certificate

URL	/security/certificates/vpn
GET	Returns all information for a VPN certificate

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	serialNr	string	Serial number
	commonName	string	Certificate name
	issuer	string	Issuer
	validFrom	string	Valid from
	validTo	string	Valid to
	key	integer	Key length (bits)
	signature	string	Signature method
	sha1	string	SHA-1 Fingerprint
	sha256	string	SHA-256 Fingerprint
	alternateIp1	string	Alternative name IP 1
	alternateIp2	string	Alternative name IP 2
	alternateDns	string	Alternative name DNS
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Renewing a VPN certificate

URL	/security/certificates/vpn		
PATCH	Renews the VPN certificate		
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Retrieving the settings for certificates

Retrieving the settings for certificates

URL	/security/certificates/settings		
GET	Returns all information for the certificate management settings		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	keyLen	integer	Preferred key length (bits): 1 = 2048 2 = 4096
	hashMethod	integer	Preferred hash method: 1 = SHA256 2 = SHA512
	caRenew	integer	CA certificate renewal (days before expiry)
	certValidity	integer	Validity of client certificates (days)
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Configuring settings for certificate management

URL	/security/certificates/settings		
POST	Defines the settings for certificate management		
Request	**Parameter**	**Data type**	**Values/Comments**
	keyLen (required)	integer	Preferred key length (bits): 1 = 2048 2 = 4096
	hashMethod (required)	integer	Preferred hash method: 1 = SHA256 2 = SHA512
	caRenew (required)	integer	CA certificate renewal (days before expiry)
	certValidity (required)	integer	Validity of client certificates (days)
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

4.3.8.3 Configuring VPN Connections

Retrieving the OpenVPN settings

Retrieving the OpenVPN settings

URL	/security/openvpn
GET	Returns all information for the OpenVPN basic settings

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	status	integer	Status 0 = deactivated 1 = activated
	tcp	integer	TCP port
	udp	integer	UDP port
	keepAlive	integer	Keep alive interval (s)
	timeout	integer	Connection timeout (s)
	key	integer	DH key length (bits) 1 = 1024 2 = 1536 3 = 2048 4 = 4096
	cipher	integer	Cipher 1 = AES-128 2 = AES-192 3 = AES-256 4 = DES-EDE 5 = DES-EDE3
	hash	integer	Hash method 1 = SHA 1 2 = SHA 256 3 = SHA 512
	tls	integer	TLS version 0 = 1.0 1 = 1.1 2 = 1.2
	interface	integer	The interface that forms the local VPN endpoint. Via this interface the OpenVPN connection to the OpenVPN partner (SINEMA RC Client, device) is established. 0 = WAN: Connection only via the WAN interface 1 = LAN 1-n: Connection via available LAN interfaces 2 = WAN + LAN 1-n: Connection via all interfaces
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Configuring OpenVPN settings

URL	/security/openvpn
POST	Defines the OpenVPN settings

Request	**Parameter**	**Data type**	**Values/Comments**
	status (required)	integer	Status 0 = deactivated 1 = activated
	tcp (optional)	integer	TCP port
	udp (optional)	integer	UDP port
	keepAlive (optional)	integer	Keep alive interval (s)
	timeout (optional)	integer	Connection timeout (s)
	key (optional)	integer	DH key length (bits) 1 = 1024 2 = 1536 3 = 2048 4 = 4096
	cipher (optional)	integer	Cipher 1 = AES-128 2 = AES-192 3 = AES-256 4 = DES-EDE 5 = DES-EDE3
	hash (optional)	integer	Hash method 1 = SHA 1 2 = SHA 256 3 = SHA 512
	tls (optional)	integer	TLS version 0 = 1.0 1 = 1.1 2 = 1.2
	interface (optional)	integer	The interface that forms the local VPN endpoint. Via this interface the OpenVPN connection to the OpenVPN partner (SINEMA RC Client, device) is established. 0 = WAN: Connection only via the WAN interface 1 = LAN 1-n: Connection via available LAN interfaces 2 = WAN + LAN 1-n: Connection via all interfaces
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY	error	Invalid entry

