



## **Interworking Proxy Entity(IPE): OCF**

**Software Version: 1.1.0**

**-Base on IoTivity 1.2.1**

---

## **Develop Guide v1.1**

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.

# Contents

- 1. Introduction.....4
  - 1.1. oneM2M OCF IPE.....4
  - 1.2. oneM2M OCF IPE Environment.....4
- 2. Development Environment.....5
  - 2.1. Download.....5
  - 2.2. Compilation.....6
  - 2.3. Execution.....8
  - 2.4. Notice: ..... 10

# 1.Introduction

## 1.1. oneM2M OCF IPE

oneM2M is an IoT server platform standard that supports devices as well as cloud server, while OCF(Open Connectivity Foundation) is an IoT service platform that supports devices. IPE(Interworking Proxy Entity) defined in oneM2M enables OCF device to work with oneM2M server side platform.

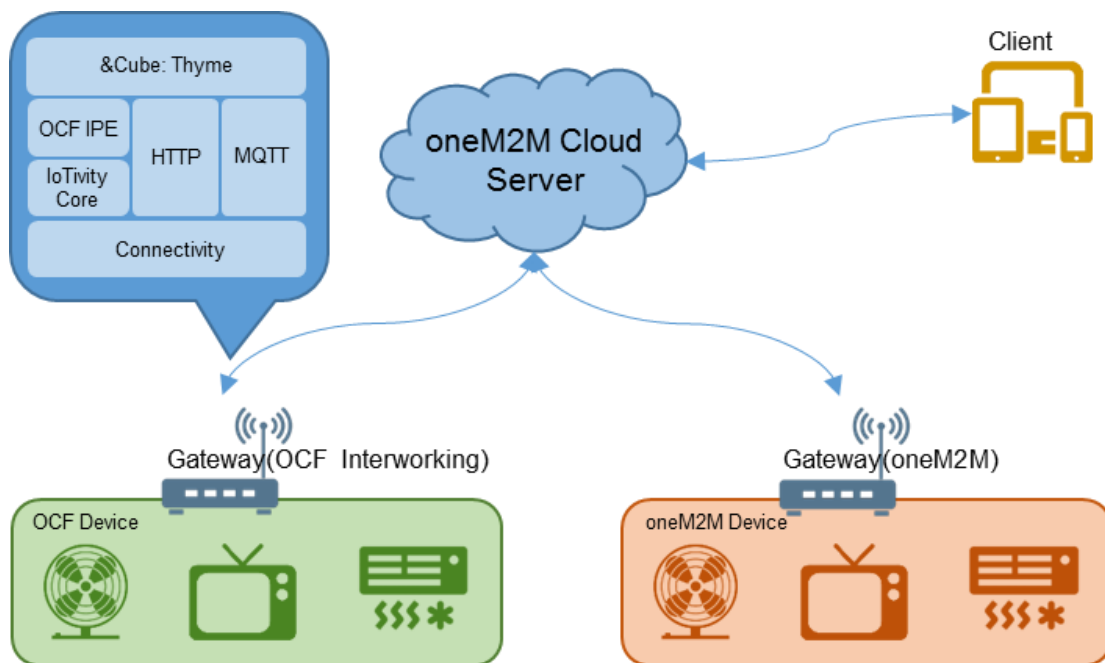


Figure 1 oneM2M OCF Interworking

## 1.2. oneM2M OCF IPE Environment

OCF IPE uses IoTivity source code as basis which is aligned with OCF standard. It is implemented in C++ and C. A gateway is required to run IoTivity core and oneM2M device platform which is implemented in Java or Node.js. For instance, IPE needs OS that can run JVM(Java Virtual Machine) or Node.js and IoTivity core that can run on Raspberry pi.

