LifeSmart OpenDEV Advanced Interface Description_1.10

Version	Revise Date	Revisor	Revise Contents
1.0	2020/05/30	Ye ZhengQiang	As the completement of document《 Local Interfaces of LifeSmart Smart Station》
1.1	2020/07/23	Ye ZhengQiang	Add remote unlock and pairing of Yale doorlock module 2.1.5 remote unlock 2.3.2 pairing
1.2	2022/07/03	Ye ZhengQiang	Add DEFED Smart Station network commands 2.3.2 DEFED Smart Station WiFi configuration commands 2.3.3 DEFED Smart Station SIM card configuration commands 2.3.4 DEFED Smart Station network status query command 2.3.5 DEFED Smart Station Network Function Commands
1.4	2022/08/05	Ye ZhengQiang	DEFED smart station LED control command
1.5	2022/12/01	Ye ZhengQiang	infrared learning and infrared coding sending interface
1.6	2023/05/12	Ye ZhengQiang	The new interfaces are as follows: 2.3.6 DEFED Smart Station WiFi Network Function Command 2.5 All interfaces for OTA management of sub devices
1.7	2023/07/14	Ye ZhengQiang	The new interfaces are as follows: 2.3.2 WiFi network configuration command of defed smart station Configuration of supplementary BSSID 2.3.8 Ethernet network configuration command of defed smart station 2.3.9 MTU configuration commands for network cards in the defed smart station
1.8	2023/08/18	Ye ZhengQiang	Add 2.3.2 DEFED Smart Center WiFi Network Configuration Command Support for WPA3 2.3.10 DEFED Smart Center Network Status Query Command - WiFi List
1.9	2023/09/15	Ye ZhengQiang	Modify 2.1.5 Remote Unlock The new content is as follows 2.1.6 Setting LifeSmart V3 type door locks 2.1.7 SWIFT door lock module control interface
1.10	2023/12/15	Ye ZhengQiang	Add SwifteOta and HwInfo command in 2.1.7 SWIFT door lock module control interface

Catalog

1. Instruction

As the completement of document《Local Interfaces of LifeSmart Smart Station》, we provide more advanced API which are not included in that. The format of the interface and security signature are based on《Local Interfaces of LifeSmart Smart Station》.

2. API Details

2.1 Doorlock User Management

LifeSmart Doorlock series related to the management of temporary password users, so the valid time of all the user management request is 5 seconds.
So the timestamp of sys.ts in request and utc timestamp of Smart Station should less than 5 seconds.

2.1.1 Get user list of a specified doorlock

obj	doorlock
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"me": “2711",
	"cmd": “getuser”,
	}
	"obj": "doorlock",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	"msg": {
	"P_R_1": {
	"name": “XXX",
	"id": “P_R_1”,
	“user_type": “tempuser”,
	“user_info": {},
	“cnt": 1,
	“sts": 1571976095,
	“ets": 1571977095,
	“sprd": “0 0 8 ? * 1,2,6,7 *”,
	“eprd": “0 30 8 ? * 1,2,6,7 *”,
	},
	}
	}

Args	Currently SET->doorlock command： cmd:getuser —> Get user list of a specified doorlock in Smart Station, need to include parameter me(required) and userid(optional)； “me”: Doorlock device id, which is get by “GET-ep” or “GET-eps” “userid”: temporary useid, take“P_R_1” for instance which is returned by get a specified doorlock user list. Notes：”userid” is optional, if the userid of user is already be known which you want to get, please add userid in request, then the user information of this userid will be returned. If no userid parameter, then all the user list will be returned.

Response Msg	Msg Explanation： Msg is a key-value list. key: user list; Value: user information of this userid; user information include： 1.General information property(name、id、user_type、user_info)： “name”: optional, set it to user name ； “id”: userid ； “user_type”: user type, including finger(fingerprint user), password(password user), NFC(NFC card user), tempuser(temporary user) ； “user_info”: the customized info of user, set by third-party. Support numbers, strings, lists, and key-value list; 2.Temporary password specified information property (cnt、sts、ets、sprd、eprd)： “cnt”:=1, means this password is a one-time password; > 1, indicates the time when the password has been use; “sts/ets”: sts/ets is the validity period of the password, sts and ets must be exist at the same time. Sts and ets are UTC timestamp, sts is start time, ets is the end time; “sprd/eprd”: sprd and eprd are the time representation in CORN format, sprd is start time, eprd is the end time; sprd and eprd must be exist at the same time, format is”0 0 8 ? * 1,2,3,4,5,6,7 *”, the first three digits respectively represent seconds, minutes and hours in 24-hour time system. And the subsequent digits represent the validity period of 7 days a week, 1 represents Sunday, 7 represents Saturday. The weekly time settings of sprd and eprd must be consistent. For instance sprd=”0 30 8 ? * 1,3,5 *”, eprd=”0 30 18 ? * 1,3,5 *” , Indicates that the password is valid from 8:30 a.m. to 18:30 a.m. every Sunday, Tuesday, and Thursday.

2.1.2 Set user info of a specified doorlock

obj	doorlock
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"me": “2711",
	"cmd": “setuser”,
	"userid": “P_P_2”,
	"name": “test001”,
	}
	"obj": "doorlock",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	"agtid": "mga",
	"msg": “success",
	}

Args	Currently SET->doorlock command： “cmd”: setuser —> Set user info of a specified doorlock in Smart Station, need to include the parameters me(required), userid(required),name(optional), user_info(optional), newpw(temporary pw optional), oldpw(temporary pw optional), cnt(temporary pw optional), sts(temporary pw optional), ets(temporary pw optional), sprd(temporary pw optional), eprd(temporary pw optional); “me”: Doorlock device id; “userid”: temporary useid, which is returned by get a specified doorlock user list. “user_info”: the customized info of user, set by third-party. Support numbers, strings, lists, and key-value list; “name”: optional, set it to user name ; Notes: The preceding parameters can be set for all users. Notes: The following parameters can be set only for temporary users. “newpw/oldpw”: Temporary password of the temporary user to be set , newpw and oldpw must be exsit at the same time. The new password is set only when the old password is successfully authenticated. The password contains 6 to 12 digits before it is encoded in Base64. When sending a password, the password must be encoded in Base64. For example, the password 678901 is encoded in Base64 and its value is Njc4OTAx. “cnt”:=1, means this password is a one-time password; **>** 1, indicates the time when the password has been use;=0, indicates clear the original cnt parameter. “sts/ets”: sts/ets is the validity period of the password, sts and ets must be exist at the same time. Sts and ets are UTC timestamp, sts is start time, ets is the end time; Set sts/ets=0, indicates clear the original sts/ets parameter. “sprd/eprd”: sprd and eprd are the time representation in CORN format, sprd is start time, eprd is the end time; sprd and eprd must be exist at the same time, format is”0 0 8 ? * 1,2,3,4,5,6,7 *”, the first three digits respectively represent seconds, minutes and hours in 24-hour time system. And the subsequent digits represent the validity period of 7 days a week, 1 represents Sunday, 7 represents Saturday. The weekly time settings of sprd and eprd must be consistent. Set sprd/eprd =0, indicates clear the original sprd/eprd parameter. For instance sprd=”0 30 8 ? * 1,3,5 *”, eprd=”0 30 18 ? * 1,3,5 *” , Indicates that the password is valid from 8:30 a.m. to 18:30 a.m. every Sunday, Tuesday, and Thursday. Note: If the week is not limited, the latter half is empty, the whole is "0, 30, 8? * * ";If there is no CNT, STS/ETS, SPRD/EPRD parameter, the password is valid until it is deleted or disabled.
Response Msg	Msg Explanation： Msg=success, indicates send successfully;

