Service Provisioning for Constrained Devices

(draft-vasu-core-ace-service-provisioning-00.txt)

Authors: Vasu, Rahul & Yangneng

Presenters: Jayaraghavendran / Zuojing

Huawei Technologies

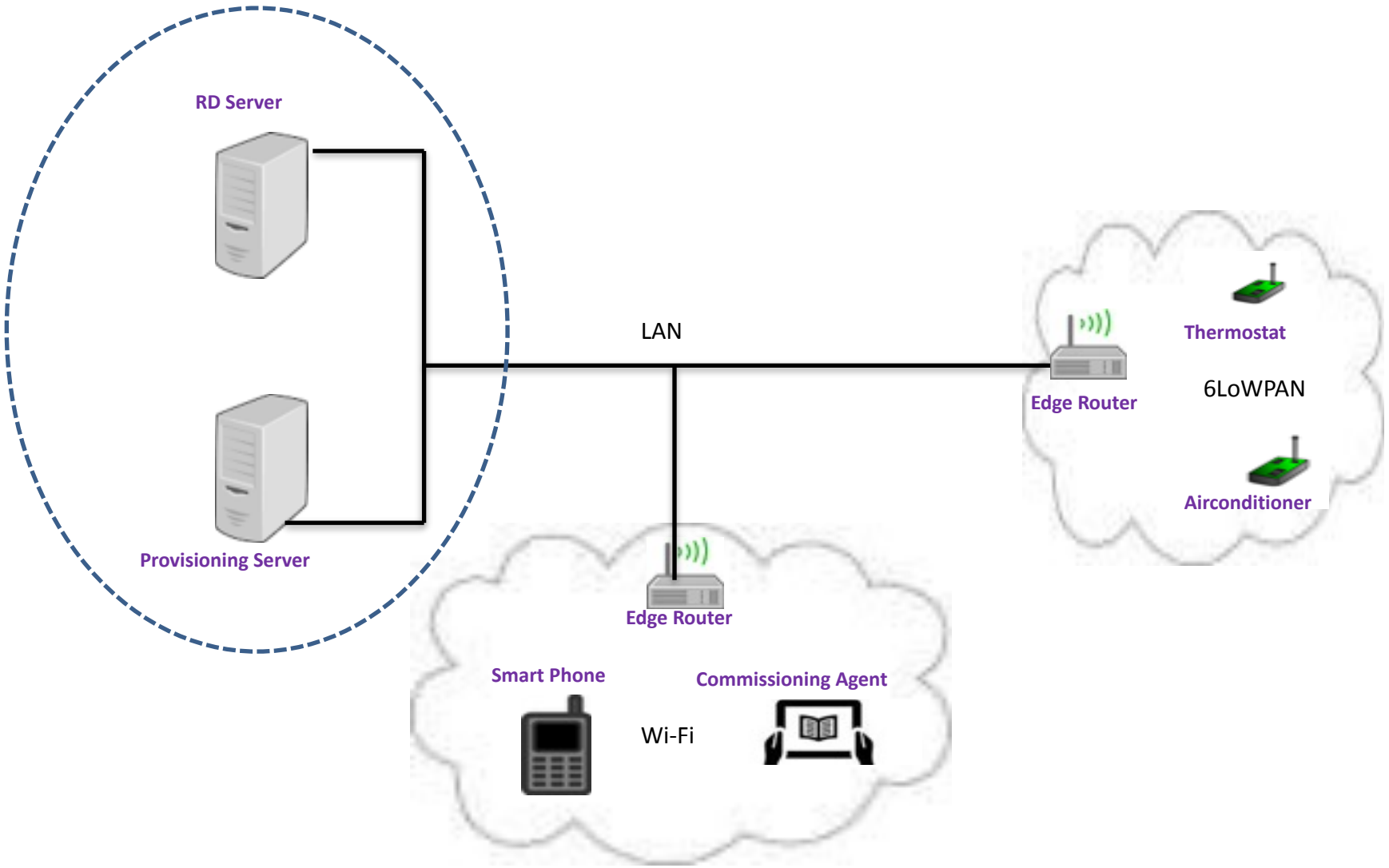
Motivation

- Core-RD : Supports Centralized Operations for Service Registration, Updation, Removal & Lookups initiated by End Points & Clients.
- Core-RD doesn't handle the Provisioning of Services.
- Centralized Provisioning of Services based on following parameters is necessary
  - Authorizing which devices / client applications can access the service
  - Allowing the user to set-up rules about usage permissions for a new service through a commissioning agent
  - Allowing the setting up of service level agreement for QoS & Priority of access to service
- Logical Entities like a Commissioning Agent & Provisioning Server are needed for achieving this goal.

