IoTivity on NS-3 Simulator

Aug 26, 2015.
Ikjun Yeom
(Presenter : Gyuhong Choi)
NS-3 Simulator

- **NS-3** is an open-source network simulator for research and educational use.

- **NS-3** works on Linux systems, and is written in C++.

- **NS-3** supports to test various network modules.
  (Ex, TCP/IP, Wi-Fi, WiMax, LTE, 802.15.4, etc)
Research papers using NS-3

- About 750 results for ns-3 simulator on IEEE and ACM Digital Libraries

- Recent publications examples*

* [www.nsnam.org/overview/publications/](http://www.nsnam.org/overview/publications/)
NS-3 Emulation?

- NS-3 also supports emulation.
  - Real code, device and virtual network
  - pros: code-level validation
  - cons: scalability ↓, flexibility ↓
Suggesting IoTivity Simulation

Advantages
- Make ease to test and develop new features. (For developers)
- Can conduct large scale test without devices. (For researchers)
- Can participate with just a PC. (For Newbies)
IoTivity module consists of 3 major parts.
- Application
- Basestack
- CoAP

IoTivity Application
- ThingsManager + Groupserver App
- Lightserver, Bookmark App

IoTivity Basestack
- OCStack, OCResource et al.

CoAP
- Reference implementation of RFC 7252.

NS-3:: UDP/IP

NS-3:: CH. (Wi-Fi, CSMA, LTE etc)
~/ns-3.23/src : modules are here (internet, wifi, wimax, lte, lr-wpan, etc.)

/iotivity : IoTivity module

/examples

  groupaction-wireless : GroupAction scenario file

/helper : Helper classes for scenario

/model

/app : IoTivity applications

  iotivity-application : abstract class for client role

  group-server : group manager app which inherited iotivity-application

  iotivity-server-application : abstract class for server role

  light-server : light app which inherited iotivity-server-application

  bookmark : bookmark app which inherited iotivity-server-application
Module Directory Tree (2)

/internal: IoTivity internal stack, CoAP

oc-stack: class for IoTivity C SDK

oc-action: action, actionset class for group resource

coap-header: CoAP header class

coap-option: class for variable length options of CoAP header

oc-coap: CoAP implementation class

iotivity-base-stack: callback setting class among oc-stack, oc-coap, and apps
NS-3 IoTivity Module Methods (1)

• Application
  - RegisterResource
    : register resource with specific resource URI, type, interface, etc.
  - EntityHandler
    : take a specific action depending on method
  - Find/Put/PostResource
    : request for find/put/post
NS-3 IoTivity Module Methods (2)

- **Oc-stack**
  - **CreateResource**
    : create Resource with parameters from application layer
  - **DoResource**
    : send information for request to the CoAP layer
  - **HandleStackRequests**
    : match a proper handler to received request
  - **HandleStackResponses**
    : perform callback of application layer when response received
  - **HandleResourceWithEntityHandler**
    : call back EntityHandler of application layer
  - **DoResponse**
    : send information for response to the CoAP layer
NS-3 IoTivity Module Methods (3)

- Oc-coap
  - Send
    : make CoAP packet and send it to UDP layer
  - Receive
    : determine whether Con/Non/Ack/Rst
  - RcvCon/Non/Ack/Rst
    : Handle error if any error code
  - HandleRequests/Responses
    : remove CoAP header and forward up requests/responses
NS-3 IoTivity Module Methods (4)

Create
- App: RegisterResource
- Stack: CreateResource
- CoAP: Send
- UDP: Send

Request
- App: Find/Put/Post Resource
- Stack: DoResource
- CoAP: Send
- UDP: RcvCon/Non/Ack/Rst

Handle Request & Response
- App: EntityHandler or Callback Functions
- Stack: HandleStack Requests/Responses
- CoAP: Handle Requests/Responses
- UDP: Receive

Handle
- Stack: DoResponse
We implemented and tested ThingsManager among various IoTivity services.
Step 0. Configure Topology

5 nodes-star topology
- 2 Bulbs, 1 Bookmark, and 1 GroupServer(Phone)
- 1 router of NS-3
Step 1. Find bulbs

Groupserver discovers the bulbs using multicast.
Bulb1 and Bulb2 response to the Groupserver using unicast, then Groupserver sets the Bulbs as "Group".
Step 2. Create ActionSet

Groupserver multicasts to find Group resource and receives response from itself.
Then ActionSet (AllBulbOn/Off) is created on found Group resource.
Groupserver executes ActionSet (AllBulbOff). The off requests are sent to each bulb using unicast.
Step 4. Find Bookmark

**Groupserver** discovers the bookmark using multicast.

Bookmark responses to the **Groupserver** using unicast, then **Groupserver** observe the bookmark’s openness.
Bookmark notifies its openness to the Groupserver, then the Groupserver executes ActionSet (AllBulbOn).
Additionally, using the inherent tools like Flow-Monitor users can get the important figures (delay, overhead, throughput, etc.) of IoTivity test scenarios.
Conclusion

- **NS-3** is widely used network simulator for research
  - Supported for various network modules

- We implemented **IoTivity** as a module in the NS-3 Simulator
  - Make ease to test large-scale
  - Enable fast IoTivity service development through the simulation