Table of Contents

• Technical Principles for an Internet of Things Ecosystem
• Introduction to the Open Connectivity Foundation
• OCF Specification Overview
  - Core Framework
  - Security
  - Bridging
  - Resource Type
  - OCF to AllJoyn Mapping
  - Smart Home Device Profile
TECHNICAL PRINCIPLES FOR AN INTERNET OF THINGS ECOSYSTEM
Scope of IoT

Vertical Profiles
- Smart Home
- Industrial
- Healthcare
- ... (indicated by ellipsis)

Baseline Functionality
- Group management
- ID & Addressing
- Protocol Bridge/GW
- Common Resource Model
- CRUDN
- Device management
- Messaging
- Streaming
- ... (indicated by ellipsis)

Connectivity
- Wi-Fi
- BT/BLE
- Thread
- ... (indicated by ellipsis)

Local Control
- Controller
- Cloud Interface
- Cloud Servers

Remote Control
- Controller
- Controller App
- Cloud Servers

Service #1 Domain

Service #2 Domain

Remote Access

JULY 24, 2017
COPYRIGHT © 2017 OPEN CONNECTIVITY FOUNDATION, INC. ALL RIGHTS RESERVED.
Approaches to definition of various Things

- By defining resources of things and its properties
  - *Fixed set of verbs (CRUDN) from transport layer will be used*
  - Resource model in RESTful Architecture
    (e.g., W3C, CSEP, etc.)

- By defining functions/operations of things
  - (Verbs + Objects)
  - RPC model
Support of Constrained Things
Class 2 Devices as Defined by RFC 7228

- Less overhead/ Less Traffic
  - Minimize CPU Load, Memory impacts, Traffic and Bandwidth
    - Compact header
    - Binary protocol
    - Compressed encoding of payload

- Low Complexity
  - Simple Resource Model
    - Short URI (Late Binding w/ resource type defined)
    - Broad and Shallow Hierarchy
Support of Multiple Verticals

- Legacy vertical services usually designed as silos
  → No common way to communicate among them

- A common platform provides a foundation for vertical services to collaborate and interwork by providing common services and data models
Conformance & Certification

- Conformance test - Each device proves conformance to specifications

- Certification Scope

- Mandatory (in spec, cert & committed in Open Source Project)
- Tested Optional Spec Features
- Optional Spec Features
- Optional Open Source Features
- Optional Open Source Features

Open Source

Specification

Certificate Issue & Logo Licensing

CERTIFIED
Licensing

• For Intellectual Property Rights (IPR) Policy: RAND-Z > RAND >> no IPR policy
• For Open Source: Apache 2.0 > Internet Systems Consortium (ISC)

• Due to the common nature of IoT connecting everything over the Internet, it’s most critical for manufacturers to avoid a licensing risk
  - Everything connected could be at potential risk

• Offering manufacturer-friendly Licensing and IPR Policy enables growth of market by attracting both start-ups and large enterprises; such an IPR policy must be clear and readily understandable ensuring that the terms are offered by all IP holders.
INTRODUCTION TO THE OPEN CONNECTIVITY FOUNDATION
Introduction to OCF - Optimized for IoT

- RESTful Architecture
- Common Platform
- Certification Program
- CoAP for Constrained Devices
- Best In Class Security
OCF Areas of Technology Development

• Core Architecture
  • Fundamental resource framework
  • Discovery
  • CRUDN

• Security

• Resource Models (vertical agnostic)

• Device Profiles
  • Smart Home
  • Health
  • Automotive

• Transport Binds
OCF Key Concepts (1/2)

- **Dedicated and optimized protocols for IoT (e.g. CoAP)**
  - Specific considerations for constrained devices
  - Fully compliant towards RESTful architecture
  - Built-in discovery and subscription mechanisms

- **Standards and Open Source to allow flexibility creating solutions**
  - Able to address all types of devices, form-factors, companies and markets with the widest possibility of options
  - Open Source is just one implementation to solve a problem
OCF Key Concepts (2/2)

- **Certification testing for interoperability**
  - Formal conformance testing for device validation to specifications
  - Plugfest testing for product interoperability

- **Certification and Logo program**
  - Products with the OCF Logo ensure OCF specifications are met
  - Logo reflects being part of an ecosystem of interoperable products
OCF Specification Overview
OCF Deliverables

**Normative Specifications**
- See next slide

**Resource Models via oneIoTa**
- Domain agnostic resources
- Derived models for Ecosystem Mapping
  - To date: OCF-AllJoyn (CDM 16.4)

**Certification Procedures**
- Test Policy (Certification Procedure Requirements Document)
- Test Plans and Test Cases (Certification Test Requirements Document)
Specification Structure

Infrastructure

• Core Framework
• Security
• Bridging
• Device Specification

Resource Model

• Resource Specification (reflects OneIoTa content)
• OCF Resource to AllJoyn Interface Mapping Specification (reflects OneIoTa content)
Where can I find the specifications and Resource Type definitions?

OCF Specifications:

- [https://openconnectivity.org/developer/specifications](https://openconnectivity.org/developer/specifications)

Resource Type Definitions

- Core Resources: [https://github.com/openconnectivityfoundation/core](https://github.com/openconnectivityfoundation/core)
- Bridging Resources: [https://github.com/openconnectivityfoundation/bridging](https://github.com/openconnectivityfoundation/bridging)
- Security Resources: [https://github.com/openconnectivityfoundation/security-models](https://github.com/openconnectivityfoundation/security-models)
- Vertical Resources and Derived Models: [https://oneiota.org/documents?filter%5Bmedia_type%5D=application%2Byaml](https://oneiota.org/documents?filter%5Bmedia_type%5D=application%2Byaml)
OneloTa Tool

- Web based (see: http://oneiota.org) development tool
- Supports RAML, JSON, and Swagger2.0 syntax
- Populated to date with all OCF Resources, Swagger2.0 versions of all such Resources, and OCF-AllJoyn derived models.
- Supports multiple organizations
  - Each submitting organization defines their own license terms
Core Framework Objectives

- Core Framework Specification Scope
  - Specifies the technical specification(s) comprising of the core architectural framework, messaging, interfaces and protocols based on approved use-case scenarios
  - Enables the development of vertical profiles (e.g. Smart Home) on top of the core while maintaining fundamental interoperability