## 2. Development Environment Setup

### 2.1. Download

1. Download IoTivity open source from [IoTivity home page](https://www.iotivity.org).

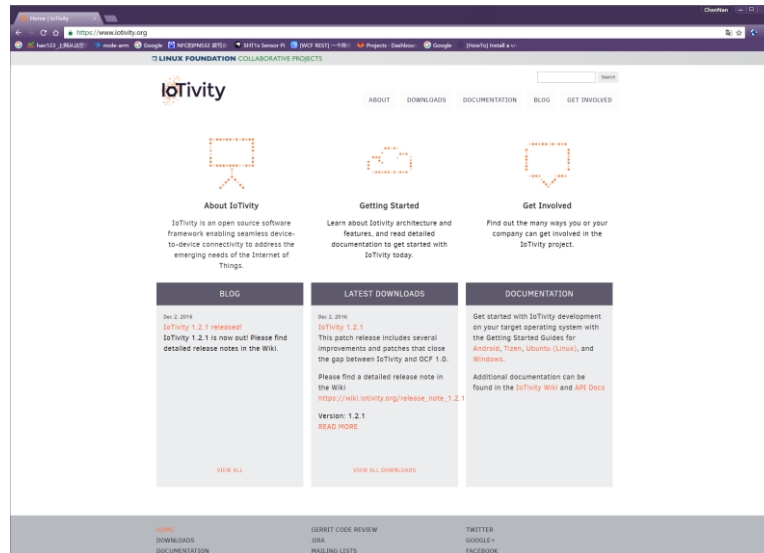


Figure 2 IoTivity home page

2. Download OCF IPE Sample source from [IoT Ocean home page](http://www.iot-ocean.org).

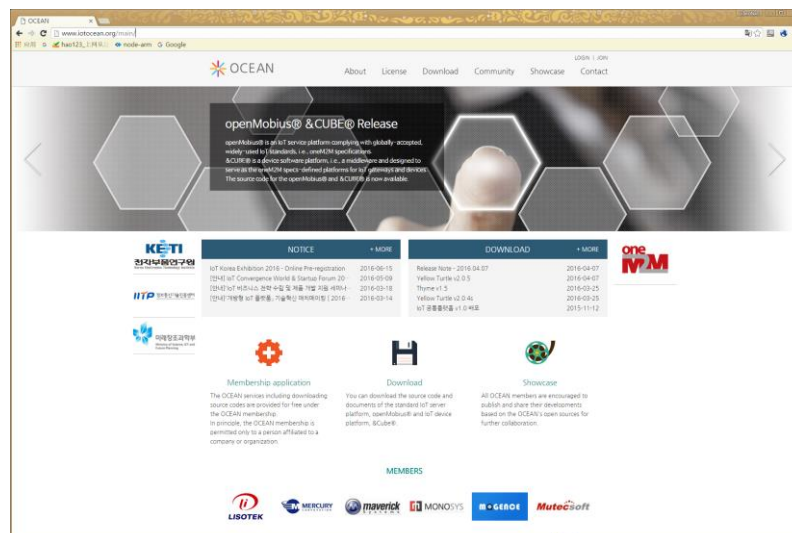


Figure 3 IoT Ocean home page

## 2.2. Compilation

1. Unzip OCF IPE sample file and then you will get a set of source files and libraries.

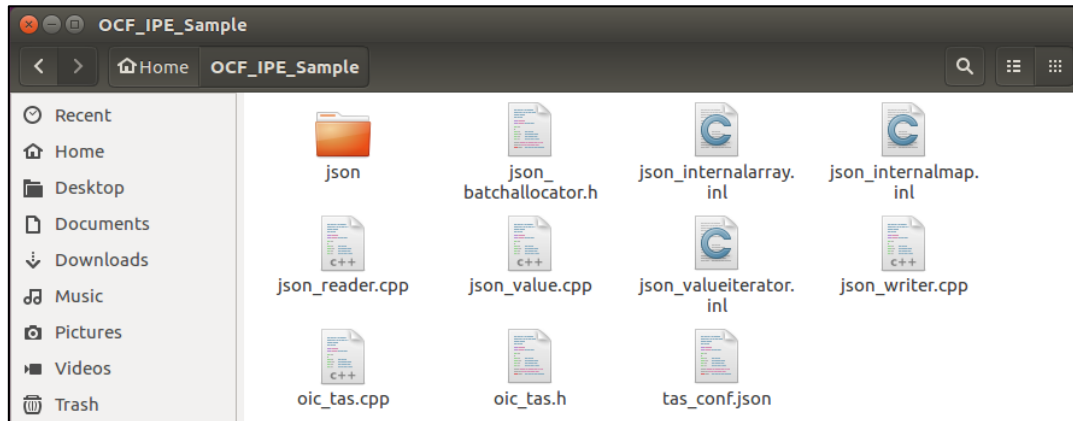


Figure 4 OCF IPE sample source file

