|  |
| --- |
| [Company name] |
| DELI offline SDK protocol |
| [Document subtitle] |

|  |
| --- |
| 1-25-2025 |

Table of contents

[Device control interface 4](#_Toc188693339)

[Interface address 4](#_Toc188693340)

[Request message format 4](#_Toc188693341)

[Response message format 4](#_Toc188693342)

[Command successful response 5](#_Toc188693343)

[Error Response 5](#_Toc188693344)

[Security Configuration 6](#_Toc188693345)

[SetSecurityConfig 6](#_Toc188693346)

[User synchronization 7](#_Toc188693347)

[GetUserIdList 7](#_Toc188693348)

[GetUserInfo 8](#_Toc188693349)

[SetUserInfo 9](#_Toc188693350)

[DeleteUserInfo 10](#_Toc188693351)

[Prohibit or allow the use of the machine 11](#_Toc188693352)

[LockDevice 11](#_Toc188693353)

[Remote registration of features 12](#_Toc188693354)

[BeginEnrollFace 12](#_Toc188693355)

[BeginEnrollFP 12](#_Toc188693356)

[BeginEnrollCard 12](#_Toc188693357)

[BeginEnrollPalm 13](#_Toc188693358)

[QueryJobStatus 13](#_Toc188693359)

[CancelJob 14](#_Toc188693360)

[CancelAllJobs 15](#_Toc188693361)

[PhotoToFacedata 15](#_Toc188693362)

[Attendance record management 16](#_Toc188693363)

[GetAttendLog 16](#_Toc188693364)

[EraseAttendLog 17](#_Toc188693365)

[Automatic reporting of attendance records 18](#_Toc188693366)

[ConfigAttendLogUploader 18](#_Toc188693367)

[GetAttendLogUploaderStatus 19](#_Toc188693368)

[Time Synchronization 20](#_Toc188693369)

[GetDeviceTime 20](#_Toc188693370)

[SetDeviceTime 20](#_Toc188693371)

[Device network configuration 21](#_Toc188693372)

[GetNetworkConfig 21](#_Toc188693373)

[SetNetworkConfig 22](#_Toc188693374)

[Device information query and settings 24](#_Toc188693375)

[GetVersionInfo 24](#_Toc188693376)

[GetCapacityLimit 24](#_Toc188693377)

[GetChangeUserInfoOnDeviceAllowed 25](#_Toc188693378)

[SetChangeUserInfoOnDeviceAllowed 25](#_Toc188693379)

[GetCurrentUsage 25](#_Toc188693380)

[GetDeviceUid 26](#_Toc188693381)

[GetDeviceCapabilities 26](#_Toc188693382)

[GetBellSetting, SetBellSetting 27](#_Toc188693383)

[GetDeviceId, SetDeviceId 28](#_Toc188693384)

[GetSoundVolume, SetSoundVolume 29](#_Toc188693385)

[GetReverifyTime, SetReverifyTime 29](#_Toc188693386)

[GetVerifyMode, SetVerifyMode 30](#_Toc188693387)

[GetFaceVerifySetting, SetFaceVerifySetting 31](#_Toc188693388)

[DeviceControl 32](#_Toc188693389)

[Shift settings and user shift settings 32](#_Toc188693390)

[GetShiftMode, SetShiftMode 32](#_Toc188693391)

[GetAttendaceTolerance, SetAttendaceTolerance 33](#_Toc188693392)

[GetShiftSetting, SetShiftSetting 33](#_Toc188693393)

[GetUserShift, SetUserShift 35](#_Toc188693394)

[Access control settings 36](#_Toc188693395)

[GetAcTimeZone, SetAcTimeZone 36](#_Toc188693396)

[GetAcUserInfo, SetAcUserInfo 37](#_Toc188693397)

[GetAcSetting, SetAcSetting 38](#_Toc188693398)

[Department Management 39](#_Toc188693399)

[GetDeptList 39](#_Toc188693400)

[SetDept 39](#_Toc188693401)

# Device control interface

## Interface address

The device can be controlled via http or https interface. The URL of the device control interface is as follows.

|  |
| --- |
| http(s)://device IP address/control?api\_key=API key |

**Parameter Description**

|  |  |
| --- | --- |
| *Device IP address* | The IP address or DNS hostname of the device. |
| API Key | The key is used for authentication. The device will only accept commands if the key matches the currently set value. The API key can be set via the SetSecurityConfigcommand. |

In the factory settings, the device is configured to enable the http interface and disable the https interface by default. These can be changed through the SetSecurityConfigcommand. All requests are sent in the form of http POST messages.