- Architect a core framework that is scalable from resource constrained devices to resource rich devices

- Reuse open standards solutions (e.g. IETF) where they exist

- Ensure alignment with Iotivity open source releases
**RESTful Architecture**

**REST Architecture Style**

- Addressable resources
- A uniform, constrained interface
- Representation based manipulation
- Communicate statelessly
- Hypermedia State Engine

**RESTful Architecture (Representational State Transfer)**

- Resource based operation
  - Real world ‘entity’ is represented as ‘Resource’
- Resource manipulation via Request/Response: CRUDN

**Diagram**

- Client
- Server
- Request
- Response
- CRUD & N operation

```
{  
"n": "myRoomTemperature",
"rt": "oic.r.temperature",
"if": "oic.if.a",
"id": "example_id_xyz",
"temperature": 23,
"units": "C",
"setValue": 25
}
```
Current OCF Architecture defines 2 logical roles that devices can take:
- **OCF Server**: A logical entity that exposes hosted resources, is discoverable, and responds to client initiated transactions.
- **OCF Client**: A logical entity that interacts with resources on an OCF Server via discovery and CRUDN actions.

An OCF Device implements one or both roles.
Organization of an OCF Device

- OCF Device concept

```
Physical Platform

OCF Device 1
/oic/p
/oic/d
/oic/res

OCF Device 2
/oic/p
/oic/d
/oic/res

Resource URI: /oic/p
- rt: oic.wk.p
- if: oic.if.r
- n: homePlatform
- pi: at1908
- mmn: Samsung
```
Device example: light device (oic.d.light)

- Example overview
  - Smart light device with i) binary switch & ii) brightness resource

- Device type: Light device (oic.d.light) [Defined by the domain]

- Associated resources
  - Mandatory Core resources: oic/res, oic/p, oic/d
  - Mandatory Security Resources (not shown in the diagram)
  - Device specific resources: Binary switch (oic.r.switch.binary),
  - Other optional resources can be exposed, in this example Brightness resource (oic.r.light.brightness)

Example: Smart light device

<table>
<thead>
<tr>
<th>Device Title</th>
<th>Device Type</th>
<th>Associated Resource Type</th>
<th>M/O</th>
</tr>
</thead>
<tbody>
<tr>
<td>Light</td>
<td>oic.d.light</td>
<td>oic/res (oic.wk.res)</td>
<td>M</td>
</tr>
<tr>
<td></td>
<td></td>
<td>oic/p (oic.wk.p)</td>
<td>M</td>
</tr>
<tr>
<td></td>
<td></td>
<td>oic/d (oic.d.light)</td>
<td>M</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Binary switch (oic.r.switch.binary)</td>
<td>M</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Brightness (oic.r.light.brightness)</td>
<td>O</td>
</tr>
</tbody>
</table>
OCF Spec Features - Core Framework Spec

1. **Discovery**: Common method for device discovery (IETF CoRE)
2. **Messaging**: Constrained device support as default (IETF CoAP) as well as protocol translation via bridges
3. **Common Resource Model**: Real world entities defined as data models (resources)
4. **CRUDN**: Simple Request/Response mechanism with Create, Retrieve, Update, Delete and Notify commands
5. **ID & Addressing**: OCF IDs and addressing for OCF entities (Devices, Clients, Servers, Resources)
6. **Protocol Bridge/GW**: Handled by the Bridging Spec with some implications on the Core

Security is fundamental to the OCF ecosystem and applies to all elements
OCF Core Framework Basic Operation

**Discovery**
- Discover access policies, device info and resources on the devices

**Operation**
- Get device information by retrieving resources
- Control devices by changing resources
- Observe change on the properties of resources

**Basic common services**
- Device Monitoring
- Maintenance (e.g., reboot, factory reset, etc.)
Protocol Stack

Application

Resource Model

Encoding (CBOR)

CoAP

DTLS

UDP

IPv6

L2 Connectivity

OCF Stack
End point Discovery (CoAP Discovery)

• OCF devices make use of CoAP Discovery using IANA defined OCF Service Address (not the default CoAP address).

• Multicast RETRIEVE (CoAP GET) sent to well known URI /oic/res

• Response is an array of links; each link represents a Resource hosted by the responding server

• Links provide:
  • href
  • Relationship (self link, hosted link, bridged link)
  • Endpoint binds
  • Supported interfaces
  • Observability of the Resource
Everything in OCF is a Resource.

All Resources are specified using JSON schema plus RAML to define the associated API

OCF has mandated CBOR as the default encoding scheme on the wire

<table>
<thead>
<tr>
<th>Encoding Schemes - CBOR</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Standard</strong></th>
<th>CBOR</th>
<th>JSON</th>
<th>XML/EXI</th>
</tr>
</thead>
<tbody>
<tr>
<td>IETF RFC 7049</td>
<td></td>
<td>IETF RFC 7159</td>
<td>W3C Efficient XML Interchange Format 1.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>ContentType</strong></th>
<th>CBOR</th>
<th>JSON</th>
<th>XML/EXI</th>
</tr>
</thead>
<tbody>
<tr>
<td>/application/vnd.ocf+cbor</td>
<td></td>
<td>/application/json</td>
<td>/application/exi</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>OCFM/O</strong></th>
<th>CBOR</th>
<th>JSON</th>
<th>XML/EXI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mandatory</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Collection Resources

- An OCF Resource that contains one or more references (specified as OCF Links) to other OCF Resources is an OCF Collection.
- An OCF Link embraces and extends typed “web links” as specified in RFC 5988.
- The primary example of a collection is /oic/res (Discovery Resource).
  - A small number of Resources in the Resource Model are collections.
Introspection

• Why
  • On par with existing AllJoyn framework

• What
  • Device description is available on the network
  • Device description:
    • List all end points
    • Per end point
      – Which method are implemented
        » Query parameters per method
        » Payloads definitions (request and response)

• How
  • Put the data described in RAML and JSON on wire as a CBOR encoded Swagger2.0 document.
    • Describes the payload on JSON level
      – Property names
      – Type
      – range
Introspection: Goal

- Leave the current way of working intact: e.g. use RAML+JSON as is: use it as input for the swagger definition that will go on the wire.

