Tuya uses following protocols:

- UDP(User Datagram Protocl)[1]
- TCP(Transmission Control Protocol)[2]
- HTTP(Hypertext Transfer Protocol)[3]
- MQTT(Message Queuing Telemetry Transport)[4]

Tuya uses following Encryption, Error Detection and MessageDigest algorithms:

- AES(Advanced Encryption Standard)*
- MD5(Message Digest Five)**
- CRC32(Cyclic Redundancy Check)**

*Encryption key can be recoveryed from Android device. Which is wrapped in XML(Extensible Markup Language) as JSON(Javascript Object Notation) object. *Key size of AES is 128bit*.

**MD5 to validate data integrity as signature.

***CRC32 is checksum algorithm to detect data corruption from source to destination.

1. UDP protocol is used to transfer Wifi credentials to connected device.
2. TCP protocol is used to control connected device locally.
3. HTTP protocol is used to query Tuya Cloud.
4. MQTT protocol is used to control connected device remotelly.

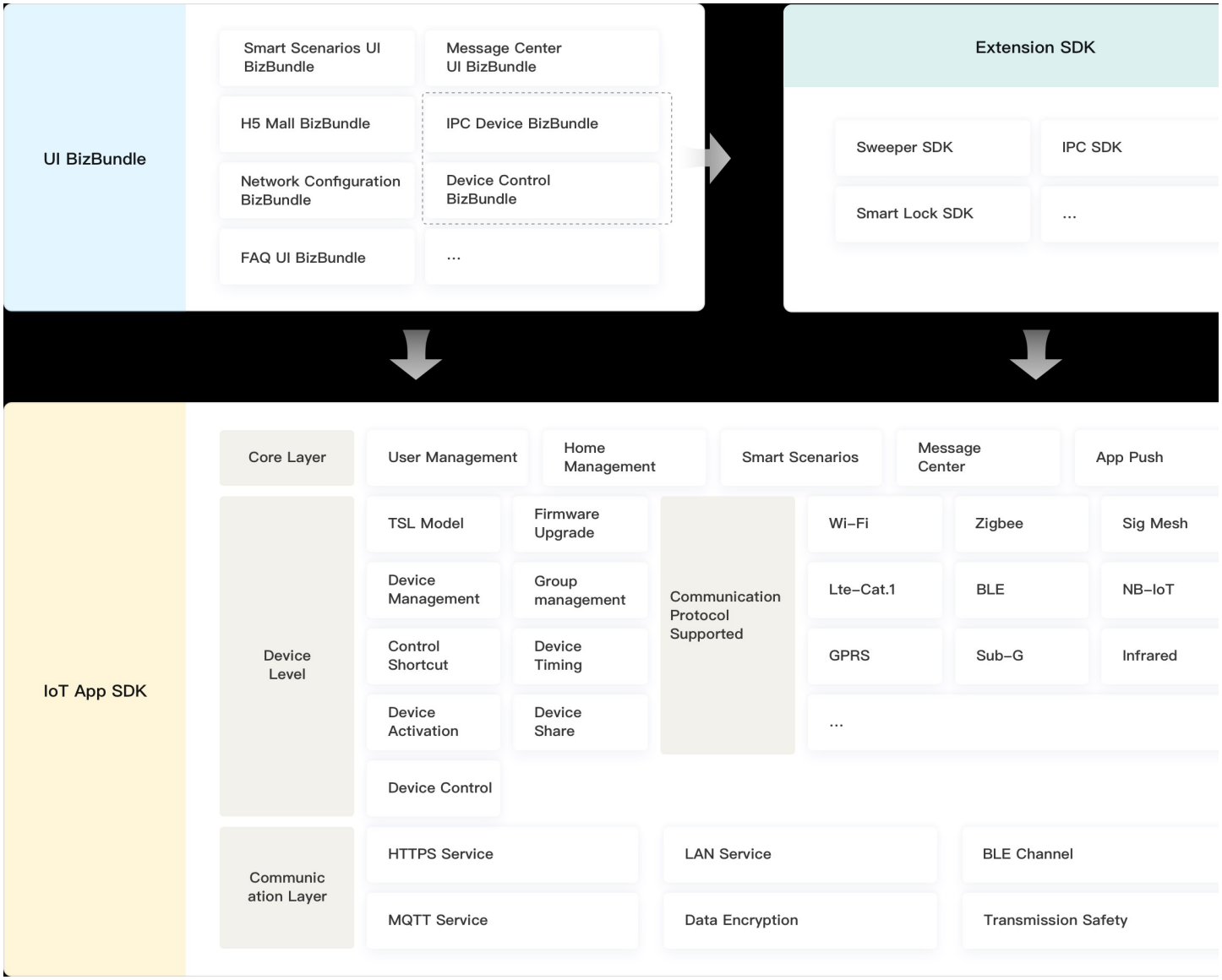
Network Pairing Management:

- Wi-Fi pairing
- Bluetooth pairing
- Bluetooth + Wi-Fi dual-module pairing

API list

Request method	API	Device type	API description
POST	/v1.0/device/paring/token	Common devices	Generate a network pairing token.
GET	/v1.0/device/paring/tokens/{token}	Common devices	Get the list of devices to be paired.
PUT	/v1.0/devices/{device_id}/enabled-sub-discovery	Zigbee devices	Open the gateway to allow the access of sub-devices.
GET	/v1.0/devices/{device_id}/list-sub	Zigbee devices	Get the list of paired devices.
GET	/v1.0/devices/{device_id}/sub-devices	Zigbee devices	Get the list of sub-devices under the gateway.

Technical Architecture



IoT App SDK

Develop mobile apps with the SDKs and components dedicated for smart home, commercial lighting, and generic industries to easily integrate IoT devices and smart scenes.

Smart Home App SDK

Empower mobile apps based on sufficient components and samples to simplify home automation.

Commercial Lighting App SDK

Bring integral multi-protocol IoT systems to commercial lighting in green buildings.

Industry App SDK

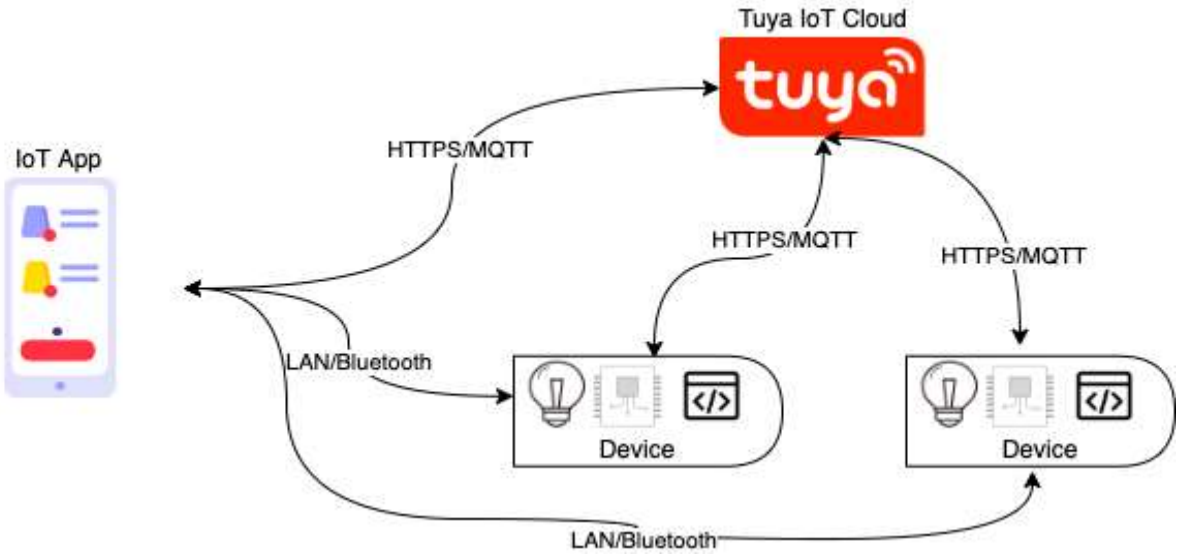
Manage abundant industry-specific cloud APIs tailored for basic needs powered by the Tuya IoT Platform.

