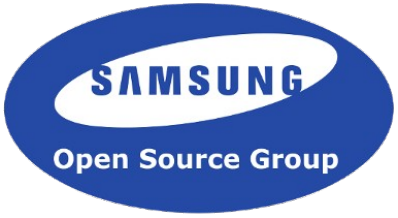


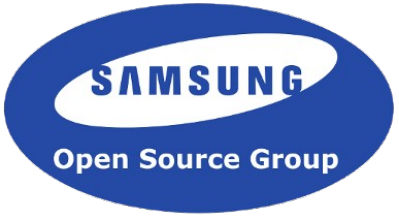
IoT meets Security

Habib Virji
habib.virji@samsung.com

Samsung Open Source Group
Samsung Research, UK

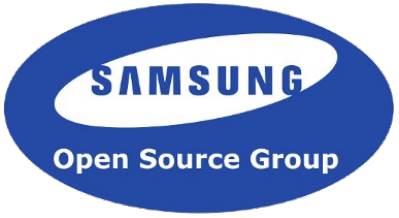
LinuxCon Europe 2015
Dublin, Ireland, October 5 – 7, 2015

Agenda



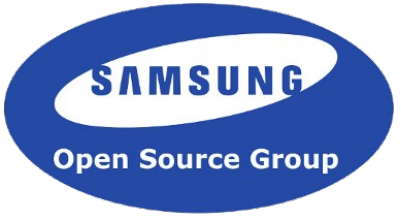
- Need for IoT Security
- Overview of IoTivity
- Device Security
  - Onboarding
  - Provisioning
  - Software Resource Manager
  - Hardware Hardening
- Connectivity
  - Local
  - Remote
- Privacy

Need for IoT Security



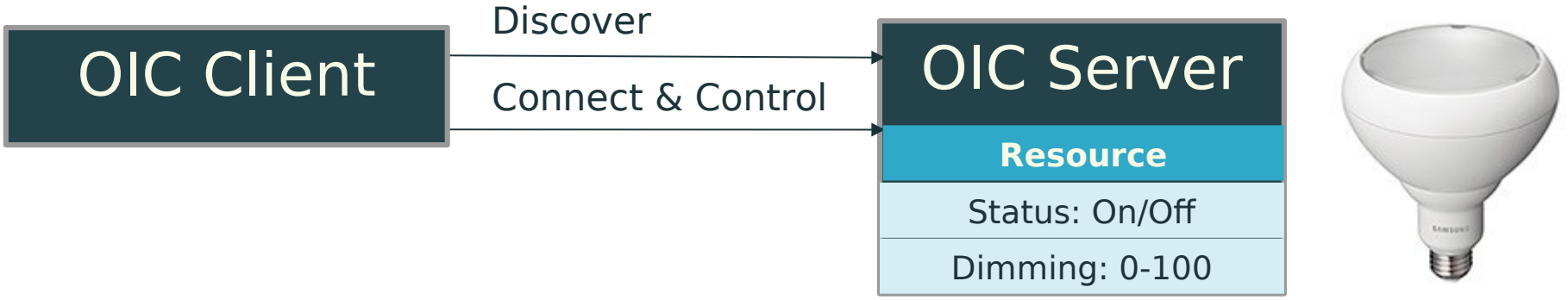
- IoT device to be around 26 billion by 2020 [1]
- Increase in IoT device require strong security.
- Lots of issues still in current IoT devices:[4]
  - 80% of devices had privacy issues.
  - 70% of devices used unencrypted network.
  - 90% of device collected personal information.
  - 70% of device along with their cloud enable attacker to identify valid user account using account renumeration.
- Need for IoT devices to have device, network and privacy concerns addressed.

IoTivity Overview



- IoTivity is Linux Foundation project to implement the OIC standard.
  - OIC is an industry consortium to define a IoT (Internet of Things) standard and certification.
  - IoTivity implementation is happening in parallel.
- Discovery of device is done by looking for a RESTful interface using multicast communication.
- Communication is done using:
  - CoAP (Constrained Application Protocol) over UDP in local scenarios.
  - XMPP is used in remote scenarios.
- Support for multiple OSs platforms – Tizen, Android, Linux, Arduino, etc.

Resource Model



- Discovery
- Control resource
- Observe

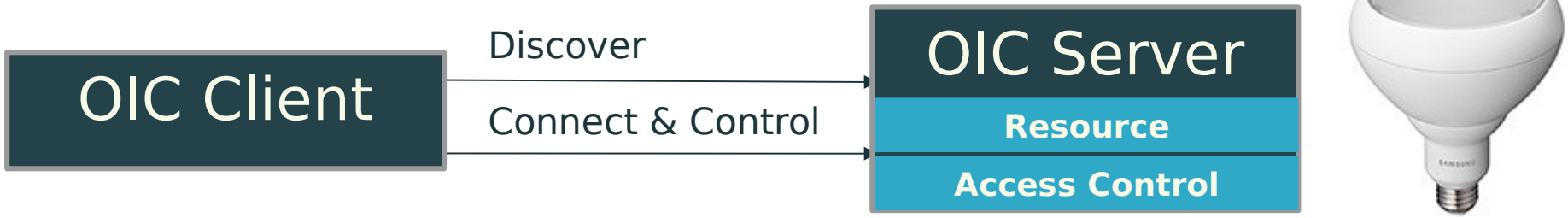
Resource Property:

rt=oic.light	(Type)
ra=192.168.1.1/a/light	(Address)
obs=1	(Observable)
acl=oic/sec/acl/1	(Access Control)