- Same restrictions as already investigated and part of the:
  - 1 file to be transferred: e.g. definition should include
    - All end points, methods, query parameters, payload definitions
    - Same kind of negotiation to download the file

- Only this time it will be a swagger2.0 file.
• Definition
  • An (OCF) Endpoint is defined as the source or destination of a request and response messages for a given Transport Protocol Suites (e.g. CoAP over UDP over IPv6). The specific definition of an Endpoint depends on the Transport Protocol Suites being used.
    • (e.g.) For CoAP/UDP/IPv6, Endpoint is identified as IP address + port number.

• Endpoint characteristics for OCF Device
  • Each OCF Device shall associate with at least one Endpoint with which it can exchange Request & Response messages.
    • When a message is sent to an Endpoint, it shall be delivered to the OCF Device which is associated with the Endpoint. When a Request message is delivered to an Endpoint, path component is enough to locate the target Resource.
  • OCF Device can be associated with multiple Endpoints.
    • E.g. OCF Device may support both CoAP & HTTP
  • An endpoint can be shared among multiple OCF Devices, only when there is a way to clearly indicate the target Resource with Request URI.
## Endpoint Information

- Endpoint Information
  - Endpoint is identified with Endpoint Information which consists of
    - 1) **ep** for Transport Protocol Suite + Endpoint locator and 2) **pri** for priority.

<table>
<thead>
<tr>
<th>Transport Protocol Suites</th>
<th>scheme</th>
<th>Endpoint Locator</th>
<th>&quot;ep&quot; Value example</th>
</tr>
</thead>
<tbody>
<tr>
<td>coap + udp + ip</td>
<td>coap</td>
<td>IP address + port number</td>
<td>coap://[fe80::b1d6]:1111</td>
</tr>
<tr>
<td>coaps + udp + ip</td>
<td>coaps</td>
<td>IP address + port number</td>
<td>coaps://[fe80::b1d6]:1122</td>
</tr>
<tr>
<td>coap + tcp + ip</td>
<td>coap+tcp</td>
<td>IP address + port number</td>
<td>coap+tcp://[2001:db8:a::123]:2222</td>
</tr>
<tr>
<td>coaps + tcp + ip</td>
<td>coaps+tcp</td>
<td>IP address + port number</td>
<td>coaps+tcp://[2001:db8:a::123]:2233</td>
</tr>
<tr>
<td>http + tcp + ip</td>
<td>http</td>
<td>IP address + port number</td>
<td>http://[2001:db8:a::123]:1111</td>
</tr>
<tr>
<td>https + tcp + ip</td>
<td>https</td>
<td>IP address + port number</td>
<td>https://[2001:db8:a::123]:1122</td>
</tr>
</tbody>
</table>
eps Parameter for Endpoint Information

• a new Parameter "eps" to embed Endpoint Information in Link
  • "eps" has an array of items as its value and each item represents Endpoint information with two key-value pairs, "ep" and "pri", of which "ep" is mandatory and "pri" is optional.

```json
{
    "anchor": "ocf://light_device_id",
    "href": "/myLightSwitch",
    "rt": ["oic.r.switch.binary"],
    "if": ["oic.if.a", "oic.if.baseline"],
    "p": {"bm": 3},
    "eps": [{"ep": "coap://[fe80::b1d6]:1111", "pri": 2}, {"ep": "coaps://[fe80::b1d6]:1122"}]
}
```

• "anchor" represents the hosting OCF Device, "href", target Resource and "eps" the two Endpoints for the target Resource.
• If the target Resource of a Link requires a secure connection (e.g. CoAPS), "eps" Parameter shall be used to indicate the necessary information (e.g. port number)
Endpoint information in /oic/res with “eps” Parameter

```
[
    {"href": "/oic/res",
     "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989/oic/res",
     "rel": "self",
     "rt": ["oic.wk.res"],
     "if": ["oic.if.ll", "oic.if.baseline"],
     "p": {"bm": 3},
     "eps": [{"ep": "coaps://[fe80::b1d6]:44444"}] },
    {"href": "/oic/p",
     "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
     "rt": ["oic.wk.p"],
     "if": ["oic.if.r", "oic.if.baseline"],
     "p": {"bm": 3},
     "eps": [{"ep": "coap://[fe80::b1d6]:44444"}, {"ep": "coaps://[fe80::b1d6]:11111"}] },
    {"href": "/oic/d",
     "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
     "rt": ["oic.wk.d", "oic.d.light"],
     "if": ["oic.if.r", "oic.if.baseline"],
     "p": {"bm": 3},
     "eps": [{"ep": "coap://[fe80::b1d6]:44444"}, {"ep": "coaps://[fe80::b1d6]:11111"}] },
    {"href": "/myLight",
     "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
     "rt": ["oic.r.switch.binary"],
     "if": ["oic.if.a", "oic.if.baseline"],
     "p": {"bm": 3},
     "eps": [{"ep": "coap://[fe80::b1d6]:44444"}, {"ep": "coaps://[fe80::b1d6]:11111"}] }
]
```
**Versioning**

**Payload Versioning**

- **Purpose**: client and server can understand each others payload.
- **Method**: resource model & encoding information in CoAP header

**Device Level Versioning**

- **Purpose**: OCF devices can be aware of each others version
- **Method**: icv (spec version), dmv (data model version) in /oic/d resource
Payload versioning

### Content-Formats

<table>
<thead>
<tr>
<th>Media Type</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>application/cbor</td>
<td>60</td>
</tr>
<tr>
<td>application/vnd.ocf+cbor</td>
<td>10000</td>
</tr>
</tbody>
</table>

### Option Numbers

<table>
<thead>
<tr>
<th>CoAP Option Number</th>
<th>Name</th>
<th>Format</th>
<th>Length (bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2049</td>
<td>Accept Version</td>
<td>uint</td>
<td>2</td>
</tr>
<tr>
<td>2053</td>
<td>Content-Format Version</td>
<td>uint</td>
<td>2</td>
</tr>
</tbody>
</table>

### Version Representation

<table>
<thead>
<tr>
<th>Major Version</th>
<th>Minor Version</th>
<th>Sub Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blt</td>
<td>15 14 13 12 11 10 9 8 7 6</td>
<td>5 4 3 2 1 0</td>
</tr>
</tbody>
</table>

### Version Example

<table>
<thead>
<tr>
<th>OCF version</th>
<th>Binary representation</th>
<th>Integer value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0.0</td>
<td>0000 1000 0000 0000</td>
<td>2048</td>
</tr>
<tr>
<td>1.1.0</td>
<td>0000 1000 0100 0000</td>
<td>2112</td>
</tr>
</tbody>
</table>
Payload Versioning Use Case & Policies

**Round 1**
- **OCF 1.0 Client** requests `/oic/res` with `application/cbor`.
- **OIC 1.1 Server** responds with `/oic/res` per OIC 1.1.

