SL-BUS APIs

Users Manual

VADACTRO Technologies India Pvt. Ltd.

Version 1.1.7

May 12, 2023

Abstract

This document is created to address the developer guidelines with the process to understand, test/validate and use the SL-BUS Technology APIs in their respective products to enable smart control, configuration and status reporting needs for SL-BUS Technology based products.

•

1 SL-BUS Java Libs

All the configurations are related to Ubuntu 20.4 Machine

1.1 How To Build Jar TestApp?

Procedure to build Jar test App

1.1.1 Prerequisite

Check java version by

```
java -version
```

If jdk8 is not install, then install by executing following command

```
sudo apt-get install openjdk-8-jdk
```

If you have different version of JAVA, then execute following command (set correct path if it is different)

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64/
```

verify ant installation by and install Apache ant if not installed.

```
ant -version
```

1.1.2 How to install Apache ant

```
sudo apt update
sudo apt install ant
```

verify ant installation by

```
ant -version
```

1.1.3 How To Build JAR Test App

Go to directory test_app
execute build scrip to create test app

```
sh ./make_slbus_test_app.sh
```

App will be in the directory ../bin with the following format.
i.e. lib-slbus-comm-<VERSION>-testapp.jar

1.1.4 How To Run JAR Test App

```
java -jar ../bin/lib-slbus-comm-\<VERSION\>-testapp.jar
```

1.2 How To Build Android test App

Procedure to build Android test App

1.2.1 Prerequisite(JDK):

Check for JAVA version by

```
java -version
```

If jdk8 is not install, then install by executing following command

```
sudo apt-get install openjdk-8-jdk
```

If you have different version of JAVA, then execute following command (set correct path if it is different)

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64/
```

1.2.2 Install Android SDK

Go to home directory and execute below commands

```
cd ~
mkdir android
mkdir android/sdk
cd android/sdk
wget https://dl.google.com/android/repository/commandlinetools-linux-6200805_latest.zip
unzip commandlinetools-linux-6200805_latest.zip
rm -f commandlinetools-linux-6200805_latest.zip
cd ..
export ANDROID_HOME=$PWD/sdk
yes | sdk/tools/bin/sdkmanager --sdk_root=${ANDROID_HOME} --licenses
sdk/tools/bin/sdkmanager --sdk_root=${ANDROID_HOME} "tools"
sdk/tools/bin/sdkmanager "platform-tools" "platforms;android-29"
```

Ignore warning if above command gives any and continue installation

```
sdk/tools/bin/sdkmanager "build-tools;29.0.3"
```

1.2.3 Install Gradle

Go to home directory and execute below commands

```
cd ~
mkdir android/gradle
cd android/gradle
wget https://services.gradle.org/distributions/gradle-5.6.4-all.zip
unzip gradle-5.6.4-all.zip
rm -f gradle-5.6.4-all.zip
```

1.2.4 How To Build Android App

Go to directory android_app
Execute build script to create Android test app

```
sh ./make_slbus_android_app.sh
```

App will be in the directory ../bin with the following format, for ex: lib-slbus-comm–androidapp.apk

1.2.5 How To Build signed Android App

Go to directory android created in home directory

```
cd ~/android/
mkdir storekey
```

Copy .keystore file here and rename it as slbus.dat
Open file java/android_app/make_slbus_android_app.sh and modify and uncomment the line. Make sure you replace placeholder for Keyalias, KeyPassword and nstorePassword as per your own keystore file.

```
echo "keyAlias=abc\nkeyPassword=<xyz>\nstoreFile=../slbus.dat\nstorePassword=<xyz> " >
keystore.properties
```

2 SL-BUS Swift Libs

Make sure Xcode 13 should be your default xcode

2.1 How to Building macOS test App

Procedure to build macOS test App

2.1.1 Build using xcode (Recommended).

Go to api/swift folder

```
cd test_app
```

Create a link to library in SampleLibraryTestApp folder using below command

```
ln -sf ../../libs/lib-slbus-comm-$VERSION.xcframework SampleLibraryTestApp/lib-slbus-comm.xc
```

Open the SampleApp-SL-BUS-COMM-IOS.xcodeproj in xcode then build and run the application.

