



Software Version: 2.0.0

Installation Guide v2.0.0

Document Release Date: July 2017

Software Release Date: July 2017



Copyright and Disclaimer of Liability

This document may contain technical inaccuracy or type errors, and the author does not have any responsibility on this matter.

The contents of this document can be changed or added regularly, and the relevant corrected version will be added to the document under the title named “New Edition” in consecutive order. The product or program mentioned in this document may be changed or modified without any prior notice.

The source code of Mobius is distributed according to the license policy below.

- The open source code shared by OCEAN (Open allianCE for iot stANdard) is distributed based on the 3-clause BSD-style license. While maintaining copyright header in the source code file, the open source code can be used freely in the purpose of commercial or non-commercial systems.
- License of OCEAN does not force users to share the developed source code with others. The ownership of the developed source code belongs to the developer and (s)he has no obligation to share it.
- Anyone can contribute to improvement of the open source environment of OCEAN. If so, the developed source code should follow the license policy of OCEAN.

Copyright (c) 2017, OCEAN
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Table of Contents

1. Mobius	5
1.1. Introduction.....	5
1.2. Mobius Platform.....	6
1.2.1. Introduction.....	6
1.2.2. Mobius Platform Functionalities	7
1.2.3. Mobius Platform Components	7
1.2.4. Mobius Platform S/W Architecture	8
1.2.5. Mobius Platform Source Code Directory.....	9
2. Mobius Server Platform Installation.....	11
2.1. Introduction.....	12
2.2. Pre-requisites Installation (Linux).....	12
2.2.1. MySQL Installation	12
2.2.2. MQTT Server Installation	16
2.2.3. Node JS Installation	19
2.3. Mobius Installation	21
3. Running Mobius Server.....	23
3.1. Runtime Environment Configuration.....	23
3.2. Running Mobius Server	24
3.3. Test.....	24
3.3.1. oneM2MBrowser.....	24
3.3.2. Postman.....	24

1. Mobius

1.1. Introduction

Mobius platform is a oneM2M compliant IoT server platform designed to offer diverse IoT services and functionalities such as smart devices control and management, and user management etc. Users can experience the IoT services through user IoT applications. Figure 1 shows the architecture of Mobius platform structured with oneM2M entities and reference points through which any two entities can communicate with each other.

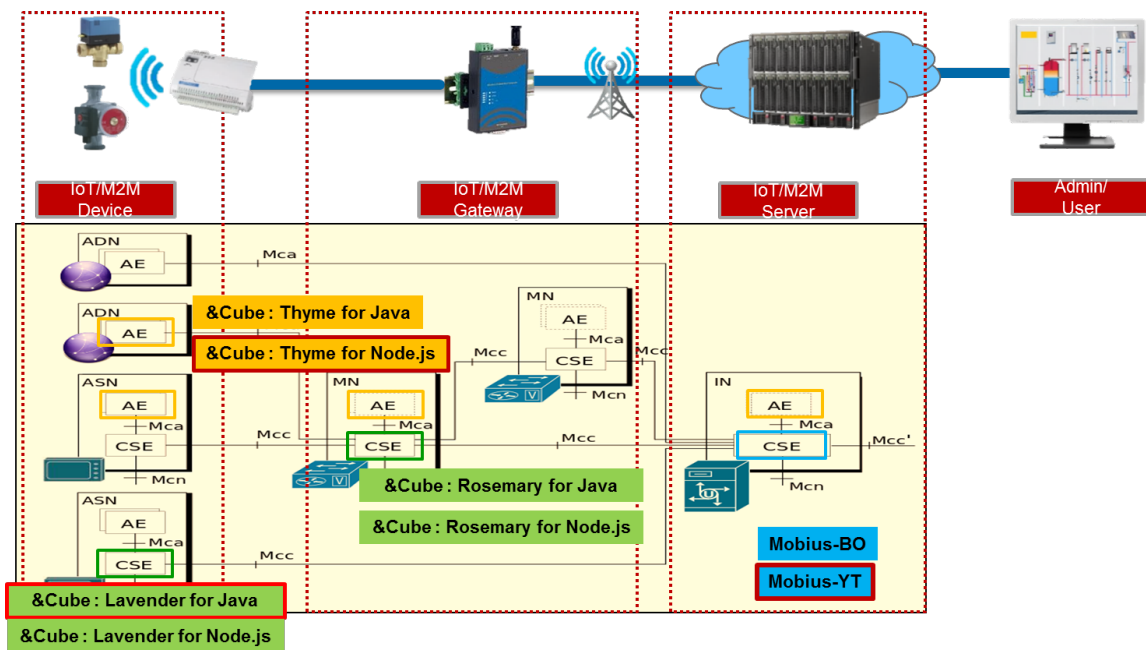


Figure 1 oneM2M complied Mobius platform architecture

Mobius platform provides a series of REST APIs for protocol HTTP, MQTT, CoAP and WebSocket to connect IoT devices together and implements interaction between each other through Mobius as shown as 오류! 참조 원본을 찾을 수 없습니다.. APIs for creation and retrieval of oneM2M resources are also provided to simplify the procedures of data management.

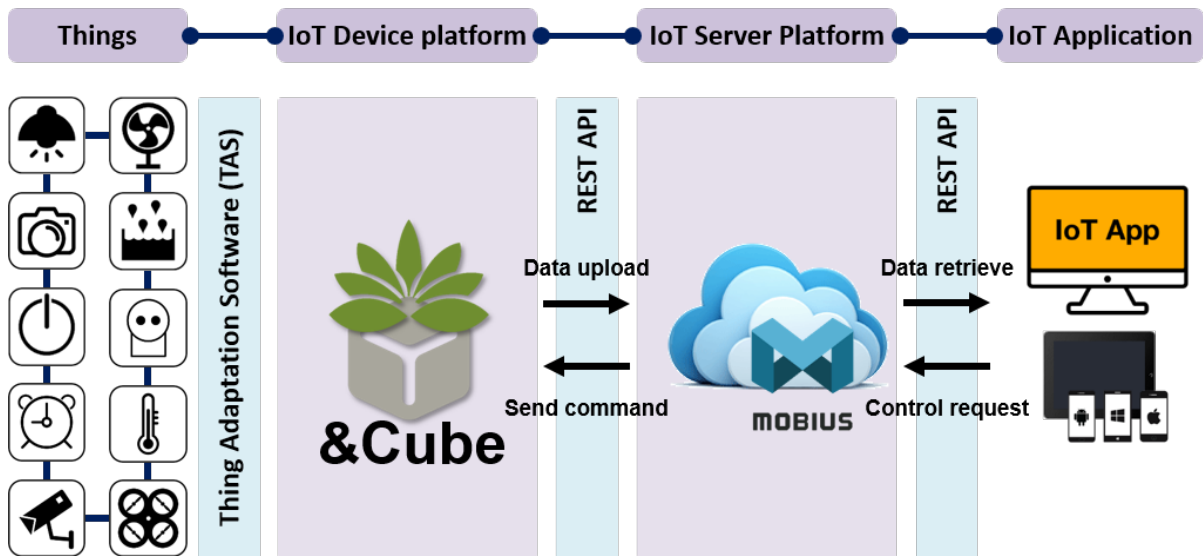


Figure 2 Interconnection between Mobius and IoT devices

1.2. Mobius Platform

1.2.1. Introduction

Mobius is a middleware server platform that connects diverse IoT devices through physical communication medias and creates virtual representations (oneM2M resources) for each IoT device to enable the interactions between each other as well as the communication between devices and IoT applications. In this way, Mobius provides an open environment and APIs for users to interconnect their own devices together and develop user specific IoT services to build an IoT ecosystem. 오류! 참조 원본을 찾을 수 없습니다. shows the components of Mobius platform i.e. IoT devices, Mobius server and IoT applications.

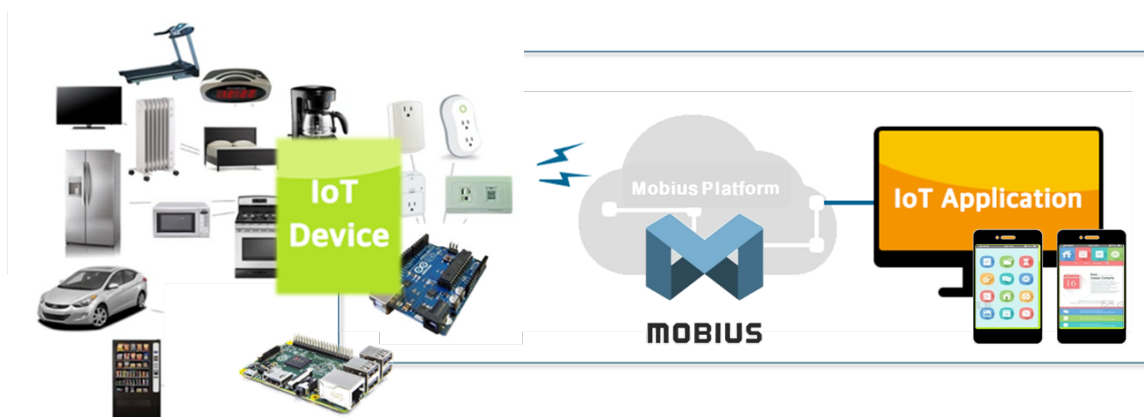


Figure 3 Mobius platform components

The IoT server platform can be implemented using diverse programming languages and the Mobius server platform is developed using Node JS. In addition, Mobius server platform uses

Node JS express modules which provides diverse modules for developers including HTTP, XML etc. instead of Node JS express framework.

The Mobius server platform is compliant to oneM2M standards and supports HTTP, CoAP, MQTT and WebSocket bindings specified in oneM2M standards. The Mobius server platform implements oneM2M Infrastructure Node Common Services Entity (IN-CSE) with structured resource architectures and provides IoT services through RESTful APIs. The Mobius server platform uses MySQL DBMS for resources storage.

1.2.2. Mobius Platform Functionalities

Mobius Platform works as the middle bridge to enable the communications and interactions between IoT devices and IoT applications as following:

- ① The Mobius server accepts the uploaded data from authenticated IoT devices and stores the data into the MySQL DB;
- ② The authenticated IoT applications can retrieve the data stored in Mobius server using Mobius open APIs for monitoring purpose;
- ③ The authenticated IoT applications are able to control remotely IoT devices that have already registered with Mobius server by sending control commands included in `<contentInstance>` resource. The Mobius server then executes control commands to the target devices.

The call flows as shown as Figure 4 indicates the communications and interactions among devices, IoT applications and Mobius server.