**Round 2**
- **OCF 1.0 Client** requests `/oic/res` with `application/cbor`.
- **OIC 1.1 Server** responds with `/oic/res` per OIC 1.1.
- **OCF 1.0 Server** returns an error.
- **OCF 1.0 Client** requests `/oic/res` with `application/vnd.ocf+cbor`.
- **OIC 1.1 Server** responds with `/oic/res` per OIC 1.1.

COPYRIGHT © 2017 OPEN CONNECTIVITY FOUNDATION, INC. ALL RIGHTS RESERVED.
Basic CoAP messages work well for the small payloads we expect from light-weight, constrained IoT devices.

It is envisioned whereby an application will need to transfer larger payloads.

CoAP block wise transfer as defined in IETF RFC 7959 shall be used by all OCF Servers that receive a retrieve request for a content payload that would exceed the size of a CoAP datagram.
Infrastructure Connectivity

- Current scope is very much addressing proximal network
- Active project activity to leverage native OCF capabilities for wider area (beyond the LAN, be it Cloud or other) connectivity:
  - Native CoAP
  - Resource Directory
  - Resource Host
  - Add Pub-Sub pattern to already supported Observe pattern
Defining OCF Components (on top of CORE)

- **OCF Servers**
  - Defined by device identifier: standardized name of the device
  - List of mandatory OCF Resource Types per device
  - Note that OCF Clients are implicitly specified as “opposite” side of an OCF Server.
    - Currently OCF does not impose interaction sequences.
    - All instances of a Resource Type are allowed to talk to/from any OCF Client at any point in time

- **OCF Resource Type**
  - Defined by resource identifier: standardized name of the resource
  - List of mandatory properties per Resource Type
  - List of allowed actions (read/readwrite/..) per Resource Type
  - All OCF Resource Type IDs are IANA registered: [http://www.iana.org/assignments/core-parameters/core-parameters.xhtml](http://www.iana.org/assignments/core-parameters/core-parameters.xhtml)
Vendor extensions

- Vendor is allowed to:
  - Create their own defined (non-OCF standardized) Resource Types
  - Create their own defined (non-OCF standardized) Device Types
  - Extend existing devices with additional (not mandated) Resource Types
    - With standardized resource types
    - With vendor defined resource types
- All vendor extensions follow an OCF defined naming scheme
INFRASTRUCTURE:  
SECURITY SPECIFICATION

Overview
OCF Security Summary

- OCF is concerned with
  - **Device Identity** (Immutable, Unique, Attestable)
  - **Onboarding** (including **Authentication, Authorization, & Auditing (AAA)**)
  - **Confidentiality** (Protect data and communications)
  - **Integrity** (Resources, device state, and transitions are all managed)
  - **Available** (not only at the device level but also secured so they don’t impact the networks within which they operate)
  - **Lifecycle Management** (Including secure software update and verifications mechanisms)
  - **Future Security** (Looking at credential types, algorithms, and adapting to changes in the security landscape as it relates to the security of OCF devices, now and in the future)

- OCF key management supports device protection and authentication
- OCF uses Access Control Lists (ACLs) to manage authorization
- Secure device ownership transfer helps prevent attacks when devices are added to the network
Security Principals

- **Resources**: a data structure that defines the types, data and interfaces of a device; each can be Created/Retrieved/Updated/Deleted or to which Notification can be set based on appropriate access control.

- **Access Control Entries (ACEs) and Access Control Lists (ACLs)** are entries and collections, respectively, of permissions granting one device access to a Resource.

- **Access Manager Service (AMS)** creates and verifies access control permissions.

- **Credential Management Service (CMS)** is the name and resource type for a device which is granted permission to create and manage security credentials.

- **Secure Virtual Resources (SVRs)** are special security resources with severely restricted permissions and access management.

- **Onboarding Tools (OBTs)** are trusted platforms that help bring OCF devices into the local network.
How OCF Security Protects Device Resources:

Server – Responding Device

/oic/d
/oic/light/3
Etc...

Application Resources

Security Layer

 acl(s)
Subject
Resource(s)
Permission

 service(s)
DeviceID
SvcType
CredID

 cred(s)
DeviceID
CredType
PrivateData

Session Layer

Connectivity Abstraction Layer

Request Access

Client – Requesting Device

1

2

Secure Channel

DTLS

Allow /Deny Resource Access

3

July 24, 2017

Copyright © 2017 Open Connectivity Foundation, Inc. All Rights Reserved.
Simplified Onboarding Sequence

• Unowned Device boots

• Discovery (unsecure)
  • DOXS sends multicast to discover unowned devices  no TLS
  • Unowned devices reply, including list of supported OTMs no TLS

• Ownership Transfer
  • DOXS selects and configures this OTM to the new device no TLS
  • DOXS & unowned Device perform OTM, inc. TLS handshake TLS
  • DOXS configs SVRs to authorize itself, CMS and AMS TLS
  • Device is now owned!