## Request message format

All messages sent from the application to the device should be json data with the following format.

|  |
| --- |
| {"mid" : "a123456","cmd" : "GetUserInfo","payload" : { ... }} |

**Parameter Description**

|  |  |
| --- | --- |
| mid | Message ID. Length must not exceed 32 characters. A response message will have the same message ID as its corresponding request message. |
| cmd | Indicates the type of operation that the device is to perform. |
| payload | Additional information required to perform the specified operation. Its format depends on the cmd parameter. |

## Response message format

After receiving the command, the device should perform the requested action and return json data in the following format.

|  |
| --- |
| {"mid" : "a123456","result" : "UserInfo","payload" : { ... }} |

**Parameter Description**

|  |  |
| --- | --- |
| mid | Message ID. A response message will have the same message ID as its corresponding request. |
| result | Indicates the kind of information this message contains. Each request has a specific set of possible response types. |
| payload | Contains data that the device wants to send to the application. Its format depends on the result parameter. |

## Command successful response

When the command completes successfully and there is no further information to send, the device will respond with the following message.

|  |
| --- |
| {"mid" : "a123456","result" : "Success","payload" : {}} |

## Error Response

When the command fails, the device will respond with the following message.

|  |
| --- |
| {"mid" : "a123456","result" : "Error","payload" : { "code" : "user\_id\_not\_exists", "arguments": [12] }} |

**Parameter Description**

|  |  |
| --- | --- |
| code | Indicates the type of error that occurred. |
| arguments | Describes detailed information about the error. The meaning of this parameter depends on the code. |

# Security Configuration

## SetSecurityConfig

This command configures communication security parameters.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "SetSecurityConfig", "payload" : { "api\_key": "API key", "enable\_http": "yes", "tls\_conf": { "enabled": "yes", "validate\_certificate": "yes", "ca\_cert": "PEM-encoded CA certificate", "device\_cert": "PEM-encoded certificate", "device\_key ": "PEM-encoded private key" } }} | {"mid" : "a123456","result" : "Success","payload" : {}} |

**Request parameter description**

|  |  |
| --- | --- |
| api\_key | After setting, all commands must specify the API key in their request URL. See [ Interface address]. |
| enable\_http | yes: to enable the device http interface, no: to disable it. |
| tls\_conf |  |
| enabled | yes: to enable the device https interface, no: to disable it. |
|  validate\_certificate | yes: The device will verify the server certificate when invoking https requests.no: The device will not verify the server certificate.(This setting is used when the device uploads attendance records.) |
|  ca\_cert | The certificate authority certificate (PEM-encoded) that the device will use when verifying the server certificate. |
|  device\_cert | The TLS certificate (PEM encoded) that the device https interface will use. |
|  device\_key | The private key corresponding to the device certificate (PEM encoded). |

# User synchronization

## GetUserIdList

Retrieve a list of user IDs.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "GetUserIdList","payload" : { "start\_pos": 0 // Optional }} | {"mid" : "a123456","result" : "UserIdList","payload" : { "user\_id" : [ "1", "5", "12" ], "next\_page\_pos": 50}} |

**Request parameter description**

|  |  |
| --- | --- |
| start\_pos | Optional, when this parameter is omitted, the device will return the first N IDs from the user list (N is device model specific, e.g. 50).When this parameter is specified, the device will start returning user IDs from this position. |

**Corresponding parameter description**

|  |  |
| --- | --- |
| user\_id | Array of user IDs. |
| next\_page\_pos | start\_pos in the next request to get the remaining user IDs. If the end of the user list is reached, this parameter is not present in the response message. |

**Communication example for obtaining a list of user IDs**

|  |  |  |  |
| --- | --- | --- | --- |
| order | request | response | description |
| 1 | {"mid": "1","cmd": "GetUserIdList","payload" : {}} | {"mid": "1","result": "UserIdList","payload" : {"user\_id" : [ "1", "5", "12", ... ],"next\_page\_pos": 50}} | The application requests a list of user IDs.The device returns the first 50 user IDs. |
| 2 | { "mid": "2", "cmd": "GetUserIdList", "payload" : { "start\_pos": 50 }} | { "mid": "2", "result": "UserIdList", "payload" : { "user\_id" : ["121", "124", "125", ...], "next\_page\_pos" : 100}} | The application requests a list of user IDs starting at position 50.The device returns the 51st to 100th user IDs. |
| 3 | {"mid": "3","cmd": "GetUserIdList","payload" : {"start\_pos": 100}} | {"mid": "3","result": "UserIdList","payload" : {"user\_id" : [ "183", "187", "190", ... ]} | The application requests a list of user IDs starting at position 100.The device returns user IDs starting from the 101th to the end of the list . |

## GetUserInfo

Get user information.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "GetUserInfo","payload" : { "id": "1" }} | { "mid" : "a123456", "result" : "UserInfo", "payload" : { "id" : "1","name" : "aaaaa","depart" : "bbbbb", "privilege": "user", "password" : "12345", "card" : "102845102","fp" : [ "base64-encoded data", ... ],"face" : "base64-encoded data","palm" : [ "base64-encoded data", ... ]}} |

**Request parameter description**

|  |  |
| --- | --- |
| id | User ID. Call the GetUserIdListcommand to obtain the user ID list. |

**Corresponding parameter description**

|  |  |
| --- | --- |
| id | User ID. |
| name | User's name. |
| depart | User's department. |
| privilege | Can be one of:* user ： ordinary user
* manager ：Administrator
* admin : super administrator
 |
| password | Password. If the device does not support password authentication, this parameter does not appear. |
| card | Card number. If the device does not support ID cards, this parameter will not appear. |
| fp | Fingerprint template. If the device does not support fingerprint, this parameter will not appear. |
| face | Face template. If the device does not support faces, this parameter will not appear. |
| palm | Palm template. If the device does not support palm, this parameter will not appear. |