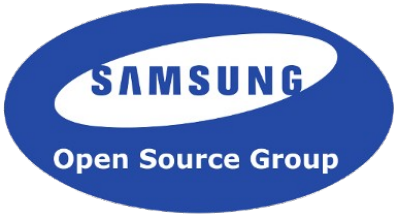
Resource Attributes:

```
{
  “status” : on
  “dimming” : 35
}
```

IoTivity Security

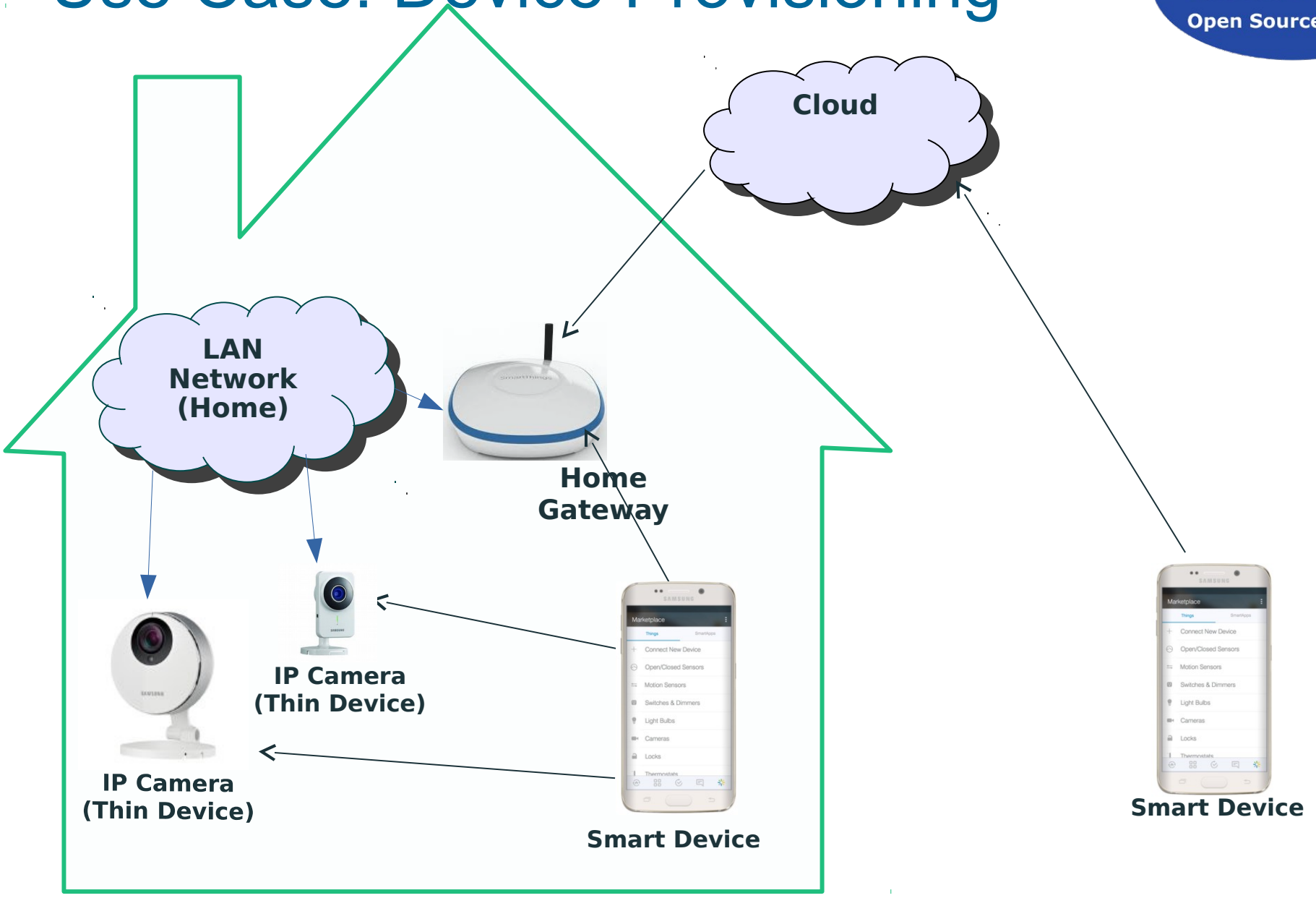


- Protection of resources.
- Three step in the security mechanism
  - Connectivity.
  - Secure channel.
  - Privacy permission.
- Device needs to be onboarded and provisioned.

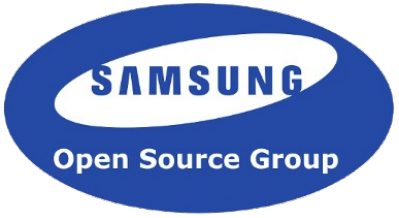


Device Security

Use Case: Device Provisioning

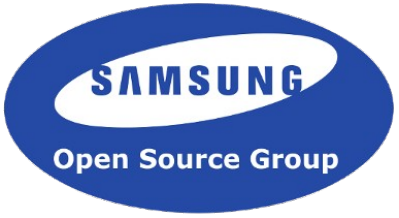


OWASP Device Security Risks



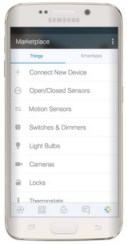
- Physical
  - Poor physical security
- Software
  - Insecure cloud interface
  - Insecure mobile interfaces
  - Insufficient security configuration
  - Insecure software/firmware

Onboarding



- Establishes device ownership.
  - Device becomes part of the user network.
  - Device cannot onboard other device ownership.
- It is a two step process:
  - Isolated secure communication between physical device and onboarding tool (OBT).
  - Then it assigns ownership key and second carrier key
- Onboarding relies on ownership transfer protocol.
  - Ownership credential (OC) establishes OBT and device communication and authenticate each other.
- Ownership protocols
  - Just Work
  - Random PIN
  - Asymmetric (Certificate)