Communications

- In an IoT solution, IoT devices can receive data and commands from the cloud and report local data to

the cloud. Example:

- A thermometer can be configured to report temperature data to the cloud on an hourly basis.
- The cloud can send commands to the air conditioner in a room to enable the cooling or heating mode.
- Abundant computing resources are provided for mobile phones and other terminals. However, it is not the case for IoT devices that have the following limits:
- Limited capabilities and computing resources are available.
- Networking might be unstable or costly.
- Dedicated, customized, or industry-specific application protocols are required in certain scenarios.
- To fix these issues, SDKs bring you on the right path to build secure and reliable connections between smart devices and the cloud.
- Tuya IoT App SDK and Tuya IoT Cloud support communications over Message Queuing Telemetry Transport (MQTT). MQTT is a publish-subscribe-based messaging protocol that runs over TCP/IP. The protocol is designed for bandwidth-efficient connections with memory-limited IoT devices in unreliable networking conditions. MQTT is a perfect fit for IoT communications.



Java classes for User Datagram preparation and transfer:

- import java.net.DatagramPacket
- import java.net.DatagramSocket
- import java.net.InetAddress
- import java.net.InetSocketAddress
- import java.net.MulticastSocket
- import java.net.SocketException
- import java.nio.channels.DatagramChannel

Linking a Tuya device with Smart Link

This method requires you to create a developer account on iot.tuya.com. It doesn't matter if the device(s) are currently registered in the Tuya Smart app or Smart Life app or not.

Create a new account on iot.tuya.com and make sure you are logged in. Select United States as your country when signing up. This seems to skip a required verify step.

Go to Cloud -> Developement in the left nav drawer. If you haven't already, you will need to "purchase" the Trial Plan before you can proceed with this step. You will not have to add any form of payment, and the purchase is of no charge. Once in the Projects tab, click "Create". Make sure you select "Smart Home" for the "Industry" field and SaaS for the development method. Select your country of use in the for the location access option, and feel free to skip the services option in the next window. After you've created a new project, click into it. The access ID and access key are equivalent to the API key and API secret values need in step 7.

Go to Cloud -> Development -> "MyProject" -> API -> "Go to authorize". "Select API" > click subscribe on "Smart Home Devices Management" in the dropdown. Click subscribe again. Click "basic edition" and "buy now" (basic edition is free). Check if the "Smart Home Devices Management" API is listed under Cloud -> Projects -> "MyProject" -> API. If not, click "New Authorization" and select it.