## SetUserInfo

Change user information.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "SetUserInfo","payload" : { "id" : "1", "name" : "aaaaa", "depart" : "bbbbb", "privilege": "user", "password" : "12345" "card" : "102845102", "fp" : [ "base64-encoded data", ... ], "face" : "base64 encoded data", "palm" : [ "base64-encoded data", ... ]s }} | {"mid" : "a123456","result" : "Success","payload" : {}} |

**Request parameter description**

|  |  |
| --- | --- |
| id | User ID. If the ID does not exist, a new user will be created. |
| name | User name. Can be omitted to keep current name. |
| depart | User department. Can be omitted to keep current department. |
| privilege | Can be one of:* user ： ordinary user
* manager ：Administrator
* admin : super administrator

May be omitted to preserve the current user level. |
| password | Password. Can be omitted to keep the current password.If the device does not support password authentication, this parameter is ignored. |
| card | Card number. Can be omitted to keep the current card number.If the device does not support ID cards, this parameter is ignored. |
| fp | Fingerprint template. Can be omitted to keep current information.If the device does not support fingerprint, this parameter is ignored. |
| face | Face template. Can be omitted to keep current information.If the device does not support face recognition, this parameter is ignored. |
| palm | Palm template. Can be omitted to keep current information.If the device does not support palm, this parameter is ignored. |

(Please refer to "[Prohibit or allow the use of the machine](#_Prohibit,_allow_the)".)

## DeleteUserInfo

Delete a user .

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "DeleteUserInfo", "payload" : { "id": "1" }} | {"mid" : "a123456","result" : "Success","payload" : {}} |

**Request parameter description**

|  |  |
| --- | --- |
| id | User ID . Call the GetUserIdListcommand to obtain the user ID list . |

(Please refer to "[Prohibit or allow the use of the machine](#_Prohibit,_allow_the)".)

# Prohibit or allow the use of the machine

## LockDevice

Prohibit or allow the use of the machine.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "LockDevice","payload" : {"is\_locked": "yes"}} | {"mid" : "a123456","result" : "Success","payload" : { }}} |

**Request parameter description**

|  |  |
| --- | --- |
| is\_locked | "yes", prohibit the use of the machine."no" allows the use of the machine. |

When changing user information, if the user performs verification casually, it will lead to abnormal status.

Suggestion: Before changing the user information, prohibit the use of the machine. After changing the user information, allow the use of the machine.

# Remote registration of features

## BeginEnrollFace

Start the face registration process.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "BeginEnrollFace","payload" : {}} | {"mid" : "a123456","result" : "JobCreated","payload" : {"job\_id": 1}} |

**Response Parameter Description**

|  |  |
| --- | --- |
| job\_id | Unique job ID assigned by the device. This ID will be used in QueryJobStatusand CancelJobcommands. |

## BeginEnrollFP

Start the fingerprint registration process.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "BeginEnrollFp","payload" : {}} | {"mid" : "a123456","result" : "JobCreated","payload" : {"job\_id": 1}} |

**Response Parameter Description**

|  |  |
| --- | --- |
| job\_id | Unique job ID assigned by the device. This ID will be used in QueryJobStatusand CancelJobcommands. |

## BeginEnrollCard

Start the card registration process.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "BeginEnrollCard","payload" : {}} | {"mid" : "a123456","result" : "JobCreated","payload" : {"job\_id": 1}} |

**Response Parameter Description**

|  |  |
| --- | --- |
| job\_id | Unique job ID assigned by the device. This ID will be used in QueryJobStatusand CancelJobcommands. |

## BeginEnrollPalm

Start the palm registration process.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "BeginEnrollPalm","payload" : {}} | {"mid" : "a123456","result" : "JobCreated","payload" : {"job\_id": 1}} |

**Response Parameter Description**

|  |  |
| --- | --- |
| job\_id | Unique job ID assigned by the device. This ID will be used in QueryJobStatusand CancelJobcommands. |

## QueryJobStatus

Gets the current status of the registered job.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "QueryJobStatus","payload" : { "job\_id": 1 }} | { "mid" : "a123456", "result" : "JobStatus", "payload" : { "job\_id" : 1, "state" : "succeeded", "face\_data" : "base64 encoded data" "fp\_data": "base64 encoded data" "card\_data": "decimal integer" }} |

**Request parameter description**

|  |  |
| --- | --- |
| job\_id | Unique job ID assigned by the device. Call the [ BeginEnrollFace] or [ BeginEnrollFP] or [ [BeginEnrollCard](#_BeginEnrollCard) ] or [ [BeginEnrollPalm](#_BeginEnrollPalm) ] command to start the enrollment process and obtain the job ID. |

**Response Parameter Description**

|  |  |
| --- | --- |
| job\_id | Unique job ID assigned by the device. |
| state | Can be one of:* pending : This job has not yet been completed.
* succeeded : The job succeeded.
* failed : The job has failed or has been canceled.
 |
| face\_data | Base64-encoded face template.This parameter exists only when the job is created by the BeginEnrollFacecommand and the job has been completed (state =succeeded ). |
| fp\_data | Base64-encoded fingerprint template.This parameter exists only when the job is created through the [ BeginEnrollFP] command and the job has been completed (state =succeeded ). |
| card\_data | Decimal integer, card ID.This parameter exists only when the job is created through the [[BeginEnrollCard](#_BeginEnrollCard) ] command and the job has been completed (state =succeeded ). |
| palm\_data | Base64-encoded palm template.This parameter exists only when the job is created through the [[BeginEnrollPalm](#_BeginEnrollPalm) ] command and the job has been completed (state =succeeded ). |