Ownership Transfer – Just Works



Onboarding Tool

Enrolling Device

Ownership Transfer Start

Discovery

GET /oic/sec/doxm?Owned=”False”

RSP [{“OxmType”: “oic.sec.doxm.jw”, “DeviceId”: “UUID”}]

Preparing for Ownership transfer using Just Works

Set Ownership transfer Method

PUT /oic/sec/doxm [{“OxmSel”: “oic.sec.doxm.jw”}]

RSP 2.04

SRM enables TLS_ECDH_anon_WITH_AES_12SHA256 cipher suite

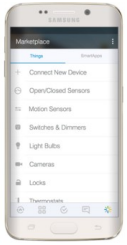
DTLS Connection

PUT /oic/sec/doxm [{“Owned”: “T”, “Owner”: “Admin0””}]

RSP 2.04

Ownership Transfer Stops

Ownership Transfer – Random PIN



Onboarding Tool

Enrolling Device

Ownership Transfer Start

Discovery

GET /oic/sec/doxm?Owned=”False”

RSP [{“OxmType”: “oic.sec.doxm.rdp”, “DeviceId”: “UUID”}]

Preparing for Ownership transfer using Random PIN

Set Ownership transfer Method

PUT /oic/sec/doxm [{“OxmSel”: “oic.sec.doxm.rdp”}]

RSP 2.04

SRM enables TLS_ECDHE_PSK_WITH_AES_128_CBC_SHA256 cipher suite

DTLS Connection

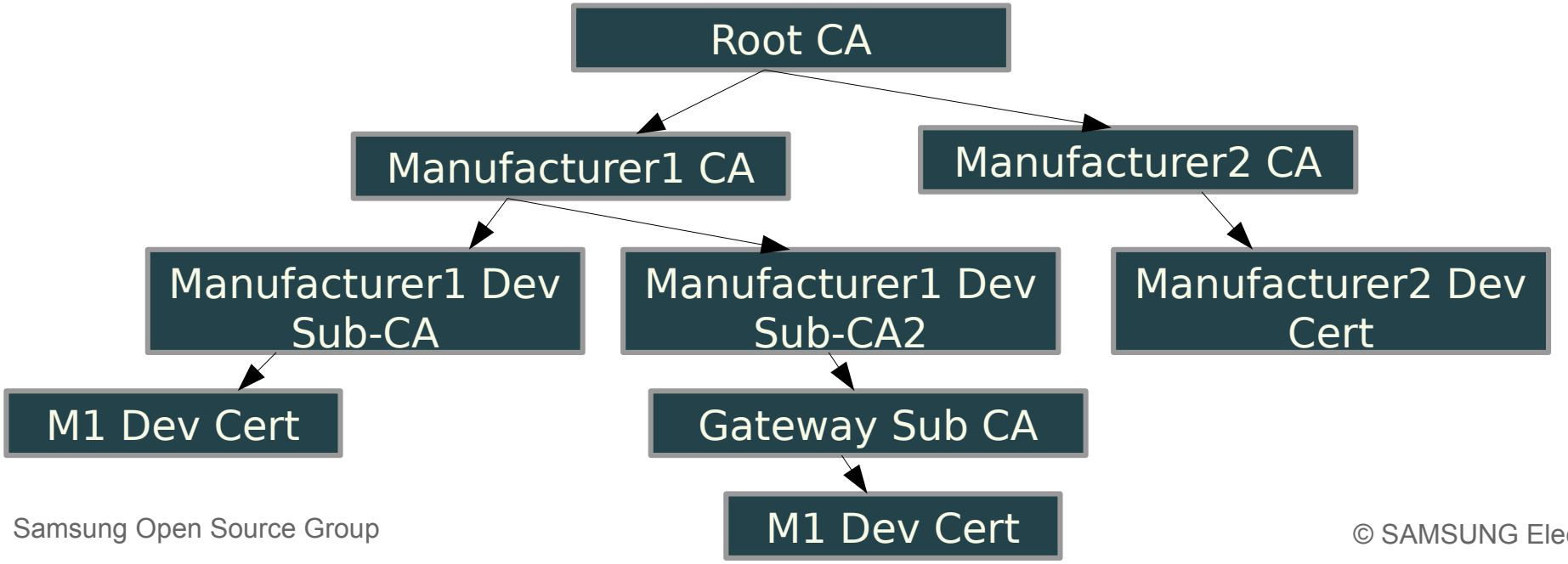
PUT /oic/sec/doxm [{“Owned”: “T”, “Owner”: “Admin0””}]

RSP 2.04

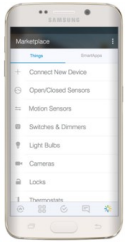
Ownership Transfer Stops

Owner transfer protocol - Asymmetric certificate

- Minimum certificate size (292 bytes) and minimal parser.
- Certificate generated with signed certificate and asymmetric key pair.
  - OBT binary app signed trusted CA to communicate with above certificate.
  - Device and OBT authenticate each other using ECDSA.
  - Authenticate successful then link exchange over ECDH.