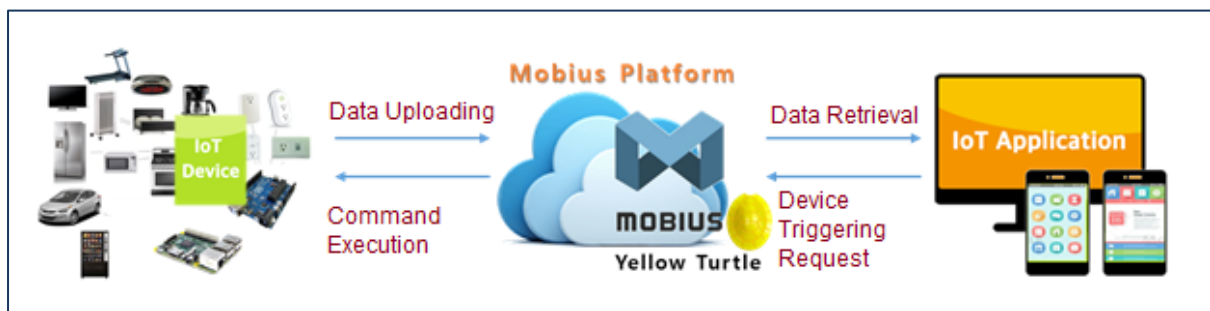


Figure 4 Mobius interaction with IoT devices and applications

1.2.3. Mobius Platform Components

Mobius server consists of HTTP, CoAP, MQTT, WebSocket server and MySQL DBMS while IoT applications implement HTTP, CoAP and/or MQTT clients in order to communicate with Mobius server.

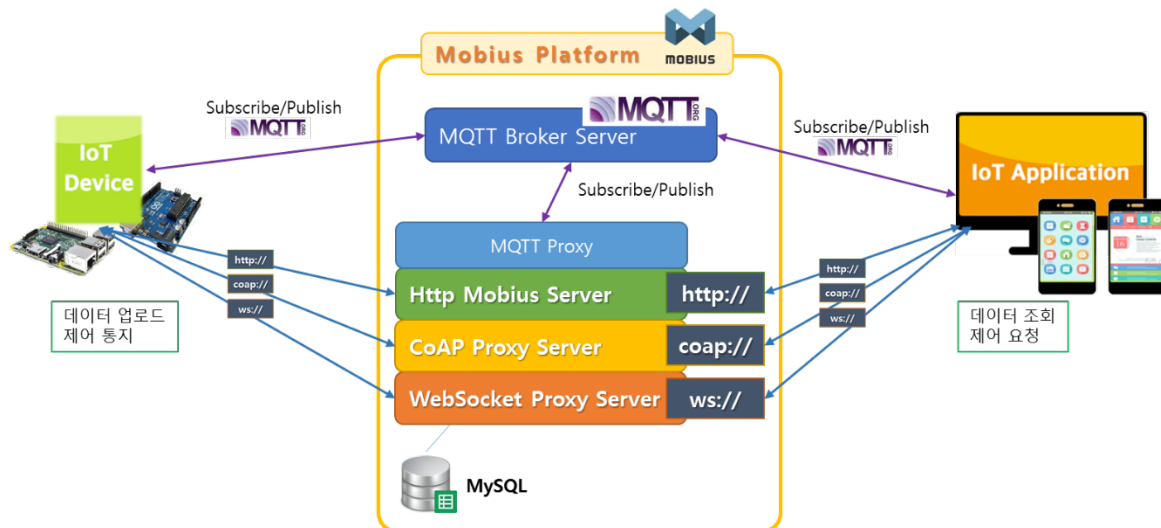


Figure 5 Mobius platform components

1.2.4. Mobius Platform S/W Architecture

For protocol binding support, Mobius has MQTT broker, CoAP server, WebSocket server that is bound to HTTP server internally. For example, MQTT protocol binding is supported by implementing MQTT to HTTP proxy. CoAP and WebSocket are designed in the same way. Mainly it consists of requester and responder. The requester contains the DB access component. Every HTTP request is go through requester component, parser, actor and then create SQL query to data access (e.g. retrieval, discovery) with DB connector. When it gets access result, the responder creates the response in XML, JSON or CBOR serialization.

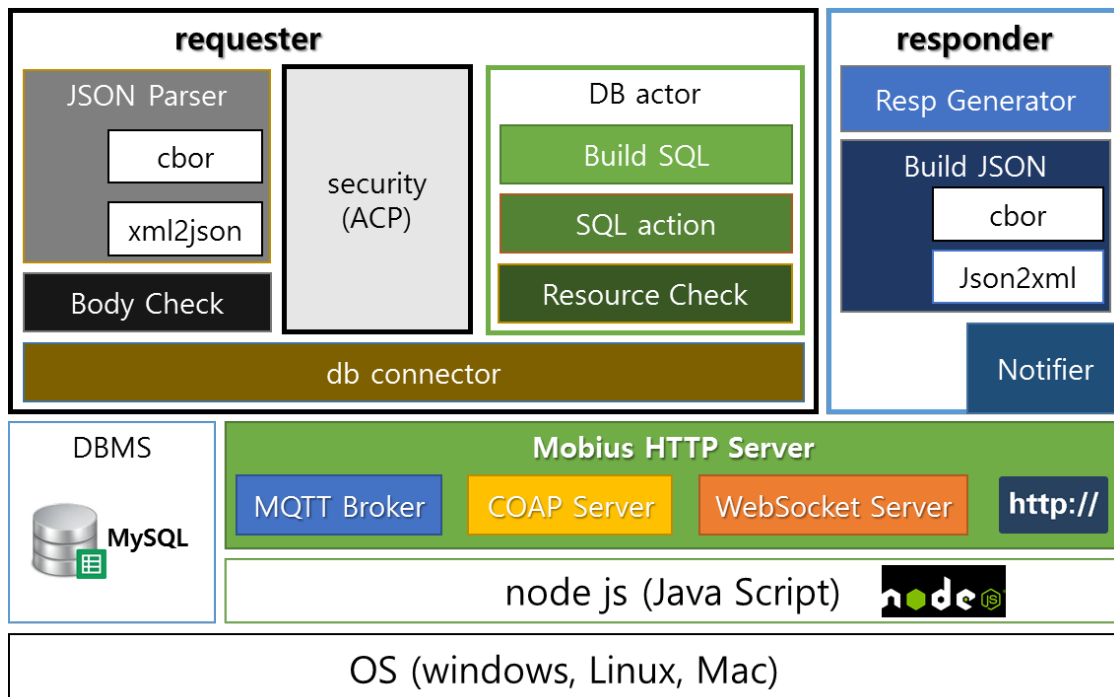


Figure 6 Mobius platform S/W architecture

1.2.5. Mobius Platform Source Code Directory

The figure below shows the Mobius Node JS source directory. For the detailed functions and roles for each Node JS file, please refer to the Table 1.

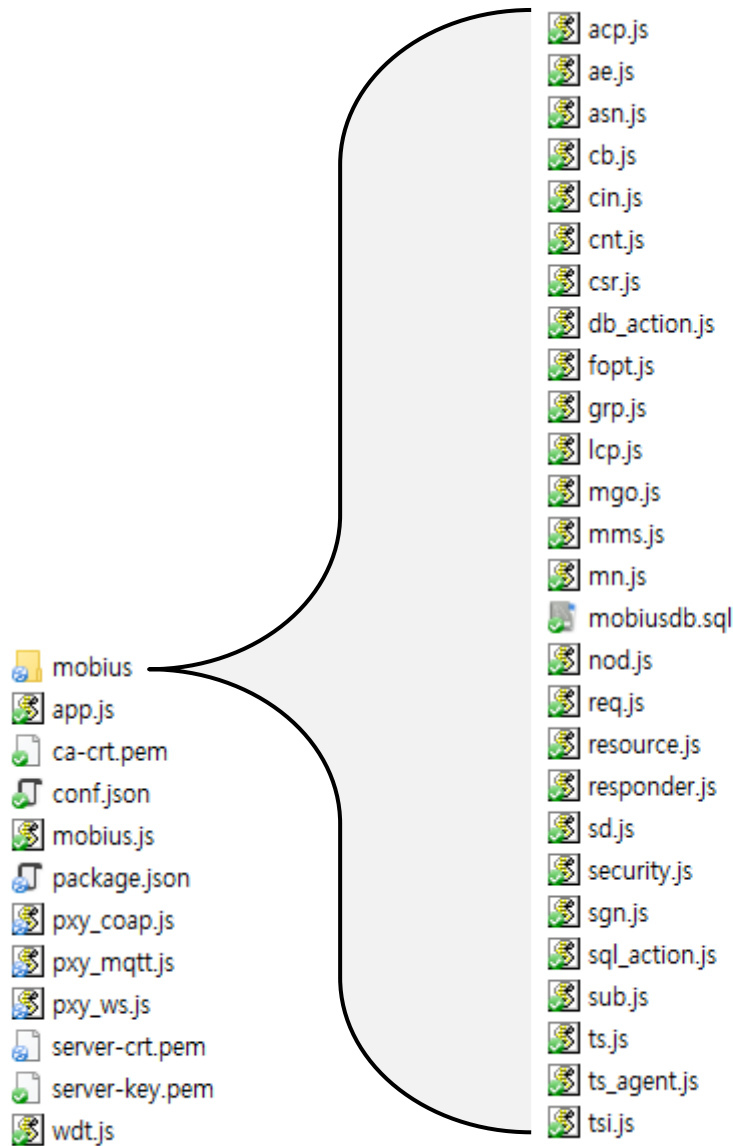


Figure 7 Mobius Node JS source code directory

Table 1 Function Reference Table for Node JS Files

Source File	Role and Function
mobius.js	This file initiates Mobius server and helps loading main Node JS files. It also contains configuration parameters for Mobius server such as <i>defaultbodytype</i> indicating the serialization, <i>usecsetype</i> indicating CSE type (either IN-CSE, MN-CSE or ASN-CSE), <i>usecsebase</i>

	indicating CSEBase name, <i>usecseid</i> indicating CSEID, <i>usedbhost</i> indicating the host address, and <i>usedbpass</i> indicating the password for MySQL etc. Users can modify those configuration parameters.
app.js	<p>This file acts as role of flow router and it is the main code running Mobius server.</p> <ul style="list-style-type: none"> ① It handles initial processing of received packets. ② It initiates HTTP server with 'listening' mode to wait for HTTP requests target to the Mobius HTTP server. ③ It handles the parsing of URL of packets and evaluate the correctness of the request body resulted of parsing. It then sends the request to resource.js to continue the processing if the request is valid one, otherwise throws exceptions. <p>This file also implements the server clustering algorithms to improve the performance of HTTP server.</p>
mobius/resource.js	<p>It is core file to process the CREATE, RETRIEVE, UPDATE, DELETE, NOTIFY and DISCOVERY operations for oneM2M resource primitives.</p> <p>This file undertakes the processing of parsed request URI and request body received from app.js according to corresponding operation. It converts the data into a format to process the data and connect to mysql database.</p> <p>The mysql database is initialized and handled by db_action.js and sql_action.js module.</p>
mobius/responder.js	<p>It is responsible for handling the response process.</p> <p>It receives processing results from app.js and resource.js modules and generates responses from the processing results following format requested by originator either XML or JSON serialization.</p>
mobius/db_action.js	This file contains parameters used to connect and access to the database and parameters for returning response results from the database.
mobius/sql_action.js	This file contains functions to receive data and parameters required for a series of database operations and functions to call db_action.js module to return data from database.
mobius/sgn.js	<p>This module file is responsible for checking the existence of <subscription> child resource under the requested target resource.</p> <p>If it exists, the module checks the event type and retrieves the field value of attribute <i>notificationUri</i>. Then it generates and sends a notification message to the address indicated by field value of attribute <i>notificationUri</i>.</p>
mobius/security.js	<p>This file contains functions to check the access privileges of originators for a requested resource when the resource applies with <accessControlPolicy> resource.</p> <p>The originator ID is abstracted from request header field of X-M2M-Origin. The process checks the access right of current originator ID to a target resource and responds with ACCESS_DENIED error when the originator ID has no privileges to access to the resource.</p>
mobius/fopt.js	<p>This file contains function to handle operations target to <fanOutPoint> virtual resource of a <group> resource.</p> <p>It transmits operation request to all group members contained in <group> resource and aggregates responses from all group members into an aggregated response.</p>