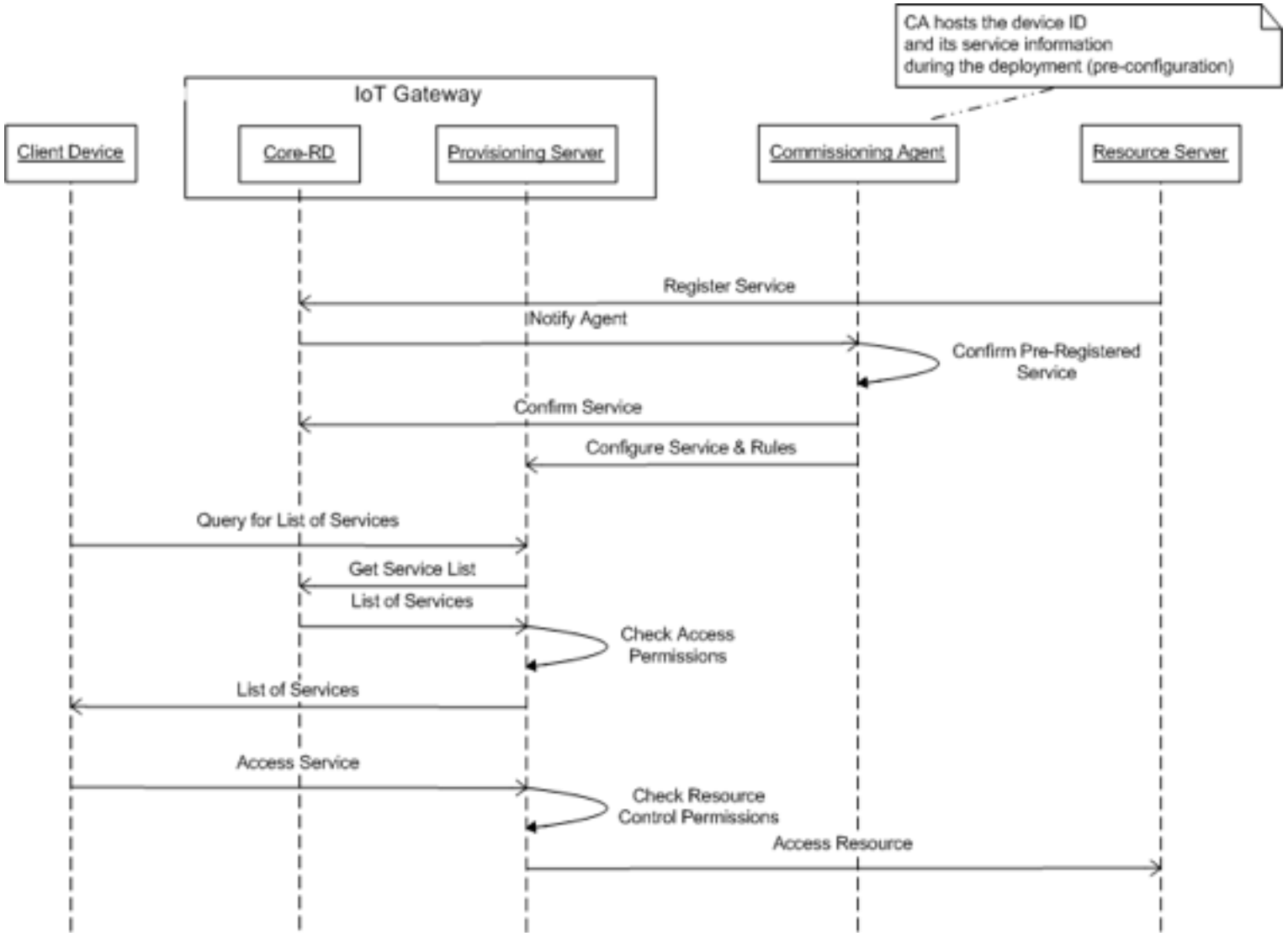
Key Idea

- Apart from the Core-RD, 2 New Logical (or Physical) Entities, Provisioning Server & Commissioning Agent must be added for realizing Service Provisioning
- Every Service Registered with Core-RD is notified to the Commissioning Agent
- Commissioning Agent
  - Checks if the Service is pre-configured with it by User
  - Configures the rules / conditions for accessing the service by different users to the Provisioning Server
- Provisioning Server
  - Stores the rules mapping which user has what privileges for each service
  - Provides information based on the rules when contacted by either the service or the user of the service.