Owner transfer protocol - Asymmetric certificate



Onboarding Tool

Enrolling Device

Ownership Transfer Start

Discovery

GET /oic/sec/doxm?Owned=”False”

RSP [{“OxmType”: “oic.sec.doxm.mfgcert”, “DeviceId”: “UUID”}]

Preparing for Ownership transfer using Certificate

Set Ownership transfer Method

PUT /oic/sec/doxm [{“OxmSel”: “oic.sec.doxm.mfgcert”}]

RSP 2.04

SRM enables TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8_SHA256 cipher suite

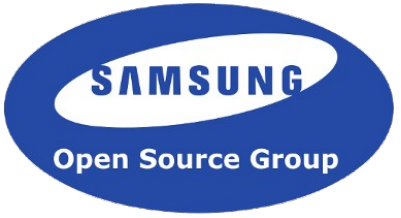
DTLS Connection

POST /oic/sec/doxm [{“credid”: “..”, “sub”: “..”, “credType”: “8”, “pdbdata”: “device and CA in base 64”, “pvdata”: {“x”: “x position of elliptic curve in base 64”, “y”: “y position of elliptic curve in base 64”, “ownrs”: “” }}]

RSP 2.04

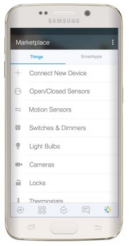
Ownership Transfer Stops

Provisioning



- Credential are transferred from OBT to device.
- Device needs to engage with bootstrap server to provision
  - Client directed: Client update server is in need of provisioning.
  - Server directed: Server self checks if it is provisioned.
- Proper security credential and parameters.
- Parameters include:
  - Security credentials through credential management service.
  - Access control policies and ACL
  - Devices are self aware about security provision status.

Provisioning



Onboarding Tool

Enrolling Device

ACL Provisioning Start

Status

GET /oic/sec/pstat

RSP [{“IsOp”: “False”, “Sm”: “0x11”}]

Client Mode

PUT /oic/sec/pstat [{“Om”: “0x11”}]

RSP 2.04

DTLS with Owner PSK

POST /oic/sec/acl [{“Subject”: “Admin0”, “Permissions”: “CRUDN”, “Owner”: “Admin0”}]

RSP 2.04

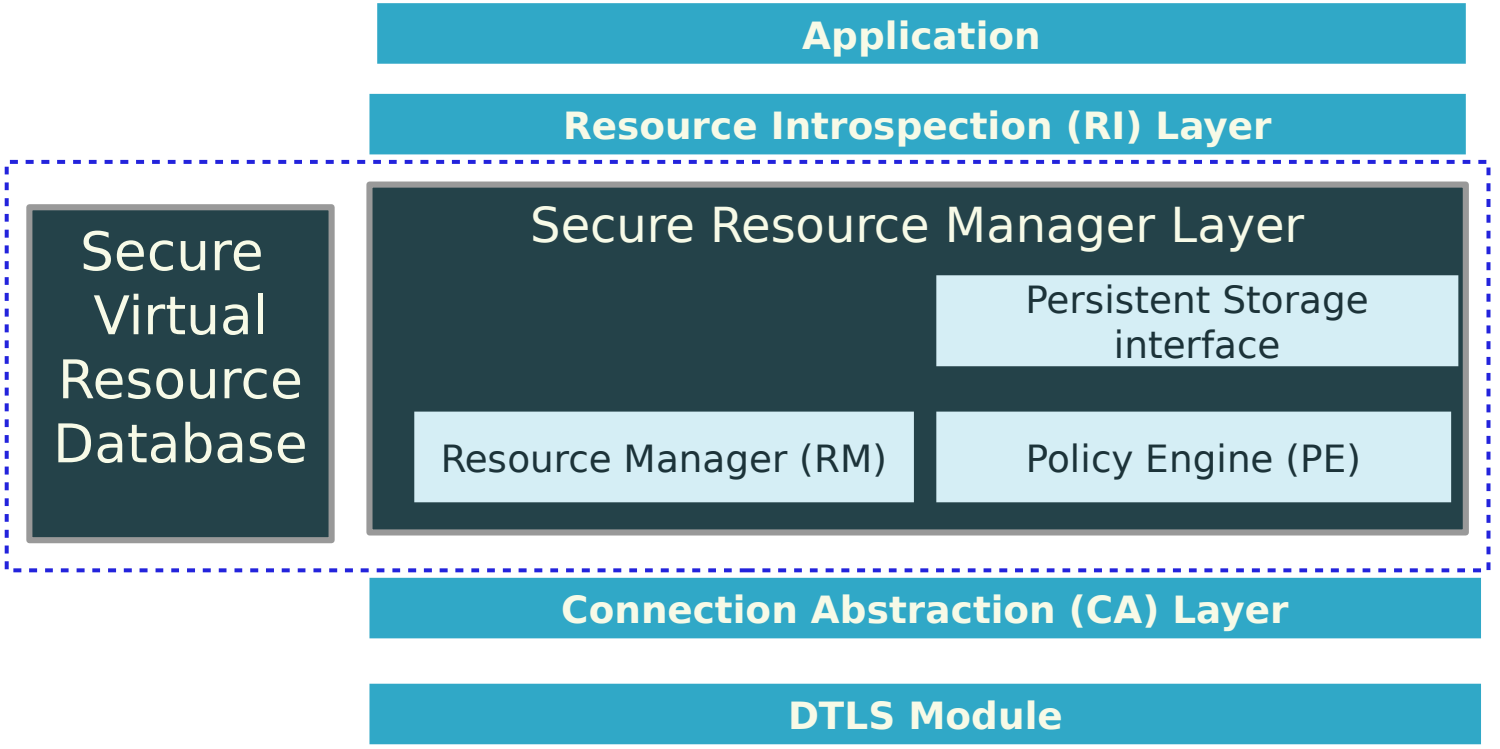
POST /oic/sec/pstat [{“Tm”: “0”, “CommitHash”: “Base64”}]

RSP 2.04

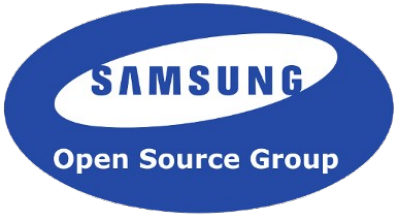
ACL Provisioning Stop

Secure Resource Manager (SRM)

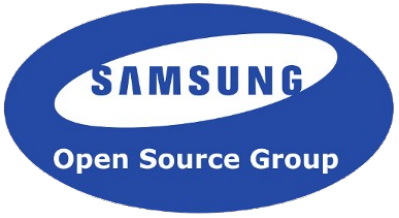
- Management of the secure virtual resource and ACL [3].



