Embedded Linux Conference: OIC Security Model and Vision

Ned Smith
Intel

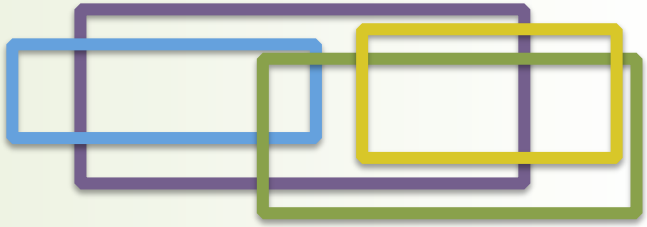
Day-in-the-Life Scenario

http://www.thankyouverymuchinc.com
http://smarthomeautomationva.com
http://i2.cdn.turner.com
http://gcn.com
http://cnet3.cbsistatic.com

intel

Security Objectives

- Crossing domain boundaries



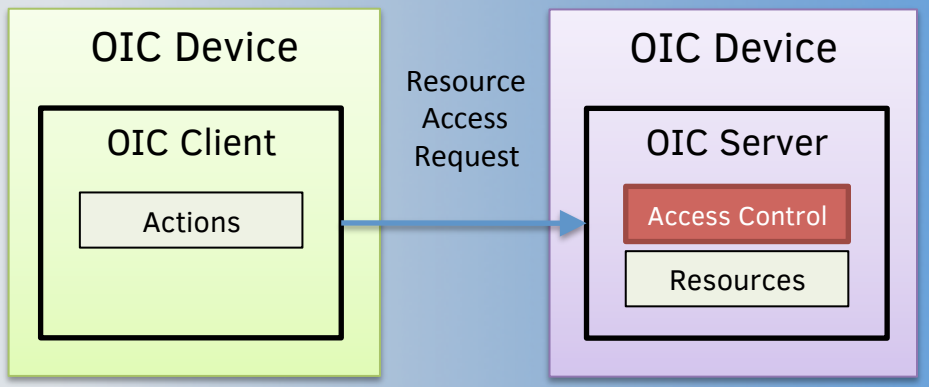
- Ad-hoc introductions

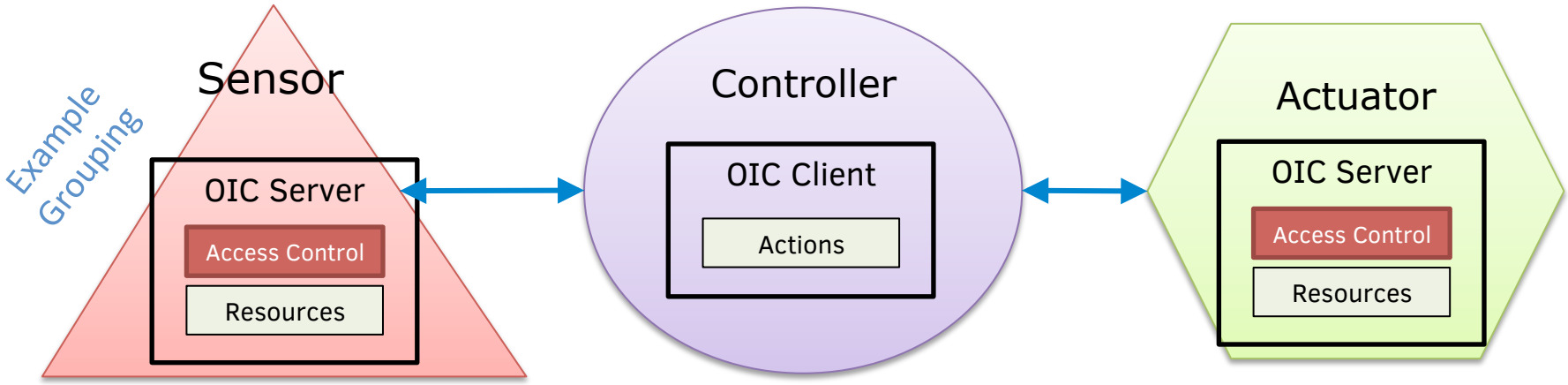
- Ensuring access

- Establishing ownership

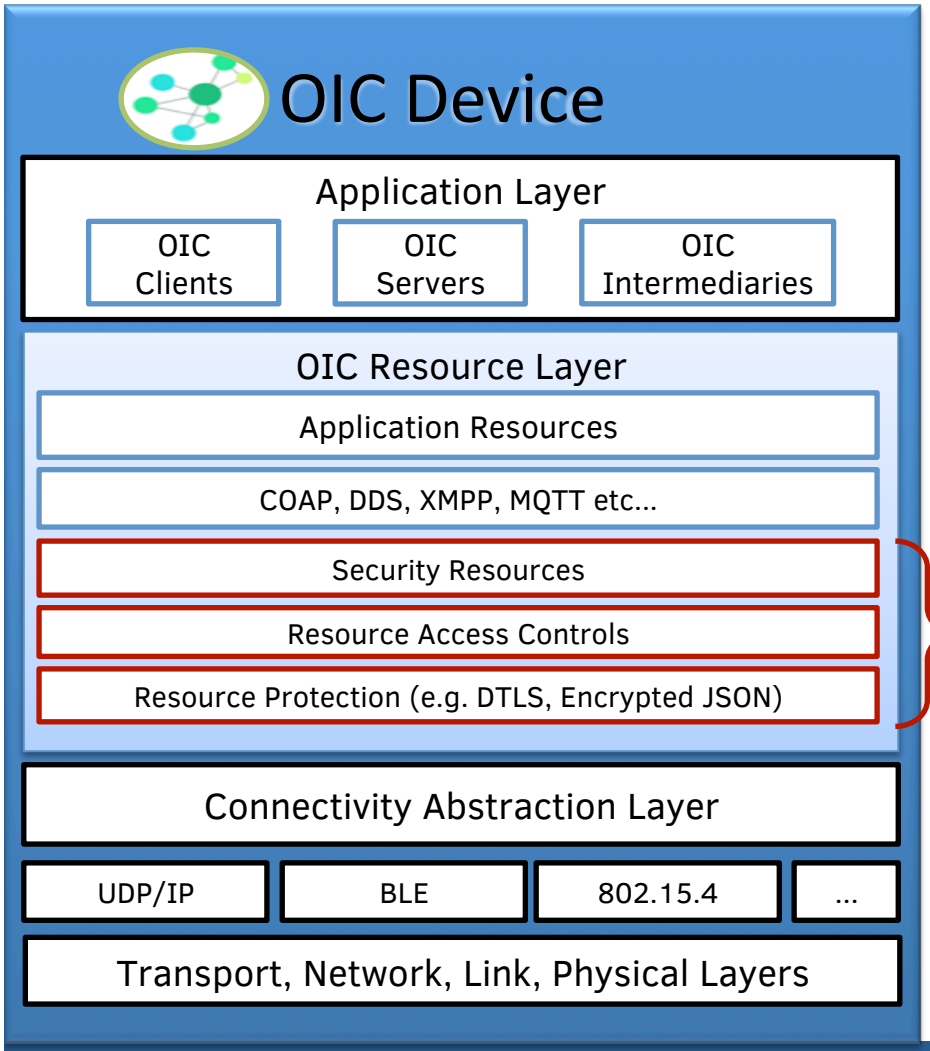
OIC Terminology

- A Device is an OIC stack instance
- Devices implement roles: Client, Server, Intermediary