4.3.8.4 Managing Syslog certificates

Syslog CA certificate

Retrieving a Syslog CA certificate

URL	/security/syslog/ca?count={integer_value}		
GET	Returns all Syslog CA certificates with ID and the general name as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	count (optional)	integer	Number of results that are to be returned
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	certificates	ListOf<id, commonName>	Lists all Syslog CA certificates with IDs and names
	count	integer	When the "count" parameter is specified, the first found entries are listed in the specified quantity
	previous	string	
	next	string	
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/security/syslog/ca/<ID>		
GET	Returns all information for a Syslog CA certificate		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the Syslog CA certificate
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	commonName	string	Certificate name
	status	string	Status
	certType	string	Certificate type
	subject	string	
	issuer	string	Issuer
	validFrom	string	Valid from
	validTo	string	Valid to
	fingerprint	string	Fingerprint
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Importing a Syslog CA certificate

URL	/security/syslog/ca
POST	Imports a CA certificate for authentication of a Syslog server

Request	**Parameter**	**Data type**	**Values/Comments**
	ca (required)	file upload	Syslog CA certificate
	cert (required)	file upload	Syslog client certificate
	format (required)	integer	Certificate format 1 = PEM/DER 3 = PKCS12
	password (optional)	string	Password for the p12 certificate if PKCS12 is selected
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	ID of the Syslog CA certificate
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	Syslog CA certificate already exists
	422 - UNPROCESSABLE ENTRY		Invalid settings

Deleting a Syslog CA certificate

URL	/security/syslog/ca/<ID>		
DELETE	Deletes the Syslog CA certificate with the specified ID		
Request	**Parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the Syslog CA certificate
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Syslog certificate

Retrieving a Syslog certificate

URL	/security/syslog/certificate?count={integer_value}		
GET	Returns all Syslog certificates with ID and the general name as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	count (optional)	integer	Number of results that are to be returned

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	certificates	ListOf<id, commonName>	Lists all Syslog certificates with IDs and names
	count	integer	When the "count" parameter is specified, the first found entries are listed in the specified quantity
	previous	string	
	next	string	
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/security/syslog/certificate/<ID>		
GET	Returns all information for a Syslog certificate		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the Syslog certificate
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	commonName	string	Certificate name
	status	string	Status
	certType	string	Certificate type
	subject	string	Owner of the private key assigned in the certificate
	issuer	string	Issuer Certificate authority that issued the certificate
	validFrom	string	Valid from
	validTo	string	Valid to
	fingerprint	string	Fingerprint
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Importing a Syslog certificate

URL	/security/syslog/certificate		
POST	Imports a certificate to a Syslog server that can be used for the client authentication		
Request	**Parameter**	**Data type**	**Values/Comments**
	password (optional)	string	Password for the p12 certificate if PKCS12 is selected
	cert (required)	file upload	Syslog client certificate file if PKCS12, unencrypted PEM/DER or encrypted PEM/DER is selected
	key (required)	file upload	Syslog client key file if unencrypted PEM/DER or encrypted PEM/DER is selected
	format (required)	integer	Certificate format 1 = Unencrypted PEM/DER 2 = Encrypted PEM/DER 3 = PKCS12

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	id	integer	ID of the Syslog certificate
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY		Invalid settings

Deleting a Syslog certificate

URL	/security/syslog/certificate/<ID>		
DELETE	Deletes the Syslog CA certificate with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the Syslog certificate
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Revoking a Syslog CA certificate

Retrieving the Syslog revocation list

URL	/security/syslog/revocation		
GET	Returns all revocation list IDs with the issuer as a list		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	server	ListOf<id, issuer>	Lists all revocation list IDs with the issuer with IDs and names
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/security/syslog/revocation/<ID>		
GET	Returns all information for a revocation list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Revocation list ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	issuer	string	Issuer
	serialNr	string	Revoked serial number
	lastUpdate	string	Last update
	nextUpdate	string	Next update
	origin	string	Origin of the certificate revocation list • File: The certificate revocation list was imported • URL: The certificate revocation list is stored at the distribution point

Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	Revocation request for the Syslog certificate not found