## CancelJob

Cancels a feature registration job.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "CancelJob","payload" : {"job\_id": 1}} | {"mid" : "a123456","result" : "Success","payload" : {}} |

**Request parameter description**

|  |  |
| --- | --- |
| job\_id | Unique job ID assigned by the device. Call the [ BeginEnrollFace] or [ BeginEnrollFP] or [[BeginEnrollCard](#_BeginEnrollCard) ] or [ [BeginEnrollPalm](#_BeginEnrollPalm) ] command to start the enrollment process and obtain the job ID. |

## CancelAllJobs

Cancel any feature registration job, if one exists.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "CancelAllJobs","payload" : {}} | {"mid" : "a123456","result" : "Success","payload" : {}} |

## PhotoToFacedata

Convert photo into face template. (For visible light face recognition only.)

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid": "a123456","cmd": "PhotoToFacedata","payload": {"photo": "base64 encoded data(jpg)"}} | {"mid": "a123456","result": "PhotoToFaceData""payload": {"face\_data": "base64 encoded data","state": "succeeded""state": "process\_fail""state": "decode\_fail""state": "dimension\_overflow"},} |

**Request parameter description**

|  |  |
| --- | --- |
| photo | Base64-encoded photo data, must be a jpg file.Maximum dimension is 480x640. |

**Response Parameter Description**

|  |  |
| --- | --- |
| state | Can be one of:* succeeded : The conversion succeeded.
* process\_fail: The conversion failed.
* decode\_fail: Invalid jpg file.
* dimension\_overflow: The dimension overflow.
 |
| face\_data | Base64-encoded face template.This parameter exists only when the conversion succeeded (state =succeeded ). |

# Attendance record management

## GetAttendLog

Example

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "GetAttendLog","payload" : {"start\_pos": 0 // Optional}} | {"mid" : "a123456","result" : "AttendLog","payload" : {"start\_pos": 0,"next\_pos": 50,"logs" : [{"user\_id": "1","time": "2020-04-01T08:10:05","mode": "face "} ...]}} |

**Request parameter description**

|  |  |
| --- | --- |
| start\_pos | Optional, when this parameter is omitted, the device will return the first N records from the attendance record (N is device model specific, e.g. 100).When this parameter is specified, the device will start returning attendance records from this position. |

**Response Parameter Description**

|  |  |
| --- | --- |
| start\_pos | The starting position of the attendance record currently returned. |
| next\_pos | The end position of the currently returned attendance records. Specify this position in the next request to continue reading the remaining attendance records. |
| logs |  |
|  user\_id | The authenticated user ID. |
|  time | Verification time, in iso8601 format. |
|  mode | Authentication method. Can be a comma-separated list of the following values:* face ：Face verification
* fp : Fingerprint verification
* card: ID card verification
* pwd : Password verification
 |

**Incremental download of attendance records**

When downloading attendance logs, the application can continue from the last download position and only get the newly generated logs. To do this, the application should save the next\_pos parameter of the previous response message and provide the value to the next incremental download request.

## EraseAttendLog

Clear attendance records until the specified location.

**Example**

|  |  |
| --- | --- |
| Request | response |
| {"mid" : "a123456","cmd" : "EraseAttendLog","payload" : {"end\_pos": 10}} | {"mid" : "a123456","result" : "Success","payload" : {}} |

**Request parameter description**

|  |  |
| --- | --- |
| end\_pos | All previous attendance records for this position will be deleted. |

can be called after [ GetAttendLog] to delete the downloaded attendance records.

# Automatic reporting of attendance records

## ConfigAttendLogUploader

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "ConfigAttendLogUploader","payload" : {"target\_uri" : "http://10.80.0.1/upload\_log","interval" : 60}} | {"mid" : "a123456","result" : "Success","payload" : {}} |

**Request parameter description**

|  |  |
| --- | --- |
| target\_uri | The service address to which the attendance records will be uploaded. |
| Interval | Attendance record upload interval (in seconds). Should be an integer between 5 and 3600. |

**Attendance record reporting process**

Once configured, the device will first send an empty json object to the configured URI. The server should return the log position where the device should start sending, or an empty json object if no logs were received from the device.

|  |  |
| --- | --- |
| Request (Device 🡪Server) | Response (Server 🡪Device) |
| {} | {"next\_upload\_pos": 50} |

After that, the device will start sending attendance logs periodically from this position. After receiving each request, the server will return the next upload position. Usually, it should be equal to the "next\_pos" attribute in the request.

|  |  |
| --- | --- |
| Request (Device 🡪Server) | Response (Server 🡪Device) |
| {"start\_pos": 50 ,"next\_pos": 100 ,"logs" : [{"user\_id": "1","time": "2020-04-01T08:10:05","mode": "face "} ...]} | {"next\_upload\_pos*"* : 100} |

All requests are sent with the HTTP POST method.