2.1.2 Build using command line (This might affect your other xcode projects).

Open xcode 13
Open preferences from Xcode menu and go to Locations tab
Change Derived Data to relative if not set and mention DerivedData as value.
Open slbus_apis/swift/test_app folder in terminal
Execute below command to create application

```
sh ./make_slbus_test_app.sh
```

Application will be in bin directory as lib-slbus-comm-$VERSION-mactestapp.app

2.2 How to Build iOS app :-

Procedure to build iOS App

2.2.1 Build iOS App using xcode (Recommended).

Go to api/swift folder then

```
cd ios_app
```

Create a link to library using below command

```
ln -sf ../../libs/lib-slbus-comm-$VERSION.xcframework SL-BUS-APP/lib-slbus-comm.xcframework
```

Open the SL-BUS-APP.xcodeproj in xcode then build and run the application.

2.2.2 Build iOS App using command line (This might affect your other xcode projects).

Open xcode 13
Open preferences from Xcode menu and go to Locations tab
Change Derived Data to relative if not set and mention DerivedData as value
Open slbus_apis/swift/ios_app folder in terminal
Open make_slbus_ios.sh modify YOUR_TEAM_NAME with your team name in below command, please make sure you replace YOUR_TEAM_NAME placeholder with respective input.

```
xcodebuild -project SL-BUS-APP.xcodeproj -scheme SL-BUS-APP -archivePath
SL-BUS-APP.xcarchive archive DEVELOPMENT_TEAM="\<YOUR_TEAM_NAME\>"
```

Execute below command to create application

```
sh ./make_slbus_ios_app.sh
```

Application will be in bin directory as lib-slbus-comm-$VERSION-iosapp.ipa

3 HTTP API

3.1 HTTP API: Login

Item	Description
Command	Login and Get the list of SL-BUS Devices on your local wireless network
HTTP API	`https://oath2.vadactro.org.in/slbus/api`
POST Data	`{"cmd":{"login":{"user":"\<VDCloud_user_email\>", "password":"\<VDCloud_password\>"}}}`
Response	This command will return the complete JSON object of a typical installation for a customer that includes, list of online devices, their ipaddress and configuration (groups, scenes, nodes details)
Example curl command	`curl https://oath2.vadactro.org.in/slbus/api -H "Content-Type: application/json" -d '{"cmd":{"login": {"user":"\<VDCloud_user_email\>", "password":"\<VDCloud_password\>"}}}' -c cookie.txt -b cookie.txt -k`
Response received for the command from server	`{"login":{"status":"pass", "msg":"success", "data": {"dnsListData": [{"uuid":"c861a4181b934925accf2320f3b2", "type":"SL-WSW-2CH-TR-V5", "dname":"SL-WSW-2CH-V5-TR-F3B2", "ipAddress":"2502a8c0", "rmacid":"a4:2b:b0:dd:fa:0c", "bus":0, "groupList":[{}]}, {"uuid":"d9664c4107616e2cb072bf1c88c3", "type":"SL-WAIRC-LGVRF9-V12", "dname":"SL-WAIRC-V7-LGVRF9-88C3", "ipAddress":"2d02a8c0", "rmacid":"a4:2b:b0:dd:fa:0c", "bus":0, "groupList":[{},{}, {}]}]}}}`
Explanation	

3.2 HTTP API: GrantAccess

Item	Description
Command	To Grant the device access license to a SL-BUS Device
HTTP API	`https://oath2.vadactro.org.in/slbus/api`
POST Data	`{"cmd":{"grantAccess": {"user":"\<dlm_user_email\>", "password":"\<dlm_user_VDCloud_password\>", "deviceList":[{"uuid":"c861a4181b934925accf2320f3b2"}, {"uuid":"d9664c4107616e2cb072bf1c88c3"}], "assignee":"\<assignee_details\>"}}}`
Response	This command will return licensing information for the requested devices in the JSON format