mobius/ts_agent.js	This file contains functions to manage <code><timeSeriesInstance></code> resource. It monitors the missing data in the <code><timeSeriesInstance></code> resource and then stores the missing data.
pxymqtt.js	<p>This file contains functions implementing mqtt proxying function to handle mqtt protocol messages with http protocol module as following procedures:</p> <ol style="list-style-type: none"> ① It creates a oneM2M mqtt topic with configuration information of Mobius server and then subscribe to the topic to receive mqtt requests targeted to this topic later; ② Whenever receiving mqtt protocol messages, it generates and sends a http request wrapping mqtt request message to Mobius server and waits for http response from Mobius server. It then abstracts http response and generates a mqtt response message correspondingly. ③ It responds with mqtt protocol message.
pxy_coap.js	This file contains functions implementing coap proxying function to handle coap protocol messages with http protocol module.
pxy_ws.js	This file contains functions implementing websocket proxying function to handle websocket protocol messages with http protocol module.
mobius/acp.js	It contains functions to parse accessControlPolicy request. The accessControlPolicy resource is used to manage the access control privileges to resources that apply with access control policy
mobius/ae.js	It contains functions to parse AE request
mobius/cb.js	It contains functions to parse CSEBase request. The CSEBase resource contains configuration information of Mobius server
mobius/cin.js	It contains functions to parse contentInstance request
mobius/cnt.js	It contains functions to parse container request
mobius/csr.js	It contains functions to parse remoteCSE request
mobius/grp.js	It contains functions to parse group request
mobius/lcp.js	It contains functions to parse locationPolicy request
mobius/mms.js	It contains functions to parse multimediaSession request
mobius/mn.js	It contains functions to register this CSE to the other CSE.
mobius/sd.js	It contains functions to parse semanticDescriptor request
mobius/sub.js	It contains functions to parse subscription request.
mobius/ts.js	It contains functions to parse timeSeries request.
mobius/tsi.js	It contains functions to parse timeSeriesInstance request.

2. Mobius Server Platform Installation

2.1. Introduction

Mobius Yellow Turtle server platform deploys MySQL DBMS as a database so MySQL is required to be installed as a basic and then MQTT server and Mobius server have to install one by one. Note that it is unnecessary to install MQTT server if MQTT protocol and MQTT related functions are not supported.



Figure 7 Mobius server installation procedures

2.2. Pre-requisites Installation (Linux)

2.2.1. MySQL Installation

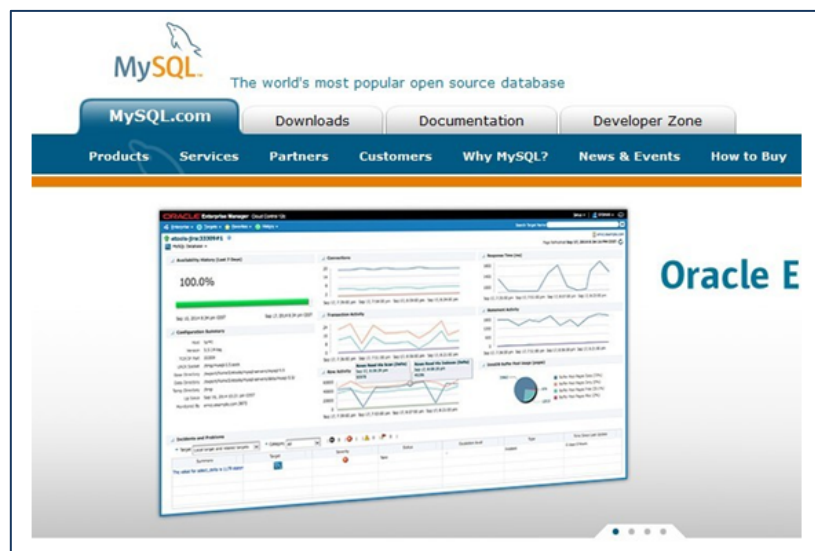


Figure 8 MySQL Introduction

Mobius server platform stores the data uploaded from devices into MySQL database. MySQL is an open source RDBMS and the installation procedures are introduced as follows:

```

keti@ubuntu: ~
File Edit View Search Terminal Help
keti@ubuntu:~$ wget http://dev.mysql.com/get/mysql-apt-config_0.6.0-1_all.deb
--2018-01-25 17:17:44-- http://dev.mysql.com/get/mysql-apt-config_0.6.0-1_all.d
eb
Resolving dev.mysql.com (dev.mysql.com)... 137.254.60.11
Connecting to dev.mysql.com (dev.mysql.com)|137.254.60.11|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://dev.mysql.com/get/mysql-apt-config_0.6.0-1_all.deb [following]
--2018-01-25 17:17:45-- https://dev.mysql.com/get/mysql-apt-config_0.6.0-1_all.
deb
Connecting to dev.mysql.com (dev.mysql.com)|137.254.60.11|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://repo.mysql.com/get/mysql-apt-config_0.6.0-1_all.deb [following]
--2018-01-25 17:17:46-- https://repo.mysql.com/get/mysql-apt-config_0.6.0-1_all.de
b
Resolving repo.mysql.com (repo.mysql.com)... 23.35.221.187
Connecting to repo.mysql.com (repo.mysql.com)|23.35.221.187|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 18384 (18K) [application/x-debian-package]
Saving to: 'mysql-apt-config_0.6.0-1_all.deb'

mysql-apt-config_0. 100%[=====] 17.95K --.-KB/s in 0s

```

Figure 10 Download MySQL deb File

Different versions of MySQL are provided for downloading at below link:

<http://dev.mysql.com/downloads/mysql>.

MySQL 5.7 version can be installed with "`sudo apt-get install mysql-server`" command because Ubuntu version 16.04 or later is done without installing any additional package. Version 14.04 of Ubuntu requires that you download the deb file directly from the server and add it to the package installation list for MySQL 5.7 installation. The download will proceed through "`wget http://dev.mysql.com/get/mysql-apt-config_0.6.0-1_all.deb`".

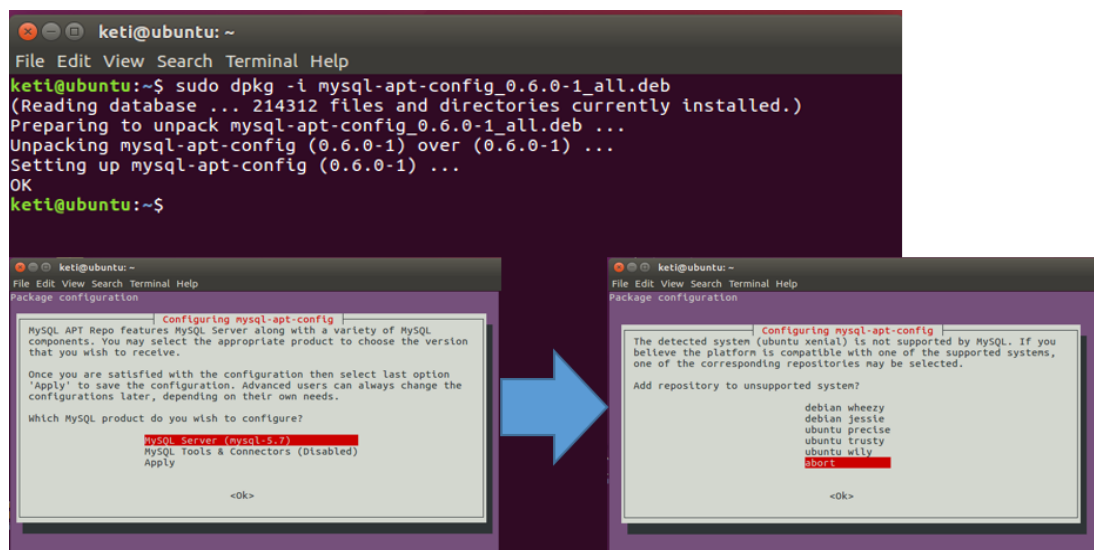


Figure 11 Downloaded MySQL deb file Installation

The downloaded deb file is installed by using "dpkg" command. For the next step, select MySQL Server (mysql-5.7) in the Select MySQL Version screen, then select Apply to proceed.

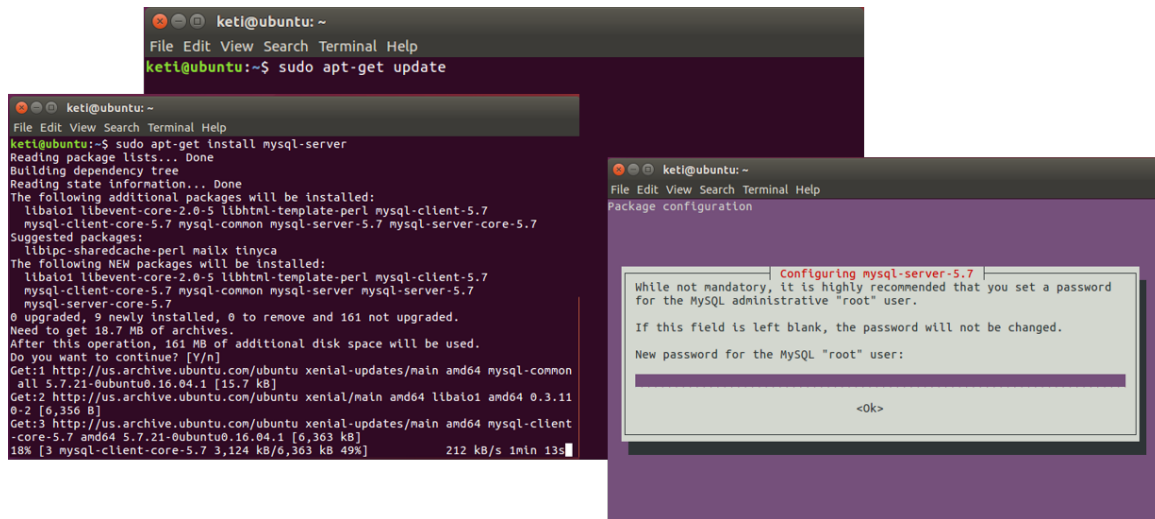


Figure 12 MySQL installation procedures

Updating the installation package list can be done by typing "`sudo apt-get update`" command. And type "`sudo apt-get install mysql-server`". The root password will be displayed when you proceed the installation. This password is necessary for interworking with Mobius in the future, so you must remember to set the password.