- Devices have Resources and perform Actions
- Resources have Properties





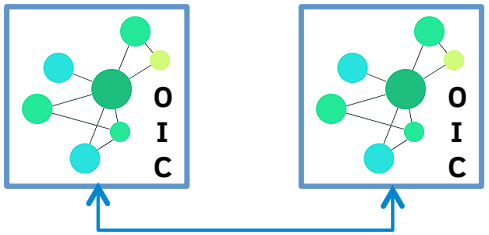
To Cross a Boundary We Must Define the Endpoint



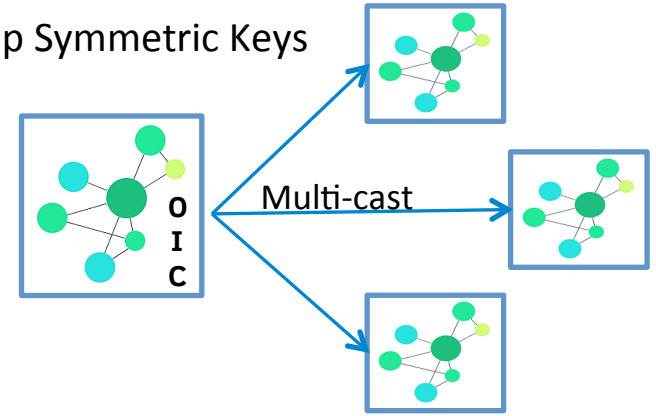
- An OIC *device* is the endpoint
- ...more specifically it is the OIC resource layer
- OIC resources define how device capabilities are exposed to other OIC devices
- Resources are accessed securely through a secure channel such as DTLS
  - End-to-end message encryption, integrity and replay protection
- OIC does not define endpoint hardening techniques
  - Resource layer hardening is implied