Item	Description
Example curl command	`curl https://oath2.vadactro.org.in/slbus/api -H "Content-Type: application/json" -d '{"cmd": {"grantAccess": {"user":"\<dlm_user_email\>", "password":"\<dlm_user_VDCloud_password\>", "deviceList":[{"uuid":"c861a4181b934925accf2320f3b2"}, {"uuid":"d9664c4107616e2cb072bf1c88c3"}], "assignee":"\<assignee_details\>"}}}' -c cookie.txt -b cookie.txt -k`
Response received for the command from server	`{"grantAccess":{"deviceList": [{"uuid":"c861a4181b934925accf2320f3b2", "license":"newly_assigned"}, {"uuid":"d9664c4107616e2cb072bf1c88c3", "license":already_assigned"], "access_token":"\<dlm_access_token\>", "licenses_granted":1, "licenses_consumed" 265, "licenses_purchased": 2000, "status":"pass"}}`
Explanation	Application gets entire licensing information, the access token (DLM Access Token) received in the call to be used by for device access for command API

3.3 HTTP API: Get User Access Token

Item	Description
Command	Get User Access Token from VDCloud for Cloud base Operation
HTTP API	`https://oath2.vadactro.org.in/slbus/api`
POST Data	`{"cmd":{"getAccessToken": {"user": "\<VDCloud_user_email\>", "password": "\<user_VDCloud_password\>"}}}`
Response	This command will return licensing information for the requested devices in the JSON format
Example curl command	`curl https://oath2.vadactro.org.in/slbus/api -H "Content-Type: application/json" -d '{"cmd": {"getAccessToken": {"user":"\<VDCloud_user_email\>", "password":"\<user_VDCloud_password\>"}}}' -c cookie.txt -b cookie.txt -k`
Response received for the command from server	`{"getAccessToken": {"access_token":"\<user_access_token\>", "token_type":"smartphone", "expires_in":1209600, "refresh_token":"\<refresh_token\>", "status":"pass"}}`
Explanation	This access token is essentail for cloud base operation along with DLM Access Token

3.4 HTTP API: SignUp

Item	Description
Command	Add user to the VDCloud (User Registration)
HTTP API	`https://oath2.vadactro.org.in/slbus/api`
POST Data	`{"cmd":{"signUp":{"user":"\<SI_user_email\>", "password":"\<SI_user_VDCloud_password\>", "cname":"\<Customer Full Name\>", "cemail":"\<Customer Email ID\>", "cmobile":"\<Customer Mobile no\>", "caddress":"\<Optional: Postal Address of Customer\>", "ccity":"\<Optional: City for Postal Address\>", "cpin":"\<Optional: Pin code for Postal Address\>"}}}`
Resposne	This command will return the JSON object with user status.
Example curl command	`curl https://oath2.vadactro.org.in/slbus/api -H "Content-Type: application/json" -d '{"cmd":{"signUp": {"user":"\<SI_user_email\>", "password":"\<SI_user_VDCloud_password\>", "cname":"\<Customer Full Name\>", "cemail":"\<Customer Email ID\>", "cmobile":"\<Customer Mobile no\>", "caddress":"\<Optional: Postal Address of Customer\>", "ccity":"\<Optional: City for Postal Address\>", "cpin":"\<Optional: Pin code for Postal Address\>"}}}' -c cookie.txt -b cookie.txt -k`
Response received for the command from server	`{"signUp":{"status":"pass","data":"Created User Account for \<Customer Email ID\>, please check email and set password"}}`
Explanation	This process will add user to VDCloud.
Post Process	User has to validate VDCloud account with the email received and need to set password to complete registration.

3.5 HTTP API: Assign UUID to User

Item	Description
Command	Assign UUID to registered User
HTTP API	`https://oath2.vadactro.org.in/slbus/api`
POST Data	`{"cmd":{"manageUuid":{"user":"\<VDCLoud_user_email\>", "password":"\<VDCLoud_user_pass\>", "uuids":[{"uuid":"\<product uuid0\>", "action":"add"}, {"uuid":"\<product uuidn\>", "action":"add"}]}}}`
Reposne	This command will return the JSON object with UUID registration status with user.