Hardware Hardening

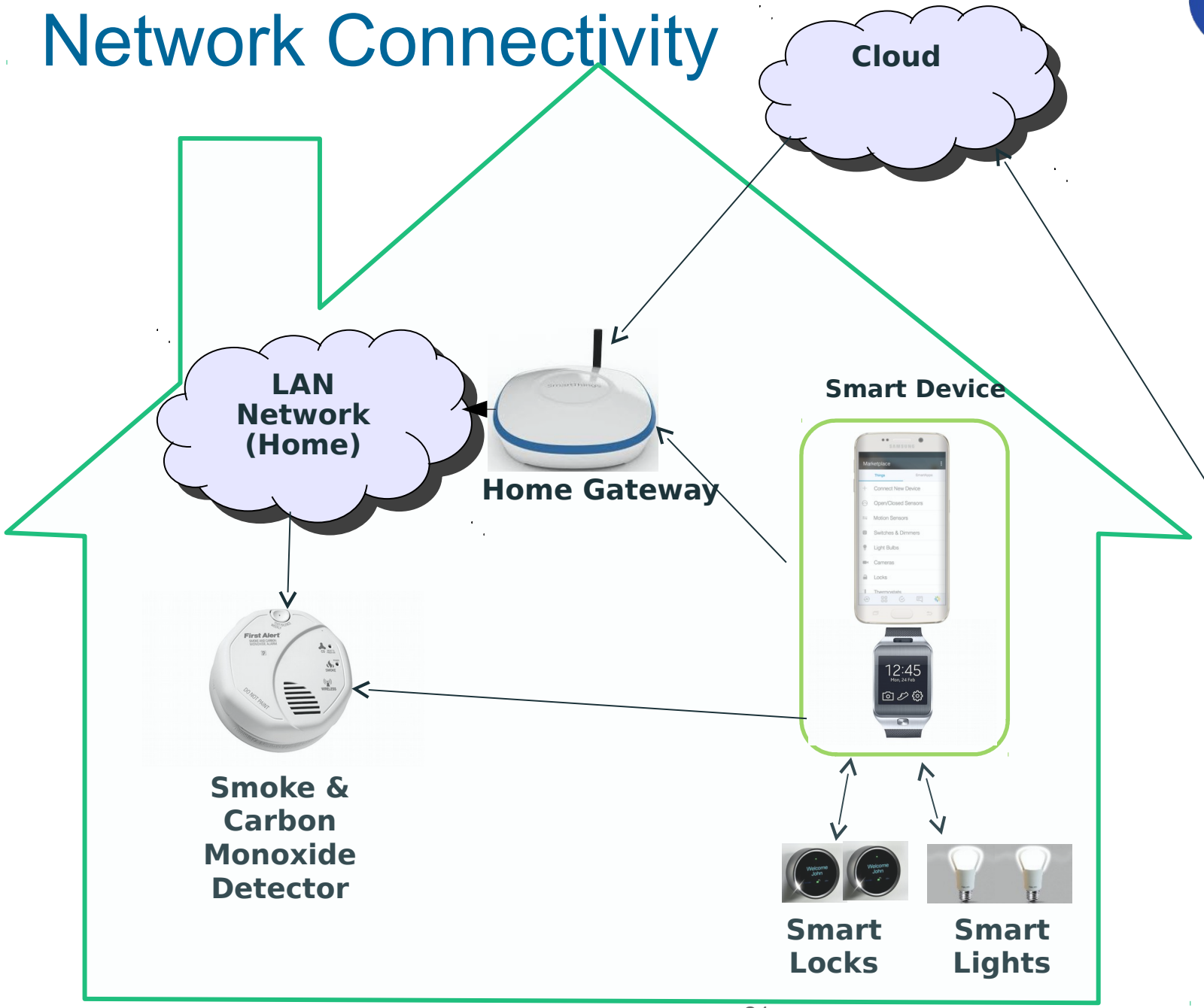


- Secure storage is to provided using encryption and hardware security.
- Secure execution environment:
  - Secure storage
  - Secure execution engine
  - Trusted I/O paths
  - Secure Time Source/Clock
  - Random number generator
  - Cryptographic algorithm
  - Hardware tampering