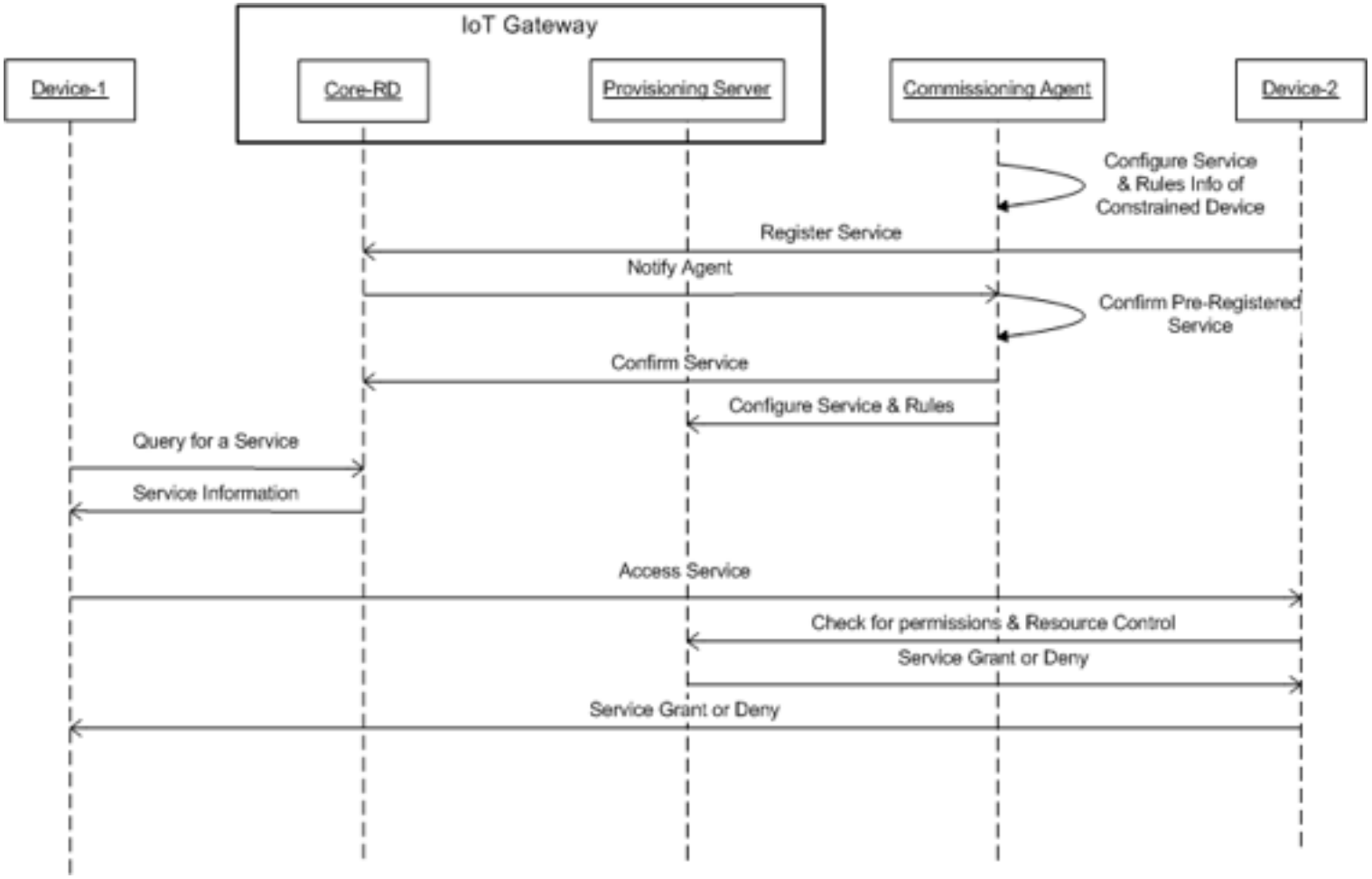
Network Topology



Scenarios


CA hosts the device ID
and its service information
during the deployment (pre-configuration)
IoT Gateway
Client Device
Core-RD
Provisioning Server
Commissioning Agent
Resource Server
Register Service
Notify Agent
Confirm Pre-Registered Service
Confirm Service
Configure Service & Rules
Query for List of Services
Get Service List
List of Services
Check Access Permissions
List of Services
Access Service
Check Resource Control Permissions
Access Resource

Scenarios


IoT Gateway
Device-1
Core-RD
Provisioning Server
Commissioning Agent
Device-2
Configure Service & Rules Info of Constrained Device
Register Service
Notify Agent
Confirm Pre-Registered Service
Confirm Service
Configure Service & Rules
Query for a Service
Service Information
Access Service
Check for permissions & Resource Control
Service Grant or Deny
Service Grant or Deny

Resource Control

Once the hosted service has been verified by commissioning agent (CA), the CA must create a service registry with the provisioning server.

The provisioning server should send a service ID as a response back to the commissioning agent after creating the service entry. This service ID can be later used by the commissioning agent to permanently DELETE the service entry (if required).

The commissioning agent must create some admission control policies such as read (R), write (W), read/ write (R/W), delete (D), number of simultaneous connection on resource etc. on the registered service.

Once the admission control policies has been set on a specific device, the resource control policies such as conditional access of a service, quality of service agreements (based on the priority levels set for clients) can be set on that registered service.

```
Cn
{
        Gm
        {
                Allow {x1,…xk}€{R,W,R/W,U,D};    k is number of permissions allowed for client
                Priority {p1,p2,….pr};    r is total priority classes of client
                QoS {Q1,Q2,…..Qs}; s is total QoS classes allowed for client
                Operations {O1,O2,…….On};n is total possible operations for client
        }
}
```

Resource Control

Example Statement

ST**1**: If the client belongs to G**1** then it is allowed with permissions {R, R/W, U}, priority {P**1**}, QoS {Q**1**}, and operations {turn it up, read}; else if the client belongs to G**2** then it is allowed with permissions {R, W, R/W}, priority {P**3**}, QoS {Q**2**}, and operations {turn it up, read}; else if the client belongs to G**3** then it is allowed with permissions {D}, priority {P**2**}, QoS {Q**3**}, and operations {turn it down}.