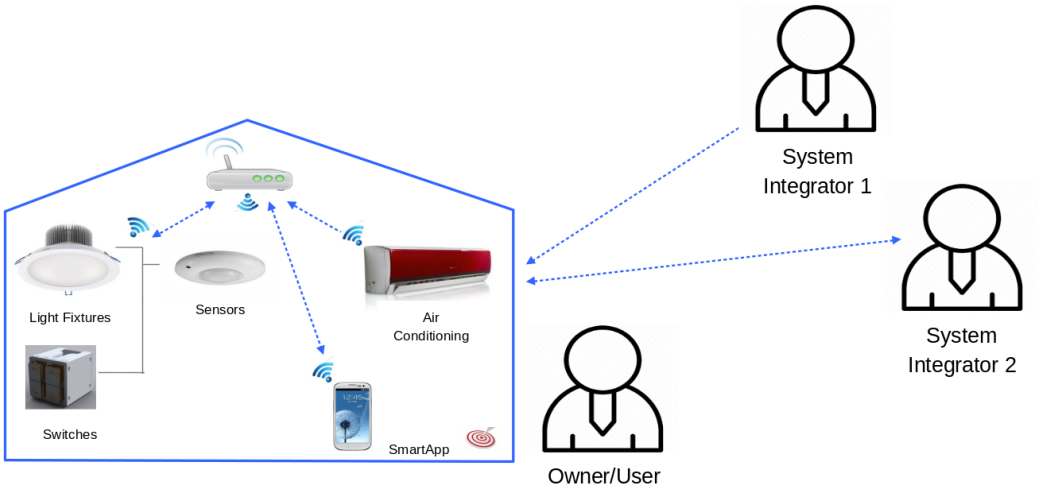
Item	Description
Example curl command	`curl https://oath2.vadactro.org.in/slbus/api -H "Content-Type: application/json" -d '{"cmd": {"manageUuid":{"user":"\<VDCLoud_user_email\>", "password":"\<VDCLoud_user_pass\>", "uuids":[{"uuid":"\<product uuid0\>", "action":"add"}, {"uuid":"\<product uuidn\>", "action":"add"}]}}}' -c cookie.txt -b cookie.txt -k`
Response received for the command from server	`{"manageUuid":{"msg":"uuids Assigned","data":{"added":["\<uuid0\>", "\<uuid1\>"],"not added":["\<uuid2\>","\<uuid3\>"]},"status":"pass"}`
Explanation	This process will add given UUIDs to registered user

3.6 HTTP API: Check UUID with User

Item	Description
Command	Check the given UUIDs assigned to user or not
HTTP API	`https://oath2.vadactro.org.in/slbus/api`
POST Data	`{"cmd":{"checkUuid":{"uuids":[{"\<uuid0\>":"?", "\<uuid1\>":"?"}]}}}`
Response	This command will return the JSON object with UUID registration status with user.
Example curl command	`curl https://oath2.vadactro.org.in/slbus/api -H "Content-Type: application/json" -d '{"cmd":{"checkUuid": {"uuids":[{"\<uuid0\>":"?", "\<uuid1\>":"?"}]}}}' -c cookie.txt -b cookie.txt -k`
Response received for the command from server	`{"checkUuid": {"msg":"Success", "data": {"uuids":[ {"\<uuid0\>":"\<CID\>"}, {"\<uuid1\>":"\<CID\>"}, {"\<uuid0\>":""}]}, "status":"pass"}`
Explanation	This is open command and will provide if the CID (customer ID) associated with device UUIDs provided

3.7 HTTP APIs: Assigning System Integrator to manage User Account.

System Integrator (SI) is a person that provides services to the end customer for installation and configuration of SL-BUS Technology based devices on need basis. Their could be multiple system integrators that can provide their services to the same end customer, where as end customer is the true owner of installation with full rights. If there is a need for any system integrator to manage any user account for installation and configuration then the need can be achieved with below sets of APIs in the same session managed by cookees.



3.7.1 HTTP API: SI Login

Item	Description
Command	Login to VDCloud with SI Credential.
HTTP API	`https://oath2.vadactro.org.in/slbus/api`
POST Data	`{"cmd":{"authSI":{"user":"<si_user_emailid>", "password":"<si_user_pswd>"}}}`
Response	This command will return the JSON object with SI details.
Example curl command	`curl https://oath2.vadactro.org.in/slbus/api -H "Content-Type: application/json" -d '{"cmd":{"authSI":{"user":"<si_user_emailid>", "password":"<si_user_pswd>"}}}' -c cookie.txt -b cookie.txt -k`
Response received for the command from server	`{"authSI": {"msg":"login successful", "data":null, "rights":"<system_rights>", "company":"<company_name>", "membership": {"validtill":"xxyy", "type":"<membership_type>", "startdate":"xxyy"}, "status":"pass"}}`
Explanation	To start registration process to register a user on VDCloud, SI login is the first step
Post Process	Hold the response cookie to be use with next request.