2. Copy all sample source files to ".../<IoTivity home>/resource/examples" folder.

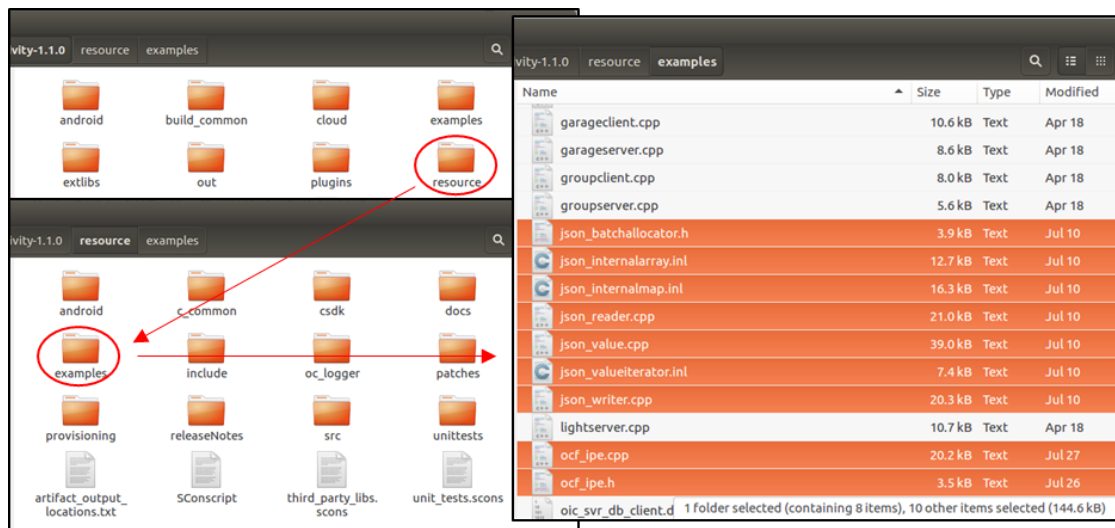


Figure 5 Copy OCF IPE sample to IoTivity

- Open SConscript file in ".../<IoTivity home>/resource/examples" folder and add the code below at "Source file and Target".

```
conf = examples_env.Install(examples_env.get('BUILD_DIR') + '/resource/examples/',
examples_env.get('SRC_DIR') + '/resource/examples/' + 'conf.json')
ocf_ipe = examples_env.Program('ocf_ipe', ['ocf_ipe.cpp', 'json_reader.cpp', 'json_value.cpp',
'json_writer.cpp'])
examples += [conf, ocf_ipe]
```