2.1.3 Add temporary user of a curtain doorlock

obj	doorlock
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"me": “2711",
	"cmd": “addtempuser”,
	"id": 3,
	"name": “test002”,
	“password": “Njc4OTAx”,
	}
	"obj": "doorlock",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	"msg": {
	“userid": “P_R_3”,
	}
	}

Args	Currently SET->doorlock command： cmd:addtempuser —> Add temporary user of a curtain doorlock in Smart Station, these parameters should be included： me(required),id(required),password(required),name(required),cnt(optional),sts(optional),ets(optional),sprd(optional),eprd(optional)； “me”: doorlock device id; “id”: the number of the temporary password user to be added, range [1-255]； “password”: The temporary pw of temporary user. Password contains 6 to 12 digits before it is encoded in Base64. When sending a password, the password must be encoded in Base64. For example, the password 678901 is encoded in Base64 and its value is Njc4OTAx. “name”: optional, the name of temporary password user; “cnt”:=1, means this password is a one-time password; > 1, indicates the time when the password has been use; “sts/ets”: sts/ets is the validity period of the password, sts and ets must be exist at the same time. Sts and ets are UTC timestamp, sts is start time, ets is the end time; “sprd/eprd”: sprd and eprd are the time representation in CORN format, sprd is start time, eprd is the end time; sprd and eprd must be exist at the same time, format is”0 0 8 ? * 1,2,3,4,5,6,7 *”, the first three digits respectively represent seconds, minutes and hours in 24-hour time system. And the subsequent digits represent the validity period of 7 days a week, 1 represents Sunday, 7 represents Saturday. The weekly time settings of sprd and eprd must be consistent. For instance sprd=”0 30 8 ? * 1,3,5 *”, eprd=”0 30 18 ? * 1,3,5 *” , Indicates that the password is valid from 8:30 a.m. to 18:30 a.m. every Sunday, Tuesday, and Thursday. Note: If the week is not limited, the latter half is empty, the whole is "0, 30, 8? * * ";If there is no CNT, STS/ETS, SPRD/EPRD parameter, the password is valid until it is deleted or disabled.
Response Msg	Msg Explanation： “userid”: The newly generated temporary password user number;

2.1.4 Delete temporary user of a curtain doorlock

obj	doorlock
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"me": “2711",
	"cmd": “deltempuser”,
	“userid": “P_R_3”,
	}
	"obj": "doorlock",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	"msg": “success",
	}

Args	Currently SET->doorlock command： “cmd”: deltempuser —> Delete temporary user of a curtain doorlock in Smart Station, parameters me(required) and userid(required) need to be added. “me”: doorlock device id; “userid”: the temporary password user number, which is returned by the user list interface to obtain the specified lock in Smart Station. Only temporary password user can be deleted, format is “P_R_+ digit”.
Response Msg	Msg Explanation： Msg=success, indicates send successfully;

2.1.5 Remote Unlock(only for Yale doorlock module)

Remote unlocking function, supporting door lock device SL_ P_ BDLK , SL_ LK_ SG), SL_ LK_ YL , SL_ LK_ LS_ V3(doorlock pro X), SL_ LK_ TY (C100 door lock), SL_ LK_ DJ (C200/C210 door lock). Among them, SL_ LK_ TY does not support remote door unlocking in sleep mode, and must be awakened to remotely open the door.

obj	doorlock
Pkg_type	SET

Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"me": “2711",
	"cmd": “openlock”,
	}
	"obj": "doorlock",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	"msg": “success",
	}

Args	Currently SET->doorlock command： “Cmd”:openlock —> To run the remote unlock command, add the parameter me(required)； “me”: doorlock device id;
Response Msg	Msg Explanation： Msg=success, indicates send successfully;

2.1.6 Setting LifeSmart V3 type door locks

obj	ctldoorlockv3
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"me": “2711",
	"cmd": “ctldoorlockv3”,
	“act": “DelUser”,
	“actargs": { “userid": 4128 },
	}
	"obj": "doorlock",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	"msg": “success",
	}

Args	Current SET ->doorlock command instruction: cmd: ctldoorlockv3->The setting control command for LifeSmart V3 door locks requires adding parameters me (necessary), act (necessary), and actargs (optional); Me:device id of the smart station's door lock, representing the corresponding LifeSmart V3 door lock; The act parameter is a control action, with a type of string. For details, please refer to the list of act and actargs parameters in 2.1.6.1; The actargs parameter is a control action parameter, which is a serialized string of JSON list objects or JSON objects, related to the parameter act. For details, refer to 2.1.6.1 act and actargs parameter lists;
Response Msg	Return Data Description: When the code is 0, it indicates that the command was successful. Please refer to the parameter list of act and actargs in 2.1.6.1 for specific return results

2.1.6.1 The list of act and actargs parameters

Function	act Operation command , string	actargs Operation command parameter, a serialized string of a JSON object	Response

Access to door lock information	Info	None	{...} ● passwordAdminN indicates the number of Admin password users. ● fingerAdminN indicates the number of Admin fingerprint users. ● nfcAdminN indicates the number of Admin NFC users. ● passwordN indicates the number of password users (including Admin users). ● fingerN indicates the number of fingerprint users (including Admin users). ● nfcN indicates the number of NFC users (including Admin users).
Get a list of users	GetUsers	{fromAID, maxcnt}. ● fromAID indicates the serial number of the UserID at the start of the search, is numeric, e.g. for a password user, the starting serial number is 4096. ● maxcnt indicates the number of entries to be fetched at one time, and is numeric and cannot be greater than 10; its default value is 10.	{users:[{userid,flag}]} The users array indicates the corresponding list of users to be returned. ● userid indicates the UserID serial number of the user. ● flag=3 indicates that it is an Admin user. ● flag=1 indicates that it is a normal user.
Create user	AddUser	{userid, userpwd}. ● userid indicates the new UserID serial number and is of type numeric, e.g. for password users, the starting serial number is 4096. note: 1. if a UserID serial number already exists, it cannot be created again, i.e. the password of a UserID cannot be changed, it must be deleted before the same UserID can be created again; 2. password users can only be created in groups of up to 32 including Admin password users), so the userid must be less than 4096+32. ● userpwd indicates the user's password, which is a string of type, the content of which can only be numeric characters, and its length cannot be less than 6 or greater than 24.	"SUCCESS"

Delete user	DelUser	{userid}. ● userid specifies the serial number of the UserID to be deleted, and is of type numeric, e.g. for password users, the starting serial number is 4096. note: Admin users can also be deleted, but only if the current Admin user of the system is greater than 2, otherwise deletion is not allowed.	"SUCCESS"

2.1.7 SWIFT door lock module control interface

Only for Lifrsmart V3(Lock pro X,devtype is SL_LK_SWIFTE)

obj	ctldoorlockswifte
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"me": "2711",
	"cmd": "ctldoorlockswifte",
	"act": "GetMode",
	}
	"obj": "doorlock",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	"agtid": "mga",
	"msg": {
	"motorVoltage": 0,
	"primaryAndSecondaryLockSetting": 0,
	"magnetic": 1,
	"unlockMode": 0,
	"bleSetting": 1,
	"keepOpenAfterTurningKnob": 1,
	"volume": 3,
	"ocButton": 1,
	"time": 1658671960,
	"direction": 1,
	"lenSetting": 32,

	} }
Args	Current SET ->doorlock instruction: Cmd: ctldoorlockswifte ->The setting control command for the SWIFT door lock module requires the addition of parameters me (necessary), act (necessary), and actargs (optional); Me: the device id of the smart station's door lock, representing the corresponding SWIFT door lock module; The act parameter is a control action, with a type of string. For details, refer to 2.1.7.2 List of act and actargs parameters; The actargs parameter is a control action parameter, which is a JSON list object or a serialized string of JSON objects. It is related to the parameter act, as detailed in 2.1.7.2 Act and actargs parameter lists;
Response Msg	Return Data Description: When the code is 0, it indicates that the command was successful. For specific return results, please refer to the parameter list of act and actargs in 2.1.7.2

2.1.7.1 UserID description

- For SWIFTE door locks the user ID value is 2 byte.