3.7.2 HTTP API: Request OTP

Item	Description
Command	Send a request to VDCloud to issue an OTP for end user account to whom he/she wish to provide service. As a result an OTP will be send to registered end users email ID.
HTTP API	`https://oath2.vadactro.org.in/slbus/api`
POST Data	`{"cmd":{"requestOtp":{"email":"<VDCloud EndUser Email ID>"}}}`
Response	This command will return the JSON object with OTP status.
Example curl command	`curl https://oath2.vadactro.org.in/slbus/api -H "Content-Type: application/json" -d '{"cmd":{"requestOtp":{"email":"<VDCloud EndUser Email ID>"}}}' -c cookie.txt -b cookie.txt -k`
Response received for the command from server	`{"requestOtp":{"status":"pass", "msg": "OTP send to <VDCloud EndUser Email ID>"}}`
Explanation	It will send OTP to the VDCloud User Email ID.
Post Process	Hold the response cookie to be use with next request. Check VDCloud User Email ID for OTP received.

3.7.3 HTTP API: Register With OTP

Item	Description
Command	SI need to contact end user and get the OTP received on his/her email id and put it in this API, as a result system integrator will be assigned as admin to manage user account using OTP based authorization. End user can launch VDCLoud support request in case if he/she with to remove admin rights provided to any SI. Thus end user can maintain security of his/own installations.
HTTP API	`https://oath2.vadactro.org.in/slbus/api`
POST Data	`{"cmd":{"registerWithOtp":{"email":"<VDCloud EndUser Email ID>", "OTP":"XXXYYY"}}}`
Response	This command will return the JSON object showing the status of the user registration.
Example curl command	`curl https://oath2.vadactro.org.in/slbus/api -H "Content-Type: application/json" -d '{"cmd":{"registerWithOtp":{"email":"<VDCloud EndUser Email ID>", "OTP":"XXXYYY"}}}' -c cookie.txt -b cookie.txt -k`
Response received for the command from server	`{"registerWithOtp":{"status":"pass","msg":"New Customer <Customer Full Name> added"}}`
Explanation	It will send information with status as `"pass"` or `"fail"`.

3.8 HTTP API: Command

Item	Description
Command	To issue a communication command to Cloud for control, configuration & status of SL-BUS Device
HTTP API	`https://oath2.vadactro.org.in/slbus/api`
POST Data	`{"access_token":"\<user_access_token\>", "dlm_access_token":"\<dlm_access_token\>, "uuid":"\<uuid of the device to be access\>, "cmd": \<JSON Object from SL-BUS API doc\>}`
Response	JSON Object, response from device.
Example curl command	`curl https://oath2.vadactro.org.in/slbus/api -H "Content-Type: application/json" -d '{"access_token":"\<user_access_token\>", "dlm_access_token":"\<dlm_access_token\>, "uuid":"\<uuid of the device to be access\>, "cmd":{"islands":[{"bus_id":0, "groups":[{"nodes":[{"address":63, "SL-SW": {"state":"?"}}]}]}]}' -c cookie.txt -b cookie.txt -k`
Response received for the command from device	`{"islands": [{"groups": [{"nodes":[{"address":63, "SL-SW": {"state":"off"}}]}], "bus_id":0}], "status":"pass"}` If status is “pass” means API gets executed succesfully. If status is “fails” means there is error. For details check the response.
Explanation	Commands are explained in detail in the API document

4 LIB_BUSCOM_API

4.1 LIB_BUSCOM_API: Login

Item	Description
COMMAND	Login and Get the list of SL-BUS Devices assigned to user HTTP API Will be provided by Library callback layer
JSON Input	`{"login":{"user":"\<VDCloud_user_email\>",` `"password":"\<VDCloud_password\>"}}`
Response	This command will return the complete JSON object of a typical installation for a customer that includes, list of online devices, their IP address and configuration (groups, scenes, nodes details)
Example Java command	`String loginResponse =` `busCom.CloudCommand(JSON_Input);`