Key Management Objectives

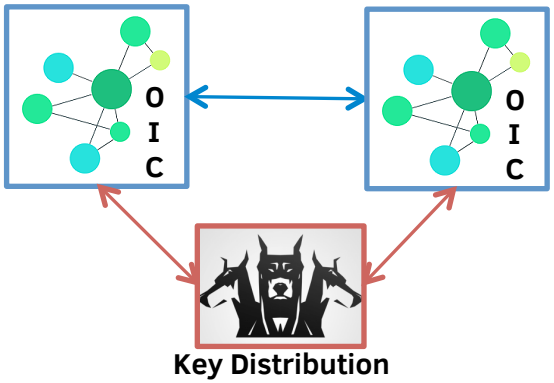
Pair-wise Keys



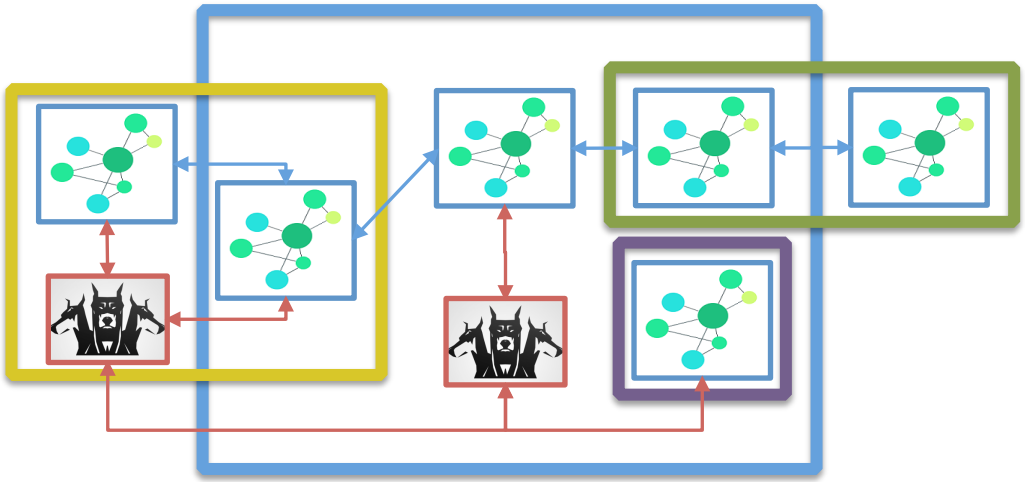
Group Symmetric Keys



Dynamic Provisioning of Keys

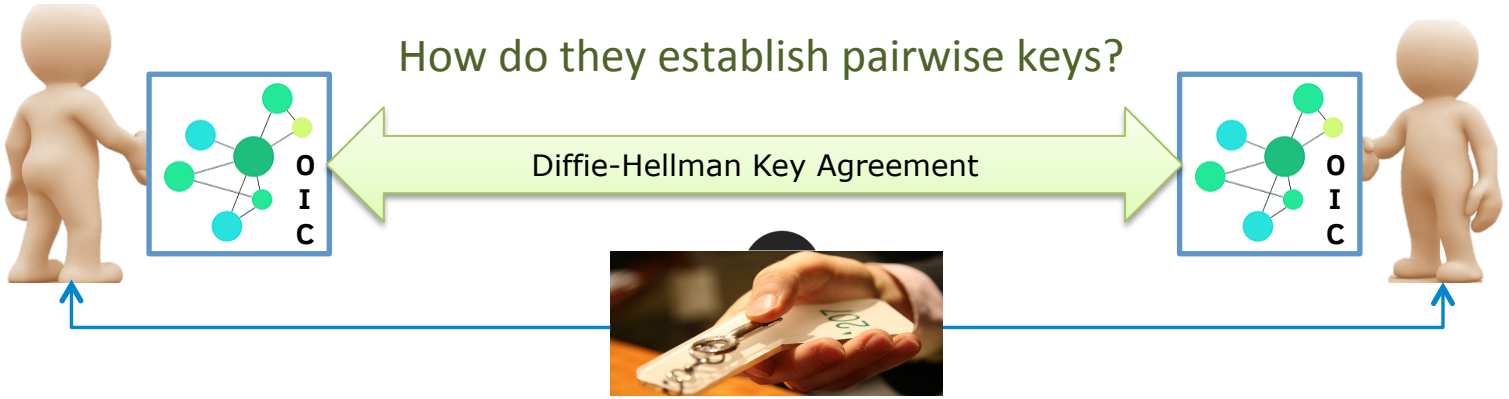


Localized Autonomy



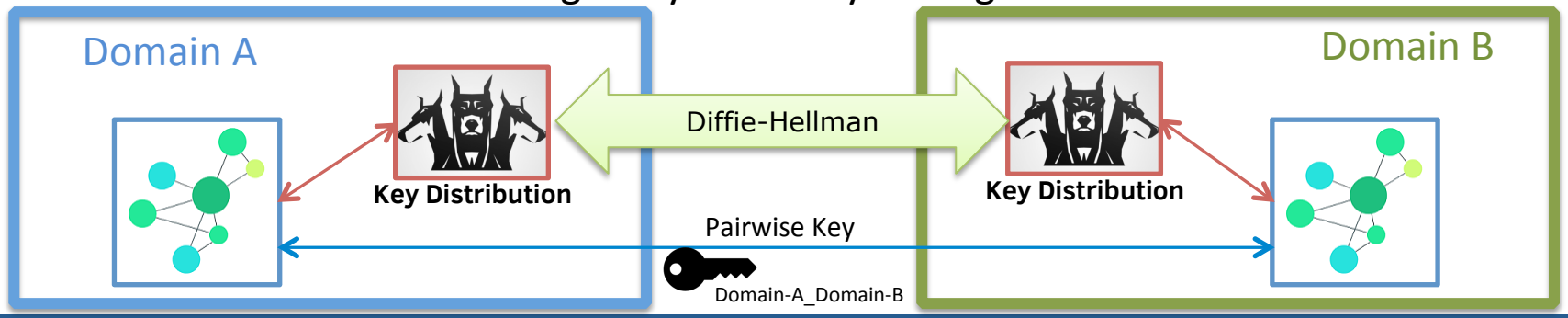