## GetAttendLogUploaderStatus

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "GetAttendLogUploaderStatus", "payload" : {}} | { "mid" : "a123456", "result" : "AttendLogUploaderStatus", "payload" : { "status" : "active", "pending\_count" : 12 }} |

**Response Parameter Description**

|  |  |
| --- | --- |
| status | If the attendance log is being uploaded, it is "active" . Otherwise, it is "inactive" . |
| pending\_count | The number of attendance records stored in the device but not uploaded yet. |

# Time Synchronization

## GetDeviceTime

Get the device time.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "GetDeviceTime","payload" : {}} | {"mid" : "a123456","result" : "DeviceTime","payload" : {"time" : "2020-04-01T16:05:11"}} |

**Response Parameter Description**

|  |  |
| --- | --- |
| time | Current device time in iso8601 format. |

## SetDeviceTime

Set the device time.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "SetDeviceTime","payload" : {"time" : "2020-04-01T16:05:11"}} | {"mid" : "a123456","result" : "Success","payload" : {}} |

**Request parameter description**

|  |  |
| --- | --- |
| time | Set the time in iso8601 format. |

# Device network configuration

## GetNetworkConfig

Get the network configuration information of the device.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456", "cmd" : "GetNetworkConfig", "payload" : {}} | { "mid" : "a123456", "result" : "NetworkConfig", "payload" : { "ethernet": { "running": { "address": "192.168.0.5", "netmask": "255.255.255.0", "gateway": "192.168.0.1", "nameservers": [ "8.8.8.8", "4.4.4.4" ] }, "config": { "dhcp" : "off", "address": "192.168.0.5", "netmask": "255.255.255.0", "gateway": "192.168.0.1", "nameservers": [ "8.8.8.8", "4.4.4.4" ] } }, "wlan": { "running": { "address": "10.242.1.9", "netmask": "255.0.0.0", "gateway": "10.0.0.2", "nameservers": [ "10.0.0.2" ] },"config": {"dhcp": "on*"* ,"access\_points": [{ "ssid": "AP1", "key": "1234" }...]}}}} |

**Response Parameter Description**

|  |  |
| --- | --- |
| ethernet | Wired network parameters and status. If the device does not support wired network, this parameter will not exist. |
|  running | Current wired network status. |
|  address | IP address. |
|  netmask | Subnet mask. |
|  gateway | Default gateway. |
|  nameservers | DNS server address. |
|  config | Wired network configuration information. |
|  dhcp | "on" means the IP address is obtained dynamically via DHCP, otherwise "off". |
|  address | Static IP address. This parameter is only valid if dhcp is set to "off". |
|  netmask | Subnet mask. This parameter is only valid when dhcp is set to "off". |
|  gateway | Default gateway. This parameter is only valid when dhcp is set to "off". |
|  nameservers | DNS server addresses, up to 2. This parameter is only valid when dhcp is set to "off". |
| wlan | Wifi parameters and status. If the device does not support Wifi , this parameter will not exist. |
|  running | Current wifi status . |
|  address | IP address. |
|  netmask | Subnet mask. |
|  gateway | Default gateway. |
|  nameservers | DNS server address. |
|  config | Wifi configuration information. |
|  dhcp | "on" means the IP address is obtained dynamically via DHCP, otherwise "off". |
|  address | Static IP address. This parameter is only valid if dhcp is set to "off". |
|  netmask | Subnet mask. This parameter is only valid when dhcp is set to "off". |
|  gateway | Default gateway. This parameter is only valid if dhcp is set to "off". |
|  access\_points | Configured Wi ifi access point. |
|  ssid | Wi-Fi access point name. |
|  key | Wifi password. |

## SetNetworkConfig

Set up the device network configuration.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "SetNetworkConfig","payload" : {"ethernet": {"dhcp": "off*"* , "address": "192.168.0.5", "netmask": "255.255.255.0", "gateway": "192.168.0.1", "nameservers": [ "8.8.8.8", "4.4.4.4" ] }, "wlan": { "dhcp" : "on","access\_points": [{ "ssid": "AP1", "key": "1234" }...]}}} | {"mid" : "a123456","result" : "Success","payload" : {}} |

**Request parameter description**

|  |  |
| --- | --- |
| ethernet | Wired network parameters and status. If the device does not support wired network, this parameter is ignored. |
|  dhcp | "on" means the IP address is obtained dynamically via DHCP, otherwise "off". |
|  address | Static IP address. This parameter is only valid if dhcp is set to "off". |
|  netmask | Subnet mask. This parameter is only valid when dhcp is set to "off". |
|  gateway | Default gateway. This parameter is only valid if dhcp is set to "off". |
|  nameservers | DNS server addresses, up to 2. This parameter is only valid when dhcp is set to "off". |
| wlan | Wifi parameters and status. If the device does not support Wifi , this parameter is ignored. |
|  dhcp | "on" means the IP address is obtained dynamically via DHCP, otherwise "off". |
|  address | Static IP address. This parameter is only valid if dhcp is set to "off". |
|  netmask | Subnet mask. This parameter is only valid when dhcp is set to "off". |
|  gateway | Default gateway. This parameter is only valid if dhcp is set to "off". |
|  access\_points | Configured Wi ifi access point. |
|  ssid | Wi-Fi access point name. |
|  key | Wifi password. |

# Device information query and settings

## GetVersionInfo

Get version information.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "GetVersionInfo","payload" : {}} | {"mid" : "a123456","result" : "VersionInfo","payload" : {"firmware\_version": "13750C v3.11.802","face\_algorithm\_version": "GreenFace v2.2","fp\_algorithm\_version": "deli SMK v3.0"}} |