Response	received for the command from server
	
	`{"login": {"status": true, "msg": "success", "data":`
	`{"dnsListData": [{"uuid":`
	`"c861a4181b934925accf2320f3b2", "type":`
	`"SL-WSW-2CH-TR-V5", "dname": "SL-WSW-2CH-V5-TR-F3B2",`
	`"ipAddress": "192.168.2.128", "rmacid":`
	`"a4:2b:b0:dd:fa:0c", "bus": 0, "groupList": [ {}]},`
	`{"uuid": "d9664c4107616e2cb072bf1c88c3", "type":`
	`"SL-WAIRC-LGVRF9-V12", "dname":`
	`"SL-WAIRC-V7-LGVRF9-88C3", "ipAddress":`
	`"192.168.2.129", "rmacid": "a4:2b:b0:dd:fa:0c",`
	`"bus":0, "groupList": [{},{},{}], "status":`
	`"pass"}]}}}`

Explanation	

4.2 LIB_BUSCOM_API: GrantAccess

Item	Description
Command	To Grant the device access license to a SL-BUS Device and get DLM Access Token
HTTP API	Will be provided by Library callback layer

JSON Input	`{"grantAccess":{"user":"\<dlm_username\>",`
	`"password":"\<dlm_password\>",`
	`"deviceList":[{"uuid":"c861a4181b934925accf2320f3b2"},`
	`{"uuid":"d9664c4107616e2cb072bf1c88c3"}],`
	`"assignee":"\<assignee_details\>"}}`

Response	This command will return licensing information for the requested devices in the JSON format
Example Java command	`String grantList = busCom.CloudCommand(JSON_Input)`

Item	Description
Response received for the command from server	`{"deviceList":[ {"uuid":"c861a4181b934925accf2320f3b2","license":"newly_assigned"}, {"uuid":"d9664c4107616e2cb072bf1c88c3","license":already_assigned" ], "access_token":"46200ac02f69d60c79f07092ffxxxxxx", "licenses_granted":1,"licenses_consumed" 265,"licenses_purchased": 2000, "status":"pass" }`
Explanation	Application gets entire licensing information, the access token received in the call to be used by for device access for command API
Post Process	Assign DLM Access Token using folllowing method to SL-BUS Library busCom.setCid(dlmAccessToken); If it is not set, both local and Cloud command execution will be "fail"

4.3 LIB_BUSCOM_API: Get User Access Token

Item	Description
Command	Get User Access Token from Cloud for Cloud Operation
HTTP API	Will be provided by Library callback layer
JSON Input	`{"getAccessToken":{"user":"\<user_email_id\>", "password":"\<user_password\>"}}`
Response	This command will return the JSON object with User Access Token
Example Java command	`String accessTokenResponse =busCom.CloudCommand(JSON_Input);`
Response received for the command from server	`{"access_token":"\<user_access_token\>", "status":"pass"}`
Explanation	This access token is essentail for cloud operation along with DLM Access Token
Post Process	Assign User Access Token using folllowing method to SL-BUS Library. `busCom.setUserAccessToken(userAccessToken);` If it is not set, cloud command execution will be "fail"

4.4 LIB_BUSCOM_API: SignUp

Item	Description
Command	Add user to the VDCloud (User Registration)
HTTP API	Will be provided by Library callback layer
JSON Input	`{"signUp": {"user": "<SI_user_email>", "password": "<SI_user_VDCloud_password>", "cname": "<Customer Full Name>", "cemail":"<Customer Email ID>", "cmobile": "<Customer Mobile no>", "caddress":"<Optional: Postal Address of Customer>", "ccity": "<Optional: City for Postal Address>", "cpin": "<Optional: Pin code for Postal Address>"}}`
Response	This command will return the JSON object with user status.
Example Java command	`String signUpResponse =busCom.CloudCommand(JSON_Input);`
Response received for the command from server	`{"signUp": {"status":"pass", "data": "Created User Account for <Customer Email ID>, please check email and set password"}}`
Explanation	This process will add user to VDCloud
Post Process	User has to validate VDCloud account with the email received and need to set password to complete registration.

