# Issues for TPAC Plugfest

September 27th, 2017

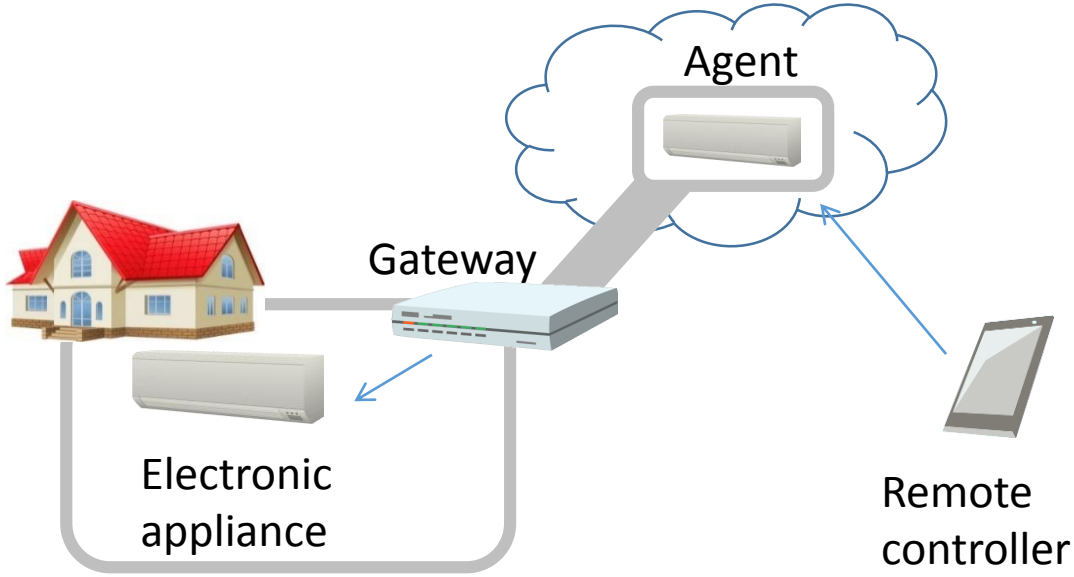
Ryuichi Matsukura

Fujitsu Laboratories Ltd. / Fujitsu Ltd.

See IG issue #346, https://github.com/w3c/wot/issues/346

# Use Case (example)

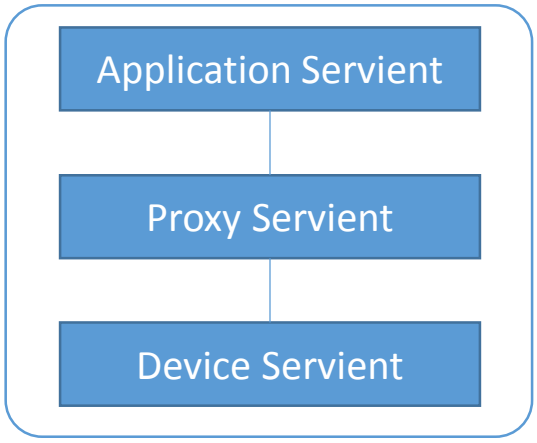
- New scenario: Two and more Servients collaborations, especially between the Internet and local networks
  - Application on Smartphone connected to Internet can operate air conditioner on local network. Agent and gateway relay messages between application and device.



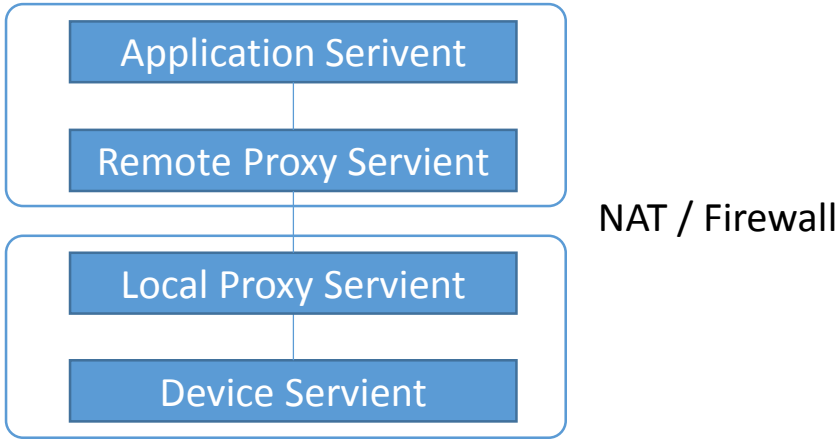
# Functional roles of WoT Servients

- 3 types of WoT Servients
  - Device Servient: WoT Servient with ExposedThing
  - Application Servient: WoT Servient with ConsumedThing
  - Proxy Servient: WoT Servient with both of ExposedThing and ConsumedThing
- Integration model