**Response Parameter Description**

|  |  |
| --- | --- |
| firmware\_version | Firmware version. |
| face\_algorithm\_version | Face algorithm version. If the device does not support face, this parameter will not exist. |
| fp\_algorithm\_version | Fingerprint algorithm version. If the device does not support fingerprint, this parameter will not be present. |

## GetCapacityLimit

Get the capacity limit of the device.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "GetCapacityLimit","payload" : {}} | {"mid" : "a123456","result" : "CapacityLimit","payload" : {"max\_user": 5000 ,"max\_face": 3000 ,"max\_fp" : 15000 ,"max\_attend\_log": 400000,"max\_palm": 1000}} |

**Response Parameter Description**

|  |  |
| --- | --- |
| max\_user | Maximum number of users. |
| max\_face | The maximum number of faces. If the device does not support faces, this parameter will not exist. |
| max\_fp | Maximum number of fingerprints. If the device does not support fingerprints, this parameter will not be present. |
| max\_attend\_log | Maximum number of attendance records. |
| max\_palm | Maximum number of palms. If the device does not support palms, this parameter will not be present. |

## GetChangeUserInfoOnDeviceAllowed

Get the device's local editing user information switch setting.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "GetChangeUserInfoOnDeviceAllowed","payload" : {}} | {"mid" : "a123456","result" : "ChangeUserInfoOnDeviceAllowed","payload" : {"is\_allowed": "yes"}} |

**Response Parameter Description**

|  |  |
| --- | --- |
| is\_allowed | "yes" if local user information editing is allowed, otherwise "no" . |

## SetChangeUserInfoOnDeviceAllowed

Set the device to edit user information locally.

**Example**

|  |  |
| --- | --- |
| request | response |
| {"mid" : "a123456","cmd" : "SetChangeUserInfoOnDeviceAllowed","payload" : {"is\_allowed": "yes"}} | {"mid" : "a123456","result" : "Success","payload" : {}} |

**Request parameter description**

|  |  |
| --- | --- |
| is\_allowed | "yes" allows local user information editing, otherwise, "no". |

## GetCurrentUsage

Get the registration information of the device.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "GetCurrentUsage", "payload" : {}} | { "mid" : "a123456", "result" : "CurrentUsage", "payload" : { "attend\_log\_count": 26, "card\_count": 0, "face\_count": 1, "fp\_count": 1, "manager\_count": 0, "palm\_count": 1, "password\_count": 0,"user\_count": 1 }} |

**Parameter Description**

It’s obvious.

## GetDeviceUid

Get the unique ID of the device.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "GetDeviceUid", "payload" : {}} | { "mid" : "a123456", "result" : "DeviceUid", "payload" : {"device\_uid": "e09c1d1fd6ac34a7" }} |

**Parameter Description**

|  |  |
| --- | --- |
| device\_uid | Unique ID of the device. 8-byte hexadecimal string. |

## GetDeviceCapabilities

Get the capabilities of the device.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "GetDeviceCapabilities", "payload" : {}} | { "mid" : "a123456", "result" : "DeviceCapabilities", "payload" : { "access\_control": false, "card": false, "ethernet": true, "face": true, "fingerprint": true, "input\_device": true, "palm": true, "password": true, "vrface": true,"wifi": true }} |

**Parameter Description**

|  |  |
| --- | --- |
| vrface | Whether to support visible light face recognition. |
| others | It’s obvious. |

## GetBellSetting, SetBellSetting

Get and write ring settings.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "GetBellSetting", "payload" : {}} | { "mid" : "a123456", "result" : "BellSetting", "payload" : { "bells": [ { "duration\_index": 6, "start": "08:00", "type\_index": 0, "weekmask": 62 },...24 ring settings }} |

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "SetBellSetting","payload" : { "bells": [ { "duration\_index": 6, "start": "08:00", "type\_index": 0, "weekmask": 62 },...24 ring settings}} | { "mid" : "a123456", "result" : "Success", "payload" : {}} |

**Parameter Description**

|  |  |
| --- | --- |
| start | Ringing start time |
| duration\_index | Ringing interval 0 - "None", 1 - "3s", 2 - "5s", 3 - "10s", 4 - "15s", 5 - "20s", 6 - "30s", 7 - "45s", 8 - "1min" |
| type\_index | Type 0 - "Inside", 1 - "Outside", 2 - "In/Out" |
| weekmask | Weekday bit mask Sun = bit 0, Mon = bit 1, Tue = bit 2, Wed = bit 3, Thu = bit 4, Fri = bit 5, Sat = bit 6 |

## GetDeviceId, SetDeviceId

Get and write device id.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "GetDeviceId", "payload" : {}} | { "mid" : "a123456", "result" : "DeviceId", "payload" : {"device\_id": 1 }} |

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "SetDeviceId","payload" : {"device\_id": 1}} | { "mid" : "a123456", "result" : "Success", "payload" : {}} |

**Parameter Description**

|  |  |
| --- | --- |
| device\_id | device id, 1~255 |

## GetSoundVolume, SetSoundVolume

Get and write the device sound volume.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "GetSoundVolume", "payload" : {}} | { "mid" : "a123456", "result" : "SoundVolume", "payload" : {"sound\_volume": 1 }} |

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "SetSoundVolume","payload" : {"sound\_volume": 1}} | { "mid" : "a123456", "result" : "Success", "payload" : {}} |

**Parameter Description**

|  |  |
| --- | --- |
| sound\_volume | device sound volume, 1~10 |

## GetReverifyTime, SetReverifyTime

Get and write the invalid time of repeated attendance (reverify time).

