Nexus Channel

Open DC Grid Meeting
11 August 2020

What is Nexus Channel?

- An application layer for device <=> device communication on constrained hardware:
  - ... facilitating interoperability between devices/appliances produced by different manufacturers
  - ... agnostic to underlying link/transport layer
  - ... with optional application layer security which can be controlled by external platforms
    - ... which is critical for PAYG applications, among others

Requirements (from a PAYG perspective)

- PAYG “accessories” can be enabled/disabled according to the state of a PAYG “controller” device (usually an SHS).
- Accessories can be restricted to function only when linked to one specific controller.
- Communication between controllers and accessories is “secure”.
- Messages from the PAYG account platform to controllers can be encoded in existing communication types (namely, keycode/token).

The more general problems

- How can constrained devices communicate with each other in a secure way?
- How can devices from different manufacturers communicate in a shared language, i.e. be interoperable?*
    - *assuming they can communicate at all

Do we want to reinvent the wheel?
Is this applicable only to PAYG situations?

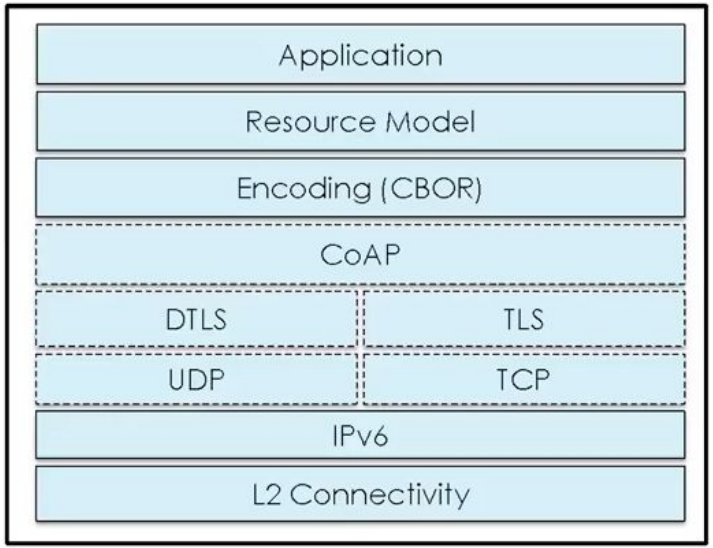
Let’s use existing open standards

ISO/IEC 30118-1:2018

Open Connectivity Foundation (OCF) Specification

(available at no cost on the OCF website!)