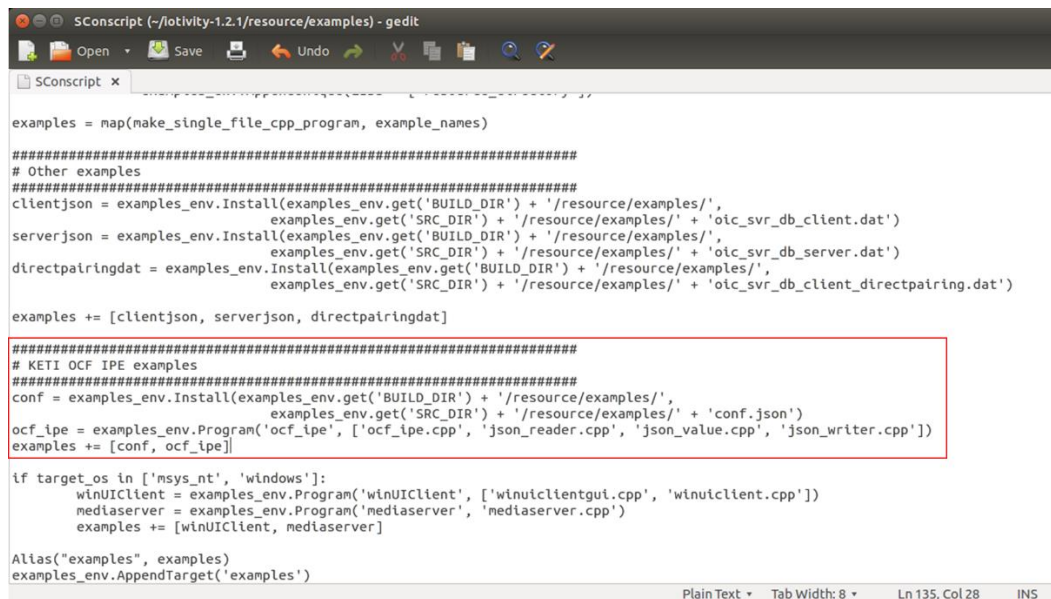


Figure 6 Add compilation script part

- Go to ".../<IoTivity home>" folder and compile the source with "scons" command line in Ubuntu Terminal.

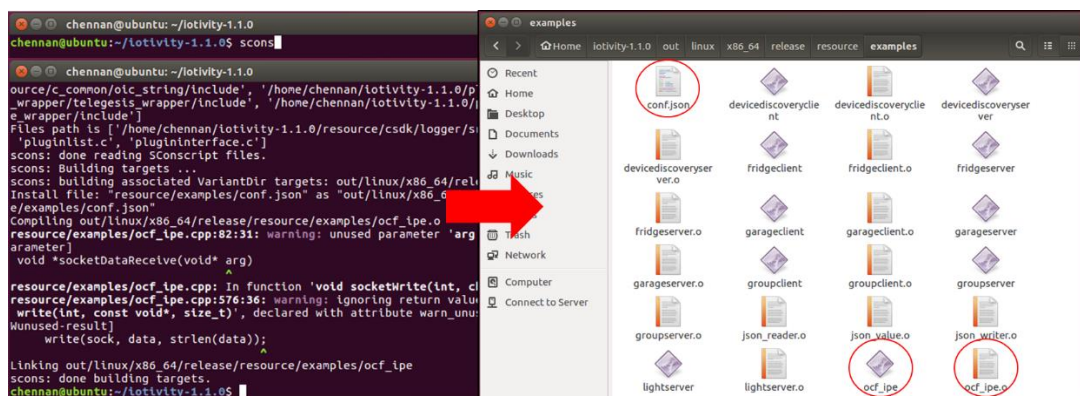


Figure 7 Compile result

## 2.3. Execution

In case of testing without real OCF device, we recommend to run [simpleserver] as a virtual OCF light device in ".../<loTivity home>/out/<target os>/<target arch>/release/resource/examples" folder.

OCF IPE sample code is designed to communicate with Mobius platform using Thyme(&Cube).