Importing the Syslog revocation list

URL	/security/syslog/revocation		
POST	Imports a Syslog revocation list (CRL)		
Request	**Parameter**	**Data type**	**Values/Comments**
	file (required)	file upload	CRL file
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY		Invalid settings

Deleting a Syslog revocation certificate

URL	/security/syslog/revocation/<ID>		
DELETE	Deletes the desired Syslog revocation certificate with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the Syslog revocation certificate
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Syslog settings

Retrieving settings of Syslog certificate management

URL	/security/syslog		
GET	Returns all information of the settings for managing Syslog certificates		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	crlChecking	boolean	CRL check activated (true/false)
	interval	integer	CRL update interval (min)
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Defining settings of Syslog certificate management

URL	/security/syslog/ca
POST	Specifies the settings for the Syslog certificate. When no interval is defined, the default value is used.

Request	**Parameter**	**Data type**	**Values/Comments**
	crlChecking (required)	boolean	CRL check activated (true/false)
	interval (optional)	integer	CRL update interval (min)
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY		Invalid settings

4.3.8.5 Managing PKI Certificates

PKI CA certificate

Retrieving PKI CA certificates

URL	/security/pki/ca?count={integer_value}		
GET	Returns all PKI CA certificates with ID and the general name as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	count (optional)	integer	Number of results that are to be returned
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	certificates	ListOf<id, commonName>	Lists all PKI CA certificates with IDs and names
	count	integer	When the "count" parameter is specified, the first found entries are listed in the specified quantity
	previous	string	
	next	string	
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/security/pki/ca/<ID>		
GET	Returns all information on a PKI CA certificate with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	PKI CA certificate ID

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	commonName	string	Certificate name
	status	string	Status
	certType	string	Certificate type
	subject	string	
	issuer	string	Issuer
	validFrom	string	Valid from
	validTo	string	Valid to
	fingerprint	string	Fingerprint
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No PKI CA found

Importing a PKI CA certificate

URL	/security/pki/ca		
POST	Imports the selected PKI CA certificate as a file		
Request	**Parameter**	**Data type**	**Values/Comments**
	ca (required)	file upload	PKI CA certificate file
	cert (required)	file upload	PKI p12 certificate file if the PKCS12 format is selected
	format (required)	integer	Certificate format 1 = PEM/DER 3 = PKCS12
	password (optional)	string	Password for the p12 certificate if PKCS12 is selected
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY		Invalid settings

Deleting a PKI CA certificate

URL	/security/pki/ca/<ID>		
DELETE	Deletes the PKI CA certificate with the specified ID		
Request	**Parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the PKI CA certificate
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

PKI Revocation Lists

Retrieving revocation lists

URL	/security/pki/revocation?count={integer_value}		
GET	Returns all revocation list IDs with the issuer as a list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	count (optional)	integer	Number of results that are to be returned
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	server	ListOf<id, issuer>	Lists all certificate revocation list IDs with the issuer with IDs and names
	count	integer	When the "count" parameter is specified, the first found entries are listed in the specified quantity
	previous	string	
	next	string	
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/security/pki/revocation/<ID>		
GET	Returns all information about a certificate revocation list		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	Certificate revocation list ID
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	issuer	string	Issuer
	serialNr	string	Revoked serial numbers
	lastUpdate	string	Last update
	nextUpdate	string	Next update
	origin	string	Origin of the certificate revocation list • File: The certificate revocation list was imported • URL: The certificate revocation list is stored at the distribution point
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	Revocation request for PKI certificate not found

URL	/security/pki/blacklist
GET	Returns all DN filter rules with disabled users as a list

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	DNBlacklist	List<id,dnFilter, deactivatedUsers>	Lists all DN filter rules with deactivated users
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Importing a certificate revocation list

URL	/security/pki/revocation		
POST	Imports a certificate revocation list (CRL)		
Request	**Parameter**	**Data type**	**Values/Comments**
	file (required)	file upload	CRL file
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY		Invalid settings

Deleting a certificate revocation list

URL	/security/pki/revocation/<ID>		
DELETE	Deletes the certificate revocation list (CRL) with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the CRL
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Importing a PKI DN Blacklist