• Provisioning:
  • CMS provisions credentials, AMS provisions access policies TLS
  • Device is now provisioned and can commence normal operation

• Normal Operation! TLS or no TLS
  • Credentials and/or access policies can be updated by returning to Provisioning
Device Provisioning States

- **RESET**: Hard reset
  - Owned? No
  - Provisioned? No
  - Permitted operations: Device prepares SVRs for OTM

- **RFOTM**: Ready For Ownership Transfer Mechanism
  - Owned? No → Yes
  - Provisioned? No
  - Permitted operations: DOXS configures /doxm to claim ownership. Configures owners of SVRs & their credentials in /cred

- **RFPRO**: Ready For Provisioning
  - Owned? Yes
  - Provisioned? No/Partly → Yes
  - Permitted operations: CMS updates /cred, AMS updates /acl2

- **RFNOP**: Ready For Normal Operation
  - Owned? Yes
  - Provisioned? Yes
  - Permitted operations: Normal Device-to-Device Operation subject to SVRs

During RFNOP, there are two services available to the device serving lifecycle management functions:

1: Secure Check for Software Update Availability
2: Trigger Secure Software Update

Device can transition to **RESET** from any state (these transitions are not shown)
Credential Management

- OCF devices can support the use of both symmetric and asymmetric credentials for establishing secure communication
  - Symmetric Key is mandatory
  - Certificates public/private keys are supported.

- Missing credentials could be procured from a CMS

- Credentials may have an expiration period
  - Expired credentials can be refreshed
Access Control

Is Light On?

Turn Light Off

Request → acl[0]

Accept

Request → acl[0]

Reject

subject: DeviceID

rsrc: [/a/light]

pms: R

subject: DeviceID

rsrc: [/a/light]

pms: R
Access Control

- Protect Resources of the OCF Server to control CRUDN access for entity requesting access
  - Any request to the OCF Server is subject to ACL (Access Control List) policy check
  - ACE (Access Control Entry) policy applies to a OCF Server hosted Resource
  - Each ACE has a permission which allows read or write operation

- Two type of access control mechanism are supported:
  - Subject-based access control (SBAC)
    - ACE specifies the identity of requestor
  - Role-based Access Control (RBAC)
    - ACE specifies the role to accept of the entity requesting access

- ACL can be changed/updated via the AMS
- ACL policies applies only at the OCF server side
OCF defines SVRs (Security Virtual Resource) to perform OCF security related functionality.

- **Device Ownership Transfer Resource (/oic/sec/doxm)** manage Device Ownership status.
- **Provisioning Resource (/oic/sec/pstat)** manage Device Provisioning status.
- **Credential Resource (/oic/sec/cred)** manages Device credentials.
  - Credential Resource is used for establishing secure communication.
  - Roles Resource (/oic/sec/roles) manage credentials based on the Role.
  - Certificate Signing Request Resource (/oic/sec/csr) is used to signed certificate by DOXS.
  - Security hardening applies to /oic/sec/cred Resource.

- **Access Control List (/oic/sec/acl)** manages the Access Control Entry for the Resource Server.
  - Access Manager ACL (/oic/sec/amacl) Resource specified an AMS to enforce ACL.
  - Signed ACL (/oic/sec/sacl) Resource to sign ACL policies.
## Security Virtual Resource (SVR)

- **oic.r.acl2 Resource**
  - aclist2
  - rowneruuid

- **oic.r.acl Resource**
  - aclist
  - rowneruuid

- **oic.r.aacl Resource**
  - resources

- **oic.r.creds Resource**
  - creds
  - rowneruuid

- **oic.r.doxm Resource**
  - oxm
  - oxmsel
  - sct
  - owned
  - deviceuuid
  - devowneruuid
  - rowneruuid

- **oic.r.pstat Resource**
  - dos
  - isop
  - cm
  - tm
  - om
  - sm
  - rowneruuid

- **oic.r.roles Resource**
  - roles

- **oic.r.crl Resource**
  - crlid
  - thisupdate
  - crldata
Message Integrity and Confidentiality

- Secured communications between clients and servers are protected against eavesdropping, tampering, and message replay.
- Unicast messages are secured using DTLS or TLS. Multicast messages are not secured.
- All secured communications are signed and encrypted.
- Communicating devices are required to authenticate each other. Communicating devices need to have useable credentials to talk to each other. If they are missing, the devices could contact the CMS to get them.
- The sending device encrypts and authenticates messages as defined by the selected cipher suite and the receiving device verifies and decrypts the messages.
- Secured unicast messages use the specified cipher suites during device ownership transfer and normal operation (for symmetric keys and asymmetric credentials).
INFRASTRUCTURE: BRIDGING SPECIFICATION

Overview
Bridging Specification

• Specifies a framework for bi-directional translation between devices in OCF and non-OCF ecosystems.

• Specifies general requirements for translation between OCF and non-OCF ecosystems
  • Requirements for resource discovery, message translation, security, and handling of multiple bridges.

• Specifies specific requirements for translation between OCF and AllJoyn ecosystems
  • Requirements for mapping core resources, propagating errors, and algorithmically translating custom resource types.
  • Refers to OCF to AllJoyn Mapping specification for translating well-known resource types.
OCF Bridge - Definition

- An OCF Bridge is a device that represents one or more non-OCF devices (bridged devices) as virtual OCF devices on the OCF network.
- The bridged devices themselves are out of the scope of OCF.
- The only difference between a ‘regular’ OCF device and a virtual OCF device is that the latter is encapsulated in an OCF Bridge device.
- An OCF Bridge device is indicated on the network with an “rt” of “oic.d.bridge”. When such a device is discovered, its discoverable resources would describe the devices that it bridges.
Bridging Concept – Bidirectional Operation

OCF Ecosystem

OCF Server

OCF Client

Partner Ecosystem

Partner Ecosystem Client

Partner Ecosystem Server

Bridge (oic.d.bridge)

OCF Bridging Security Filter & Mapping
Bridging Concept - Data Model & Security

OCF Resource Model

Derived Model

Partner Ecosystem Resource Model

Resource Spec
Device Spec

Bridging Spec
Mapping Spec
Security Spec

Other Ecosystem Spec
• OCF Bridge needs to be a trusted entity as it translates message payloads.

• OCF Bridge itself and all virtual devices that it exposes must be onboarded (transfer of ownership) and provisioned for secure operation.

• Each virtual device exposed by the OCF Bridge must implement the security requirements of the ecosystem that it is connected to.