The high 4 bits of the first byte are 0 and the low 4 bits indicate the AdminUser (both Owner and Master), the default AdminUser ID value is 0 (Owner) and takes the value range [0-9].

The second byte value is the SubUserID under that AdminUser and takes a value in the range [0-99] which means the ID of the common user of the SWIFTE door lock consists of AdminUserID and SubUserIID.

- Range of values for SubUserID:

When AdminUserID=0: 10 to 99

When AdminUserID>0: 0 to 99

When creating a normal User, you must ensure that the AdminUser already exists, otherwise it returns a failure, e.g. to create 322 users, you must ensure that AdminUser 3 already exists

- AdminUser needs to be configured on the door lock to add

An AdminUser can create up to 100 SubUserIDs (AdminUser=0 can only create 90)

If a UserID already exists, you can choose whether to modify the pwd when creating it. For details, see the AddUser instruction description.

2.1.7.2 The list of act and actargs parameters

Function Function	act Operation comm , string	actargs Operation command parameter, a serialized string of a JSON object	Response
Synchronised user list	Sync	None. Note: 1. It is best to call Sync once before calling GetUsers,Info, otherwise the information will be out of sync. If you call GetUsers directly after AddUser/DelUser without calling Sync, you will find that the users are out of sync. Therefore, it is best to call Sync once before calling GetUsers.2: If the number of users is greater than 10 and multiple calls are required, it is not necessary to call Sync once before each call to GetUsers, but only once before the first call. 3: You will need to wait 2 seconds after calling Sync before calling the GetUsers,Info command.	"SUCCESS"
Access to door lock information	Info	None	{...} ● userN indicates the number of users of the door lock; (includes Admin users) ● bat Indicates the battery level. The normal valid value range is [0-100]. If the value is something else it means that the battery information is not available at the moment.
Get a list of users	GetUsers	{fromAID, maxcnt}. ● fromAID indicates the serial number of the UserID at the start of the search, and is of type numeric, refer to 2.5.1 for details. Note: the UserID returned by the query does not include fromAID, i.e. it is greater than fromAID. ● maxcnt indicates the number of entries to be fetched at one time, and is of type numeric and cannot be greater than 7; its default value is 7.	{users:[{userid,flag}]} ● The users array indicates the corresponding list of users to be returned. ● userid indicates the UserID serial number of the user. ● flag=1 indicates that it is a normal user.

Create user	AddUser	{userid, userpwd}. ● userid indicatess the new UserID serial number, the type is numeric, please refer to 2.5.1 for UserID. Note:If 0xNFF is used then the door lock will automatically assign an available SubUserID under this AdminUser with AdminUserID=N. ● userpwd indicates the user's password and is of type string, the content of the string can only be numeric characters and its length cannot be less than 4 or greater than 8. • userpwdRaw indicates the original user authentication data, which is of type byte array and can be used to add IC card users, although password users can still use it. The first byte indicates the authentication type, the second byte indicates the length of authentication data, and the remaining data is authentication data. For example, adding a regular password user with the password "123456" would result in userpwdRaw values of \ [5,6,49,50,51,52,53,54 \]. Note: userpwdRaw has higher priority than userpwd. If userpwdRaw is specified, the userpwd parameter will be ignored. Example: IC card information: 04EFCCE2021290(Needn to get it by phone which support NFC) If the IC card is less than 16 bits (16Byte), zero should be added at the end. After completion, the IC card information is: 0x04EFCCE2021290000000 Divide each byte into: 04, EF, CC, E2, 02, 12, 90, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00,	"SUCCESS"

		00 Convert to decimal as: 4, 239, 204, 226, 2, 18, 144, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 "Actargs": "{\" userid \ ": 255, \" userpwdRaw \ ": [4, 16, 4, 239, 204, 226, 2, 18, 144, 0, 0, 0, 0, 0, 0, 0, 0, 0]}" ● UserTimeLimitRaw (Optional): byte [], user time limited array, if provided, the length must be 10 bytes; ● Update indicates whether to modify the user if the userid already exists. The type is numeric. 1 indicates to modify the user if it exists, and 0 indicates to return an error if it exists. The default value is 0	
Delete user	DelUser	{userid}. ● userid indicates the serial number of the UserID to be deleted, numeric which is the value of the UserID returned by GetUsers. Tip: If the second byte of userid, SubUserID, has a value of 0xFF, all users under Admin indicated by the first byte, (AdminUserID) will be deleted. If the value of AdminUserID is 0 and the value of SubUserID is 0xFF, all users under the owner will be deleted.	"SUCCESS"
Search for user information	GetUser	{userid}. ● userid indicates the serial number of the UserID to be queried, numeric, whcih is the value of the UserID returned by GetUsers.	{password,authMode,daySetting,effectiveTs1,effectiveTs2,id} ● password is the password for that user. ● authMode is the authentication mode for the user, currently the value is 5. ● daySetting is the date setting for this user, the current default value is 65535. ● effectiveTs1 is the effective start time, the current default value is 0. ● effectiveTs2 is the effective termination time, the current default value is 0. ● id is the internal user ID.

Setting the door lock	SetMode	● {time, direction, volume, unlockMode, ocButton, primaryAndSecondaryLockSetting, magnetic, keepOpenAfterTurningKnob, motorVoltage, bleSetting, lenSetting} ● Please refer to the Response section of the GetMode command for the meaning of each parameter and the value to be set. Note: All of the parameters listed above must be provided when setting a mode, if some parameters are not provided then the default values will be used, please refer to the Response section of the GetMode command for the default values. ●	"SUCCESS"
Query door lock settings	GetMode	None	{...} ● Time: The current UTC time of the door lock, in seconds. ● Direction: Direction of door lock rotation. 0: left; 1: right (default) ● Volume 0: Mute; 1: Low volume; 2: Medium volume (default); 3: High volume ● unlockMode: 0: Auto lock S (default); 1: Auto lock L; 2: Manual spindle; 3: Unlock same ● ocButton OC button. 0: Invalid; 1: Valid (default) One byte represents the meaning of a value. Low 4 bit configuration ocButton:0x?0:Invalid;0x?1:Valid (default) High 4bit configuration A contact:0x0?:Valid (default);0x1?:Invalid ● primaryAndSecondaryLockSetting: primary and secondary lock settings. 0: primary (default); 1: secondary ● magnetic door. 0: no; 1: yes (default); 2: yes (automatic locking only, no lock picking alarm) ● keepOpenAfterTurningKnob Keep the lock open after turning the knob. 0: ON (default, keep lock always open); 1: OFF (auto-lock) ● motorVoltage Motor voltage. 0: 6V