URL	/security/pki/blacklist		
POST	Imports a PKI DN Blacklist		
Request	**Parameter**	**Data type**	**Values/Comments**
	dnFilter (required)	string	PKI DN filter rule
Successful call	200 - OK	-	-
	Result	**Data type**	**Values/Comments**
	id	integer	ID of the PKI DN filter rule

Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	409 - CONFLICT	error	This PKI DN filter rule already exists
	422 - UNPROCESSABLE ENTRY	error	Invalid settings

Deleting a PKI DN Blacklist

URL	/security/pki/blacklist/<ID>		
DELETE	Deletes the PKI DN Blacklist with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the PKI DN filter rule
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

PKI Settings

Retrieving PKI settings

URL	/security/pki		
GET	Returns all information on the PKI settings		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	crlChecking	boolean	Enable CRL checking (true/false)
	interval	integer	CRL update interval (min)
	crlMissing	boolean	Allow missing CRL (true/false)
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

Configuring PKI settings

URL	/security/pki		
POST	Defines the PKI settings		
Request	**Parameter**	**Data type**	**Values/Comments**
	crlChecking (required)	boolean	Enable CRL checking (true/false)
	interval (optional)	integer	CRL update interval (min)
	crlMissing (optional)	boolean	Allow missing CRL (true/false)
Successful call	200 - OK	-	-

Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	403 - FORBIDDEN	error	The boxes "crlMissing" and "interval" cannot be changed when "crlChecking" is disabled.
	422 - UNPROCESSABLE ENTRY	error	Invalid settings

4.3.8.6 UMC certificate

UMC CA certificate

Retrieve UMC CA certificate

URL	/security/umc/ca		
GET	Returns all information on the UMC CA certificate		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	certificates	ListOf<id, commonName>	Lists all UMC CA certificates with IDs and names
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/security/umc/ca/<ID>		
GET	Returns all information on a UMC CA certificate		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the UMC CA certificate
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	commonName	string	Certificate name
	status	string	Status
	certType	string	Certificate type
	subject	string	
	issuer	string	Issuer
	validFrom	string	Valid from
	validTo	string	Valid to
	fingerprint	string	Fingerprint
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Import UMC CA certificate

URL	/security/umc/ca
POST	Imports the selected UMC CA certificate as file

Request	**Parameter**	**Data type**	**Values/Comments**
	ca (required)	file upload	UMC CA certificate if PEM/DER is selected
	cert (required)	file upload	UMC client certificate if PKCS12 is selected
	format (required)	integer	Certificate format 1 = PEM/DER 3 = PKCS12
	password (optional)	string	Password for the p12 certificate if PKCS12 is selected
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY		Invalid settings

Delete UMC CA certificate

URL	/security/umc/ca/<ID>		
DELETE	Deletes the UMC CA certificate with the specified ID		
Request	**Parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the Syslog CA certificate
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

UMC certificate

Retrieve UMC certificate

URL	/security/umc/certificate		
GET	Returns all UMC certificates with ID and the general name as a list		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	certificates	ListOf<id, commonName>	Lists all UMC certificates with IDs and names
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/security/umc/certificate/<ID>		
GET	Returns all information on a UMC certificate		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the Syslog certificate

Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	commonName	string	Certificate name
	status	string	Status
	certType	string	Certificate type
	subject	string	Owner of the private key assigned in the certificate
	issuer	string	Issuer Certificate authority that issued the certificate
	validFrom	string	Valid from
	validTo	string	Valid to
	fingerprint	string	Fingerprint
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Import UMC certificate

URL	/security/umc/certificate		
POST	Imports a UMC certificate that can be used for the client authentication		
Request	**Parameter**	**Data type**	**Values/Comments**
	password (optional)	string	Password for the p12 certificate if PKCS12 is selected
	cert (required)	file upload	UMC client certificate file if PKCS12, unencrypted PEM/DER or encrypted PEM/DER is selected
	key (required)	file upload	UMC client key file
	format (required)	integer	Certificate format 1 = Unencrypted PEM/DER 2 = Encrypted PEM/DER 3 = PKCS12
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY		Invalid settings

Delete UMC certificate

URL	/security/umc/certificate/<ID>		
DELETE	Deletes the UMC certificate with the specified ID		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the Syslog certificate
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