4.5 LIB_BUSCOM_API: Assign UUID to User

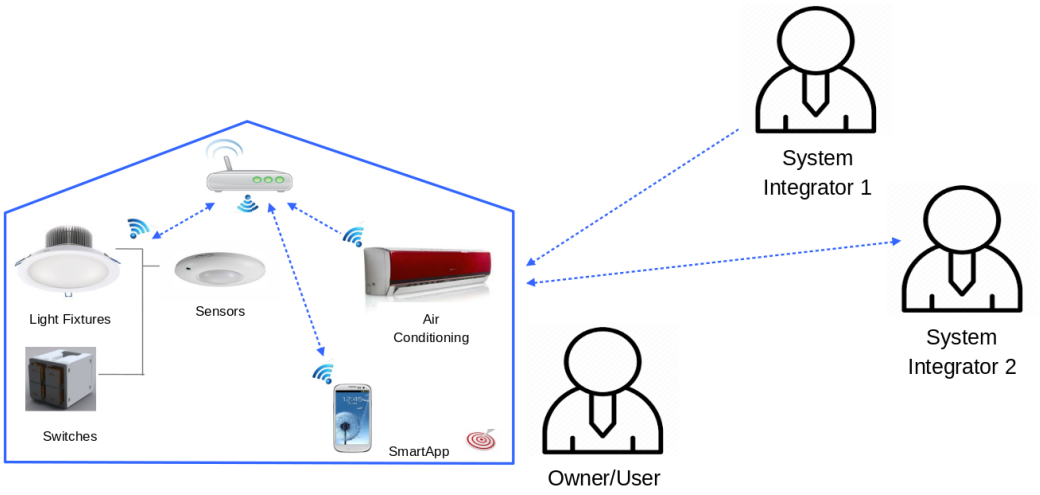
Item	Description
Command	Assign UUID to registered User
HTTP API	Will be provided by Library callback layer
JSON Input	`{"manageUuid": {"user": "<VDCLoud_user_email>", "password": "<VDCLoud_user_pass>", "uuids": [{"uuid": "<product uuid0>", "action": "add"}, {"uuid": "<product uuidn>", "action": "add"}]}}`
Response	This command will return the JSON object with UUID registration status with user.
Example Java command	`String manageUuidResponse =busCom.CloudCommand(JSON_Input);`
Response received for the command from server	`{"manageUuid": {"msg":"uuids Assigned", "data": {"added": ["<uuid0>", "<uuid1>"], "not added": ["<uuid2>", "<uuid3>"]}, "status": "pass"}`
Explanation	This process will add given UUIDs to registered user.
Post Process	

4.6 LIB_BUSCOM_API: Check UUID with User

Item	Description
Command	Check the given UUIDs assigned to user or not
HTTP API	Will be provided by Library callback layer
JSON Input	`{"checkUuid": {"uuids": [{"<uuid0>":"?", "<uuid1>":"?"}]}}`
Response	This command will return the JSON object with UUID with CID status.
Example Java command	`String checkUuidResponse =busCom.CloudCommand(JSON_Input);`
Response received for the command from server	`{"checkUuid": {"msg":"Success", "data": {"uuids": [{"<uuid0>":"<CID>"}, {"<uuid1>":"<CID>"}, {"<uuid0>":""}]}, "status":"pass"}`
Explanation	This is open command and will provide if the CID (customer ID) associated with device UUIDs provided.
Post Process	

4.7 LIB_BUSCOM_API: Assigning System Integrator to manage User Account.

System Integrator (SI) is a person that provides services to the end customer for installation and configuration of SL-BUS Technology based devices on need basis. Their could be multiple system integrators that can provide their services to the same end customer, where as end customer is the true owner of installation with full rights. If there is a need for any system integrator to manage any user account for installation and configuration then the need can be achieved with below sets of APIs in the same session managed by cookees.