			(default); 1: 8V (powerful) ● bleSetting BLE setting. 0: OFF; 1: ON (default) ● lenSetting The touchpad random code setting and the length of the IC card that can be used. Default value is 0x25
Remote Lock and Unlocking	Oper	● {operType}. ● The operType specifies the type of operation and can be "Open", "Close", "KeepOpen". The values are "unlock", "lock" and "KeepUnlock".	● "SUCCESS"
Get FW version of outer/inner/BLE	HwInfo		{IO_INNER_Ver,IO_OUTER_Ver,BLE_Ver} • IO_INNER_Ver： the FW version of inner • IO_OUTER_Ver： the FW version of outer BLE_Ver： the version of BLE

Upgrade the FW of inner/outer	SwifteOta	{tag, ver, delaysec} ● **tag** indicates the upgrade object, which is a string type. "IO_INNER" indicates the upgrade of internal locks, and "IO_outer" indicates the upgrade of outer.. ● **ver** indicates the target version number, which is a string type, such as "0.3.3.22". ● **Delaysec** indicates the delay of each packet of data during the upgrade process, with a value range of [0,0.5] and a unit of seconds. For example, 0.1 represents a delay of 0.1 seconds for each packet operation, and 0 represents the fastest upgrade speed. When set 0, the GW is busy and other CoSS devices may not be able to communicate properly. This parameter defaults to 0.	If the command is executed successfully, it returns {code: 0, message: "EBGR"} ● **Code**: 0 indicates successful command execution ● **message**: "EBGR" indicates that the command will not be completed immediately and the task execution needs to be started in the background until it is completed. Tip: To query the progress of task execution, you can call the 2.5.6. Tip: You can call the 2.5.1 to upload OTA files. Note: 1. The GW version>=1.0.93, and the CoSS module version>=14(060e) only support the OTA function of locks; 2. Outer upgrade process:GW send the FW of outer to inner then the inner forward the FW of outer to outer, finally the new firmware will cover the old one. After the GW sends the FW of outer to inner,the LED of outer will flash slowly and will be off when the covering is completed,which means you can operate the doorlock.Do not operate the lock during the upgrading.(It takes approximately 3 minutes to transfer firmware from the inner to the end of the upgrade.)If upgraded improperly, it may cause damage to the outer, and frequent upgrades are not recommended. Note: Before OTA, you can first call the Remove command of the 2.5.7 to delete the previous tasks, so as not to interfere with the subsequent OTA upgrade

2.2 SPOT(Coss)-Remoter

LifeSmart SPOT is related to the IR remoter operation, because of the resource limited of Smart Station, so the whole code library will not be placed in it. Please make sure the remote controllers have been configured on LifeSmart APP, then send request to us. The request may as below: get remoter list, get key list of a specified remoter, transmit key code to this specified remoter.

2.2.1 Get remoter list

obj	spotremote
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"cmd": “getlist”,
	},
	"obj": “spotremote",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	"msg": [
	{
	"id": “AI_IR_5bd1_1590665106”,
	“name": "CHANGHONGremotecontroller",
	"panel": “TV_A",
	"brand": “changhong"
	},
	{
	"id": “AI_IR_5bd1_1590665188",
	“name": "xiaomiremotecontroller",
	"panel": “BOX_A”,
	"brand": “xiaomi"
	}
	]
	}

Args	Currently SET->spotremote command： cmd:getlist —> Get remoter list；

Response Msg	Msg Explanation： Msg is the remoter list returned. “id”: id of remoter assigned by LS system; “name”: the name of remoter; “brand”: the brand of remoter; “panel”: the category of remoter; The category of remoter (panel) including： AC_A(Air Condition/Heat Pump), TV_A(TV Remote Control), BOX_A(Box Remote Control), BOX_APPLE(APPLE Box Remote Control), STB_A(IPTV Remote Ccontrol), DVD_A(DVD Remote Control), FAN_A(Fan Remote Control), HUMI_A(Humidifier Remote Control), CUSTOM_A(Customize)

2.2.2 Get key-value list of a specified remoter

obj	spotremote
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"id": “AI_IR_5bd1_1590665106",
	"cmd": “getkeys”,
	},
	"obj": "spotremote",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	"msg": {
	“MENU": “”,
	“4": “”,
	“1": “”,
	……,
	}
	}

Args	Currently SET->spotremote command： “cmd”:getkeys —> get key list of a specified remoter, id is required(For instance, "id": “AI_IR_5bd1_1590665106"); Parameter “id” is returned in “Get remoter list”；
Response Msg	Msg Explanation： Msg shows the key list of remoter in our category and the customized key value(produced in customized remoter) returned, we also call this list as key-value list. In key-value list，”MENU”,”4”, ”1” is the key(will be used in 2.2.3), value is empty in this sample because it’s not a customized remoter.

2.2.3 Transmit key code to this specified remoter

obj	spotremote
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"id": “AI_IR_5bd1_1590665106",
	"cmd": “sendkey”,
	"key": “9”,
	},
	"obj": "spotremote",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}

	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	"msg": “success",
	}

Args	Currently SET->spotremote command： “cmd”: sendkey —> transmit key code to this specified remoter, “id” and “key” parameters are required； Parameter “id” is returned in “Get remoter list”； Parameter “key” is returned in “Get key-value list of a specified remoter”;
Response Msg	Msg Explanation： Msg return “successs”, means transmit successfully.

2.2.4 send IR code

obj	spotremote
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"me": “2711",
	"cmd": “sendcodes”,
	“keys": [{
	“param”:{
	“type”: 1,
	“duty”: 3,
	“delay”: 1,
	“data”:
	“018C500320016E500654014D500A5540016E500654014D500E55546E50014D500A5540016E5004504D50016E500A5540FF0001AC8C036001506E0754014D500A5540016E500654014D500E55546E50014D500A5540016E5004504D50016E500A5540FF0001AC8C036001506E0754014D500A5540016E500654014D500E55546E50014D500A5540016E5004504D50016E500A5540”,
	},},]
	},
	"obj": "spotremote",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	"msg": “success",
	}

Args	Current SET ->spotremote command: Cmd: sendcodes ->Assign specific infrared transmitting equipment under the specified smart station to transmits the infrared codes in the specified infrared code list, and the parameters me (necessary) and keys (necessary) need to be added; The me parameter is the equipment number of the infrared emission equipment in the smart station, such as the Spot ， The keys parameter is a list of infrared codes to be sent. The parameters of each element in the list are as follows: Param parameter is the infrared coding data table, including the following parameters: Type: code type, generally 1;

	Duty: code duty ratio, which is generally 3; Data: encoded data string; Freq: the coding frequency, which is generally 38000. It is optional; Delay: the time to delay sending the next code after sending the first code. It is optional. It is necessary to fill in when sending multiple codes
Response Msg	Msg Explanation： Msg return “successs”, means transmit successfully.