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "GetReverifyTime", "payload" : {}} | { "mid" : "a123456", "result" : "ReverifyTime", "payload" : {"reverify\_time": 1 }} |

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "SetReverifyTime","payload" : {"reverify\_time": 1}} | { "mid" : "a123456", "result" : "Success", "payload" : {}} |

**Parameter Description**

|  |  |
| --- | --- |
| reverify\_time | the invalid time of repeated attendance (reverify time), 0~59 （minutes） |

## GetVerifyMode, SetVerifyMode

Get and write verification mode.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "GetVerifyMode", "payload" : {}} | { "mid" : "a123456", "result" : "VerifyMode", "payload" : {"verify\_mode": 1 }} |

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "SetVerifyMode","payload" : {"verify\_mode": 1}} | { "mid" : "a123456", "result" : "Success", "payload" : {}} |

**Parameter Description**

|  |  |
| --- | --- |
| verify\_mode | verification mode 0 - "Any", 1 - "Finger", 2 - "Card + Finger", 3 - "Card", 4 - "Id + Finger", 5 - "Id + Password", 6 - "Card + Password", 7 - "Finger + Password", 8 - "Finger + Card + Password", 9 - "Face", 10 - "Card + Face", 11 - "Face + Password", 12 - "Face + Card + Password", 13 - "Face + Finger", 14 - "Face + Finger + Card", 15 - "Face + Finger + Password",When writing, if the device does not support the mode, an error is returned. |

## GetFaceVerifySetting, SetFaceVerifySetting

Get and write face recognition settings.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "GetFaceVerifySetting", "payload" : {}} | { "mid" : "a123456", "result" : "FaceVerifySetting", "payload" : {"antispoofing\_thre": 2,"multi\_veri\_enabled": true,"use\_antispoofing": true }} |

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "SetFaceVerifySetting","payload" : {"antispoofing\_thre": 2,"multi\_veri\_enabled": true,"use\_antispoofing": true}} | { "mid" : "a123456", "result" : "Success", "payload" : {}} |

**Parameter Description**

|  |  |
| --- | --- |
| use\_antispoofing | Liveness detection switch |
| antispoofing\_thre | Liveness detection threshold 0 - "Lower", 1 - "Low", 2 - "Normal", 3 - "High", 4 - "Higher"， |
| multi\_veri\_enabled | Multi-face detection switch |

## DeviceControl

Control the device.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "DeviceControl","payload" : { "Action": "ClearAttendLog"}} | { "mid" : "a123456", "result" : "Success", "payload" : {}} |

**Parameter Description**

|  |  |
| --- | --- |
| Action | Clear attendance records - "ClearAttendLog"Clear management records - "ClearAdminLog"Clear users - "ClearUsers"Clear managers - "ClearAdmins"Clear all data - "ClearAllData" |

# Shift settings and user shift settings

## GetShiftMode, SetShiftMode

Get and write the shift scheduling mode.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "GetShiftMode", "payload" : {}} | { "mid" : "a123456", "result" : "ShiftMode", "payload" : {"shift\_mode": "user" }} |

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "SetShiftMode","payload" : {"shift\_mode": "user"}} | { "mid" : "a123456", "result" : "Success", "payload" : {}} |

**Parameter Description**

|  |  |
| --- | --- |
| shift\_mode | User shift mode - "User",Department shift mode - "Department" |

## GetAttendaceTolerance, SetAttendaceTolerance

Get and write attendance statistics parameters.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "GetAttendaceTolerance", "payload" : {}} | { "mid" : "a123456", "result" : "AttendaceTolerance", "payload" : {"afternoon\_in\_tolerance": 0,"afternoon\_out\_tolerance": 0,"morning\_in\_tolerance": 0,"morning\_out\_tolerance": 0 }} |

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "SetAttendaceTolerance","payload" : {"afternoon\_in\_tolerance": 0,"afternoon\_out\_tolerance": 0,"morning\_in\_tolerance": 0,"morning\_out\_tolerance": 0}} | { "mid" : "a123456", "result" : "Success", "payload" : {}} |

**Parameter Description**

|  |  |
| --- | --- |
| morning\_in\_toleranceafternoon\_in\_tolerance | How many minutes late is recorded as late for work?， 0~120（minutes） |
| morning\_out\_toleranceafternoon\_out\_tolerance | How many minutes early should I leave work?， 0~120（minutes） |

## GetShiftSetting, SetShiftSetting

Get and write shift settings.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "GetShiftSetting", "payload" : {}} | { "mid" : "a123456", "result" : "ShiftSetting", "payload" : { "shifts": [ [ { "end": "12:00", "start": "09:00" }, { "end": "18:00", "start": "13:00" }, { "end": "", "start": "" } ],...24 shifts }} |

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "SetShiftSetting","payload" : { "shifts": [ [ { "end": "12:00", "start": "09:00" }, { "end": "18:00", "start": "13:00" }, { "end": "", "start": "" } ],...24 shifts}} | { "mid" : "a123456", "result" : "Success", "payload" : {}} |

**Parameter Description**

There are 24 shifts in total, each with 3 time periods, each with a start time and an end time.