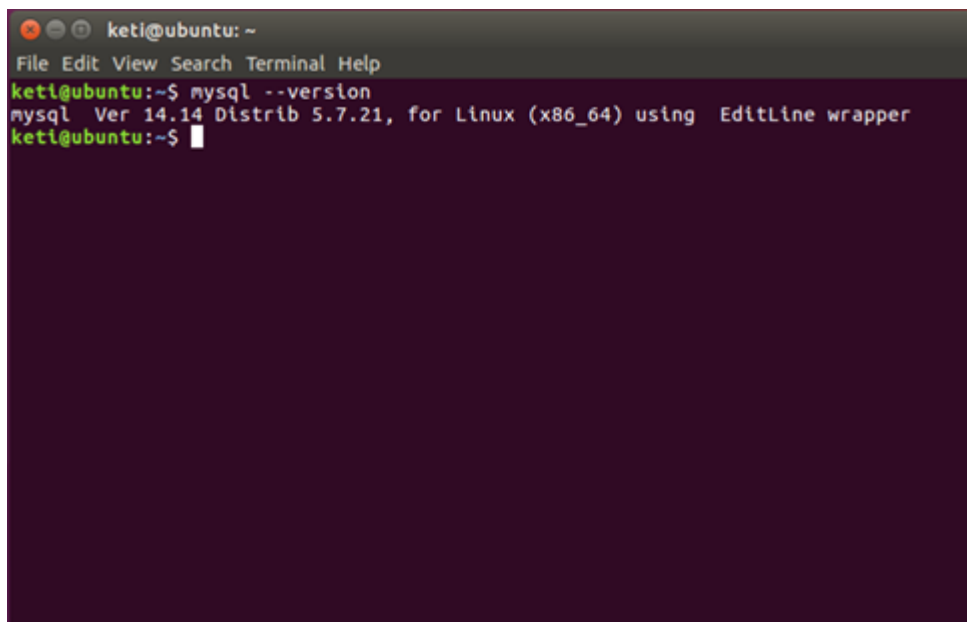
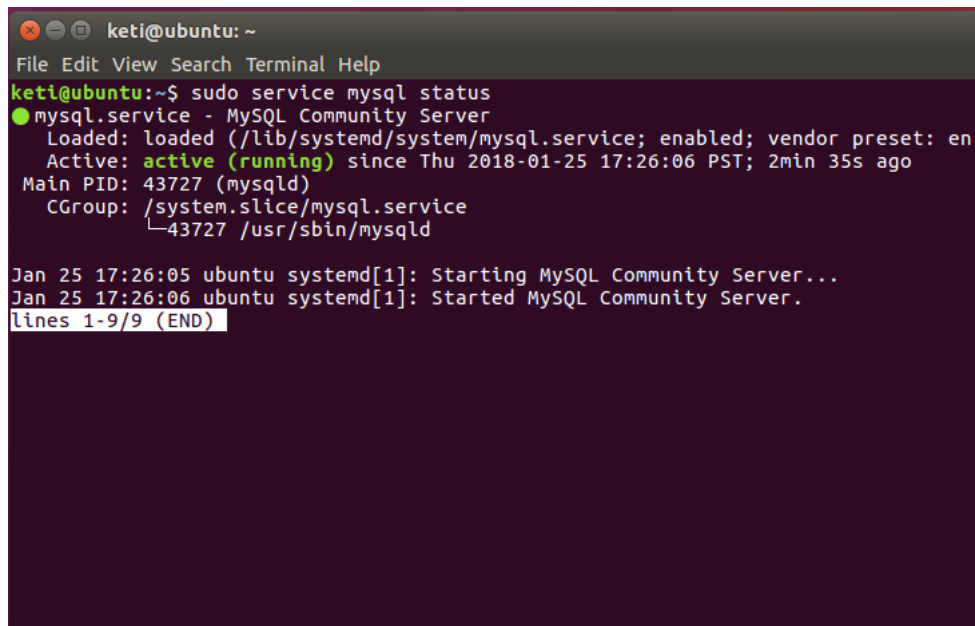


Figure 13 Check MySQL version

You can check whether the version of MySQL installed is version 5.7 or not by using "`mysql --version`" command.

A terminal window titled 'keti@ubuntu: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The user has entered the command 'sudo service mysql status'. The output shows that the 'mysql.service - MySQL Community Server' is loaded and active (running) since Thu 2018-01-25 17:26:06 PST. It also displays the main PID as 43727 (mysqld) and the CGroup path. At the bottom, there are two log messages from systemd[1] indicating the starting and starting of the MySQL Community Server. The terminal also shows 'lines 1-9/9 (END)' at the bottom left.

```
keti@ubuntu: ~  
File Edit View Search Terminal Help  
keti@ubuntu:~$ sudo service mysql status  
● mysql.service - MySQL Community Server  
   Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: en  
   Active: active (running) since Thu 2018-01-25 17:26:06 PST; 2min 35s ago  
 Main PID: 43727 (mysqld)  
    CGroup: /system.slice/mysql.service  
            └─43727 /usr/sbin/mysqld  
  
Jan 25 17:26:05 ubuntu systemd[1]: Starting MySQL Community Server...  
Jan 25 17:26:06 ubuntu systemd[1]: Started MySQL Community Server.  
lines 1-9/9 (END)
```

Figure 14 Check the execution MySQL

MySQL will run automatically when the installation is complete. However, you can check the execution of MySQL by typing the command `"sudo service mysql status"`.

By this, we have installed MySQL successfully and then we have to configure the MySQL in terms of creation of a database for Mobius. In [OCEAN alliance](#) website download page, the MySQL database file can be downloaded to use. How to download and import the file will be explained later.

2.2.2. MQTT Server Installation

Mobius enables IoT devices to communicate with MQTT server through MQTT protocol. To use this function, an MQTT broker needs to be installed. We recommend using open source MQTT broker Mosquitto server. Note: MQTT protocol can optionally support, if users are not implement MQTT functionalities, “installation of MQTT server” step can be ignored. In addition, the installation process requires installing the version of MQTT supported by Linux.

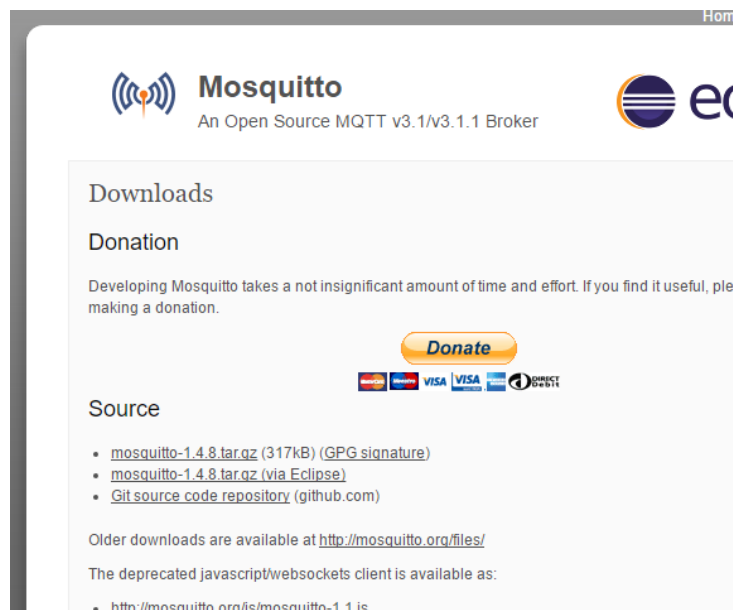


Figure 15 Mosquitto Introduction

```

keti@ubuntu: ~
File Edit View Search Terminal Help
keti@ubuntu:~$ sudo apt-get install mosquitto
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libev4 libuv1 libwebsockets7
The following NEW packages will be installed:
  libev4 libuv1 libwebsockets7 mosquitto
0 upgraded, 4 newly installed, 0 to remove and 161 not upgraded.
Need to get 253 kB of archives.
After this operation, 710 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
0% [Working]
Get:1 http://us.archive.ubuntu.com/ubuntu xenial/universe amd64 libuv1 amd64 1.8
.0-1 [57.4 kB]
Get:2 http://us.archive.ubuntu.com/ubuntu xenial/universe amd64 libev4 amd64 1:4
.22-1 [26.3 kB]
Get:3 http://us.archive.ubuntu.com/ubuntu xenial/universe amd64 libwebsockets7 a
md64 1.7.1-1 [61.0 kB]
Get:4 http://us.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 mosquitt
o amd64 1.4.8-1ubuntu0.16.04.2 [108 kB]
Fetched 253 kB in 2s (109 kB/s)
Selecting previously unselected package libuv1:amd64.
(Reading database ... 214581 files and directories currently installed.)
  
```

Figure 16 Mosquitto installation


```
keti@ubuntu: ~  
File Edit View Search Terminal Help  
keti@ubuntu:~$ sudo apt-get install mosquitto-clients  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following additional packages will be installed:  
  libc-ares2 libmosquitto1  
The following NEW packages will be installed:  
  libc-ares2 libmosquitto1 mosquitto-clients  
0 upgraded, 3 newly installed, 0 to remove and 161 not upgraded.  
Need to get 96.4 kB of archives.  
After this operation, 330 kB of additional disk space will be used.  
Do you want to continue? [Y/n] y  
Get:1 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libc-ares2 a  
md64 1.10.0-3ubuntu0.2 [34.1 kB]  
Get:2 http://us.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 libmosqu  
itto1 amd64 1.4.8-1ubuntu0.16.04.2 [31.3 kB]  
Get:3 http://us.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 mosquitt  
o-clients amd64 1.4.8-1ubuntu0.16.04.2 [31.0 kB]  
Fetched 96.4 kB in 1s (75.7 kB/s)  
Selecting previously unselected package libc-ares2:amd64.  
(Reading database ... 214626 files and directories currently installed.)  
Preparing to unpack .../libc-ares2_1.10.0-3ubuntu0.2_amd64.deb ...  
Unpacking libc-ares2:amd64 (1.10.0-3ubuntu0.2) ...  
Selecting previously unselected package libmosquitto1:amd64.
```

Figure 17 Mosquitto-clients installation

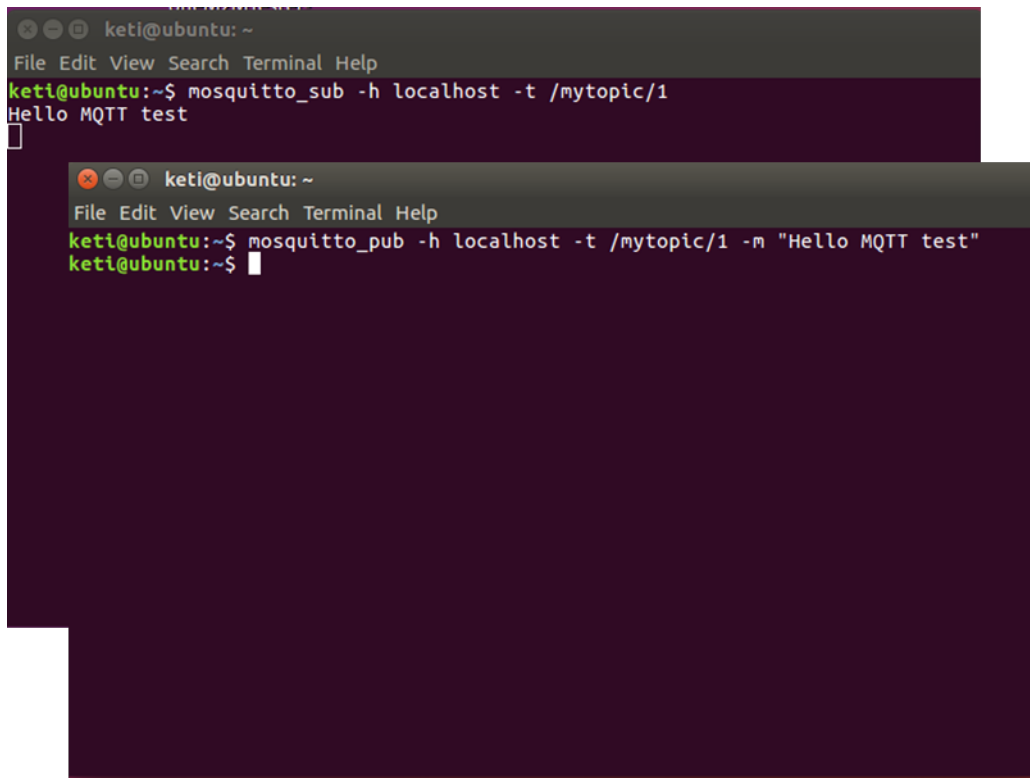
Windows mosquitto can only be downloaded and installed from the homepage, but in Linux, mosquitto and mosquitto-clients must be installed separately.