• Bridging specifies mechanisms to selectively block communications between the OCF Bridge and OCF devices and between the OCF Bridge and bridged devices. This fine-grained control enables an administrator to control communications across ecosystems that may not have similar security capabilities.
RESOURCE MODEL: RESOURCE TYPE SPECIFICATION

Overview
Resource Specification

• List of reusable resources that are used in an OCF Device
  • Total of 74 Resource Types defined as of OCF 1.0
  • Uses core definitions

• Each resource definition contains:
  • unique identifier (rt)
  • Identification of the default interface and other supported interfaces
  • List supported methods
  • List per method the JSON schema defining the supported payload
  • Detailed list of the Property(-ies) the resource exposes

Resources are specified in RESTful API Modelling Language (RAML) and Swagger2.0
### Sample Set Defined Resource Types - OIC 1.1

<table>
<thead>
<tr>
<th>Resource Types</th>
<th>Use Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Air Flow</td>
<td>Indoor Environment Control</td>
</tr>
<tr>
<td>Air Flow Control</td>
<td></td>
</tr>
<tr>
<td>Battery</td>
<td>Device Control</td>
</tr>
<tr>
<td>Binary switch</td>
<td>Device Control</td>
</tr>
<tr>
<td>Brightness</td>
<td></td>
</tr>
<tr>
<td>Colour Chroma</td>
<td>Lighting Control</td>
</tr>
<tr>
<td>Colour RGB</td>
<td></td>
</tr>
<tr>
<td>Dimming</td>
<td></td>
</tr>
<tr>
<td>Door</td>
<td>Indoor Environment Control</td>
</tr>
<tr>
<td>Energy Consumption</td>
<td>Energy Management</td>
</tr>
<tr>
<td>Energy Usage</td>
<td></td>
</tr>
<tr>
<td>Humidity</td>
<td>Indoor Environment Control</td>
</tr>
<tr>
<td>Icemaker</td>
<td>Device Control</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Resource Types</th>
<th>Use Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lock</td>
<td>Keyless Entry</td>
</tr>
<tr>
<td>Lock Code</td>
<td></td>
</tr>
<tr>
<td>Mode</td>
<td></td>
</tr>
<tr>
<td>Open Level</td>
<td>Device Control</td>
</tr>
<tr>
<td>Operational State</td>
<td></td>
</tr>
<tr>
<td>Ramp Time</td>
<td>Lighting Control</td>
</tr>
<tr>
<td>Refrigeration</td>
<td>Device Control</td>
</tr>
<tr>
<td>Temperature</td>
<td>Indoor Environment Control</td>
</tr>
<tr>
<td>Time Period</td>
<td>Device Control</td>
</tr>
</tbody>
</table>
### Sample Set Defined Resource Types - OIC 1.1 (2/2)

<table>
<thead>
<tr>
<th>Resource Type</th>
<th>Use Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio</td>
<td>TV, Home Entertainment</td>
</tr>
<tr>
<td>Auto Focus</td>
<td>IP Camera</td>
</tr>
<tr>
<td>Auto White Balance</td>
<td>IP Camera</td>
</tr>
<tr>
<td>Automatic Document</td>
<td>Scanner Support</td>
</tr>
<tr>
<td>Feeder</td>
<td></td>
</tr>
<tr>
<td>Button</td>
<td>Device Control</td>
</tr>
<tr>
<td>Colour Saturation</td>
<td>IP Camera</td>
</tr>
<tr>
<td>DRLC</td>
<td>Smart Energy</td>
</tr>
<tr>
<td>Energy Overload</td>
<td>Smart Energy</td>
</tr>
<tr>
<td>Media</td>
<td>IP Camera</td>
</tr>
<tr>
<td>Media Source List</td>
<td>TV, Home Entertainment</td>
</tr>
<tr>
<td>Movement (Linear)</td>
<td>Robot Cleaner</td>
</tr>
<tr>
<td>Night Mode</td>
<td>IP Camera</td>
</tr>
<tr>
<td>PTZ</td>
<td>IP Camera</td>
</tr>
<tr>
<td>Signal Strength</td>
<td>Proximity</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sensor Resource Type</th>
<th>Use Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acceleration</td>
<td></td>
</tr>
<tr>
<td>Activity Count</td>
<td></td>
</tr>
<tr>
<td>Atmospheric Pressure</td>
<td></td>
</tr>
<tr>
<td>Carbon Dioxide</td>
<td></td>
</tr>
<tr>
<td>Carbon Monoxide</td>
<td></td>
</tr>
<tr>
<td>Contact</td>
<td></td>
</tr>
<tr>
<td>Glass Break</td>
<td></td>
</tr>
<tr>
<td>Heart Rate Zone</td>
<td></td>
</tr>
<tr>
<td>Illuminance</td>
<td></td>
</tr>
<tr>
<td>Magnetic Field Direction</td>
<td></td>
</tr>
<tr>
<td>Presence</td>
<td></td>
</tr>
<tr>
<td>Radiation (UV)</td>
<td></td>
</tr>
<tr>
<td>Sleep</td>
<td></td>
</tr>
<tr>
<td>Smoke</td>
<td></td>
</tr>
<tr>
<td>Three Axis</td>
<td></td>
</tr>
<tr>
<td>Touch</td>
<td></td>
</tr>
<tr>
<td>Water</td>
<td></td>
</tr>
</tbody>
</table>

Extended Sensor Set
(for a Generic Sensor Device)

See [https://oneiota.org](https://oneiota.org) for the complete set of OCF defined Resource Types
# New Resource Types - OCF 1.0

<table>
<thead>
<tr>
<th>Resource Type</th>
<th>Use Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Air Quality</td>
<td>Indoor Environment Control</td>
</tr>
<tr>
<td>Air Quality Collection</td>
<td>Indoor Environment Control</td>
</tr>
<tr>
<td>Consumable</td>
<td>Device Control</td>
</tr>
<tr>
<td>Consumable Collection</td>
<td>Device Control</td>
</tr>
<tr>
<td>Delay Defrost</td>
<td>Energy Star</td>
</tr>
<tr>
<td>Ecomode</td>
<td>Device Control</td>
</tr>
<tr>
<td>Heating Zone</td>
<td>Device Control</td>
</tr>
<tr>
<td>Heating Zone Collection</td>
<td>Device Control</td>
</tr>
<tr>
<td>Selectable Levels</td>
<td>Device Control</td>
</tr>
<tr>
<td>Value Conditional</td>
<td>Notifications</td>
</tr>
</tbody>
</table>

Resource Types are Conditionally Mandatory. If an OCF Server hosts an OCF known resource then it shall follow all normative requirements in the Resource Specification applicable to that Resource.
RESOURCE MODEL:
DERIVED MODELING – OCF TO ALLJOYN MAPPING

Overview
Models the interworking between OCF and AllJoyn


Predicated on OCF being the superset model; so any Device Types and Resource Types (as equivalents to AllJoyn interfaces) that were missing from OCF were defined in the equivalent OCF Specifications.