1. Activate a local Mobius platform if you don't want to use the KETI Mobius-yt.
2. Activate Thyme after editing configuration file as below.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<m2m:conf xmlns:m2m="http://www.onem2m.org/xml/protocols"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <cse>
    <cbhost>192.168.0.8</cbhost>
    <cbport>7579</cbport>
    <cbname>mobius-yt</cbname>
  </cse>
  <ae>
    <appos>linux</appos>
    <appid>0.2.481.1.0001.001.75797579</appid>
    <appname>oic</appname>
    <appport>80</appport>
    <appprotocol>xml</appprotocol>
    <tasport>7622</tasport>
    <cilimit>0</cilimit>
  </ae>
  <cnt>
    <ctname>light</ctname>
  </cnt>
  <cnt>
    <ctname>light_ctrl</ctname>
    <subname>control_sub</subname>
    <nu>mqtt://AUTOSSET</nu>
  </cnt>
</m2m:conf>
```

3. Activate the "simpleserver" in ".../<loTivity home>/out/linux/<target>/release/resource/examples" folder.
4. Activate ocf\_ipe in ".../<loTivity home>/out/linux/<target>/release/resource/example" folder.



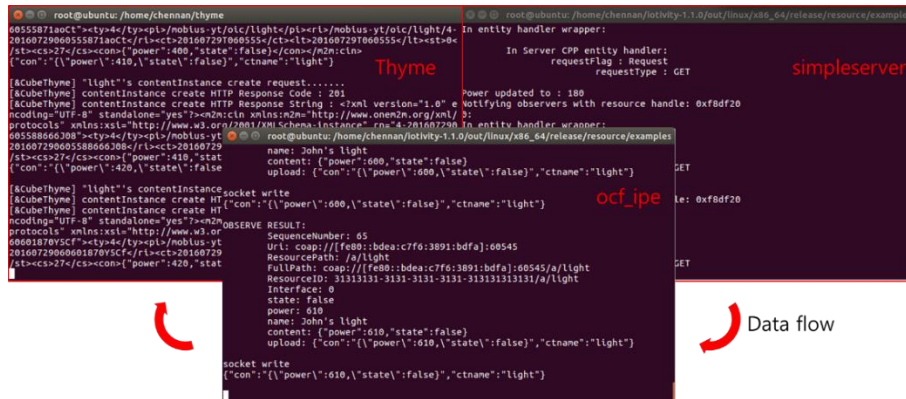


Figure 8 Active Result

- Retrieve virtual OCF device status with Mobius-yt API using postman.

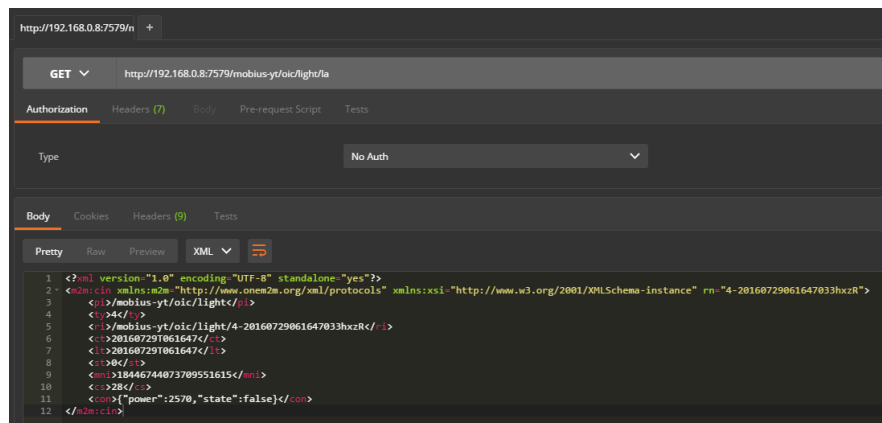


Figure 9 Retrieve Virtual OCF Device Status

- Control virtual OCF devices with Mobius-yt API using postman.

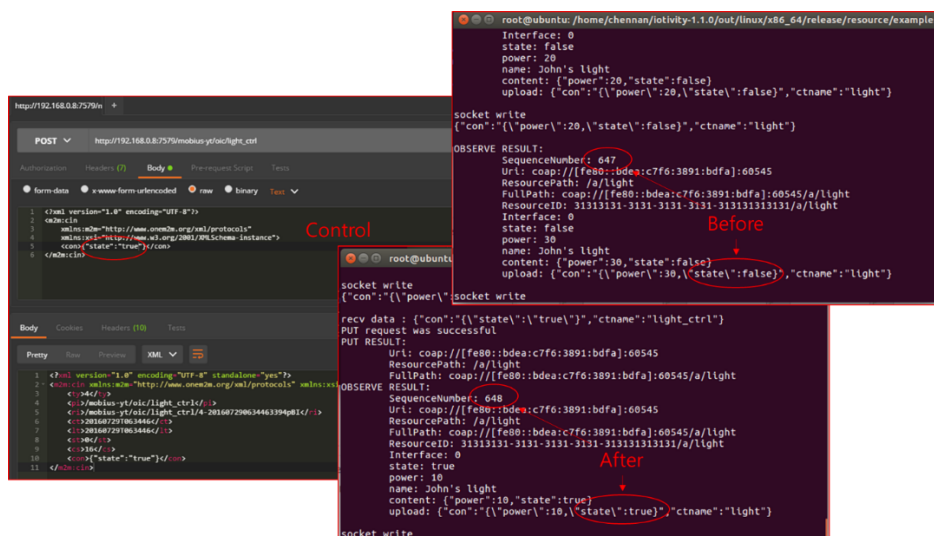


Figure 10 Control Virtual OCF Device

## 2.4. Notice

In this version(1.1.0), not all the OCF interfaces are supported by the OCF IPE. We provide implemented the OCF light interface as one sample for developers. If you have other devices with different interface types to interwork with a oneM2M platform, you need to implement corresponding codes in control part and sensing part of this OCF IPE.

In version(1.1.0), OCF IPE does not support all of OCF interfaces.