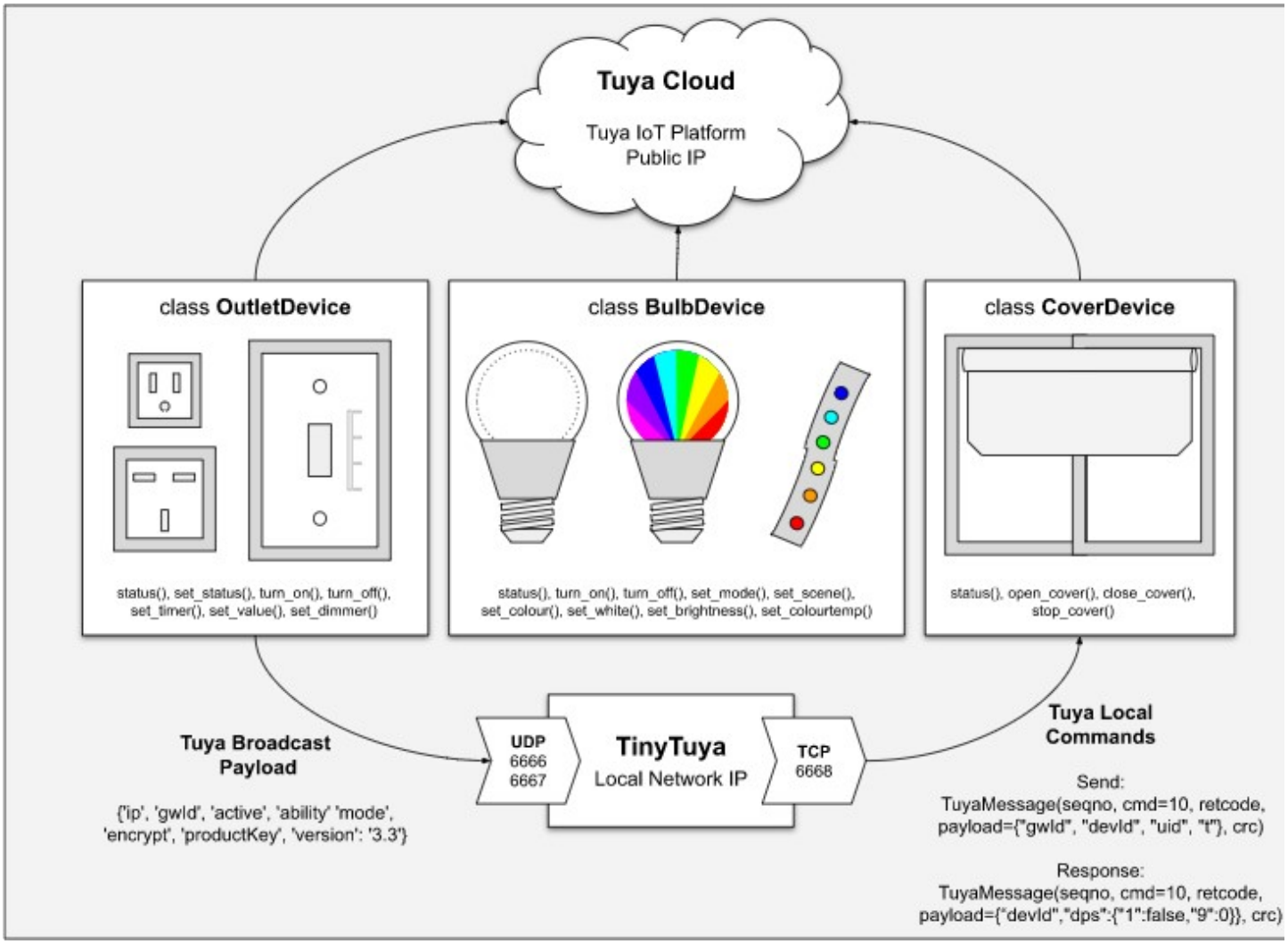
Go to App -> App SDK -> Development in the nav drawer. Click "Create" and enter whatever you want for the package names and Channel ID (for the Android package name, you must enter a string beginning with com.). Take note of the Channel ID you entered. This is equivalent to the schema value needed in step 7. Ignore any app key and app secret values you see in this section as they are not used.

Go to Cloud -> Development and click the project you created earlier. Then click "Link Device". Click the "Link devices by Apps" tab, and click "Add Apps". Check the app you just created and click "Ok".

Put your devices into linking mode. This process is specific to each type of device, find instructions in the Tuya Smart app. Usually this consists of turning it on and off several times or holding down a button.

On the command line, run tuya-cli link --api-key <your api key> --api-secret <your api secret> --schema <your schema/Channel ID> --ssid <your WiFi name> --password <your WiFi password> --region us. For the region parameter, choose the two-letter country code from us, eu, and cn that is geographically closest to you.

Your devices should link in under a minute and the parameters required to control them will be printed out to the console. If you experience problems, first make sure any smart phone/tablet app that you use with your devices is completely closed and not attempting to communicate with any of the devices.



Install

tuya-cli link --api-key 9y5wg3m3av1iz1iayejp --api-secret bd552a95e986412cb03c1fd6b3c836f4 --schema comsmarthomeuet --ssid R-One --password kics@iot --region us

tuya-cli link --api-key <your api key> --api-secret <your api secret> --schema <your schema/Channel ID> --ssid <your WiFi name> --password <your WiFi password> --region us

tuya-cli wizard

Configuration Data Saved to tinytuya.json

{

"apiKey": "9y5wg3m3av1iz1iayejp",

"apiSecret": "bd552a95e986412cb03c1fd6b3c836f4 ",

"apiRegion": "us",

"apiDeviceID": "bff34adb33aee4baebynue"

}

local_key = b8bd2daea424d12d

install homebridge

node -v

npm -v

npm install -g --unsafe-perm homebridge homebridge-config-ui-x

hb-service install

displayes

http://localhost:8581

* http://192.168.1.164:8581

* http://[fe80::c8c3:b9b6:bc9c:4e3c]:8581

default username = admin

default password= admin

Install plugin

homebridge-tuya-lan