2.2.5 IR code learning

obj	spotremote
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"me": “2711",
	"cmd": “learncode”,
	},
	"obj": "spotremote",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	"msg": {
	“type”: 1,
	“duty”: 3,
	“data”:
	“018C500320016E500654014D500A5540016E500654014D500E55546E50014D500A5540016E5004504D50016E500A5540FF0001AC8C036001506E0754014D500A5540016E500654014D500E55546E50014D500A5540016E5004504D50016E500A5540FF0001AC8C036001506E0754014D500A5540016E500654014D500E55546E50014D500A5540016E5004504D50016E500A5540”,
	},
	}

Args	Current SET ->spotremote command: Cmd: learncode ->Trigger the device under the smart station to learn the code. The maximum timeout for learning the infrared code is 30 seconds. You need to add the parameter me (necessary); The me parameter is the equipment number of the infrared emission equipment in the smart station, such as the Spot;
Response Msg	Return data description: If msg returns the infrared code data table, the learning is successful. The parameters are described as follows: Type: code type, generally 1; Duty: code duty ratio, which is generally 3; Data: encoded data string;

2.3 Add Smart Station configuration instructions

2.3.1 Set Smart Station time zone and whether to use daylight saving Time (DST)

obj	config
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"cfg": “timezone”,
	"timezone": 8,
	"summer": false,
	},
	"obj": "config”,
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	"agtid": "mga",
	"msg": {
	"timezone": 8,
	"summer": false,
	}
	}

Args	SET->config directive currently supports the following commands: “cfg”: timezone —> Set Smart Station time zone and whether to use DST, need to add parameters timezone and summer ; “timezone”: The value range is [-12,12]. The positive value is east time zone, and the negative value is West time zone. “Summer”: whether to use DST, value is true or false ; Notes: if both timezone and summer do not exist, the current timezone and DST are returned.
Response Msg	Msg Explaination : “timezone”: The value range is [-12,12]. The positive value is east time zone, and the negative value is West time zone. “Summer”: whether to use DST, value is true or false ;

2.3.2 DEFED Smart Station WiFi configuration commands

obj	config
Pkg_type	SET

Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"cfg": “net”,
	"cmd": “setifn”,
	"cmdargs": {
	"ifname": “wlan0”,
	"enable": true,
	"metric": 50,
	"C_NetworkType": “STA”,
	"C_AuthMode": “OPEN”,
	"C_EncrypType": “WPA2”,
	"C_SSID": “SSID”,
	"C_WPAPSK": “password”,
	},
	},
	"obj": "config”,
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	"agtid": "mga",
	"msg": “success",
	}

Args	The current SET->config command supports the following commands. cfg:net -> the cmd and cmdargs need to be set when configuring the WiFi network of the Smart Station. The cmd needs to be set to the character "setifn", which means it is a network configuration command. The cmdargs is a functional command parameter, which needs to contain the following parameters: Ifname: the network interface name (required), WiFi configuration command interface "wlan0"; Enable: WiFi on or off (required), boolean value; Metric: the priority of WiFi network (optional), value 0 50 80 100 (optional); C_NetworkType is the WiFi operating mode, (mandatory if enable=true), the value is "STA"/"AP"; C_AuthMode is whether to encrypt or not, (mandatory if enable=true), the value is "OPEN"/"CLOSE"; C_EncrypType is the encryption type, "WEP"/"WPA"/"WPA2"/"WPA/WPA2 "/"WPA3",WPA3 is a new support and only supports C_ NetworkType=STA; C_SSID is the WiFi hotspot name ((mandatory if enable=true); C_WPAPSK is the WiFi hotspot password; C_HWMode is whether the WiFi is used as a 5G WiFi hotspot or not when the WiFi is used as an AP hotspot; *C_ BSSID specifies the router address for WiFi, which is generally used to distinguish between duplicate APS; Note: if C_ BSSID is configurated, which must be cleared (set as an empty string) or changed to the correct address when replacing the AP;
Response Msg	Response description. When msg is success, it only means that the configuration is sent successfully. Judging whether WiFi network is successfully on needs to be judged by <2.3.4 DEFED Smart Station Network Status Query Command> interface, which is generally judged by whether IP is obtained or not.

2.3.3 DEFED Smart Station SIM card configuration commands

obj	config
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"cfg": “net”,
	"cmd": “setifn”,
	"cmdargs": {
	"ifname": “wwan0”,
	"enable": true,
	"metric": 50,
	"C_apnname": “”,
	"C_username": “XXXX@XXXX.XXX”,
	"C_password": “XXXXXXXX”,
	"C_pref": 0,
	},
	},
	"obj": "config”,
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	"agtid": "mga",
	"msg": “success",
	}

Args	The current SET->config command supports the following commands. cfg:net -> the cmd and cmdargs need to be set when configuring the WiFi network of the Smart Station. The cmd needs to be set to the character "setifn", which means it is a network configuration command. The cmdargs is a functional command parameter, which needs to contain the following parameters. Ifname: the network interface name (mandatory), SIM card configuration interface "wwan0"/"usb0"/"ppp0"; Enable: SIM card on or off (mandatory), boolean value; Metric:the priority of SIM card(optional), value 0 50 80 100(optional) When ifname is "wwan0", 4G Cat4 (generally used overseas), C_apnname, C_username, C_password, C_pref are all SIM card vendor information; The parameters are explained as follows: C_apnname is the APN, which must be set for private network cards, but not for public network cards; C_username is the authenticated user name; C_password is the authentication password; C_pref is the authentication type, which is specific to the industry-specific card, the value range is [0-3]: 0 : NONE 1 : PAP 2 : CHAP 3 : PAP or CHAP When ifname is "usb0", it is 4G Cat1 (used in China) without other parameters; When ifname is "ppp0", it is 2G SIM card with the parameters as follows: C_operator is the dial-up mode (mandatory under ppp0), the value is: "CUCC_WCDMA" "CTCC_CDMA2000" "CMCC_TD-SCDMA" "CMCC_GPRS"
Response Msg	Response description. When msg is success, it only means that the configuration is sent successfully. Judging whether SIM card network is successfully on needs to be judged by <2.3.4 DEFED Smart Station Network Status Query Command> interface, which is generally judged by whether IP is obtained or not.

2.3.4 DEFED Smart Station network status query command

obj	config

Pkg_type	SET
Direction	Device->Station
Request	{ "id": 2, "args": { "cfg": “net”, "cmd": “getifn”, "cmdargs": { "ifname": “wlan0”, “value": false, }, }, "obj": "config”, "sys": { "ver": 1, "ts": 1571976095, "sign": "dbe2076ba2a67fe886aa5098d165ac7a", "model": "OD_XXX_XXX" } }
Response	{ "code": 0, "id": 2, "agtid": "mga", "msg": { "wlan0": { “enable": true, “metric": 50, "C_NetworkType": “STA”, "C_AuthMode": “OPEN”, "C_EncrypType": “WPA2”, "C_SSID": “SSID”, "C_WPAPSK": “password”, . . . , "_info": { "ip": “192.168.33.69”, "gw": “192.168.33.1”, . . . , }, }, } }

Args	The current SET->config command supports the following commands. cfg:net -> the cmd and cmdargs need to be set when obtaining the network interface status of the Smart Station. The cmd needs to be set to the character "setifn", which means it is a network configuration command. The cmdargs is a functional command parameter, which needs to contain the following parameters. Ifname: the network interface name (mandatory), interface name "wlan0"/"wwan0"/"usb0"/" ppp0"; wlan0 : WiFi network card wwan0 : 4G Cat4 NIC usb0 : 4G Cat1 NIC ppp0 : 2G NIC value is an additional parameter for the network query interface, the parameter can be : Boolean false / true, indicates whether to get route (device routing) information; The number 2 , used by wlan0 only, indicates getting the current AP connection information of the WiFi NIC; The string "WiFiList" is only used for wlan0, and is used to retrieve the information of the searched WiFi list;

Response Msg	Response description. msg contains the contents of the configuration and the status of the NIC. The contents of the configuration contains : General configuration information attributes (enable, name, mode, metric). Enable: whether to configure the NIC on or off; Name: optional, configured as the name of the NIC; Mode: not used; Metric: optional, for the network priority. Configuration information attributes specific of each NIC. wlan0(WiFi NIC): C_NetworkType/C_AuthMode/C_EncrypType/C_SSID/C_WPAPSK/C_HW Mode,The above parameters are explained in <2.3.2 DEFED Smart Station WiFi configuration commands>; wwan0(4G Cat4): C_apnname/C_username/C_password/C_pref, The above parameters are explained in <2.3.3 DEFED Smart Station SIM card configuration commands>; usb0(4G Cat1): No specific configuration parameters; ppp0(2G): C_operator,The above parameters are explained in <2.3.3 DEFED Smart Station SIM card configuration commands>; The status of the NIC is shown in the _info parameter, which contains : Ip: the IP address of the NIC itself，the basic acquisition of an IP or the presence of an IP indicates that the NIC is booting normally. Gw：IP address of the gateway of the NIC. Ifstr：the response content of the terminal command ifconfig network card name. Route：the response content of the terminal command route -n, which exists only if value=true.