4.3.8.7 OID certificate

OAuth/OpenID CA certifiate

Retrieve OAuth/OpenID CA certificate

URL	/security/oidc/ca		
GET	Returns all information on the OAuth/OpenID CA certificate		
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	certificates	ListOf<id, commonName>	Lists all UMC CA certificates with IDs and names
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights

URL	/security/oidc/ca/<ID>		
GET	Returns all information for an OAuth/OpenID CA certificate		
Request	**URL parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the OAuth/OpenID CA certificate
Successful call	200 - OK	-	-
	Result	**Data type**	**Description**
	commonName	string	Certificate name
	status	string	Status
	certType	string	Certificate type
	subject	string	
	issuer	string	Issuer
	validFrom	string	Valid from
	validTo	string	Valid to
	fingerprint	string	Fingerprint
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

Importing OAuth/OpenID CA certificate

URL	/security/oidc/ca
POST	Imports the selected OAuth/OpenID CA certificate as a file

Request	**Parameter**	**Data type**	**Values/Comments**
	ca (required)	file upload	OAuth/OpenID CA certificate if PEM/DER is selected
	cert (required)	file upload	UMC client certificate if PKCS12 is selected
	format (required)	integer	Certificate format 1 = PEM/DER 3 = PKCS12
	password (optional)	string	Password for the p12 certificate if PKCS12 is selected
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	422 - UNPROCESSABLE ENTRY		Invalid settings

Delete OAuth/OpenID CA certificate

URL	/security/oidc/ca/<ID>		
DELETE	Deletes the OAuth/OpenID CA certificate with the specified ID		
Request	**Parameter**	**Data type**	**Values/Comments**
	ID (required)	integer	ID of the OAuth/OpenID CA certificate
Successful call	200 - OK	-	-
Failed call	401 - UNAUTHORIZED	error	The user does not have the necessary access rights
	404 - NOT FOUND	error	No entry found

JSON and data types

5

5.1 Upper and lower case

Note

JSON properties are case-sensitive

Observe the use of uppercase and lowercase in the JSON properties.

The property names of a JSON object MUST be formatted as described in the individual resources. When a property has the name `SomeProperty`, for example, it should also be spelled as such.

Do

```
{
  "SomeProperty": "SomeValue"
}
```

Don't

```
{
  "someProperty": "SomeValue"
}
```

5.2 API data types

The API server uses the following data types:

- `bool`
- `int`
- `double`
- `string`
- `datetime`
- `array`
- `object`

To show these data types in JSON, observe the instructions for serialization and deserialization.

Bool

A Boolean value. The only values that are valid in the JSON format are `true` and false. Other values, such as "`true`" (as string) or 1 (as number), are invalid.

Do

```
{
  "IsValid": true,
  "HasErrors": false
}
```

Don't

```
{
  "IsValid": "true",
  "HasErrors": 0
}
```

Int

Do

```
{
  "FirstInt": 123,
  "SecondInt": -123
}
```

Don't

```
{
  "FirstInt": "123",
  "SecondInt": 123E2,
  "ThirdInt": 0x11AB,
  "FourthInt": 3.0,
  "FifthInt": 000123,
  "SixthInt": +123
}
```

Double

A signed 64-bit floating-point value with a minimum value of -1.7976931348623157E+308 and a maximum value of 1.7976931348623157E+308. The value MUST be formatted as JSON number, which means without leading zeros, without a leading +, without string delimiters and not in octal or hexadecimal format.

In addition, three special values are permitted to map Positive Infinity, Negative Infinity and Not a Number. These values MUST be treated as JSON strings; lowercase and uppercase spelling must be observed: "Infinity", "-Infinity", "NaN".

Do

```
{
  "FirstDouble": 1.23,
  "SecondDouble": -1.23,
  "ThirdDouble": 3,
  "FourthDouble": -2E8,
  "FifthDouble": "NaN"
}
```

Don't

```
{
  "FirstDouble": "1.23",
  "SecondDouble": +1.23,
  "ThirdDouble": NaN,
  "FourthDouble": 0x11AB,
  "FifthDouble": 000123
}
```

String

A string is a sequence of Unicode code points. The value MUST be formatted according to ECMA-404 The JSON Data Interchange Standard .