Let’s use existing open standards



OCF Stack

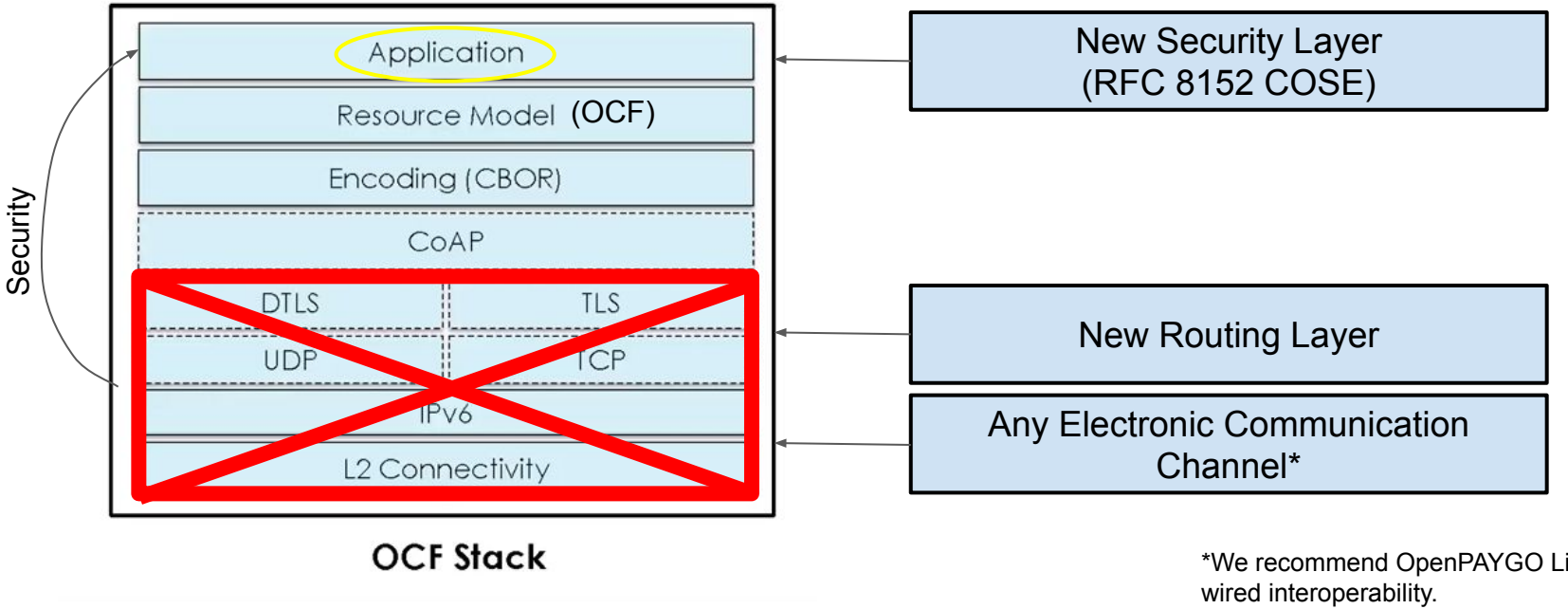
Constrained device classes

- RFC 7228

Name	Data size (e.g., RAM)	Code size (e.g., Flash)
~~Class 0, C0~~	~~<< 10 KiB~~	~~<< 100 KiB~~
Class 1, C1	~ 10 KiB	~ 100 KiB
Class 2, C2	~ 50 KiB	~ 250 KiB

Must accommodate (at a minimum) OS + Network stack + drivers + IoTivity-Constrained application

Let’s use existing open standards



*We recommend OpenPAYGO Link for wired interoperability.

Why do we need an application layer?

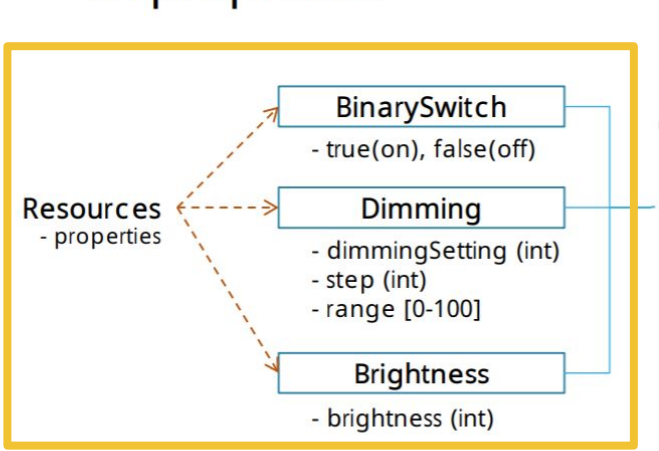
- Provide a common abstraction that works with multiple link layers
  - Ex: OpenPAYGO Link, Bluetooth, other low power local networks
- Provide security that may not be present at transport layer, consistently
- Leverage existing work and facilitate interoperability
  - Ex: Many resource models we need already exist, and Angaza is standardizing models specific to Nexus Channel security and PAYG applications

Applications

- Communicating PAYG state, i.e. enabled or disabled
- Collecting telemetry data from accessories
- Load balancing
- ... anything a device manufacturer wants!

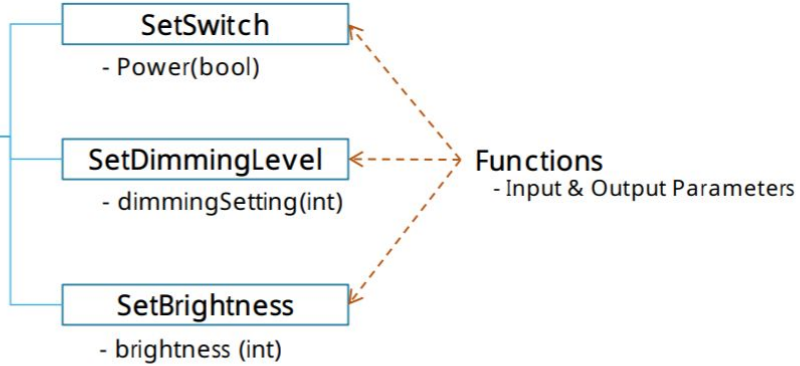
Approaches to definition of various Things

- By defining resources of things and its properties



e.g., Light bulb

- By defining functions/operations of things



- (no Verbs) + Objects
  *Fixed set of verbs (CRUDN) from transport layer will be used
- Resource model in RESTful Architecture (e.g., W3C, CSEP, etc.)