2.3.5 DEFED Smart Station Network Function Commands (send SMS by SIM card)

obj	config
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"cfg": “net”,
	"cmd": “netcmd”,
	"cmdargs": {
	"ifname": “wwan0”,
	"netcmd": “SMS”,
	"tel": “XXXXXXXXXXX”,
	"msg": “tel msg”,
	},
	},
	"obj": "config”,
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	"agtid": “mga",
	"msg": "success”
	}

Args	The current SET->config command supports the following commands. cfg:net -> the cmd and cmdargs need to be set when sending the network function commands of the Smart Station. The cmd needs to be set to the character "netcmd", which means it is a network function command. The cmdargs is a function command parameter which needs to contain the following parameters. Ifname：the name of network interface (required),which supports interfaces "wwan0" and "usb0"; netcmd ：the name of the function command (required), currently only supports "SMS"; Tel： the phone number of SMS recipient (mandatory for SMS command); Msg：the content of SMS (mandatory for SMS command);
Response Msg	Response description. When msg is success, SMS successfully.

2.3.6 DEFED Smart Station WiFi Network Function Command

obj	config
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"cfg": “net”,
	"cmd": “netcmd”,
	"cmdargs": {
	"ifname": “wlan0”,
	"netcmd": “CancelTemp”,
	},
	},
	"obj": "config”,
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	"agtid": “mga",
	"msg": "success”
	}

Args	The current SET ->config command supports the following commands: cfg:net —> sending network function commands to the smart station, the cmd and cmdargs need to be set; cmd parameter needs to be set to the character "netcmd", indicating a network function command; cmdargs parameter is a functional command parameter that needs to include the following parameters: Ifname is the network interface name (required), which supports the interface "wlan0"; Netcmd is a functional command (required), currently only supports "CancelTemp" (to cancel temporary hotspots);
Response Msg	返回数据说明： msg 为 success 时，则发送成功。

2.3.7 Command to control DEFED LED

obj	config
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"cfg": “led”,
	"on": true,
	"val": 2249772032,
	},
	"obj": "config”,
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	"agtid": “mga",
	"msg": "success”
	}

Args	The current SET ->config command supports the following commands: Cfg: led ->To control the LED of DEFED, you need to set the parameters on and val; The on parameter is the LED on or off (required), a Boolean value; Val parameter is the change parameter of LED (optional), digital; Note: The LED effect modified by the DEFED LED command has the lowest priority and will be preempted by the LED effect when the state machine changes; Note: val includes many situations, as follows: 1. When val is less than 0x80000000, RGB monochrome is used: Bit 0-7 represents B in RGB; Bit 8-15 represents G in RGB; Bit 16-23 represents R in RGB; 2. When val is greater than 0x80000000, it means dynamic color change mode is used. Refer to Table 2-3-6: Bit 0-7 represents the step of dynamic change, and the default is 0; Bit 8-15 indicates the saturation of RGB in dynamic change; Bit 16-23 represents the change rate in dynamic change; Bit 24-31 indicates the dynamic change mode;
Response Msg	返回数据说明： msg 为 success 时，则发送成功。

Effect	mode	rate	RGB saturation	sample
红-绿-蓝呼吸(高原)	0x80	0~0xFF	0~0xFF	Mode 0x80,rate 0x18,saturation 0xcc Combined to 0x8018cc00 (2149108736)
红-橘黄-黄-绿-天蓝-蓝-紫呼吸(披萨)	0x81	0~0xFF	0~0xFF	模式 0x81,变化速率 0x18,饱和度 0xcc Combined to 0x8118cc00 (2165885952)
绿色呼吸(青草)	0x82	0~0xFF	0~0xFF	模式 0x82,变化速率 0x18,饱和度 0xcc Combined to 0x8218cc00 (2182663168)
天蓝呼吸(海浪)	0x83	0~0xFF	0~0xFF	模式 0x83,变化速率 0x18,饱和度 0xcc Combined to 0x8318cc00 (2199440384)

蓝色呼吸(深蓝山脉)	0x84	0~0xFF	0~0xFF	模式 0x84,变化速率 0x18,饱和度 0xcc Combined to 0x8418cc00 (2216217600)
紫色呼吸(紫色妖姬)	0x85	0~0xFF	0~0xFF	模式 0x85,变化速率 0x18,饱和度 0xcc Combined to 0x8518cc00 (2232994816)
红色呼吸(树莓)	0x86	0~0xFF	0~0xFF	模式 0x86,变化速率 0x18,饱和度 0xcc Combined to 0x8618cc00 (2249772032)
橘黄呼吸(橙光)	0x87	0~0xFF	0~0xFF	模式 0x87,变化速率 0x18,饱和度 0xcc Combined to 0x8718cc00 (2266549248)
黄色呼吸(秋实)	0x88	0~0xFF	0~0xFF	模式 0x88,变化速率 0x18,饱和度 0xcc Combined to 0x8818cc00 (2283326464)
白色呼吸(冰淇淋)	0x89	0~0xFF	0~0xFF	模式 0x89,变化速率 0x18,饱和度 0xcc Combined to 0x8918cc00 (2300103680)
红-黄渐变(果汁)	0x8a	0~0xFF	0~0xFF	模式 0x8a,变化速率 0x18,饱和度 0xcc Combined to 0x8a18cc00 (2316880896)
红-绿渐变(温暖小屋)	0x8b	0~0xFF	0~0xFF	模式 0x8b,变化速率 0x18,饱和度 0xcc Combined to 0x8b18cc00 (2333658112)
黄-白渐变(魔力红)	0x93	0~0xFF	0~0xFF	模式 0x93,变化速率 0x18,饱和度 0xcc Combined to 0x9318cc00 (2467875840)
绿-白渐变(光斑)	0x95	0~0xFF	0~0xFF	模式 0x95,变化速率 0x18,饱和度 0xcc Combined to 0x9518cc00 (2501430272)
天蓝-白渐变(晨曦)	0x96	0~0xFF	0~0xFF	模式 0x96,变化速率 0x18,饱和度 0xcc Combined to 0x9618cc00 (2518207488)
蓝-白渐变(蓝粉知己)	0x97	0~0xFF	0~0xFF	模式 0x97,变化速率 0x18,饱和度 0xcc Combined to 0x9718cc00 (2534984704)
紫-白渐变(木槿)	0x98	0~0xFF	0~0xFF	模式 0x98,变化速率 0x18,饱和度 0xcc Combined to 0x9818cc00 (2551761920)
红-绿-蓝闪变(缤纷时代)	0x99	0~0xFF	0~0xFF	模式 0x99,变化速率 0x18,饱和度 0xcc Combined to 0x9918cc00 (2568539136)
红-绿-蓝快闪(天上人间)	0xa3	0~0xFF	0~0xFF	模式 0xa3,变化速率 0x18,饱和度 0xcc Combined to 0xa318cc00 (2736311296)
蓝快闪(魅蓝)	0xa7	0~0xFF	0~0xFF	模式 0xa7,变化速率 0x18,饱和度 0xcc Combined to 0xa718cc00 (2803420160)