3-layer model




4-layer model

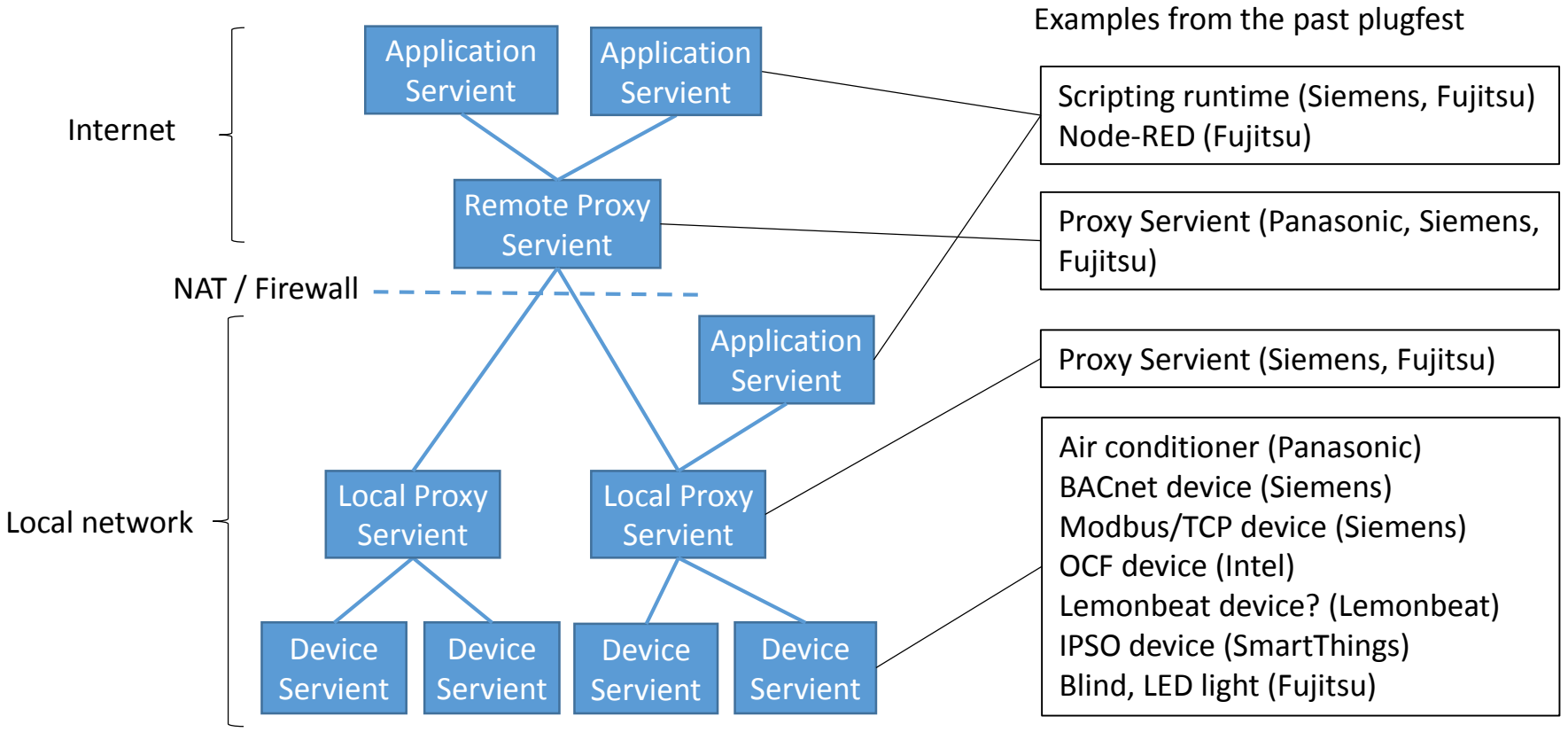


# Issues for TPAC plugfest

- Interface between Servients
  - Authentication
  - Discovery and exchange TDs
  - Firewall and NAT traversal
  - Add “Event” operation for inter-Servient interface
- Thing Description management
  - Management
    - How to create URI. A WoT Servient on the Internet cannot access Servients on local networks, because local Servients URI are assigned to the local address.
    - Who and how to manage TDs. If many servients connect to networks, the management function is necessary to easily search TD someone want to connect.

# Call for participants

- Please share the information of your Servients and legacy devices on TPAC meeting Wiki.



# Demo scenario

- Smart home, building, and factory
  - All of devices connect to the local network.
  - Applications execute on the local or the other networks.
  - Applications use Scripting API or WoT IF (e.g. NodeRED).

# Additional specifications for plugfest (overview)

- Architecture (integration model)
- Sequence diagrams among Servients
- Inter-Servient Interface
  - Authentication :  ACE (see in Current Practice)
  - Discovery and exchange TD : NEW
  - Firewall and NAT traversal : NEW
  - “Event” operation with subscription : NEW
- TD Management
  - Life cycle management
  - URL naming convention

# Basic sequence diagram (1 of 7)

- Shows interaction example between Servients
  (1) Register & Lookup (Discovery)