The start time and the end time can be empty. The validity evaluation of the shift setting is performed in the device. If it is invalid during writing, an error is returned.

## GetUserShift, SetUserShift

Get and write user shift settings.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "GetUserShift","payload" : { "id": "1" }} | { "mid" : "a123456", "result" : "UserShift","payload" : { "id": "1","shift\_start\_day": "2024-11-13T00:00:00", "shift": [ 0,...100-day shift schedule }} |

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "SetUserShift","payload" : { "shift\_start\_day": "2024-11-13T00:00:00", "shift": [ 0,...100-day shift schedule}} | { "mid" : "a123456", "result" : "Success", "payload" : {}} |

**Parameter Description**

|  |  |
| --- | --- |
| id | User ID. |
| shift\_start\_day | For the shift start time, please check the demo source code. |
| shift | 100-day shift schedule0~23: One of 24 shifts24: Business trip25: Leave100: HolidayPlease check the demo source code. |

# Access control settings

## GetAcTimeZone, SetAcTimeZone

Get and write access control time zone settings.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "GetAcTimeZone", "payload" : {}} | { "mid" : "a123456", "result" : "AcTimeZone", "payload" : { "zones": [ [ { "end": "23:59", "start": "00:00" },...6 time ranges within a day ]...A total of 255 days }} |

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "SetAcTimeZone","payload" : { "zones": [ [ { "end": "23:59", "start": "00:00" },...6 time ranges within a day ]...A total of 255 days}} | { "mid" : "a123456", "result" : "Success", "payload" : {}} |

**Parameter Description**

There are a total of 255 days’ time zones, with 6 time ranges per day.

## GetAcUserInfo, SetAcUserInfo

Get and write user access control settings.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "GetAcUserInfo","payload" : { "id": "1" }} | { "mid" : "a123456", "result" : "AcUserInfo","payload" : { "id": "1", "time\_zone": [ 0, 0, 0, 0, 0, 0, 0 ], "valid\_period": { "end": "2000-01-01T00:00:00", "start": "2000-01-01T00:00:00", "use": false} }} |

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "SetAcUserInfo","payload" : { "id": "1", "time\_zone": [ 0, 0, 0, 0, 0, 0, 0 ], "valid\_period": { "use": false, "start": "2025-01-24T18:48:22", "end": "2025-01-24T18:48:22" } }} | { "mid" : "a123456", "result" : "Success", "payload" : {}} |

**Parameter Description**

|  |  |
| --- | --- |
| id | User ID. |
| time\_zone | The user's time zone number for a week. （0~254） |
| valid\_period | Valid period. This function only takes effect when [use = true]. |

## GetAcSetting, SetAcSetting

Get and write access control settings.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "GetAcSetting", "payload" : {}} | { "mid" : "a123456", "result" : "AcSetting", "payload" : { "monitor\_tamper": false, "multi\_verify\_count": 1, "unlock\_time": 5,"wiegand\_format": 0 }} |

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "SetAcSetting","payload" : { "monitor\_tamper": false, "multi\_verify\_count": 1, "unlock\_time": 5,"wiegand\_format": 0}} | { "mid" : "a123456", "result" : "Success", "payload" : {}} |

**Parameter Description**

|  |  |
| --- | --- |
| monitor\_tamper | Anti-tamper alarm switch |
| multi\_verify\_count | Confirmed number at the same time (Muti Verify Count) |
| unlock\_time | Door opening delay, 0~200 seconds |
| wiegand\_format | Wiegand output format0 - "None", 1 - "Wiegand 26",2 - "Wiegand 34"， |

# Department Management

## GetDeptList

Get department information.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "GetDeptList", "payload" : {}} | { "mid" : "a123456", "result" : "DeptList", "payload" : { { "name": "Company", "shift": [ 100, 0, 0, 0, 0, 0, 100 ], "user\_count": 1 },...Up to 100 departments }} |

**Parameter Description**

|  |  |
| --- | --- |
| name | Department Name |
| shift | Department's 1-week shift schedule0~23: One of 24 shifts24: Business trip25: Leave100: HolidayPlease check the demo source code. |
| user\_count | Number of users |

## SetDept

Department Operations.

**Example**

|  |  |
| --- | --- |
| request | response |
| { "mid" : "a123456", "cmd" : "SetDept","payload" : { "Action": "Rename", "dept\_org": "Company", "dept\_new": "NewName", "shift": [ 100, 0, 0, 0, 0, 0, 100 ] }} | { "mid" : "a123456", "result" : "Success", "payload" : {}} |

**Parameter Description**

|  |  |
| --- | --- |
| Action | Operation types"Rearrange": rearrange, no other parameters"Check": confirm existence, use dept\_org"Add": add, use dept\_new, optional shift"Delete": delete, use dept\_org"Rename": rename, use dept\_org and dept\_new, optional shift"SetShift": configure shifts, use dept\_org and shiftSee the demo source code. |
| dept\_org | Original department name |
| dept\_new | New department name |
| shift | Department's 1-week shift schedule0~23: One of 24 shifts24: Business trip25: Leave100: HolidayPlease check the demo source code. |