It is installed through the command "`sudo apt-get install mosquitto`" and "`sudo apt-get install mosquitto-clients`".

```
keti@ubuntu: ~  
File Edit View Search Terminal Help  
keti@ubuntu:~$ mosquitto -v  
1516930283: mosquitto version 1.4.8 (build date Mon, 26 Jun 2017 09:31:02 +0100)  
starting  
1516930283: Using default config.  
1516930283: Opening ipv4 listen socket on port 1883.  
1516930283: Error: Address already in use  
keti@ubuntu:~$
```

Figure 18 Check Mosquitto version

Check the installed version of mosquito through the "`mosquitto -v`" command.



The image shows two terminal windows. The top window shows the command `mosquitto_sub -h localhost -t /mytopic/1` being executed, which results in the output `Hello MQTT test`. The bottom window shows the command `mosquitto_pub -h localhost -t /mytopic/1 -m "Hello MQTT test"` being executed, with the cursor positioned at the end of the command line.

```
keti@ubuntu: ~  
File Edit View Search Terminal Help  
keti@ubuntu:~$ mosquitto_sub -h localhost -t /mytopic/1  
Hello MQTT test  
keti@ubuntu:~$  
  
keti@ubuntu: ~  
File Edit View Search Terminal Help  
keti@ubuntu:~$ mosquitto_pub -h localhost -t /mytopic/1 -m "Hello MQTT test"  
keti@ubuntu:~$
```

Figure 19 Mosquitto broker test

After installed Mosquitto broker, we can test whether the Mosquitto broker works properly or not as follows:

Step -1: Subscription test

- ① Open the Terminal Command and input command as below

```
mosquitto_sub -h localhost -t /mytopic/1
```

Step -2: Notification test

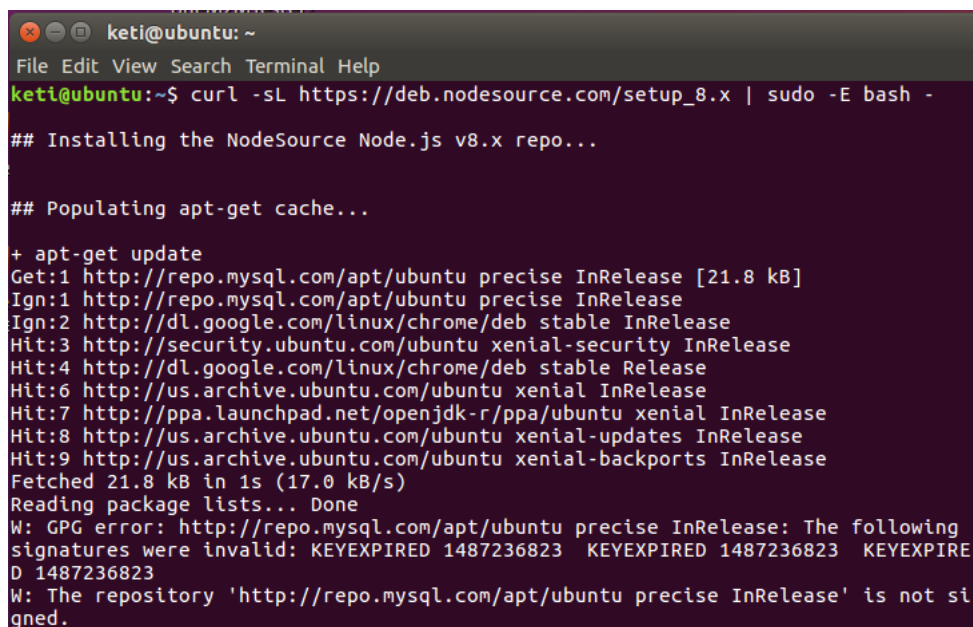
- ② Open the Terminal Command and input command as below

```
mosquitto_pub -h localhost -t /mytopic/1 -m "Hello MQTT test"
```

2.2.3. Node JS Installation

Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient. In contrast to multi-thread based servers, node.js operates on a single thread using non-blocking I/O calls and it allows to support tens of thousands of concurrent connections. With node.js, users don't need to worry about the dead-locking of processes because almost no function in node.js directly performs I/O. So there is no locking problem at all and therefore, scalable systems can be reasonably developed through node.js.

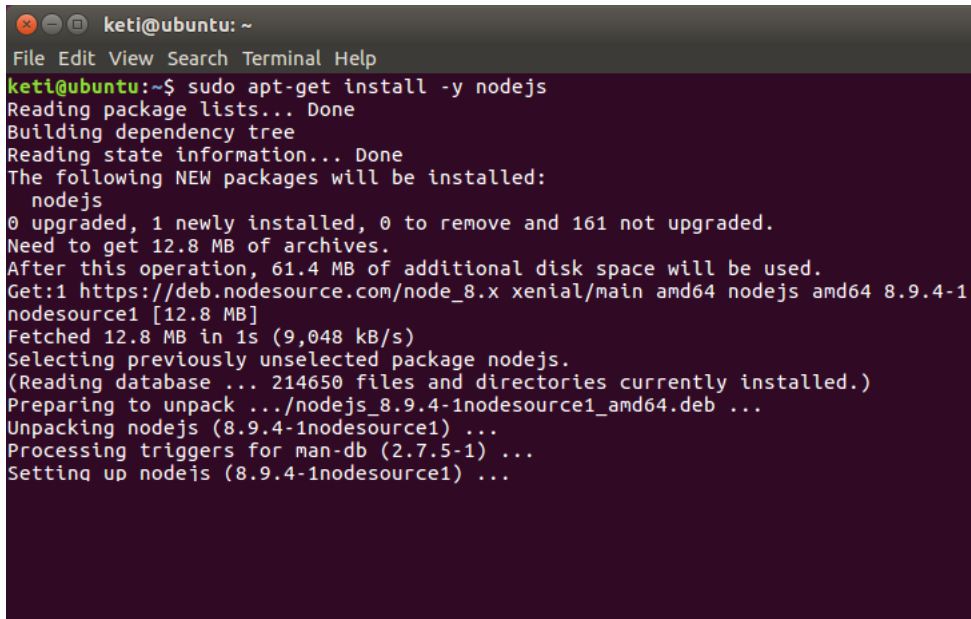
Node.js was originally written by developer Ryan Dahl in 2009 and distributed as an open source. Currently a diverse of modules developed from contributors are included such as covering file system I/O, networking, binary data (buffers), cryptography and data streams.



```
keti@ubuntu: ~  
File Edit View Search Terminal Help  
keti@ubuntu:~$ curl -sL https://deb.nodesource.com/setup_8.x | sudo -E bash -  
## Installing the NodeSource Node.js v8.x repo...  
## Populating apt-get cache...  
+ apt-get update  
Get:1 http://repo.mysql.com/apt/ubuntu precise InRelease [21.8 kB]  
Ign:1 http://repo.mysql.com/apt/ubuntu precise InRelease  
Ign:2 http://dl.google.com/linux/chrome/deb stable InRelease  
Hit:3 http://security.ubuntu.com/ubuntu xenial-security InRelease  
Hit:4 http://dl.google.com/linux/chrome/deb stable Release  
Hit:6 http://us.archive.ubuntu.com/ubuntu xenial InRelease  
Hit:7 http://ppa.launchpad.net/openjdk-r/ppa/ubuntu xenial InRelease  
Hit:8 http://us.archive.ubuntu.com/ubuntu xenial-updates InRelease  
Hit:9 http://us.archive.ubuntu.com/ubuntu xenial-backports InRelease  
Fetched 21.8 kB in 1s (17.0 kB/s)  
Reading package lists... Done  
W: GPG error: http://repo.mysql.com/apt/ubuntu precise InRelease: The following  
signatures were invalid: KEYEXPIRED 1487236823 KEYEXPIRED 1487236823 KEYEXPIRE  
D 1487236823  
W: The repository 'http://repo.mysql.com/apt/ubuntu precise InRelease' is not si  
gned.
```

Figure 20 Node source homepage adds the apt repository

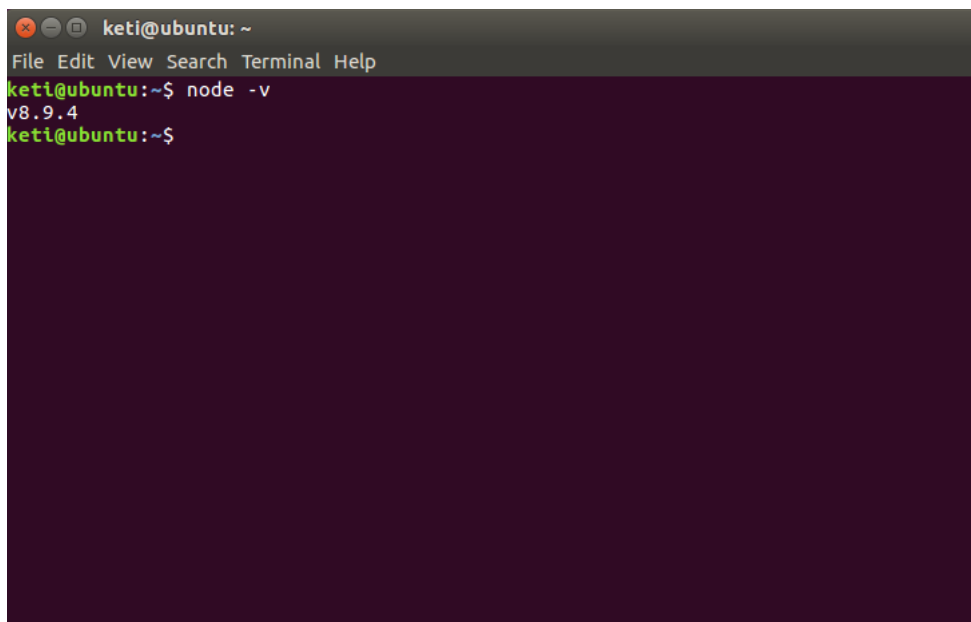
It is possible to install v6.x version through "`sudo apt-get install nodejs`" command, but we will install v8.x version here. Users of Ubuntu OS 16.04 version can proceed as shown in Figure 21. Since Ubuntu OS14.04 users cannot install Node.js v8.x directly from the package, the latest v8 LTS provided from the Node source homepage adds the apt repository through the "`curl -sL https://deb.nodesource.com/setup_8.x | sudo -E bash -`" command.