Ad-hoc Introduction

- Ad-hoc interactions suppose there isn't a trusted key distributer



Out-of-band exchange of a PIN using a smart device with a user interface!

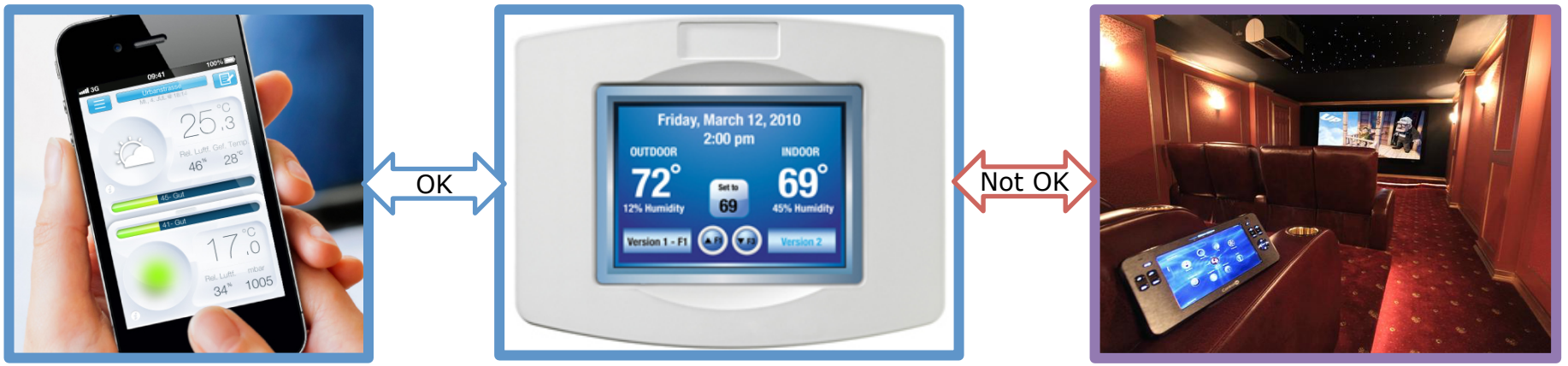
Domain crossings may enlist key management services