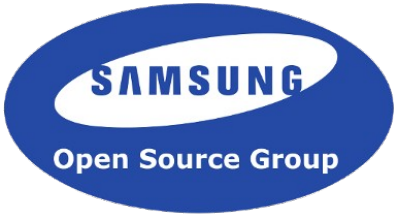


Connectivity

Use Case: Local and Remote Network Connectivity



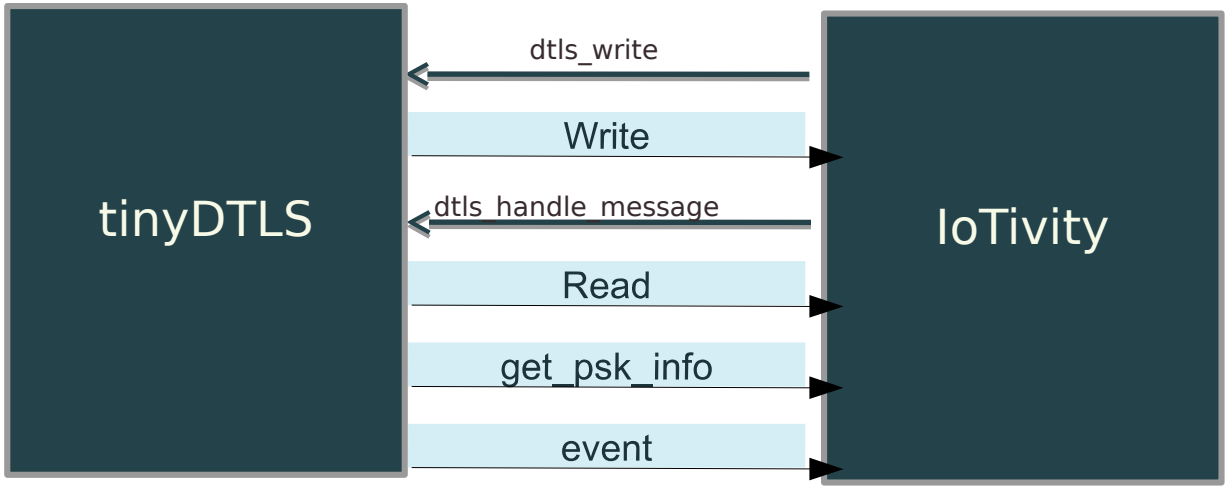
OWASP Network Security Risks



- Insecure network services
- Lack of transport encryption
- Insecure web interface
- Insufficient authentication/authorization

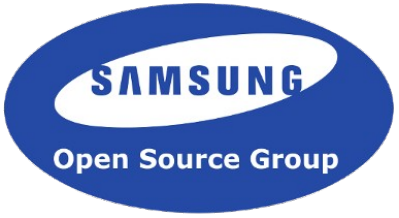
Secure Connectivity

- DTLS to provide packet by packet protection.
- OIC client and server communication should be protected using
  - Eavesdropping
  - Message replay
  - Tampering



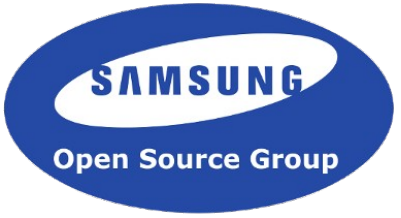
- Device authentication
  - Client verifies server using device id
  - Client if it has match sends server message
  - Server verifies message exchange

Low End Device Secure Connectivity

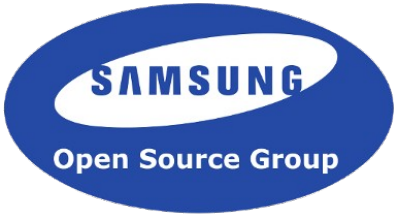


- Low end device uses extension of DTLS handshake to establish session keys.
- Based on Diffie-Hellman key agreement.
- Can be used in owner transfer protocol to establish keys.
- Breaks down further DTLS handshake to ease smaller packet transfer and fragmented PDU.
  - 6 way message protocol instead of 3 message.