- (Verbs + Objects)
- RPC model

Resources

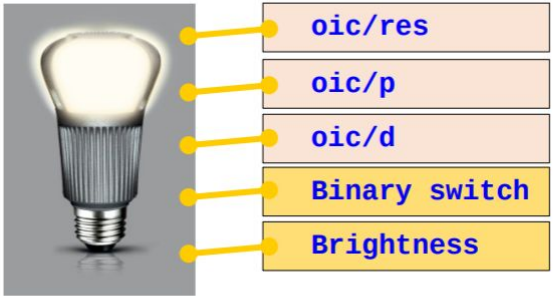
- A device model contains one or more Resources to describe a real world entity
- Each Resource contains Properties that describes an aspect that is exposed through a Resource including meta-information related to that Resource
- Each Resource contains Interface(s) that provides first a view into the Resource and then defines the requests and responses permissible on that view of the Resource

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

Device Title	Device Type	Associated Resource Type	M/O
Light	oic.d.light	oic/res (oic.wk.res)	M
		oic/p (oic.wk.p)	M
		oic/d (oic.d.light)	M
		Binary switch (oic.r.switch.binary)	M
		Brightness (oic.r.light.brightness)	O



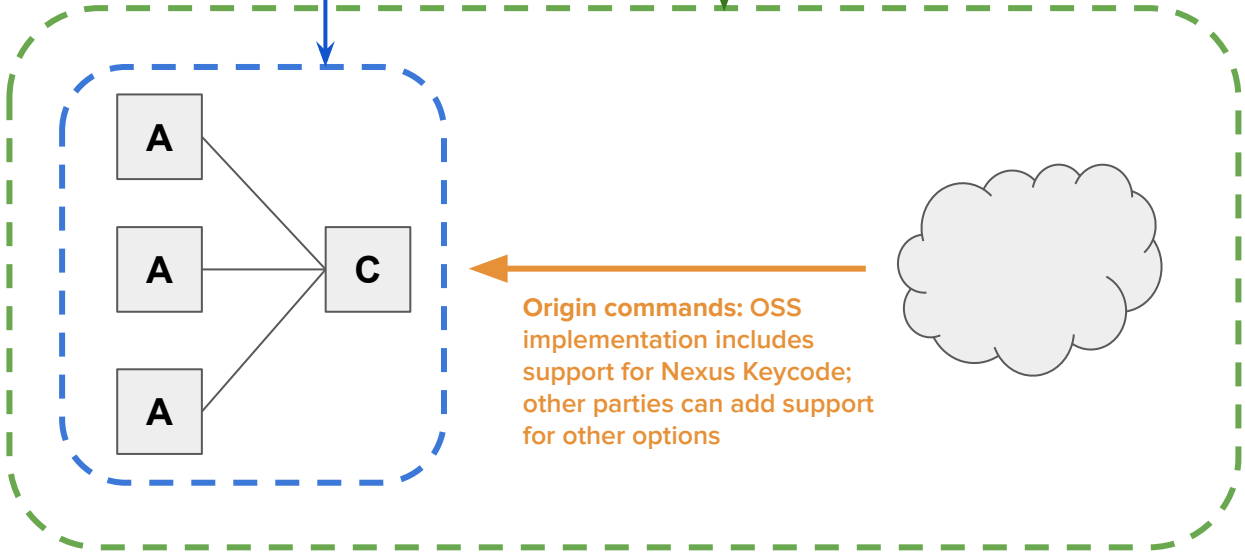
What is Nexus Channel? (more specifically)

- An implementation of a subset of OCF, inspired by IoTivity Lite, targeted to class 0 devices running on lightweight/unsecured transport layers
- An application layer security scheme based on RFC 8152 CBOR Object Signing and Encryption (COSE)
- A system for conveying “origin” commands from an external platform that mediates secured communication between devices
- An open source implementation: Angaza has released a MIT-licensed embedded library
- An open standard: Parties are free to write their own implementations