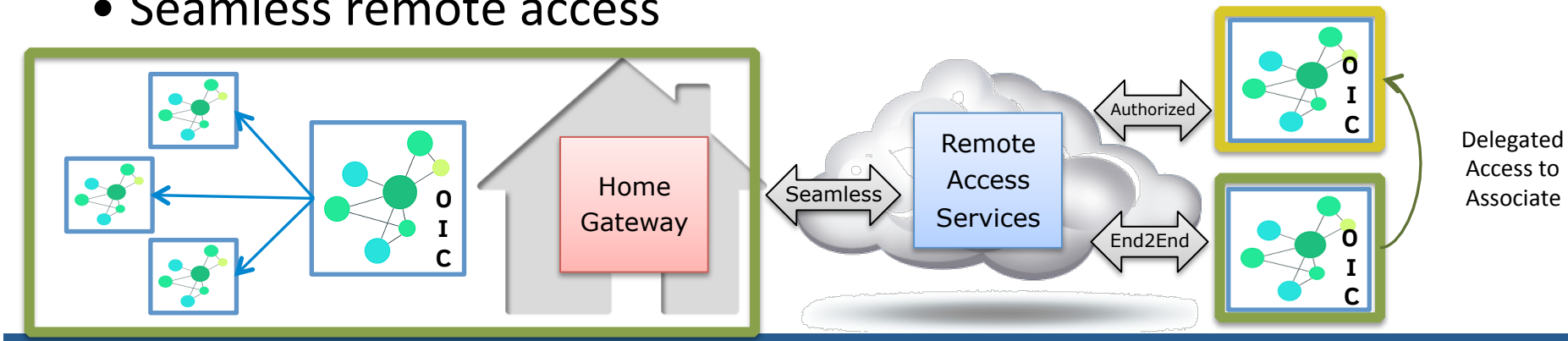
intel

Ensuring Access with Access Control

- Anticipate intended interactions
- Add friction to unintended interactions



- Seamless remote access



How To Distinguish Intended vs. Unintended?

- Access control granularity has four scoping levels
  - Group, Device, Resource and Attribute
- OIC Client actions capture interaction patterns
  - Peer-peer, Observer, Subscribe-notify, etc...
  - Actions specify intended device interactions

Device1

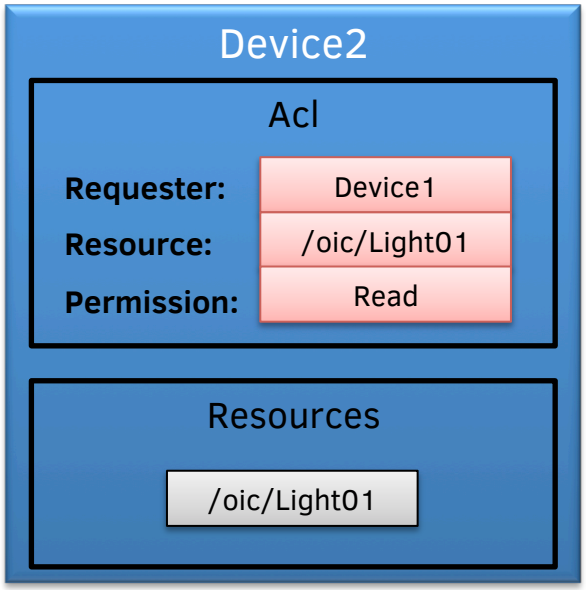
```
#%RAML 0.8
title: OIC Light set
/Collection01:
type: oic.collection
get:
   responses:
     application/json:
        schema: |  { "$schema": "http://json-schema.org/schema",
             "rt": { "type": "string", "required":true },
             "if": { "type": "string", "required":true },
             "resourceref": {
               "link": { "type":   "URI" } }
          }
        example: | {
          "rt": "oic.collection",
          "if": "oic.if.b",
          "resourcelinks": {
            "href": "Device2/oic/Light01",
            "href": "Device3/oic/Light02"   }
         }
```