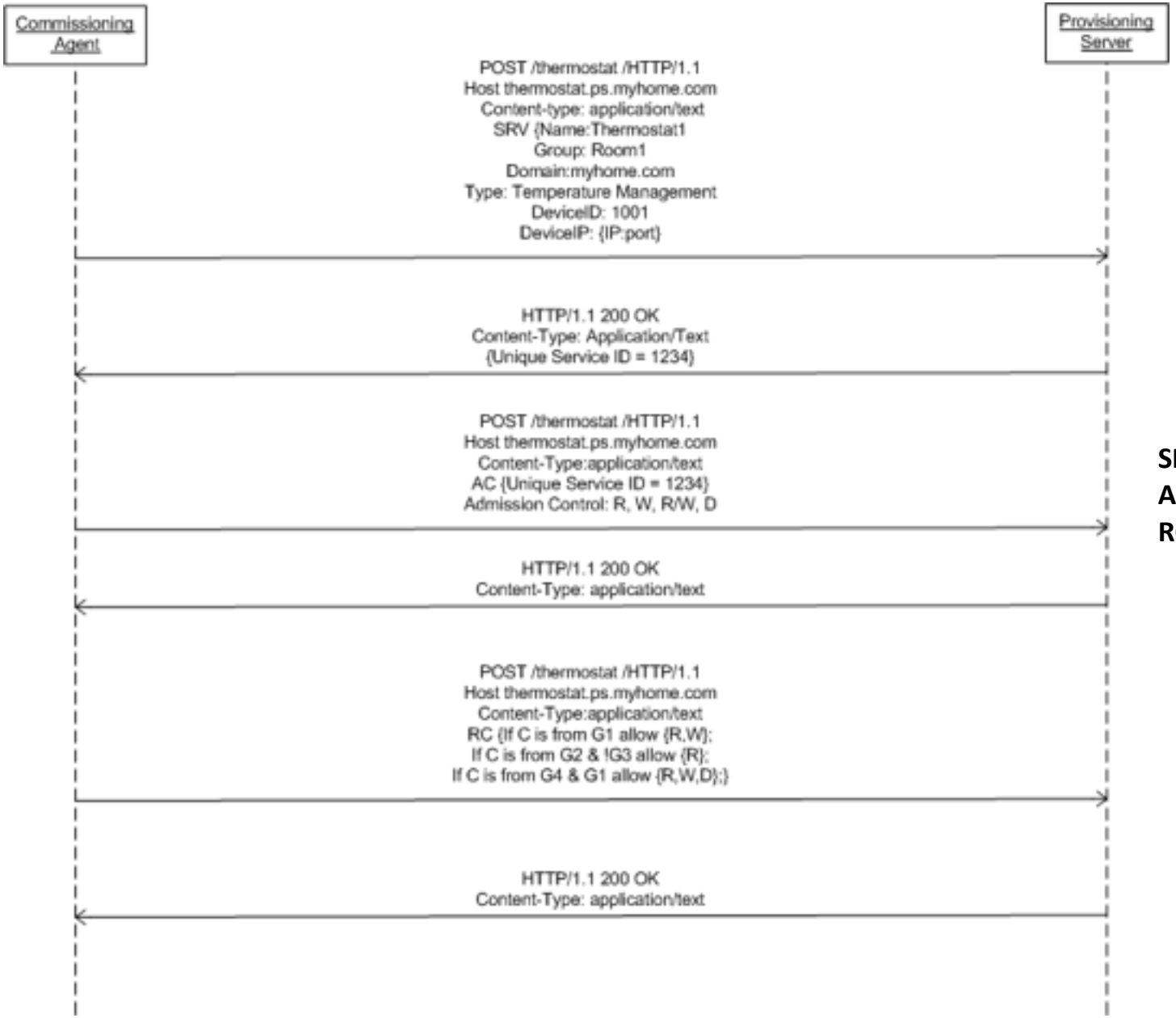
```
C
{
      G1
      {
          Allow {R, U}
          Priority {P1}
          QoS {Q1}
          Operations {turn it up, read}
       }
       G2
       {
          Allow {R,W}
          Priority {P3}
          QoS {Q2}
          Operations {turn it up, read}
       }
       G3
       {
          Allow {D}
          Priority {P2}
          QoS {Q2}
          Operations {turn it down}
       }
}
```

(a) semantic notation

```
"[
"C":{
"G1":{"Allow":"R,U","Priority":
"P1","QoS":"Q1","Operations":"turn
it up,read"},
"G2":{"Allow":"R,W","Priority":"P3",
"QoS":"Q2","Operations":"turn it up,
read"},
"G3":{"Allow":"D","Priority":"P2",
"QoS":"Q3","Operations":"turn it
down"}
}
]"
```

(b) JSON Representation

Defining Policies on Resource Control



SRV: Service Info
AC: Admission Control
RC: Resource Control

Future Thoughts

- Publishing the Provisioning Server details to all nodes? Should it be available through RD?

  ---does it through fixed/automatic discovery of provisioning sever by nodes.

- When devices query for services to RD, should the service details contain the information that they should contact provisioning server instead of device directly? or alternately, device info should only contain the provisioning server info?.
- ---security considerations.

- Add REST based core interfaces for provisioning server. Should we add a core.ps?

  -----Provisioning Server---core.ps----GET,POST,DELETE---json,link-format,cbor

Thank You!