4.7.1 LIB_BUSCOM_API: SI Login

Item	Description
Command	Login to VDCloud with SI Credential.
HTTP API	Will be provided by Library callback layer
JSON Input	`{"authSI":{"user":"<si_user_emailid>", "password":"<si_user_pswd>"}}`
Response	This command will return the JSON object with with SI details
Example Java command	`String authSIResponse =busCom.CloudCommand(JSON_Input);`
Response received for the command from server	`{"authSI": {"msg":"login successful", "data":null, "rights":"<system_rights>", "company":"<company_name>", "membership":{"validtill":"xxyy", "type":"<membership_type>", "startdate":"xxyy"}, "status":"pass"}}`
Explanation	To start registration process to register a user on VDCloud, SI login is the first step.
Post Process	Hold the response cookie to be use with next request.

4.7.2 LIB_BUSCOM_API: Request OTP

Item	Description
Command	Send a request to VDCloud to issue an OTP for end user account to whom he/she wish to provide service. As a result an OTP will be send to registered end users email ID.
HTTP API	Will be provided by Library callback layer
JSON Input	`{"requestOtp":{"email":"<VDCloud EndUser Email ID>"}}`
Response	This command will return the JSON object with OTP status.
Example Java command	`String requestOtpResponse =busCom.CloudCommand(JSON_Input);`
Response received for the command from server	`{"requestOtp":{"status":"pass", "msg": "OTP send to <VDCloud EndUser Email ID>"}}`
Explanation	It will send OTP to the VDCloud User Email ID.
Post Process	Hold the response cookie to be use with next request. Check VDCloud User Email ID for OTP received.

4.7.3 LIB_BUSCOM_API: Register With OTP

Item	Description
Command	SI need to contact end user and get the OTP received on his/her email id and put it in this API, as a result system integrator will be assigned as admin to manage user account using OTP based authorization. End user can launch VDCLoud support request in case if he/she with to remove admin rights provided to any SI. Thus end user can maintain security of his/own installations.
HTTP API	Will be provided by Library callback layer
JSON Input	`{"registerWithOtp":{"email":"<VDCloud EndUser Email ID>", "OTP":"XXXYYY"}}`
Response	This command will return the JSON object showing the status of the user registration.
Example Java command	`String registerWithOtpResponse =busCom.CloudCommand(JSON_Input);`
Response received for the command from server	`{"registerWithOtp": {"status":"pass", "msg": "New Customer <Customer Full Name> added"}}`
Explanation	It will send information with status as `"pass"` or `"fail"`.

4.8 LIB_BUSCOM_API: Local Command

Item	Description
Command	To issue a communication command for SL-BUS Java Library for control, configuration & status to local network
HTTP API	Will be provided by Library callback layer
JSON Input	JSON Object, input for device control (explained in API document), along with access information.
Response	JSON Object, response from device.

Example Java command	`busCom.setUrl("http://" + ipAddress);`
	`busCom.setUuid(uuid);`
	`String Response = busCom.Command(JSON_Input);`
	`JSON_Input =`
	`{"islands":[{"bus_id":0,"scan":"groups"}]}`

Response received for the command from device	`{"islands":[{"bus_id":0,"groups"[{"address":0. "name":"Hall"},{"address":4,"name":"Bedroom"}]}],"status":"pass"}` If status is “pass” means API gets executed succesfully. If status is “fails” means there is error. For details check the response.
Explanation	Commands are explained in detail in API document

4.9 LIB_BUSCOM_API: Cloud Command

Item	Description
Command	To issue a communication command for SL-BUS Java Library for control, configuration & status to Cloud
HTTP API	Will be provided by Library callback layer
JSON Input	JSON Object, input for device control (explained in API document), along with access information.
Response	JSON Object, response from device.

Example Java command	`busCom.setUuid(uuid);`
	`String Response = busCom.CloudCommand(JSON_Input);`
	`JSON_Input =`
	`{"islands":[{"bus_id":0,"scan":"groups"}]}`

Response received for the command from device	`{"islands":[{"bus_id":0,"groups"[{"address":0.`
	`"name":"Hall"},{"address":4,"name":"Bedroom"}]}],"status":"pass"}`
	If status is “pass” means API gets executed succesfully.
	If status is “fails” means there is error. For details check the response.

Explanation	Commands are explained in detail in API document