In addition, the JSON letter symbol `null` (without string delimiter) may be used to represent a `null string` .

Do

```
{
  "FirstString": "SomeString",
  "SecondString": "SomeOtherString",
  "EmptyString": "",
  "NullString": null
}
```

Don't

```
{
  "FirstString": SomeString,
  "NullString": "null"
```

```
}
```

Datetime

A datetime stands for a specific point in time and includes date and time information. Values of the type datetime are usually treated as UTC values by the COMOS REST API according to the Gregorian calendar. The datetime minimum value is 00:00:00.0000000 UTC, 1 January 0001. The datetime maximum value is 23:59:59.9999999 UTC, 31 December 9999.

In JSON, datetime values MUST be formatted as strings and given one of the following formats:

- yyyy
- yyyy-MM
- yyyy-MM-dd
- yyyy-MM-ddTHH
- yyyy-MM-ddTHH:mm
- yyyy-MM-ddTHH:mm:ss
- yyyy-MM-ddTHH:mm:ss.fffffff
-

The symbols have the following meaning:

- yyyy: The year in the form of 4 digits. If the year has fewer than 4 digits, leading zeros are used to complete the 4 digits. Possible values range from 0001 to 9999.
- MM: The month in the form of 2 digits. If the month has only one digit, a leading zero is used. Possible values range from 01 to 12. The default value is 01.
- dd: The day of the month in the form of 2 digits. If the day has only one digit, a leading zero is used. Possible values range from 01 to 31. The default value is 01.
- HH: The hour in the form of 2 digits in a 24-hour format. If the hour has only one digit, a leading zero is used. Possible values range from 00 to 23. The default value is 00.
- mm: The minutes in the form of 2 digits. If the minutes have only one digit, a leading zero is used. Possible values range from 00 to 59. The default value is 00.
- ss: The seconds in the form of 2 digits. If the seconds have only one digit, a leading zero is used. Possible values range from 00 to 59. The default value is 00.
- fffffff: The fraction of a second in the form of 7 digits. Possible values range from 0000000 to 9999999. The default value is 0000000.
- Letter symbols -T:.: Letter symbols that subdivide the individual areas of the datetime value.

Array

An array stands for a collection of values. In JSON, the array elements are given in brackets and are delimited by commas.

But an array can also be empty. In addition, the JSON letter symbol null (without string delimiter) may be used to represent a null.

array.

Example: An array of strings:

```
["a", "b", "c"]
```

Example: An array of objects

```
[{"SomeProperty": "SomeValue"}, {"SomeProperty": "SomeOtherValue"}]
```

Example: An empty array

```
[]
```

Object

An object is a collection of key/value pairs in parentheses ({}). A key is a string, while a value can be any type including of a different object. A colon (:) separates the name from the value. A comma (,) separates two key/value pairs from each other.

From an object-oriented perspective, an object in JSON can be considered as an object that contains the keys for the property names and the values for the property values.

An object that only consists of parentheses is a valid object. In addition, the JSON letter symbol null (without string delimiter) may be used to represent a null object.

Even though JSON objects in which the same key is used multiple time on the same hierarchy level are not excluded by the JSON standard (ECMA-404 The JSON Data Interchange Standard) , a client MAY NOT create requests that contain such objects. The processing of such duplicate keys is not defined and compatibility with regard to the processing sequence or whether the request is processed at all, for example, is not guaranteed for the existing nor any future versions of the COMOS REST API.

Example: An object

```
{
  "SomeString": "SomeValue",
  "SomeOtherString": "SomeOtherValue",
  "SomeInt": 123,
  "SomeArray": ["FirstValue", "SecondValue"],
  "SomeObject":
  {
    "YetAnotherString": "YetAnotherValue",
    "YetAnotherInt": 100
  }
}
```

Example: An empty object

```
{}
```

Don't

```
{
  "SomeProperty": "SomeValue",
  "SomeProperty": "SomeOtherValue"
}
```

See also

109764829 (https://support.industry.siemens.com/cs/ww/en/view/109764829)