Defines the mapping in terms of:

- Device Type equivalency
- Resource <-> Interface equivalency
- Detailed Property by Property mapping on a per Interface Basis (Derived Models)
Derived models use standard JSON schema syntax. Fundamentally, derived models provide a conversion mapping between OCF data models and the data models in AllJoyn.

```json
"asa.environment.targethumidity": {
  "type": "object",
  "properties": {
    "targetvalue": {
      "type": "number",
      "description": "Measured value",
      "x-ocf-conversion": {
        "x-ocf-alias": "oic.r.humidity,oic.r.selectablelevels",
        "x-to-ocf": ["if minvalue != maxvalue, ocf.desiredhumidity = targetvalue; ocf.targetlevel = selectablehumiditylevels[0].; if minvalue == maxvalue, ocf.targetlevel = targetvalue."],
        "x-from-ocf": ["if x-ocf-alias == oic.r.humidity, targetvalue = desiredhumidity; if x-ocf-alias == oic.r.selectablelevels, targetvalue = targetlevel." ]
      }
    }
  }
}
```
### Device Type Equivalency

<table>
<thead>
<tr>
<th>Classification</th>
<th>ASA Device Type</th>
<th>OCF Device Type</th>
<th>OCF Device Type ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Air Care</td>
<td>Air Conditioner</td>
<td>Air Conditioner</td>
<td>oic.d.airconditioner</td>
</tr>
<tr>
<td></td>
<td>Air Purifier</td>
<td>Air Purifier</td>
<td>oic.d.airpurifier</td>
</tr>
<tr>
<td></td>
<td>Air Quality Monitor</td>
<td>Air Quality Monitor</td>
<td>oic.d.aqm</td>
</tr>
<tr>
<td></td>
<td>Dehumidifier</td>
<td>Dehumidifier</td>
<td>oic.d.dehumidifier</td>
</tr>
<tr>
<td></td>
<td>Humidifier</td>
<td>Humidifier</td>
<td>oic.d.humidifier</td>
</tr>
<tr>
<td></td>
<td>Electric Fan</td>
<td>Fan</td>
<td>oic.d.fan</td>
</tr>
<tr>
<td></td>
<td>Thermostat</td>
<td>Thermostat</td>
<td>oic.d.thermostat</td>
</tr>
<tr>
<td>Fabric Care</td>
<td>Clothes Washer</td>
<td>Washer</td>
<td>oic.d.washer</td>
</tr>
<tr>
<td></td>
<td>Clothes Dryer</td>
<td>Dryer</td>
<td>oic.d.dryer</td>
</tr>
<tr>
<td></td>
<td>Clothes Washer-Dryer</td>
<td>Washer-Dryer</td>
<td>oic.d.washerdryer</td>
</tr>
<tr>
<td>Food Preservation</td>
<td>Refrigerator</td>
<td>Refrigerator</td>
<td>oic.d.refrigerator</td>
</tr>
<tr>
<td></td>
<td>Ice Maker</td>
<td>Ice Maker (Resource)</td>
<td>oic.r.icemaker</td>
</tr>
<tr>
<td></td>
<td>Freezer</td>
<td>Freezer</td>
<td>oic.d.freezer</td>
</tr>
<tr>
<td>Food Preparation</td>
<td>Oven</td>
<td>Oven</td>
<td>oic.d.oven</td>
</tr>
<tr>
<td></td>
<td>Cooktop</td>
<td>Cooktop</td>
<td>oic.d.cooktop</td>
</tr>
<tr>
<td></td>
<td>Cookerhood</td>
<td>Cooker Hood</td>
<td>oic.d.cookerhood</td>
</tr>
<tr>
<td></td>
<td>Foodprobe</td>
<td>Food Probe</td>
<td>oic.d.foodprobe</td>
</tr>
<tr>
<td>Dish Care</td>
<td>Dishwasher</td>
<td>Dishwasher</td>
<td>oic.d.dishwasher</td>
</tr>
<tr>
<td>Floor Care</td>
<td>Robot Cleaner</td>
<td>Robot Cleaner</td>
<td>oic.d.robotcleaner</td>
</tr>
<tr>
<td>Entertainment</td>
<td>TV</td>
<td>Television</td>
<td>oic.d.tv</td>
</tr>
<tr>
<td></td>
<td>Set Top box (STB)</td>
<td>Set Top Box</td>
<td>oic.d.stb</td>
</tr>
</tbody>
</table>