Remote connectivity

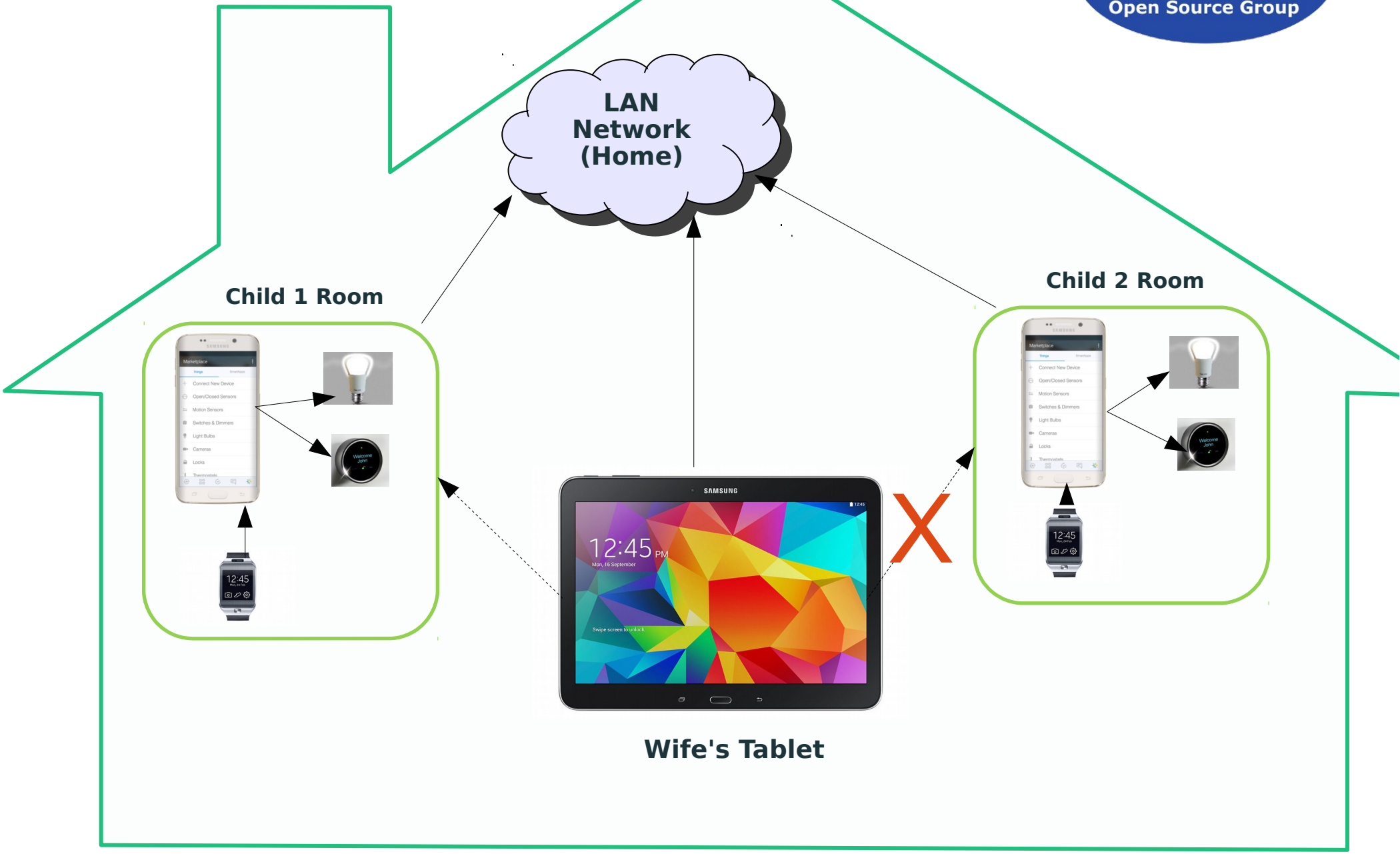


- OIC device communicate with XMPP server
  - Authenticates using XMPP roster credential
- Device identified using JID
  - Server: me@mydomain.com/oic/1.0/oic.d.light/FFFFDDDD-YYYY-4567-JADE-123456789A123
  - Client: me@mydomain.com/oic/1.0/client/FFFFDDDD-YYYY-4567-JADE-123456789A123
- Remote XMPP server and OIC server have secure connection.
- Inband bytestream is used between XMPP and OIC server.