GET implies READ access

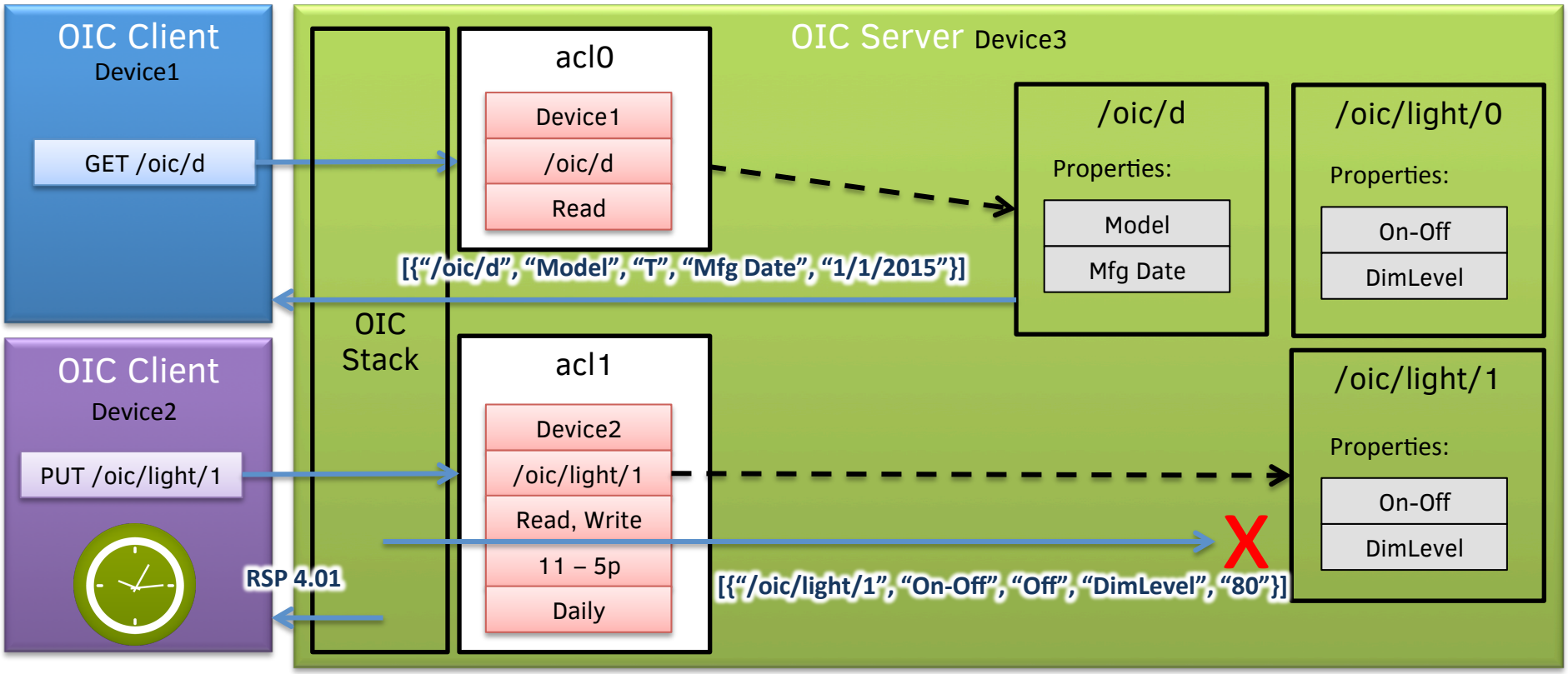
Collection href identifies which device and resource should have read access

Example

Informs



Resource Access Example



- Access is blocked if no ACL match is found
- Device1 request to get /oic/d is **accepted** due to ACL Read permission
- Device2 request to update /oic/light/1 is **denied** due to time-of-day policy
- An intermediary (Device4) may also enforce ACLs

Property-level Access

Example Resource Definitions:

Without Property Level Access Control

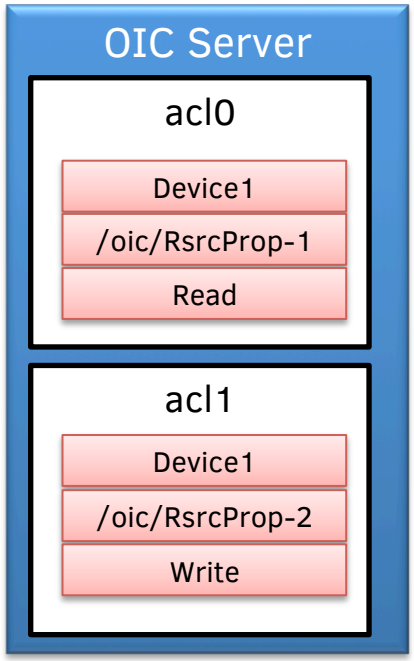
```
{"$schema": "http://json-schemas.org/schema#",
 "id": "http://openinterconnect.org oic.thing#",
 "definitions": {
  "oic.thing": {
   "type": "object",
   "properties": {
     "Property-1" {"type": "type1"}
     "Property-2" {"type": "type2"}
       ... }
  }
 }
}
```

Properties are opaque to OIC framework

With Property Level Access Control

```
{"$schema": "http://json-...
 ...
   "type": "collection",
   "resources": {
       "RsrcAtt-1",
       "RsrcAtt-2"}
 ...
 definitions": {
  "oic.RsrcProp-1": {
    "type": "object",
    "properties": {
       "Property-1" {"type": "type1"}
     } ...
  "oic.RsrcProp-2": {
    "type": "object",
    "properties": {
       "Property-2" {"type": "type2"}
     } ...
```