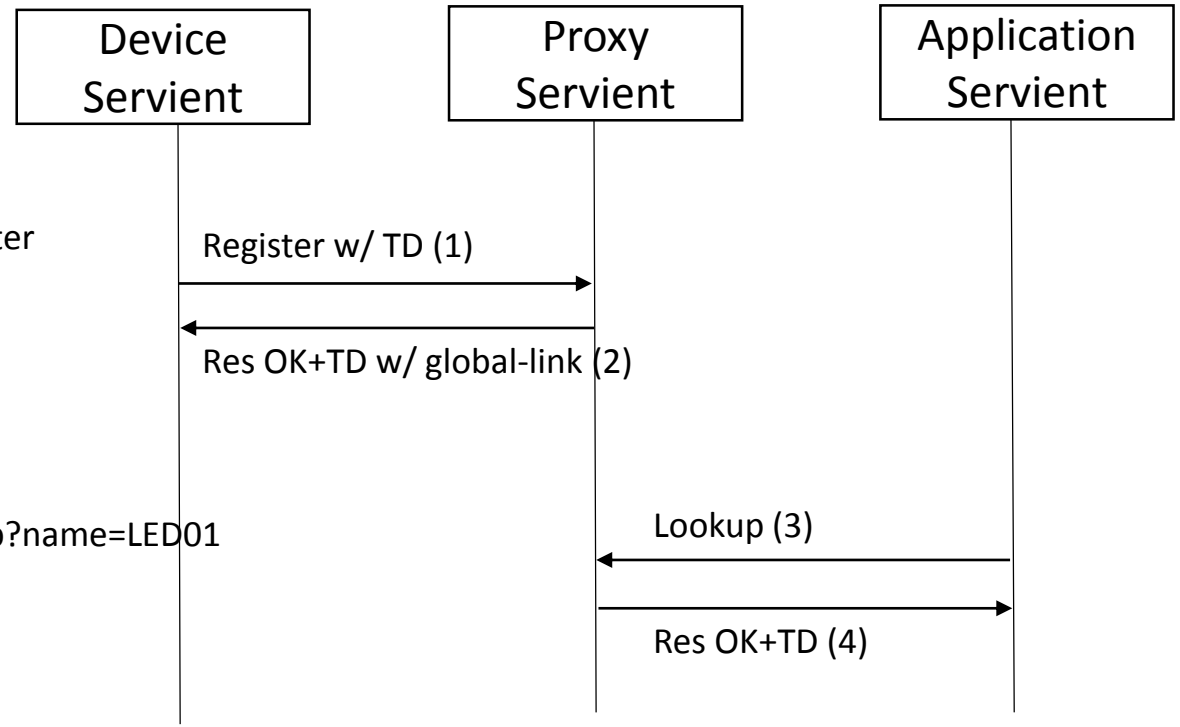
**Examples**

Message (1)
POST http(s)://ps.example.com/register
Body: TD

Message (2)
200 OK
Body: TD including global-link

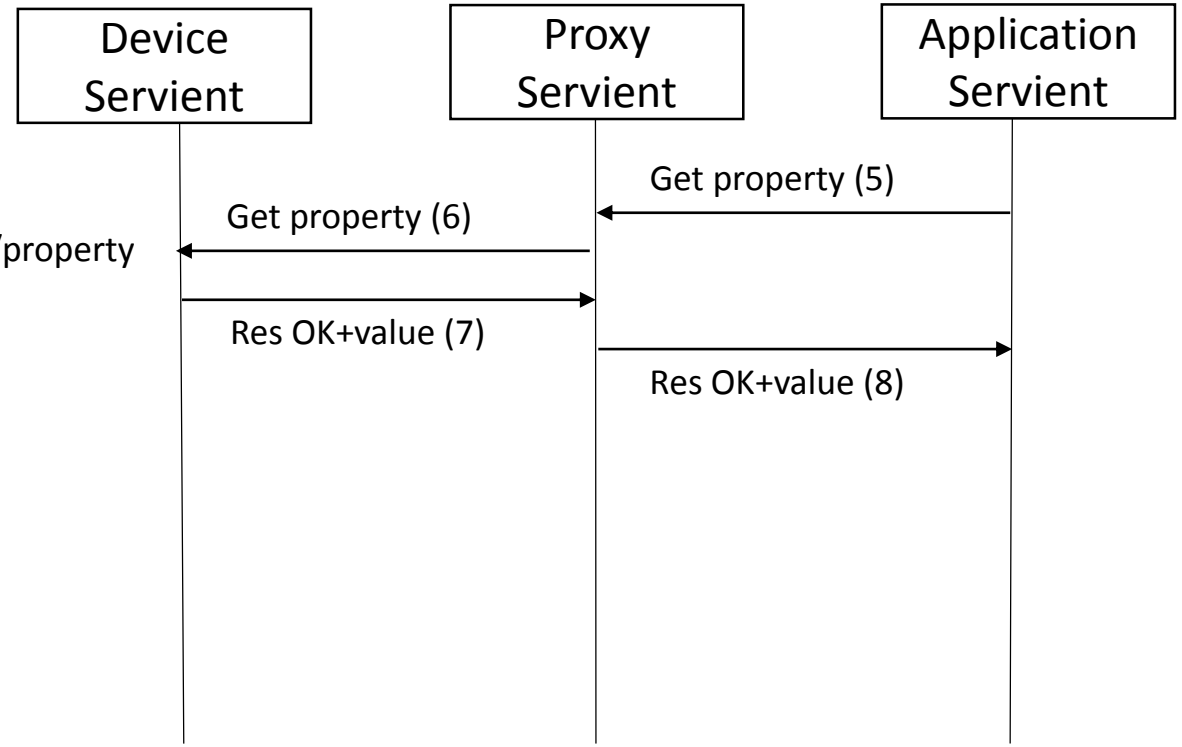
Message (3)
GET http(s)://ps.example.com/lookup?name=LED01
Body:

Message (4)
200 OK
Body: TD



# Basic sequence diagram (2 of 7)

(2) “Get” operation




**Examples**

Message (5)
GET http://ps.example.com/device1/property
Body: none

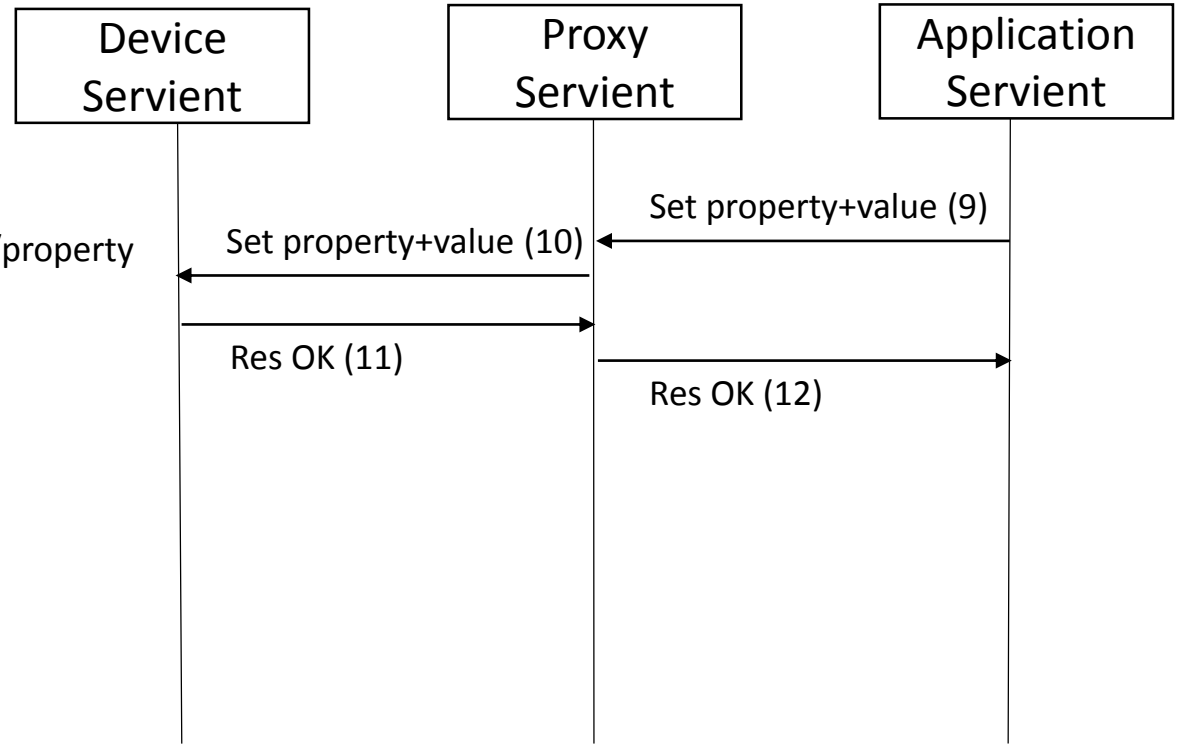
Message (6)
GET http://192.168.0.1/property
Body: none

Message (7)
200 OK
Body: 123(value)

Message (8)
200 OK
Body: 123(value)

# Basic sequence diagram (3 of 7)

(3) “Set” operation




**Examples**

Message (9)
PUT http://ps.example.com/device1/property
Body: ON (value)

Message (10)
PUT http://192.168.0.1/property
Body: ON (value)

Message (11)
200 OK
Body: none

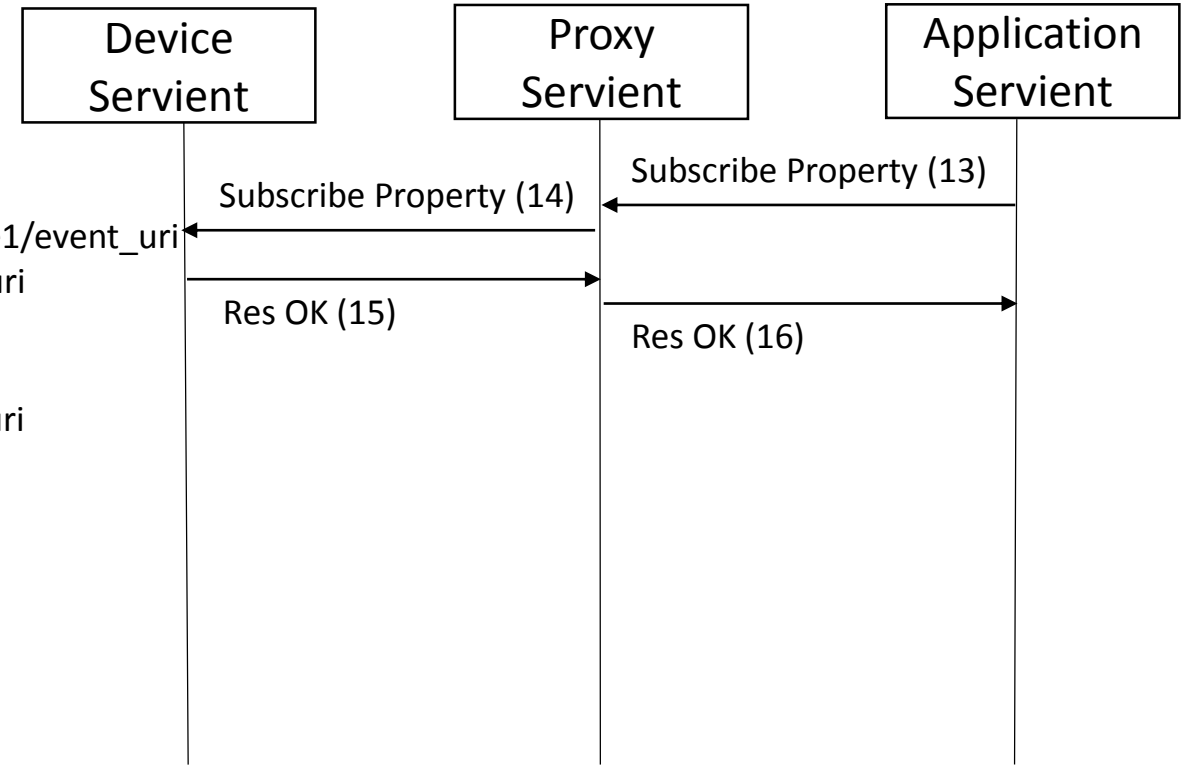
Message (12)
200 OK
Body: none

# Basic sequence diagram (4 of 7)

(4) “Subscribe” operation

**Examples**

Message (13)
POST http://ps.example.com/device1/event_uri
Body: http://as.example.com/app_uri
Message (14)
POST http://192.168.0.1/event_uri
Body: http://as.example.com/app_uri
Message (15)
200 OK
Body: SubscriptionID
Message (16)
200 OK
Body: SubscriptionID