- Yellow highlights identify Device Types that were added to support equivalency.
<table>
<thead>
<tr>
<th>AllJoyn Interface</th>
<th>OCF Resource Type Name</th>
<th>OCF Resource Type ID</th>
<th>OCF Interface(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Environment.CurrentAirQuality</td>
<td>Air Quality Collection</td>
<td>oic.r.airqualitycollection</td>
<td>oic.if.s</td>
</tr>
<tr>
<td>Environment.CurrentAirQualityLevel</td>
<td>Air Quality Collection</td>
<td>oic.r.airqualitycollection</td>
<td>oic.if.s</td>
</tr>
<tr>
<td>Environment.CurrentHumidity</td>
<td>Humidity</td>
<td>oic.r.humidity</td>
<td>oic.if.s</td>
</tr>
<tr>
<td>Environment.CurrentTemperature</td>
<td>Temperature</td>
<td>oic.r.temperature</td>
<td>oic.if.s</td>
</tr>
<tr>
<td>Environment.TargetHumidity</td>
<td>Humidity</td>
<td>oic.r.humidity,</td>
<td>oic.if.a</td>
</tr>
<tr>
<td>Environment.TargetTemperature</td>
<td>Temperature</td>
<td>oic.r.temperature</td>
<td>oic.if.a</td>
</tr>
<tr>
<td>Operation.AudioVolume</td>
<td>Audio Controls</td>
<td>oic.r.audio</td>
<td>oic.if.a</td>
</tr>
<tr>
<td>Operation.Channel</td>
<td>Not mapped</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operation.ClimateControlMode</td>
<td>Mode</td>
<td>oic.r.mode</td>
<td>oic.if.a</td>
</tr>
<tr>
<td>Operation.ClimateControlMode</td>
<td>Operational State</td>
<td>oic.r.operational.state</td>
<td>oic.if.s</td>
</tr>
<tr>
<td>Operation.ClosedStatus</td>
<td>Door</td>
<td>oic.r.door</td>
<td>oic.if.s</td>
</tr>
<tr>
<td>Operation.ClimateControlMode</td>
<td>Operational State</td>
<td>oic.r.operational.state</td>
<td>oic.if.s</td>
</tr>
<tr>
<td>Operation.FanSpeedLevel</td>
<td>Air Flow</td>
<td>oic.r.airflow</td>
<td>oic.if.a</td>
</tr>
<tr>
<td>Operation.HeatingZone</td>
<td>Heating Zone Collection</td>
<td>oic.r.heatingzonecollection</td>
<td>oic.if.s</td>
</tr>
<tr>
<td>Operation.HvacFanMode</td>
<td>Mode</td>
<td>oic.r.mode</td>
<td>oic.if.a</td>
</tr>
<tr>
<td>Operation.OnOffStatus</td>
<td>Binary Switch</td>
<td>oic.r.switch.binary</td>
<td>oic.if.s</td>
</tr>
<tr>
<td>Operation.OvenCyclePhase</td>
<td>Operational State</td>
<td>oic.r.operationalstate</td>
<td>oic.if.s</td>
</tr>
</tbody>
</table>
PER VERTICAL: SMART HOME DEVICE SPECIFICATION

Overview
Higher Layer Specifications

- Specifications are split into 2 documents:
  - Device specification (per vertical if needed)
  - Resource specification (vertical agnostic)

The Device specification uses the resources defined in the resource specification
Device Specification

- Contains profiles of
  - Core specification
  - Security specification

- Contains list of smart home devices

- Each Smart home device definition contains:
  - unique identifier (rt)
  - a list of mandatory resources

Exposure of an OCF Device Type is Mandatory. If an OCF Server hosts an OCF known device then it shall follow all normative requirements in the Device Specification applicable to that Device.
# Smart Home Device Type (1/2)

<table>
<thead>
<tr>
<th>Device Type</th>
<th>Minimum Resource Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Air Conditioner</td>
<td>Binary Switch, Temperature</td>
</tr>
<tr>
<td>Air Purifier</td>
<td>Binary Switch</td>
</tr>
<tr>
<td>Air Quality Monitor</td>
<td>Air Quality Collection</td>
</tr>
<tr>
<td>Blind</td>
<td>Open Level</td>
</tr>
<tr>
<td>Camera</td>
<td>Media</td>
</tr>
<tr>
<td>Clothes Dryer</td>
<td>Binary Switch, Mode</td>
</tr>
<tr>
<td>Clothes Washer</td>
<td>Binary Switch, Mode</td>
</tr>
<tr>
<td>Clothes Washer/Dryer</td>
<td>Binary Switch, Operational State</td>
</tr>
<tr>
<td>Cooker Hood</td>
<td>Airflow Control, Binary Switch, Mode</td>
</tr>
<tr>
<td>Cooktop</td>
<td>Heating Zone Collection</td>
</tr>
<tr>
<td>Dehumidifier</td>
<td>Binary Switch, Humidity</td>
</tr>
<tr>
<td>Dishwasher</td>
<td>Binary Switch, Mode</td>
</tr>
<tr>
<td>Door</td>
<td>Open Level</td>
</tr>
<tr>
<td>Fan</td>
<td>Binary Switch</td>
</tr>
<tr>
<td>Food Probe</td>
<td>Temperature</td>
</tr>
<tr>
<td>Freezer</td>
<td>Temperature (2)</td>
</tr>
<tr>
<td>Garage Door</td>
<td>Door</td>
</tr>
<tr>
<td>Generic Sensor</td>
<td>Sensor</td>
</tr>
<tr>
<td>Humidifier</td>
<td>Binary Switch</td>
</tr>
<tr>
<td>Light</td>
<td>Binary Switch</td>
</tr>
<tr>
<td>Oven</td>
<td>Binary Switch, Temperature (2)</td>
</tr>
<tr>
<td>Printer</td>
<td>Binary Switch, Operational State</td>
</tr>
<tr>
<td>Printer (Multi-Function)</td>
<td>Binary Switch, Operational State (2), Automatic Document Feeder</td>
</tr>
<tr>
<td>Receiver</td>
<td>Binary Switch, Audio Media Source List (2)</td>
</tr>
</tbody>
</table>
### Smart Home Device Type (2/2)

<table>
<thead>
<tr>
<th>Device Type</th>
<th>Minimum Resource Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refrigerator</td>
<td>Binary Switch, Refrigeration, Temperature (2)</td>
</tr>
<tr>
<td>Robot Cleaner</td>
<td>Binary Switch, Mode</td>
</tr>
<tr>
<td>Scanner</td>
<td>Binary Switch, Operational State, Automatic Document Feeder</td>
</tr>
<tr>
<td>Security Panel</td>
<td>Mode</td>
</tr>
<tr>
<td>Set Top Box</td>
<td>Binary Switch</td>
</tr>
<tr>
<td>Smart Lock</td>
<td>Lock Status</td>
</tr>
<tr>
<td>Smart Plug</td>
<td>Binary Switch</td>
</tr>
<tr>
<td>Switch</td>
<td>Binary Switch</td>
</tr>
<tr>
<td>Television</td>
<td>Binary Switch, Audio, Media Source List</td>
</tr>
<tr>
<td>Thermostat</td>
<td>Temperature (2)</td>
</tr>
<tr>
<td>Water Valve</td>
<td>Open Level</td>
</tr>
</tbody>
</table>
Thank you!

• Access the OCF specifications
  https://openconnectivity.org/developer/specifications

• Contact OCF at admin@openconnectivity.org