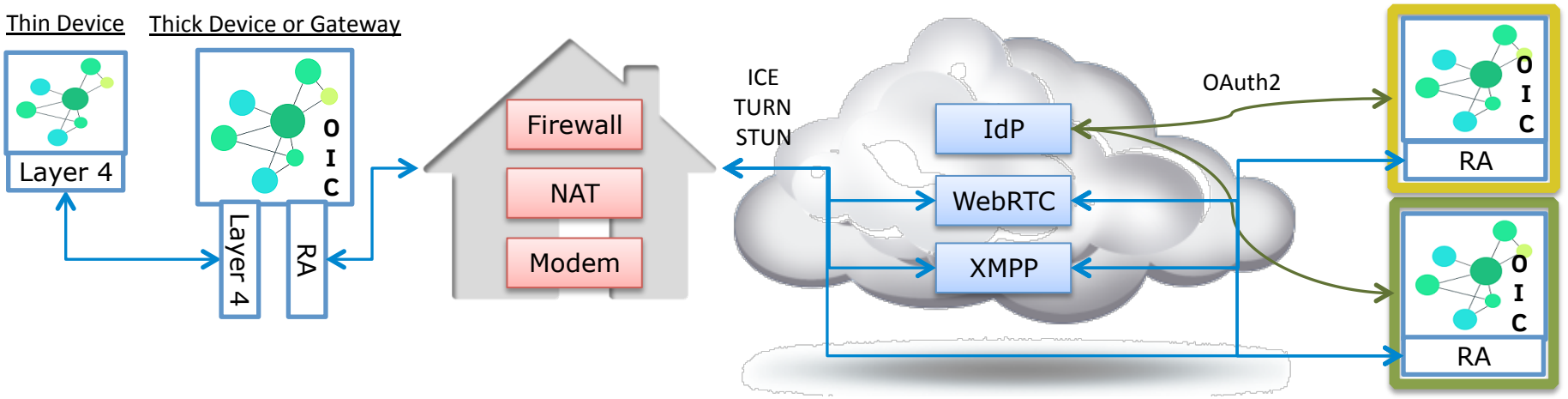
Resources with property-level granularity are NOT opaque to OIC stack



- Per property access can be achieve using a collection resource
  - A new resource is defined containing a single property
- Resource level access mechanism can satisfy property level access requirements

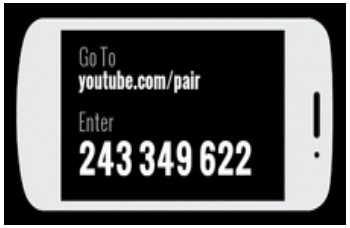
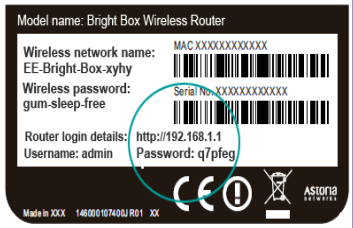
Remote Access

- OIC communications layer accommodates remote access



- Much of the remote access complexity is hidden within the OIC communication abstraction layer
- Home devices use same credentials when outside
- RA services provide a meeting place in the cloud
  - Using user credentials common to cloud services

Current Techniques for Device Ownership

Just Works	Mode Switch	Random PIN	Pre-provisioned PIN	Pre-provisioned Credential
				

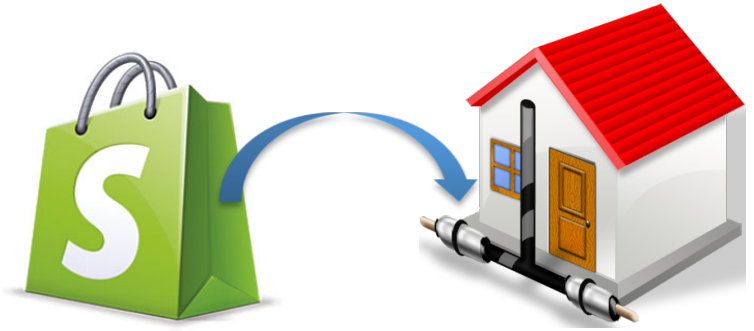
- Several techniques are in practice today, all impact manufacturing
- Problem is there is a disparity in security and attack vulnerability
- OIC members are working to standardize methods for device owner transfer