A terminal window titled 'keti@ubuntu: ~' showing the command 'sudo apt-get install -y nodejs' and its output. The output indicates that the package 'nodejs' will be installed, requiring 12.8 MB of archives and 61.4 MB of additional disk space. It shows the package being fetched from a repository and then unpacked and set up.

```
keti@ubuntu: ~  
File Edit View Search Terminal Help  
keti@ubuntu:~$ sudo apt-get install -y nodejs  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following NEW packages will be installed:  
  nodejs  
0 upgraded, 1 newly installed, 0 to remove and 161 not upgraded.  
Need to get 12.8 MB of archives.  
After this operation, 61.4 MB of additional disk space will be used.  
Get:1 https://deb.nodesource.com/node_8.x xenial/main amd64 nodejs amd64 8.9.4-1  
nodesource1 [12.8 MB]  
Fetched 12.8 MB in 1s (9,048 kB/s)  
Selecting previously unselected package nodejs.  
(Reading database ... 214650 files and directories currently installed.)  
Preparing to unpack .../nodejs_8.9.4-1nodesource1_amd64.deb ...  
Unpacking nodejs (8.9.4-1nodesource1) ...  
Processing triggers for man-db (2.7.5-1) ...  
Setting up nodejs (8.9.4-1nodesource1) ...
```

Figure 21 Node js installation

When the apt repository is complete, it is installed via the "`sudo apt-get install -y nodejs`" command.

A terminal window titled 'keti@ubuntu: ~' showing the command 'node -v' and its output 'v8.9.4'.

```
keti@ubuntu: ~  
File Edit View Search Terminal Help  
keti@ubuntu:~$ node -v  
v8.9.4  
keti@ubuntu:~$
```

Figure 22 Check Node js version

The last version of Node.js installed is checked through the "`node -v`" command.

2.3. Mobius Installation

Mobius server platform is distributed with the source code package in OCEAN alliance website (<http://www.iotocean.org>) or the Mobius GitHub (<https://github.com/loTKETI/Mobius>).

User can download the zipped package and configure the Mobius server in terms of communication port and CSE name etc.

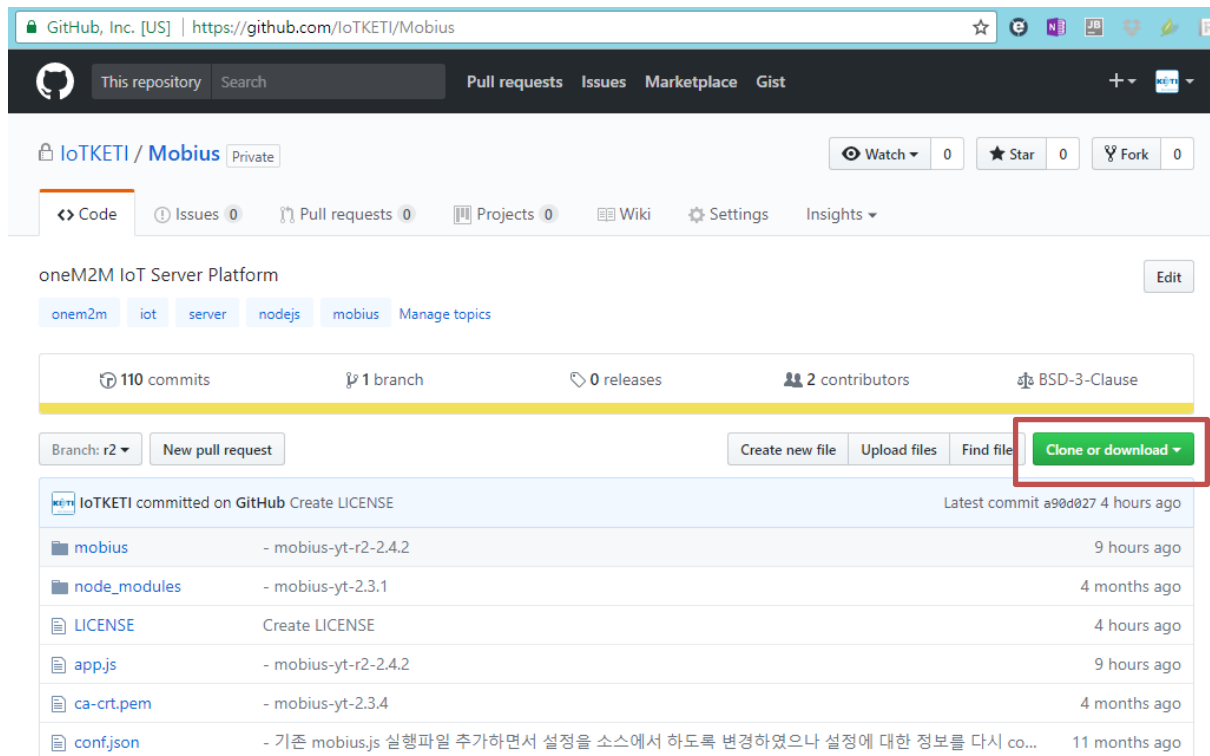


Figure 23 Download Mobius from GitHub

After downloading, unzip the package and you will see the included folders and files as Figure 24. The roles of files are introduced in Table 1.

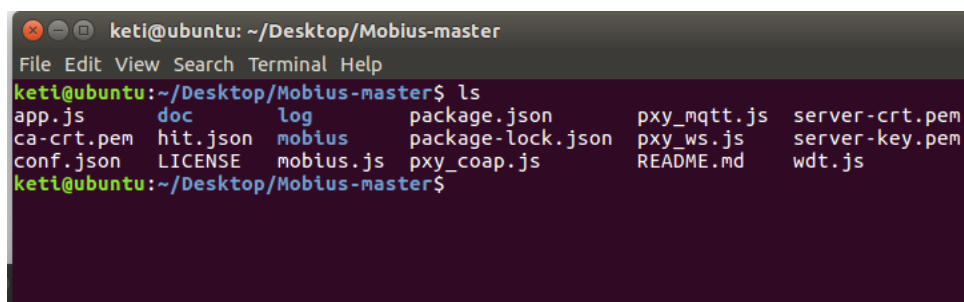


Figure 24 Unzipped Mobius server package files

The unzipped node.js files can be directly executed without any additional compiling operation. But there are several node.js modules which are not yet installed. To install the required additional node.js modules, open a Linux terminal window and go to the directory where the unzipped package and run `npm install` command as shown as Figure 25 Run `npm install` command to install additional node.js modules.

```

keti@ubuntu: ~/Desktop/Mobius-master
File Edit View Search Terminal Help
keti@ubuntu:~/Desktop/Mobius-master$ npm install
npm WARN deprecated crypto@0.0.3: This package is no longer supported. It's now
a built-in Node module. If you've depended on crypto, you should switch to the o
ne that's built-in.
> websocket@1.0.25 install /home/keti/Desktop/Mobius-master/node_modules/websocke
t
> (node-gyp rebuild 2> builderror.log) || (exit 0)

make: Entering directory '/home/keti/Desktop/Mobius-master/node_modules/websocke
t/build'
  CXX(target) Release/obj.target/bufferutil/src/bufferutil.o
  SOLINK_MODULE(target) Release/obj.target/bufferutil.node
  COPY Release/bufferutil.node
  CXX(target) Release/obj.target/validation/src/validation.o
  SOLINK_MODULE(target) Release/obj.target/validation.node
  COPY Release/validation.node
make: Leaving directory '/home/keti/Desktop/Mobius-master/node_modules/websocket
/build'
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN Mobius@2.0.0 No repository field.

added 155 packages in 10.225s
keti@ubuntu:~/Desktop/Mobius-master$

```

Figure 25 Run `npm install` command to install additional node.js modules

After running the `npm install` command, all additional required node.js modules will be generated and stored in `node_modules` folder. You can see there is a `node_modules` folder in the unzipped Mobius source folder as shown as Figure 26.

```

keti@ubuntu: ~/Desktop/Mobius-master
File Edit View Search Terminal Help
keti@ubuntu:~/Desktop/Mobius-master$ ls
app.js      hit.json   mobius.js  pxy_coap.js  server-crt.pem
ca-crt.pem  LICENSE   node_modules  pxy_mqtt.js  server-key.pem
conf.json   log       package.json  pxy_ws.js    wdt.js
doc         mobius    package-lock.json  README.md
keti@ubuntu:~/Desktop/Mobius-master$

```

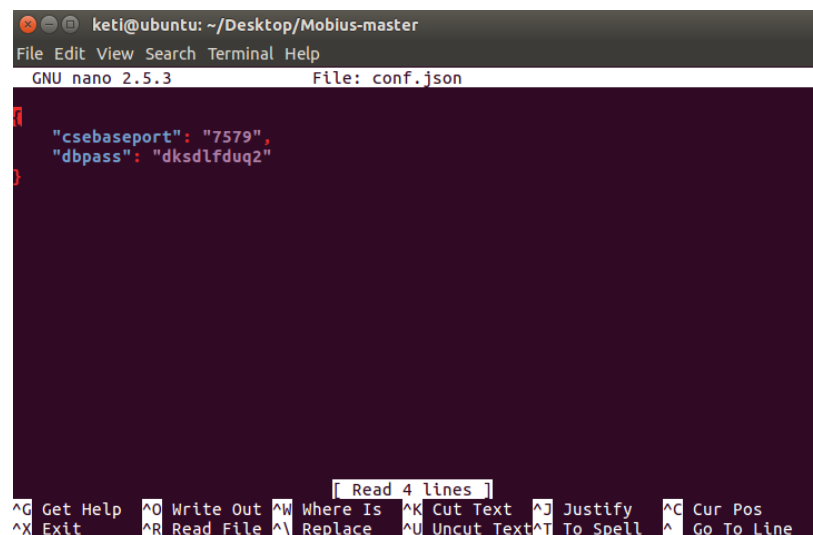
Figure 26 A `node_modules` folder created in unzipped source folder

By this, we have installed all required software and modules to run Mobius server. Users can run `node mobius.js` on the command line to activate the Mobius server. The installation of Mobius server is almost done and only a few configurations are left.