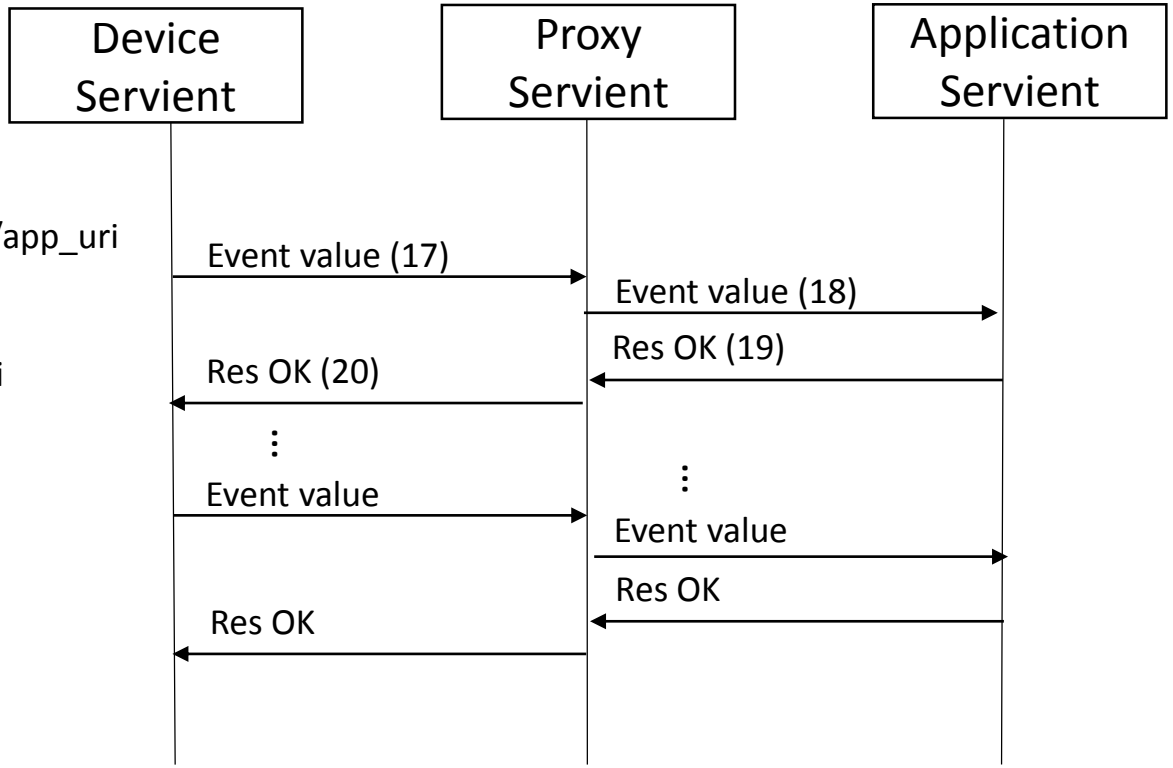


# Basic sequence diagram (5 of 7)

(5) “Event” operation

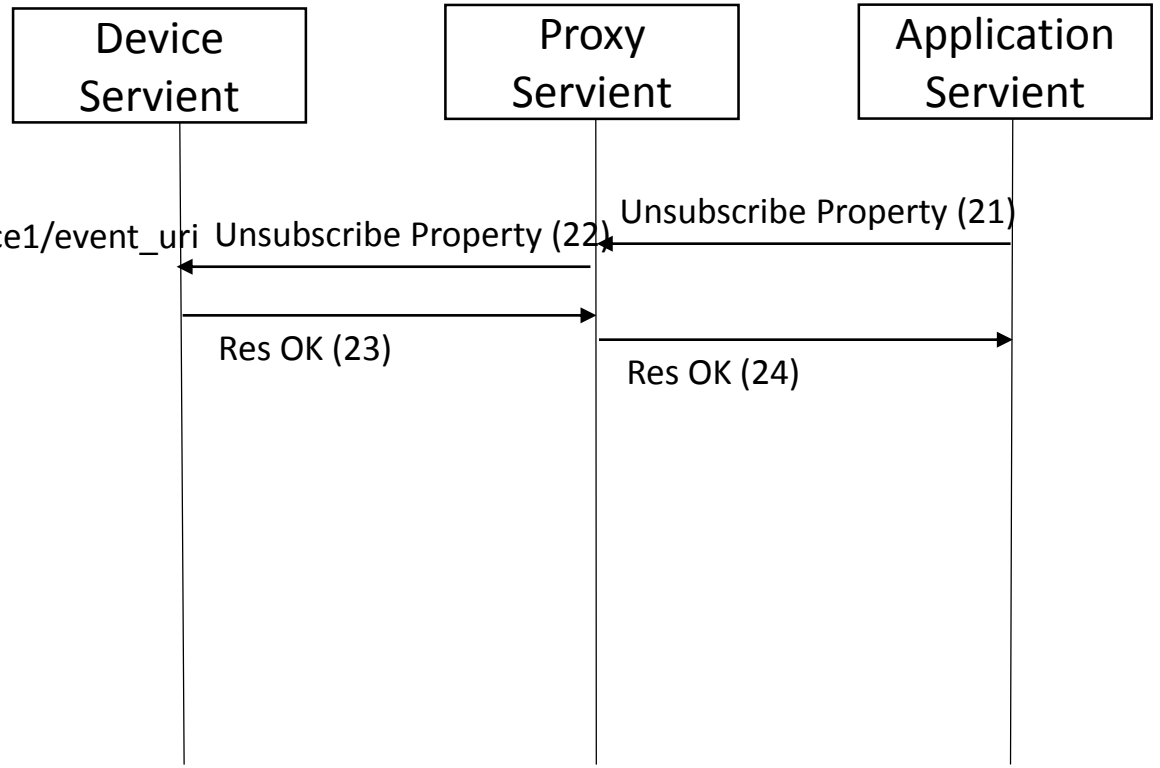
**Examples**

Message (17)
POST http://ps.example.com/device/app_uri
Body: 123 (value)
Message (18)
POST http://as.example.com/app_uri
Body: 123 (value)
Message (19)
200 OK
Body: none
Message (20)
200 OK
Body: none