*Source: http://blog.atmel.com/2014/08/12/the-abcs-of-ecdsa-part-2/

Device Owner Transfer Objectives

- Manufacturer supports secure over-the-air transfer of owner

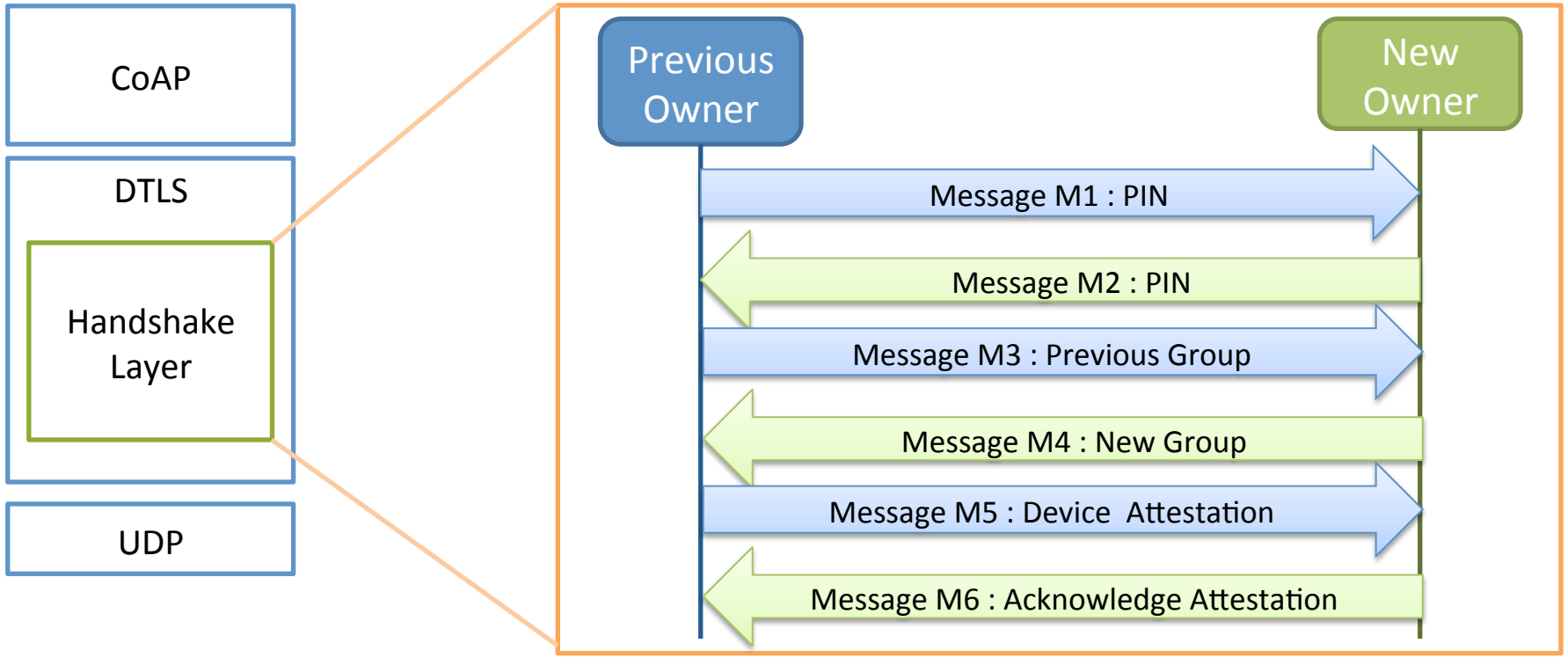
- Agreement that both parties intend to transfer ownership



- Identify the domain in which the new device is transferred

- Trust in the endpoint device performing the transfer steps

Example Device Owner Transfer Protocol



- Manufacturer PIN exchanged out-of-band authorizes intent
- Device certificate identifies owner group / domain
- Attestation of device internals to ensure trusted operation
- Diffie-Hellman for secure ad-hoc exchange of protocol messages

Conclusion

- IoT use models demand strong but flexible security
  - Devices operate in autonomous and ad-hoc ways
- OIC key management supports end-to-end device protection
- Devices from different domains can establish ad-hoc pair-wise keys using Diffie-Hellman
- Resource layer ACLS allow intended interactions while preventing unintended interactions
- Secure device ownership helps prevent attacks when devices are added to the network

Call to Action

- OIC is working toward interoperable IoT security
- Your participation in OIC is the best way to ensure your IoT products interoperate securely

Questions?

Ned M. Smith
ned.smith@intel.com
openinterconnect.org