红快闪(炫红)	0xa9	0~0xFF	0~0xFF	模式 0xa9,变化速率 0x18,饱和度 0xcc Combined to 0xa918cc00 (2836974592)

2.3.8 DEFED Smart Station Internet Network Function Command

obj	config
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"cfg": “net”,
	"cmd": “setifn”,
	"cmdargs": {
	"ifname": “eth0”,
	"enable": true,
	"metric": 80,
	"C_IsRouter": true,
	"C_RouteIfName": “wwan0”,
	"C_MTU": 1500,
	},
	},
	"obj": "config”,
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	"agtid": "mga",
	"msg": “success",
	}

Args	The current set->config command supports the following commands: Cfg:net ->when configuring the WiFi network of the smart station, you need to set the parameters CMD and cmdargs; CMD parameter needs to be set to the character "setifn", indicating network configuration command; cmdargs parameter is a function command parameter and needs to include the following parameters: Ifname： network interface name (required), Ethernet configuration command interface "wlan0"; enable： WiFi on or off (required), Boolean value; Metric： WiFi network priority (optional), and the value can be chosen from 0，50，80，100 C_ Isrouter： Ethernet working mode, Boolean value, true is the routing mode, false is the device mode; C_ Routeifname： the name of the network card that specifies the Ethernet routing data interworking (used only when C_IsRouter=true), and the value is "wlan0"/"wwan0"/"usb0". If you want to cancel the data interworking, you need to set it to an empty string "; C_ MTU is the value of the MTU to be set by the network card. The general range is [768-1500]; Note: the default state of the Ethernet card is the highest priority operation in the device mode, and mtu=1500. If you need to modify the priority, MTU or Ethernet working mode, use this interface
Response Msg	返回数据说明： msg 为 success 时，则仅表示配置发送成功。

2.3.9 MTU configuration commands for network cards in the defed smart station

obj	config
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"cfg": “net”,
	"cmd": “setifn”,
	"cmdargs": {
	"ifname": “wwan0”,
	"enable": true,
	"C_MTU": 1400,
	},
	},
	"obj": "config”,
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	"agtid": "mga",
	"msg": “success",
	}

Args	The current SET ->config command supports the following commands: cfg:net-> you need to set the cmd and amdargs when configuring the WiFi network of the smart station parameters cmd: needs to be set to the character "setifn", indicating a network configuration command; cmdargs parameter is a functional command parameter that needs to include the following parameters: Ifname: the network interface name (required), and MTU configuration supports network card eth0/ wlan0/ wwan0; Enanle：ndicates whether the turn on or off network card (required), a Boolean value. To modify the MTU, it must be true; C_ MTU is the value of the MTU to be set by the network card, generally ranging from 768 to 1500; Note 1: If MTU needs to be changed to the default value of the network card, C_MTU needs to be set to 0; Note 2: C_ MTU attribute can be used separately and can be added to the cmdargs of each network card configuration interface for distribution together. Each interface reference:

	2.3.2 DEFED Smart Station WiFi Network Configuration Command; 2.3.3 DEFED Smart Station SIM Card Network Configuration Command; 2.3.8 DEFED Smart Station Ethernet Network Configuration Command;
Response Msg	返回数据说明： msg 为 success 时，则表示配置发送成功

2.4 smart station paring command

2.4.1 Yale doorlock module pairing

obj	dopair
Pkg_type	SET

Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"period": 60,
	"chn": 1,
	"bps": 115,
	"devtype": “SL_LK_YL”,
	"isopen": 1,
	},
	"obj": "dopair",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	"msg": {
	}
	}

Args	“period”: The timeout period for CoSS protocol sub-device pairing, 60s for Yale door lock module pairing. “chn”： Yale door lock module pairing，chn=1； “bps”： Yale door lock module pairing，bps=115； “Devtype”：Yale door lock module pairing，devtype is SL_LK_YL； “isopen”：Yale door lock module pairing. If pairing remoter unlock version Yale, set isopen=1; If pairing Yale without remoter unlock function, set isopen=0;

2.4.2 Add Sub Device---optarg parameter

obj	dopair
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"period": 60,
	“optarg": {
	"cls": "SL_SC_BE",
	“exarg”: {
	"humidity_display": 3,
	"temperature_display": 2
	}
	}
	},
	"obj": "dopair",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	"msg": {
	}
	}

Args	Period: Pairing CoSS protocol sub devices timeout duration; Optarg: The parameter is an additional parameter for adding devices. Generally, it can work well without using this parameter. However, for some devices, in order to achieve flexible customization, this parameter can be used. For detailed instructions, please refer to 2.4.2.1 Optarg parameter description The optarg parameter data format is a JSON object and does not participate in signing.

2.4.2.1 optarg Parameter

optarg parameter is an additional parameter for adding a device. Generally, it can work well without using this parameter, but for some devices, in order to achieve flexible customization, this parameter can be used.
This parameter is closely related to the device type. Different devices have different parameters. If the device has no additional parameters, the value of this parameter will be ignored. The currently available additional parameters are as follows:

CUBE environmental sensor

```
optarg = {
  "cls":"SL_SC_BE",
  "exarg":{
    "humidity_display":1/2/3,
    "temperature_display":1/2/3
  }
}
```

humidity_display: attribute is used to determine the content displayed on the LCD screen of the CUBE environmental sensor, which can be displayed as humidity, illumination, humidity and illumination, corresponding to values 1, 2, and 3 respectively.
temperature_display attribute is used to determine the selection of the temperature display category on the LCD screen of the multi-function (CUBE) environmental sensor. You can select Celsius, Fahrenheit, Celsius and Fahrenheit, corresponding to values 1, 2, and 3 respectively.

CUBE motion sensor

```
optarg = {
  "cls":"SL_SC_BM",
  "exarg":{
    "warning_duration":[6-814]
  }
}
```

warning_duration: attribute is used to determine the alarm duration (unit: seconds) after the movement is detected. The default is seconds, and the optional range is 6-814 seconds,with steps increasing by 4, 6, 10, 14... 814.

Yale door lock module

```
optarg = {
  "cls":"SL_LK_YL",
  "exarg":{
    "enable_remote_unlock":1/0
  }
}
```

enable_remote_unlock: attribute is used to determine whether the Yale door lock module supports remote door opening. You can choose to support, not support, corresponding to the values 1, 0 respectively.