# Basic sequence diagram (6 of 7)

(6) “Unsubscribe” operation




**Examples**

Message (21)
DELETE http://ps.example.com/device1/event_uri
Body: none

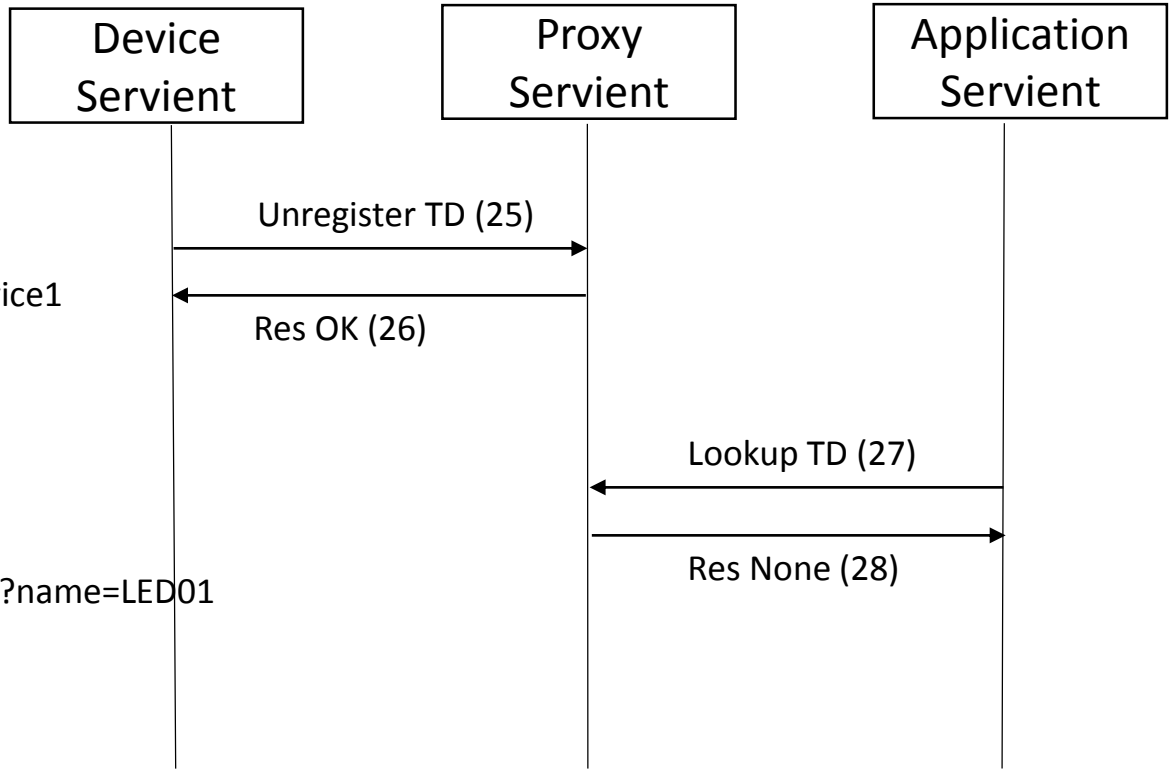
Message (22)
POST http://192.168.0.1/event_uri
Body: none

Message (23)
200 OK
Body: none

Message (24)
200 OK
Body: none

# Basic sequence diagram (7 of 7)

(7) Unregister TD




**Examples**

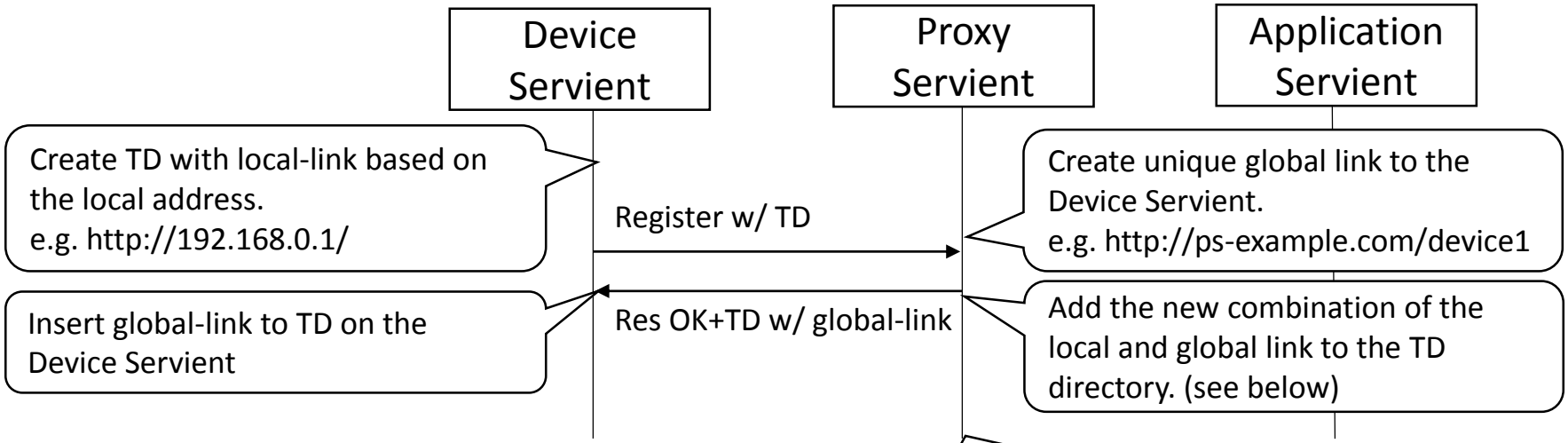
Message (23)
DELETE http(s)://ps.example.com/device1
Body:

Message (24)
200 OK
Body: none

Message (25)
GET http(s)://ps.example.com/lookup?name=LED01
Body: none

Message (26)
404 Not Found
Body: none

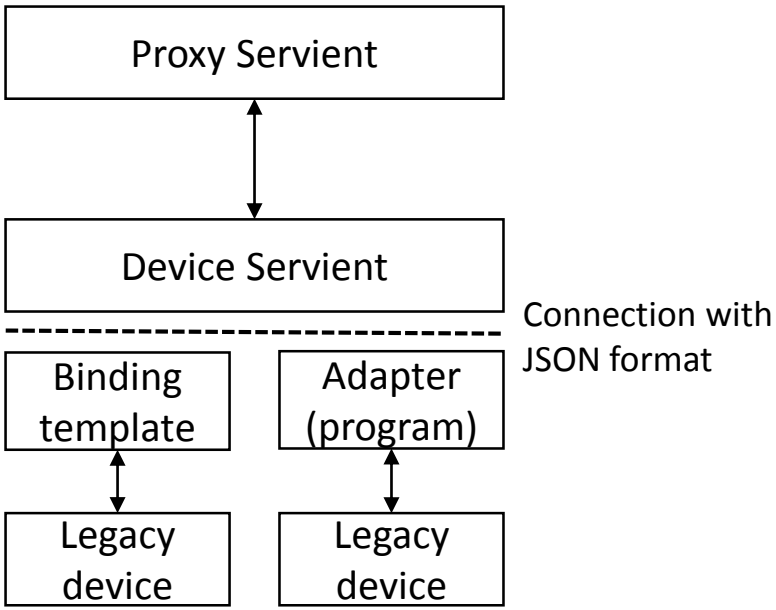
# TD directory on Proxy Servient




| ID | TD | Local-link | Global-link |
|---|---|---|---|
| device1 | Lamp | http://192.168.0.1/ | http://ps.examle.com/device1 |
| device2 | Air conditioner | http://192.168.0.2/ | http://ps.examle.com/device2 |
| application1 | Panel | http://192.168.100.1/ | http://ps.examle.com/application1 |
| | | | |

# Protocol binding for plugfest

- Specify some ways to bind legacy protocols to Servient.
  - Binding template
  - Adapter program