3. Running Mobius Server

3.1. Runtime Environment Configuration


A configuration file `conf.json` is included in the unzipped folder of Mobius package. User can customize the communication port `csebaseport` for CSEBase which is the root of all resources to be created in the hierarchical structure in Mobius server, and the MySQL database connection password `dbpass` as shown in Figure 7. These configurations are required to guarantee that the Mobius server is accessible from the CSEBase port and MySQL database with the configured password. In addition, there are configuration data included in `mobius.js` module file as shown in Figure .



```
keti@ubuntu: ~/Desktop/Mobius-master
File Edit View Search Terminal Help
GNU nano 2.5.3 File: conf.json

"dbpass": "dksdlfduq2"
"port": "7579"
}
```

Figure 27 Mobius-YT server configuration file



```
var fs = require('fs');

var data = fs.readFileSync('conf.json', 'utf-8');
var conf = JSON.parse(data);

global.defaultnntype = 'short';
global.defaultbodytype = 'json';

// my CSE information
global.usecsetype = '\n'; // select '\n' or '\r\n' or '\n\r'
global.usecsebase = 'Mobius';
global.usecseid = 'Mobius';
global.usecsebaseport = conf.csebaseport;

global.usedbhost = 'localhost';
global.usedbpass = conf.dbpass;

global.usepxywsport = '7577';
global.usepxymqttport = '7578';

global.usesagentport = '7582';

global.usemqttbroker = 'localhost'; // mqttbroker for mobius
global.usesemanticbroker = '10.10.202.114';
global.usesecure = 'disable';
if (usesecure === 'enable') {
    global.usemqttport = '8883';
} else {
    usemqttport = '1883';
}

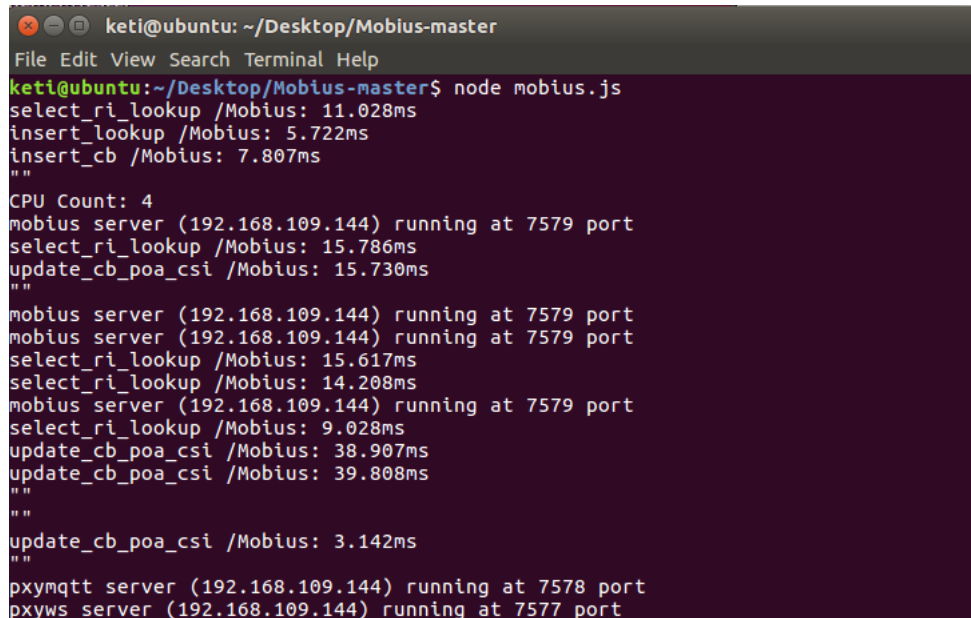
global.useaccesscontrolpolicy = 'disable';
```

Figure 28 Other configuration data in main code file

By this, we have done all installation and configurations for the Mobius server platform.

3.2. Running Mobius Server

Now users can run `node mobius.js` command to run Mobius server and it runs as shown in Figure 29.



```
keti@ubuntu: ~/Desktop/Mobius-master
File Edit View Search Terminal Help
keti@ubuntu:~/Desktop/Mobius-master$ node mobius.js
select_ri_lookup /Mobius: 11.028ms
insert_lookup /Mobius: 5.722ms
insert_cb /Mobius: 7.807ms
""
CPU Count: 4
mobius server (192.168.109.144) running at 7579 port
select_ri_lookup /Mobius: 15.786ms
update_cb_poa_csi /Mobius: 15.730ms
""
mobius server (192.168.109.144) running at 7579 port
mobius server (192.168.109.144) running at 7579 port
select_ri_lookup /Mobius: 15.617ms
select_ri_lookup /Mobius: 14.208ms
mobius server (192.168.109.144) running at 7579 port
select_ri_lookup /Mobius: 9.028ms
update_cb_poa_csi /Mobius: 38.907ms
update_cb_poa_csi /Mobius: 39.808ms
""
""
update_cb_poa_csi /Mobius: 3.142ms
""
pxymqtt server (192.168.109.144) running at 7578 port
pxyws server (192.168.109.144) running at 7577 port
```

Figure 29 Running Mobius server platform

3.3. Test

3.3.1. oneM2MBrowser

Refer to the oneM2MBrowser manual from OCEAN webpage.

<http://iotocean.org/archives/module/onem2mbrowser>

3.3.2. Postman

The Mobius server accepts valid HTTP request in terms of POST, GET, PUT and DELETE request following oneM2M service primitives. In this session, we provide several examples to test against the installed Mobius server platform using Postman Rest Client (hereafter short for Postman). Postman is a Google Chrome extension which you can use to easily test Web API methods. It can also be used against 3rd party APIs. We use Postman to test Mobius APIs.

Users can install Postman through Google Chrome web browser. Go to the Chrome Settings option and it brings you to the Chrome Extension (plugins) page where there is a 'Get more extensions' option through which more extensions can be downloaded as shown as Figure 0.

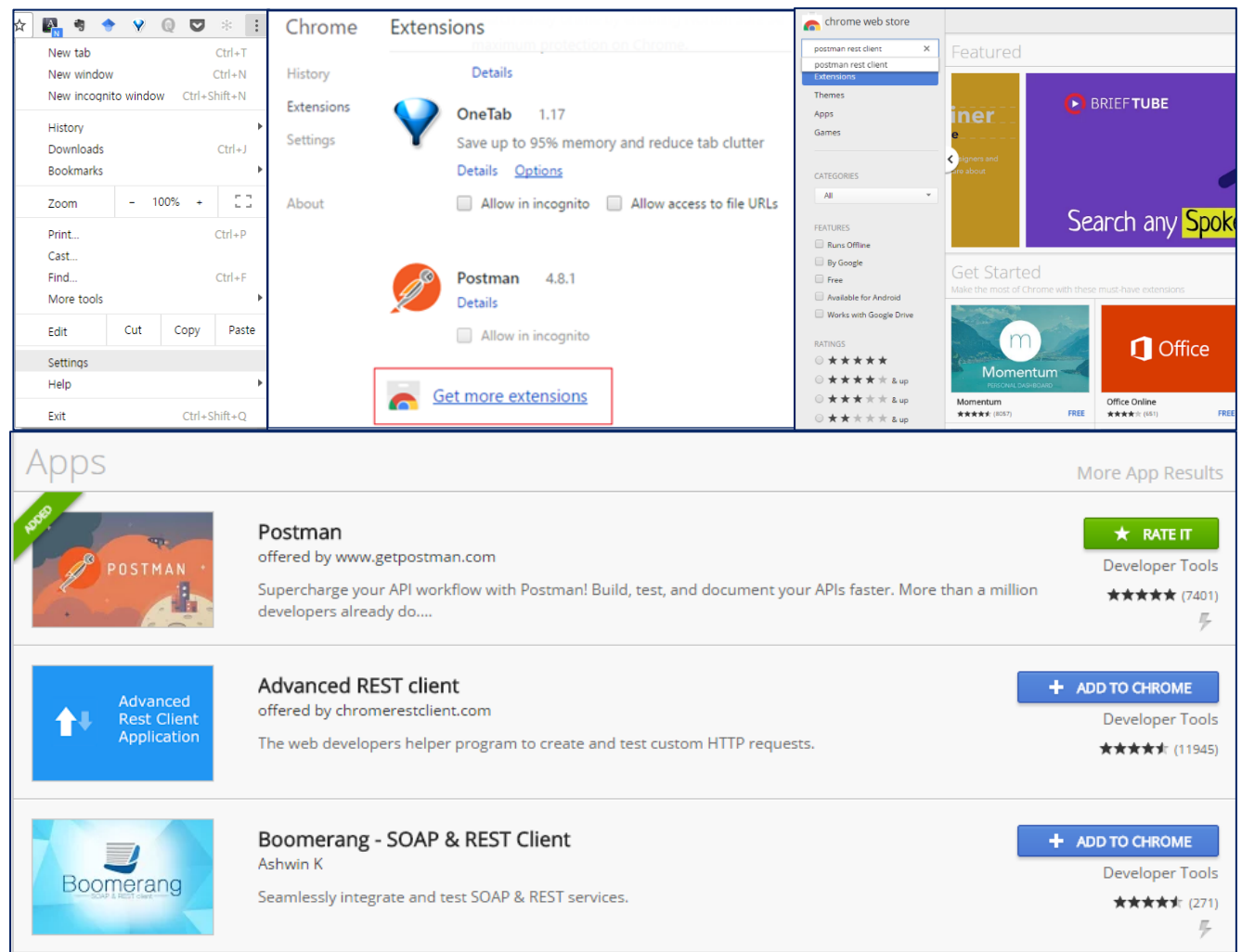


Figure 30 Download Postman REST Client

Postman provides an user interface to specify request headers and request body (if any). To simplify the test of Mobius API, we provide a collection of Postman script written for testing Mobius API. The Postman script is represented as a json file can be downloaded from OCEAN or GitHub. After download the Postman script, open the Postman and import the script as shown as Figure 9.

- ① In 'Collections' view, a group of Postman scripts are listed
- ② Click 'Import' button and
- ③ browse the downloaded script from the saved directory then the Postman script can be automatically added into the collections list.