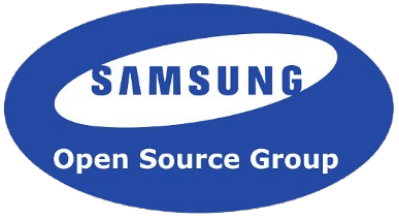


Privacy

Use Case: Controlling Access

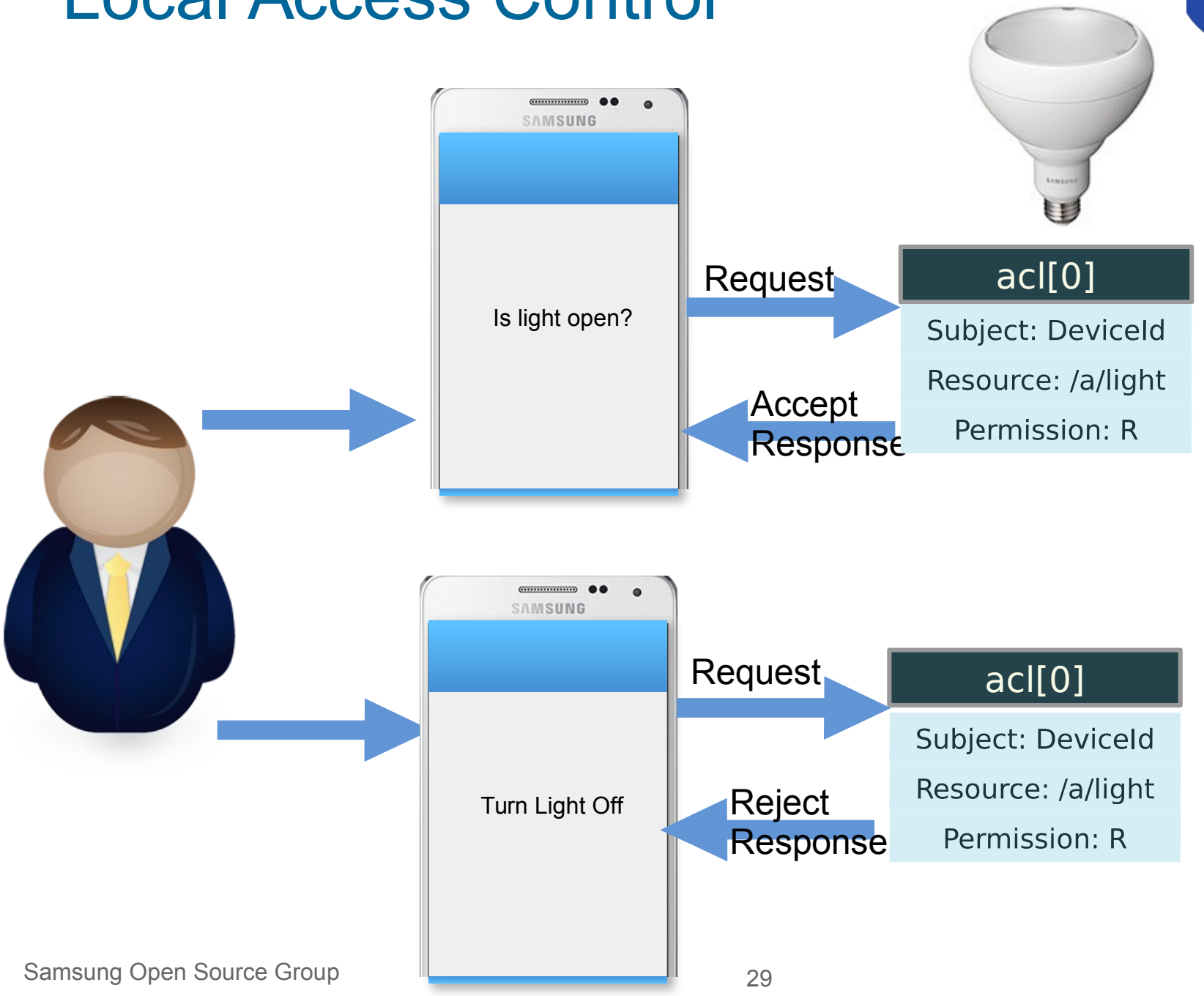


Privacy

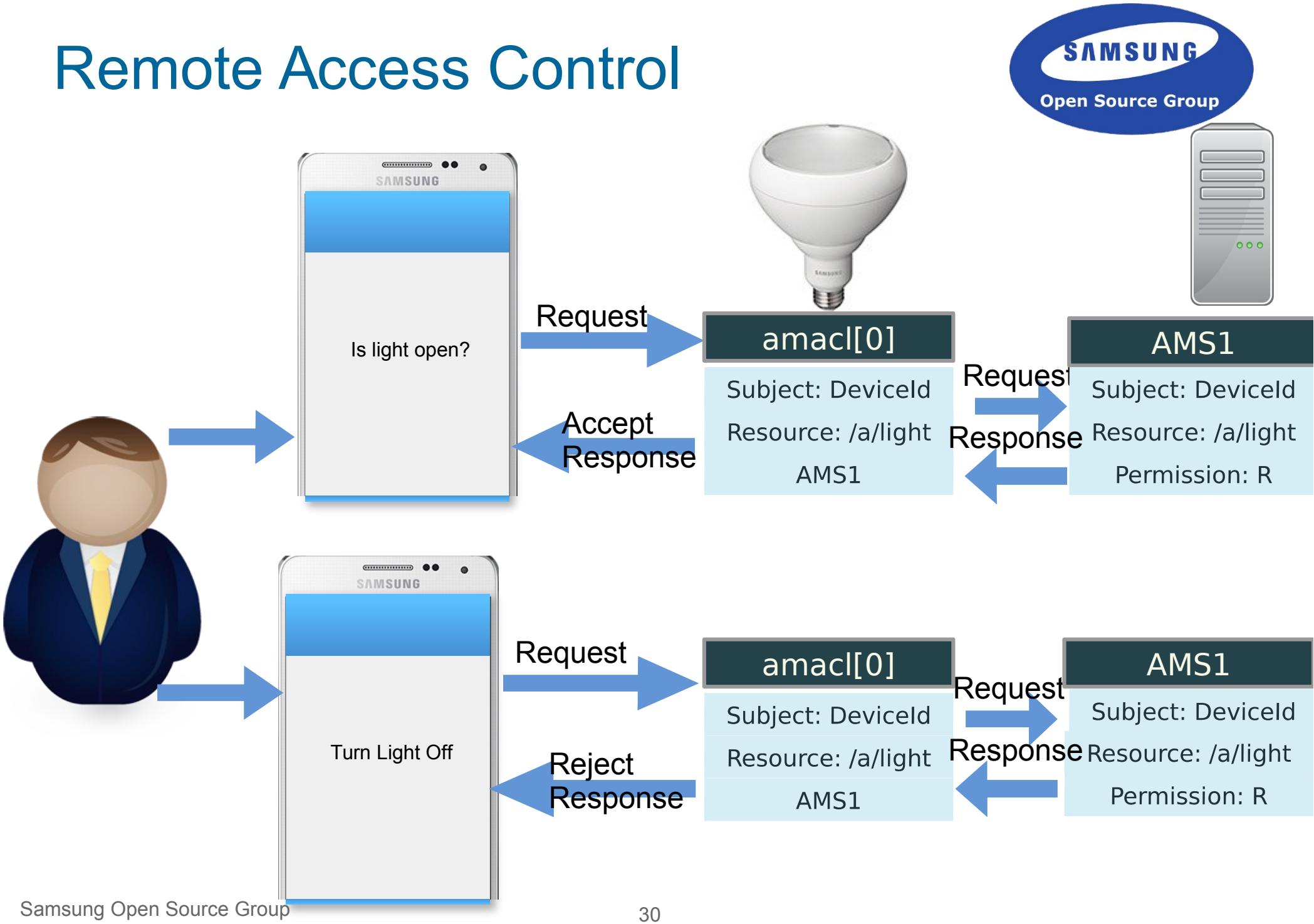


- Protects resources at the OIC server.
- ACL are defined via ACE (access control entities).
- Every resource should have an ACE.
- ACE are stored either locally or remotely on Access manager server (AMS).
- ACL needs to be secure stored and partitioned between logical OIC servers.
- Access control levels is per group, device, resource or properties.

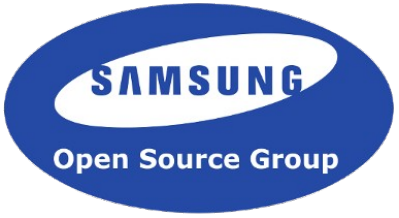
Local Access Control



Remote Access Control

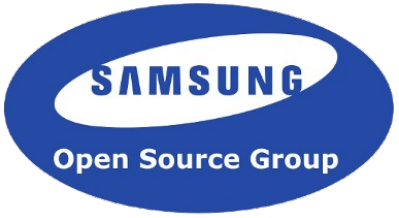


Conclusion

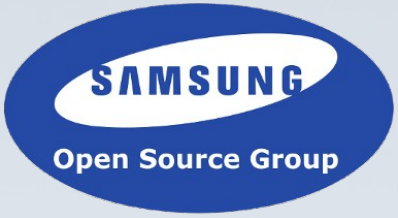


- IoTivity addresses majority of the OWASP issues.
- OIC provides following security functionality:
  - Onboarding mechanism to get device securely in user network
  - Policies control who can read/write on to the device.
  - Secure connectivity between device.
- Hardening mechanism suggested.
- SRM includes security functionality.

References



- [1] http://www.gartner.com/newsroom/id/2636073
- [2] https://www.owasp.org/images/8/8e/Infographic-v1.jpg
- [3] https://wiki.iotivity.org/iotivity_security
- [4] http://www8.hp.com/h20195/V2/GetPDF.aspx/4AA5-4759ENW.pdf



Thank You!