Nexus Channel & Nexus Channel Core

NEXUS CHANNEL CORE:
App & presentation layer,
based on/compatible with OCF

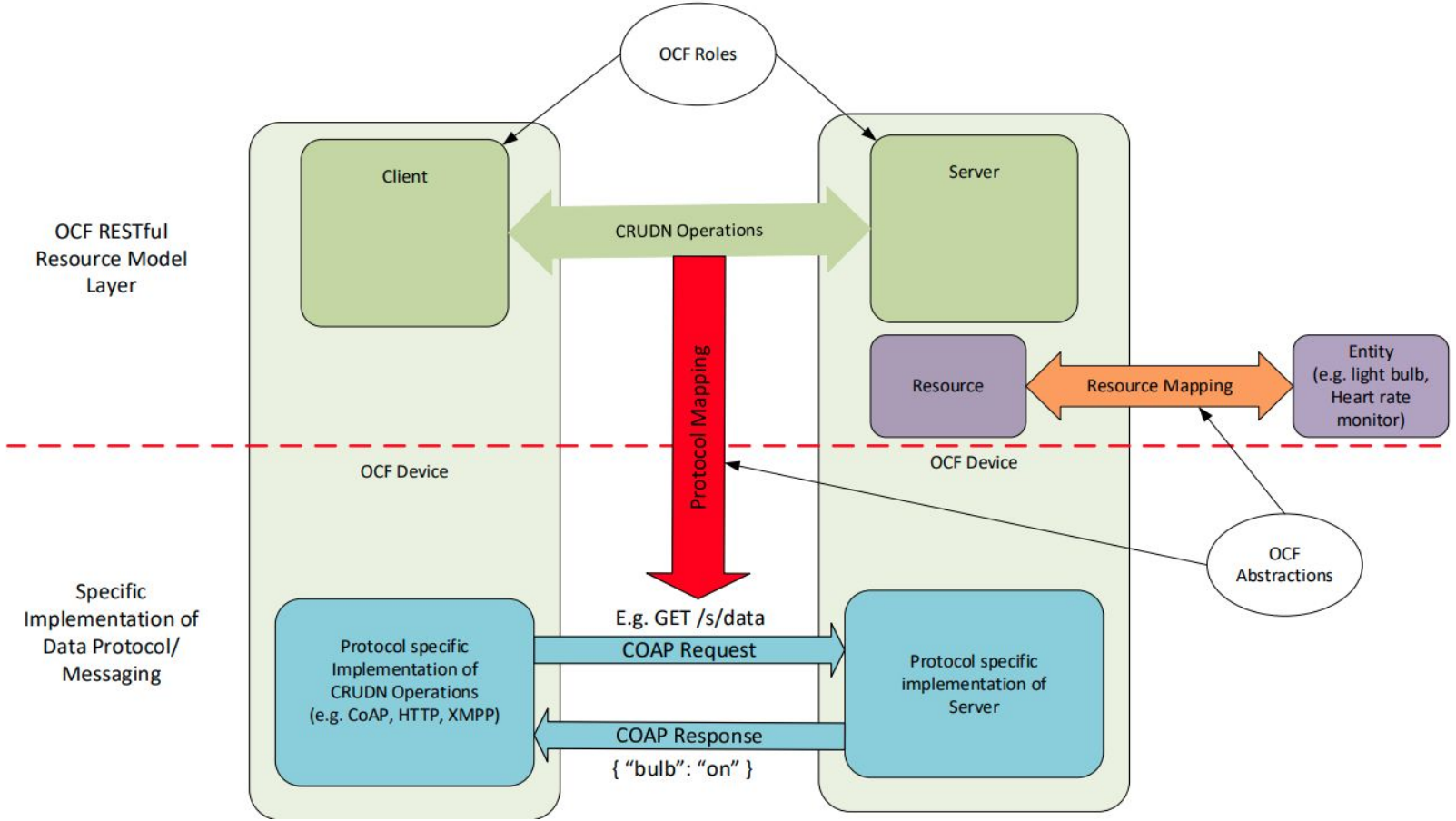
NEXUS CHANNEL:
Core +
App layer security +
Security controlled by platform



Current status

- WIP release of embedded library available on GitHub
- Aiming for a 1.0 release by the beginning of Q4 2020
- We are actively working with Solaris to make OpenPAYGO Link + Nexus Channel (Core) the standard in the PAYG industry
- Angaza has been granted funding from Efficiency for Access to bring Nexus Channel to market and establish it as an open standard

745 Figure 1 depicts the architecture.



746