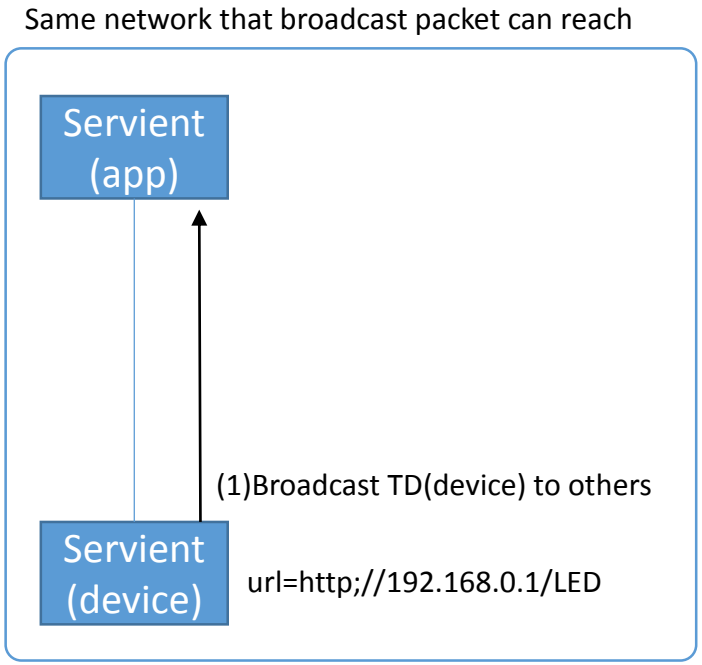
# Authentication

- IETF ACE
- Best practice 8 described in Current Practice

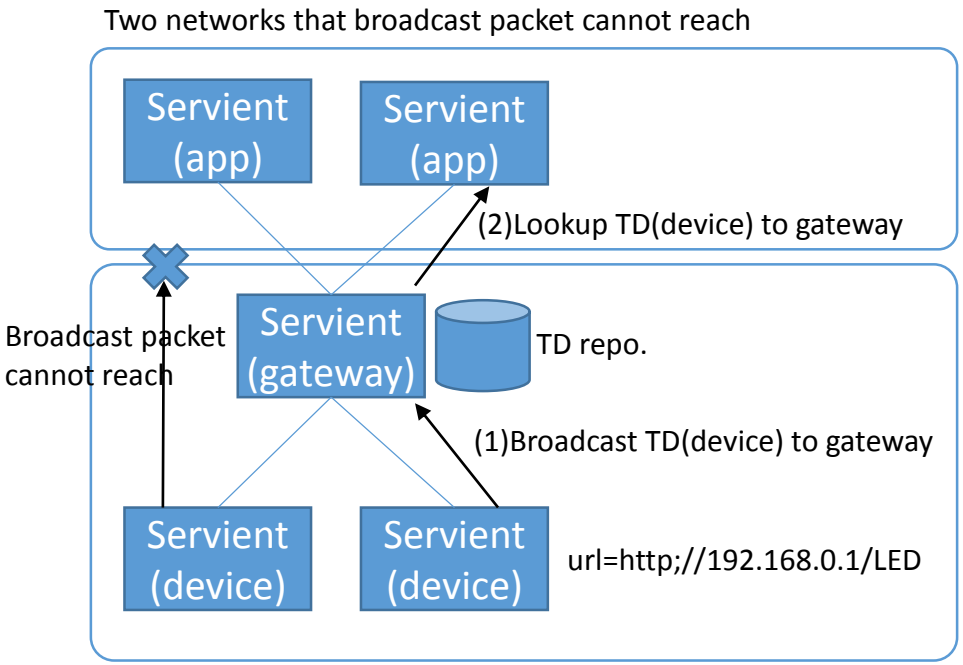
# Discovery and delivery of TD

- 2 Diagrams of PlugFest setting
  - (1) the one in the past and (2) the expected next version at TPAC
- Need for TD management in the future

(1) Diagram for the past plugfest

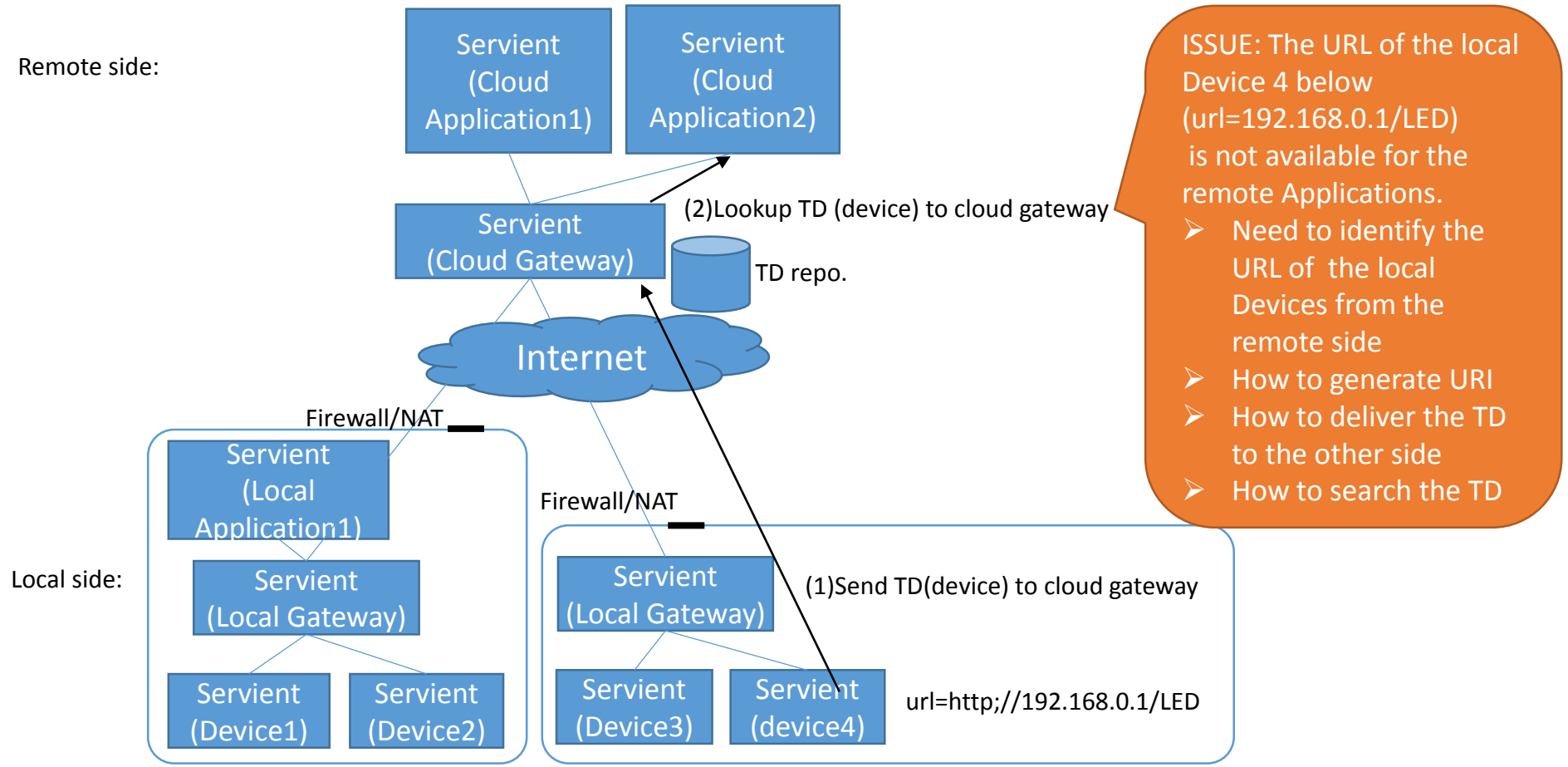



(2) Diagram for the next plugfest



# Need for managing multiple Servients

Point 1: Need to manage multiple Servients for multiple Applications on the remote side
Point 2: Need to manage multiple Servients for multiple Devices on the local side



# NAT Traversal

- TBD

# Need to collaborate

- Architecture TF
  - Add the integration model and the roles of Servients
  - Sequence diagrams for the interactions: “authentication”, “read”, “write”, “subscribe”, and “event”.
- Thing Description TF
  - Discovery and exchange TD
  - TD management