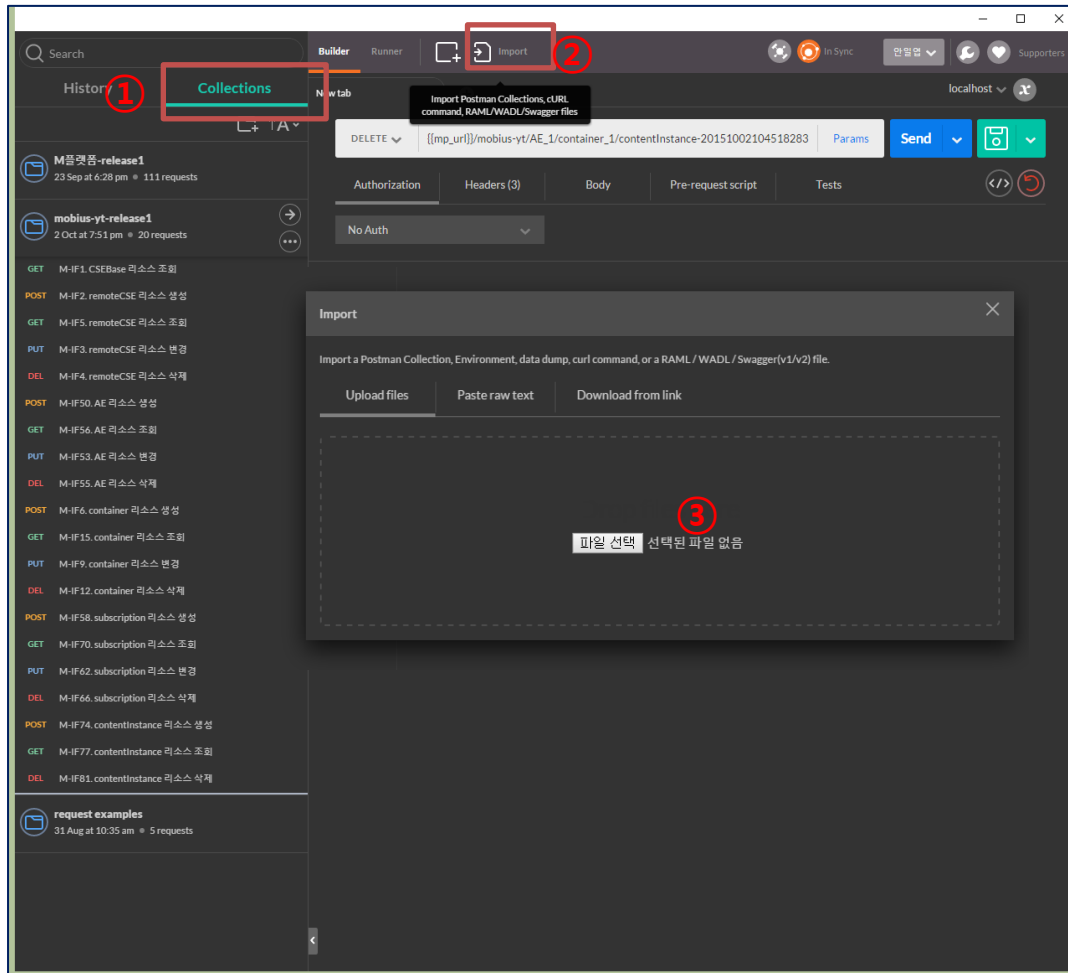



Figure 91 Import Postman Script

If the Postman script is imported successfully, a new collection will be displayed at the top of collections list shown as Figure 10. The script is designed to test Mobius-YT API in terms of oneM2M CREATE, RETRIEVE, UPDATE, and DELETE operations for oneM2M primitives represented in XML and JSON serializations.



Figure 10 A list of imported Postman Scripts

To access to the Mobius-YT server, the access configurations in terms of host and port as well as the CSEBase name for the Mobius-YT server need to be configured in Postman environment. To this end,

go to the setting  and select the 'Manage Environments' option in the option list then it pops a window where lists all existing configured environments as shown as Figure 3. User can edit existing or add a new environment by clicking 'Add' at the right bottom and in the popped window input Mobius server access configurations in terms of host and CSEBase name as shown as Figure .

- ① Name the new environment, e.g. Mobius_config
- ② {{mp_url}}: specify the host and port number of Mobius server
- ③ {{cb}}: specify CSEBase name of Mobius server
- ④ Apply the changes

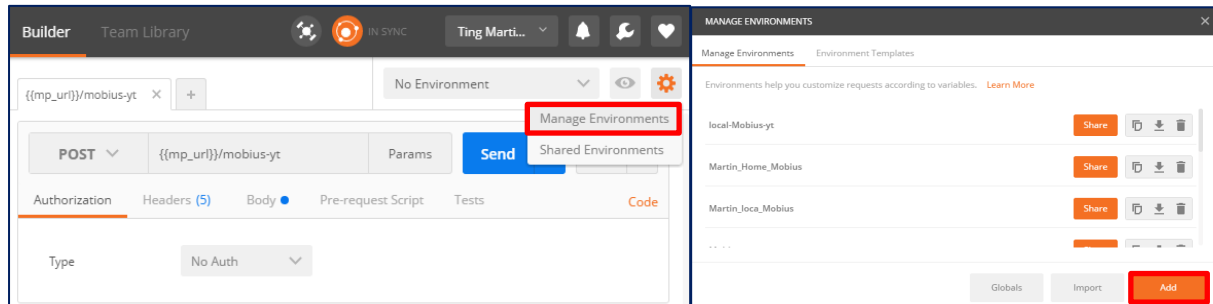


Figure 33 Manage Postman Environment

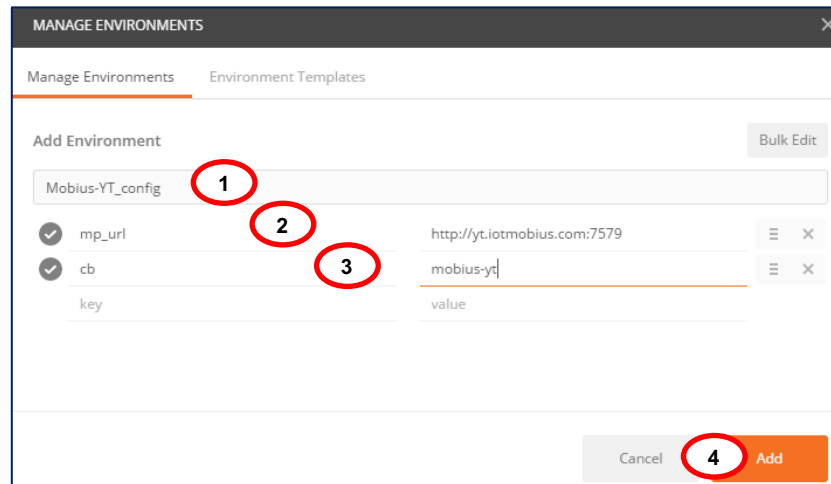


Figure 34 Add a new Environment with host and CSEBase information

Besides, there is another way to get the Postman environment, i.e. import directly an environment (POSTMAN Environment Script) that can be downloaded from OCEAN or GitHub.

After downloading the Postman environment script, import into Postman following steps as shown as Figure 35.

- ① Click 'Import' and then pops up a window to choose files
- ② Browse the downloaded Postman environment script from the saved directory
- ③ After imported successfully, you can see a new environment is added in the Manage Environments list

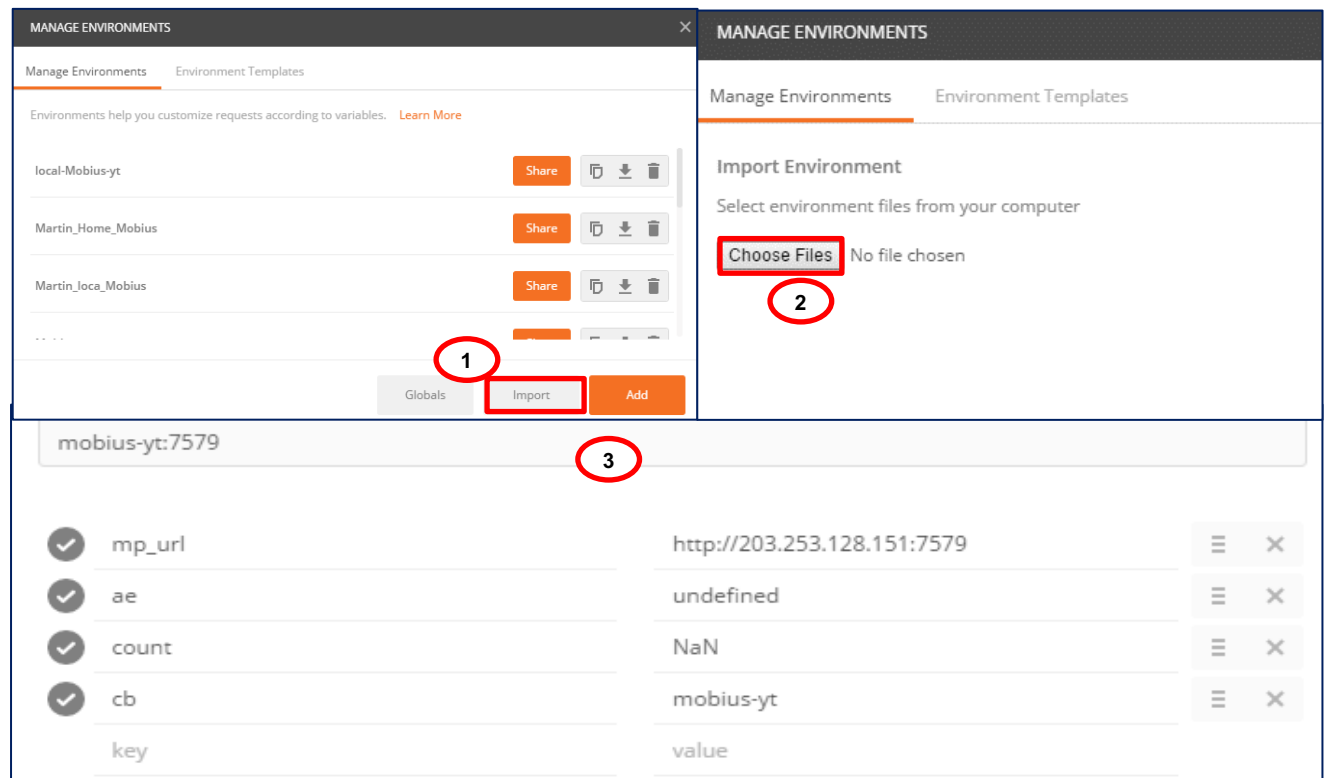


Figure 35 Import Postman Environment Script

To demonstrate how to use Postman to test Mobius API in case users don't know how to use Postman. If you are familiar with Postman, just skip this.

Here we demonstrate CSEBase RETRIEVE request which try to retrieve the CSEBase information from Mobius server. Mapping to REST request,

- ① To use HTTP GET method
- ② Specify the Mobius server access URL
- ③ Specify HTTP Headers (at least mandatory headers)
- ④ Send request to Mobius server
- ⑤ Check the response status code: 200 OK indicates request was accepted and response is returned successfully
- ⑥ Check the returned CSEBase information

TD_M2M_NH_01. CSEBase resource retrieve ▾

1 GET ▾ `{{mp_url}}/{{cb}}` 2 Params Send 4 Save ▾

Authorization Headers (3) Body Pre-request Script Tests Code

key	value
Accept	application/xml
X-M2M-RI	12345
X-M2M-Origin	50.2.481.1.1.232466

3 Bulk Edit Presets ▾

Body Cookies Headers (8) Tests 5 Status: 200 OK Time: 72 ms

Pretty Raw Preview XML ▾ Save Response

```

1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 <m2m:cb xmlns:m2m="http://www.onem2m.org/xml/protocols" xmlns:xsi="http://www.w3.org/2001
3 /XMLSchema-instance" rn="mobius-yt">
4   <ty>5</ty>
5   <ct>20160419T120301</ct>
6   <ri>/mobius-yt</ri>
7   <lt>20160419T120301</lt>
8   <lbl>mobius-yt</lbl>
9   <cst>1</cst>
10  <csi>/mobius-yt</csi>
11  <srt>1 2 3 4 10 16 23 23 24 25 26</srt>
12  <poa>http://203.253.128.151:7579 mqtt://203.253.128.151:mobius-yt</poa>
13 </m2m:cb>
  
```

6

Figure 36 Demo capture of CSEBase RETRIEVE Test

Users can test other scripts to deep understand the Mobius server API and we recommend to refer to Mobius REST Reference API User Guide to use and extend Mobius to apply with user-customized services.