Stellar Switch/Starry Switch/Polar Switch/Switch Accessory
When adding Stellar Switch/Starry Switch/Polar Switch/Switch Accessory, the specification must be specified, otherwise it cannot be added correctly

At the same time, the Stellar Switch/Starry Switch/Polar Switch can also set the working mode, respectively is: speed priority, power priority. Its configuration is as follows:

```
optarg = {
  "cls":"SL_MC_ND3_V2",
  "exarg":{
    "mode_selection":"speed"
  }
}
```

cls indicates that its Polar Switch (L 3 way)
The cls of current Stellar Switch/Starry Switch/Polar Switch/Switch Accessory are defined as follows:
●SL_SW_ND1_V1/SL_SW_ND2_V1/SL_SW_ND3_V1 Stellar Switch /Starry Switch (1 way/2way/3way)
●SL_MC_ND1_V1/SL_MC_ND2_V1/SL_MC_ND3_V1 Stellar Switch /Starry Switch Accessory (1 way/2way/3way)
●SL_SW_ND1_V2/SL_SW_ND2_V2/SL_SW_ND3_V2 Polar Switch (1 way/2way/3way)
●SL_MC_ND1_V2/SL_MC_ND2_V2/SL_MC_ND3_V2 Polar Switch Accessory (1 way/2way/3way)

mode_selection: attribute indicates the working mode. You can select "speed" and "power", which correspond to speed priority and power priority respectively. The default mode is speed priority.

Specially specified device
Some devices must specify the type when pairing the code, so that the API interface can perform better adding operations. Therefore, when adding these devices, please specify the type of device specification to be added in the optarg attribute.

The current mandatory device specifications are as follows:
●PSM: PSM series
●SL_P_IR: SPOT(MINI)

Let's take Starry Switch as an example, the parameters are as follows:

optarg = {
  "cls":"SL_SW_ND1"
}

Note:
The parameter data format is the serialized string of the JSON object, and it must participate in the method signature.

2.5 Sub-device OTA Management Interface

The OTA file management of the smart station supports downloading and saving a firmware with a maximum size of 1MByte. When the firmware is greater than 384KByte, other firmware files will be automatically cleared to ensure space.

2.5.1 download OTA file by url

obj	otamanage
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"cmd": “addfileurl”,
	“url”: “http://xxx/xx/FL01_03061000_0000ffff.ota”,
	}
	"obj": “otamanage",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	"msg": “success",
	}

Args	Url: the htth or httpsaddress of the target firmware address,necessary Key:optional,If no key is provided, the key is the file name in the download address provided by the URL; Note: Please use the key parameter with caution. The name of the key parameter must be correct, otherwise the upgrade may fail. If not necessary, there is no need to set the key parameter. Keeping Defalut is ok.
Response Msg	Download successfully when Msg is succes

2.5.2 Query the list of ota files under the current smart station

obj	otamanage
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"cmd": “queryfile”,
	}
	"obj": “otamanage",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	“msg": [
	{
	“key”: “FL01_ZG10370104_00000002.ota",
	“size”: 114614,
	}
	],
	}

Response Msg	If the query command is successful, a list of OTA file arrays will be returned, with array elements including file key and size Key indicates the firmware file name that the device needs to be upgraded; Size indicates the size of the OTA firmware file;

2.5.3 Remove ota files under the current smart station

obj	otamanage
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"cmd": “rmfile”,
	“keys”: “FL01_03061000_0000ffff.ota”,
	}
	"obj": “otamanage",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	"msg": “success",
	}

Args	Keys are necessary whose value can be set： keys=true, remove all the ota files under the smart station 若 keys=["key1", “key2"], batch delete the OTA file in the array keys="key1",remove the OTA file which is specified
Response Msg	Remove successfully when msg is success

2.5.4 Query the upgradable devices corresponding to ota files

obj	otamanage
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"cmd": “getavailableeps”,
	"keys": [“FL01_ZG10370104_00000002.ota”, . . .]
	}
	"obj": “otamanage",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	“msg": {
	“FL01_03580000_00000605.ota”: [
	{
	“me”: “a0db”,
	“ver”: “0.1.6.4”,
	“devtype”: “SL_LI_WW”,
	“fullCls”: “SL_LI_WW_V3”,
	“name”: “Dimming LED Driver”,
	“otaVer”: “6.5”,
	“supportOta”: true,
	“needOta”:: true,
	“lsid”: “A1gAACfu0f7_hDwA_____w",
	“rfic”: 3,
	}
	],
	. . .
	},
	}

Args	Keys is optional When present, its value is a string array, indicating the list of OTA files to be queried; When it does not exist, it means checking all OTA files under the query smart station; Note:only for coss device;
Response Msg	Instruction of returned parameter： If the query command is successful, the list of supported devices in the OTA file will be returned. The parameter description is as follows: Me: is the device number under the smart station, Ver: is the complete version number of the sub device, Devtype: is the sub device type, FullCls: is the complete type string of the device, Name: is the sub device name, OtaVer: is the version number of this OTA file, EpVer: is the current version number of the sub device, SupportOta: Whether the sub device supports OTA upgrade, NeedOta: Whether the sub device needs to upgrade OTA, Lsid: encoding the LSID of the sub device, Rfic: is the sub device CoSS RF type,

2.5.5 Smart station creat new OTA task

obj	otamanage
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"cmd": “addtask”,
	"me": “a0db”,
	“key”: “FL01_03061000_0000ffff.ota”,
	}
	"obj": “otamanage",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	"msg": “success",
	}

Args	Me: is the device number of the smart station to be upgraded, required; Key: Indicates the upgraded OTA file, required; Note: The type of sub devices must be consistent with the OTA file, otherwise they cannot be upgraded. You can call<2.5.4 to query the set of upgradable devices corresponding to the OTA file>to query the list of sub devices supported by the OTA file
Response Msg	Download successfully when msg is success

2.5.6 Smart station creat new OTA task

obj	otamanage
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"cmd": “querytask”,
	}
	"obj": “otamanage",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	“msg": {
	“8cc7”: {
	“id”: “8cc7”,
	“cur”: 14158,
	“size”: 165566,
	“file”: “FL01_ZG10370104_00000002.ota”,
	“tover”: “\u0000\u0000\u0000\u0002”,
	“sts”: 1577443530,
	“ts”: 1577443530,
	},
	. . .
	},
	}

Response Msg	Return Data Description: If the query command is successful, a list of OTA tasks will be returned. The list parameters are described as follows: ID: Indicates the ID of the sub device, whose value is the me attribute of the device, and the ID is the key of the rmtask interface parameter; Cur: Indicates the progress of the current OTA upgrade task. If cur is equal to size, it indicates that the upgrade has been completed; Size: The total size of the current OTA file; File: The file name of the current OTA file; tover: The firmware version number of the current OTA file; Sts: The start time of the current upgrade task, in UTC time, in seconds; Ts: The latest feedback time for the current upgrade task, in UTC time, in seconds;

2.5.7 Delete completed ota tasks under the current smart station

obj	otamanage
Pkg_type	SET
Direction	Device->Station

Request	{
	"id": 2,
	"args": {
	"cmd": “rmtask”,
	“ids”: “8cc7”,
	}
	"obj": “otamanage",
	"sys": {
	"ver": 1,
	"ts": 1571976095,
	"sign": "dbe2076ba2a67fe886aa5098d165ac7a",
	"model": "OD_XXX_XXX"
	}
	}

Response	{
	"code": 0,
	"id": 2,
	“agtid": "mga",
	"msg": “success",
	}

Args	iDs is a required parameter, and its value can be: 1. If ids=true, it indicates the deletion of all completed OTA tasks under the smart station 2. If ids=["id1", "id2"], it means to batch delete the completed OTA task set indicated by the array 3. If ids="id1", it means deleting specific completed OTA tasks Note: Only OTA tasks that have already been upgraded can be deleted
Response Msg	**返回数据说明：** code **为** 0 **时，则删除成功**