Automatic Construction of Name-Bound Virtual Networks for IoT

Kenji Fujikawa <hudikaha@nict.go.jp>
Ved P. Kafle,
Pedro Martinez-Julia,
Abu Hena Al Muktadir
Hiroaki Harai

National Institute of Information and Communications Technology (NICT)

Contents

- Backgrounds
  - Who specifies the names of IoT devices?
  - Who operates naming and addressing for VNs?
- Objectives of Name-Bound Virtual Network (NBVN)
- Proposal of Business Players ASP/VNO/InP Roles and Interactions
- Example Construction of NBVN (a Bike Race Network)
  - VNO Configuration
  - ASP’s request to VNO
  - Request from VNO (vno.rb) to InP1 (inb.rb)
  - Executed Vnn.rb Command on Each Node
  - Assigned Names/Addresses to be Used within the NBVN
- Example Construction of Multiple NBVNs on the Shared Infrastructure
- Conclusions and Future work
Backgrounds (1)

- Networks are constructed for various IoT services
  - e.g. connected vehicles and smart city, etc.

- IoT communication APIs are defined by some organizations
  - IoT devices are recognized by names
  - Who specifies the names of IoT devices when constructing IoT network using a virtual network?

**Who specifies the names of IoT devices?**

![Diagram of virtual network](image)

Backgrounds (2)

- Business players for constructing Virtual Networks (VNs) are defined [2]
  - Service Provider (SP)
  - Virtual Network Operator (VNO)
  - Virtual Network Provider (VNP)
  - Physical Infrastructure Provider (PIP)

- It is not clear who operates naming, addressing, and configuration of name resolution system for the constructed VN?
  - Is the human network manager of VNO responsible for these operations?

  → Propose a method for automatic construction of VNs with named components and name resolution system, called Name-Bound Virtual Network (NBVN)


Objectives of NBVN (Name-Bound Virtual Network)

- Construct NBVNs for IoT services such as outdoor concerts and sporting events
  - Those services are area-bound and time-bound (may last for hours or days)
  - Quick construction is required
  - Automation is important

In this presentation:
- For the automatic construction of NBVNs, re-define business players, and propose their roles and interactions between them
  - Especially, clarify which player operates naming, addressing and construction of a name resolution mechanism for NBVNs
  - Manual operations are avoided as far as possible
- Design and Implement a simple proof-of-concept system

Business Players and Information Flows

- Re-define business players
  - Application Service Provider (ASP)
  - Virtual Network Operator (VNO)
  - Infrastructure Provider (InP)
Proposal of ASP/VNO/InP Roles and Interactions

- ASP manager specifies access point locations and storage/computational server specifications
- VNO specifies network nodes and servers with names to be used in NBVN
  - Addresses/names are automatically assigned

2017/11/16

Clarify the way to specify AP locations for area-bound event network services
Automatic construction of NBVNs realizes time-bound event network services
  - Clear definitions of the roles of ASP/VNO/INP and interactions between them enable automatic construction of NBVN
Example Construction of NBVN
(a Bike Race Event Network)

Network node names are assigned by VNO
- e.g. ap101.race1 sv101.race1

IoT device names are assigned according to the AP names
- e.g. mb1.ap101.race1 mb2.ap101.race1

2017/11/16

Infrastructure Networks

Each InP assigns names to network nodes within the InP
- Note that ASP can assign different names in NBVN

2017/11/16
Developed Programs (Ruby Scripts)

- **vnn.rb**: Virtual Network Node Script
- **inp.rb**: Infrastructure Provider Server Script
- **vno.rb**: Virtual Network Operator Server Script

➢ Read JSON- or YAML-formated data

---

VNO Configuration (vno.rb)

- **vno.rb** is configured with information about InP servers (inp.rb), AP names and locations, computational/storage server names and specs
  ➢ This information should be automatically sent from InP to VNO in the future

```
vno1# cat vno1.conf
inp:
  - { name: inp1.inp1, port: 4001 }
  - { name: inp2.inp1, port: 4001 }
accesspoint:
  - { name: ap1.inp1, latitude: 3, longitude: 1 }
  - { name: ap2.inp1, latitude: 1, longitude: 1 }
  - { name: ap3.inp1, latitude: 1, longitude: 2 }
  - { name: ap4.inp1, latitude: 3, longitude: 2 }
  - { name: ap5.inp1, latitude: 2, longitude: 3 }
  - { name: ap6.inp1, latitude: 2, longitude: 4 }
  - { name: ap1.inp2, latitude: 3, longitude: 5 }
  - { name: ap2.inp2, latitude: 1, longitude: 5 }
  - { name: ap3.inp2, latitude: 1, longitude: 6 }
  - { name: ap4.inp2, latitude: 3, longitude: 6 }
  - { name: ap5.inp2, latitude: 2, longitude: 7 }
  - { name: ap6.inp2, latitude: 2, longitude: 8 }
server:
  - { name: sv1.inp1, memory: 4G, hdd: 10G }
  - { name: sv1.inp2, memory: 4G, hdd: 10G }
vno1# ./vno.rb --port 4000 --conf vno1.conf
```

(1) Inform in advance:
- names and locations of APs
- names and specs of servers
- rough information of network resources
ASP’s request to VNO (vnn.rb)

- ASP sends a request to VNO
  - The network manager of ASP grasps the required locations for the access points, however does not know the names of the access points in those locations

```bash
asp# cat race1.conf
event: race1
accesspoint:
- { latitude: 3, longitude: 1 }
- { latitude: 3, longitude: 2 }
- { latitude: 2, longitude: 3 }
- { latitude: 2, longitude: 4 }
- { latitude: 1, longitude: 5 }
- { latitude: 1, longitude: 6 }
- { latitude: 2, longitude: 7 }
- { latitude: 2, longitude: 8 }
server:
- { memory: 4G, hdd: 10G }
- { memory: 4G, hdd: 10G }
duration:
from: 2016-05-31 06:00:00 +09:00
to: 2016-05-31 12:00:00 +09:00
```

- Include location info
- Does not include names of APs or servers

(2) Request:
- AP locations
- specs of servers
- network resources
- duration
Request from VNO (vno.rb) to InP1 (inb.rb)

- Request includes actual names of access point and server names in InP1

  domain: race1  
  vnno: 3  
  accesspoint:  
  - { nodename: ap101, name: ap1.inp1 }  
  - { nodename: ap102, name: ap4.inp1 }  
  - { nodename: ap103, name: ap5.inp1 }  
  - { nodename: ap104, name: ap6.inp1 }  
  server:  
  - nodename: sv101  
  - name: sv1.inp1  
  - memory: 4G  
  - hdd: 10G  
  - offer: yes  
  duration:  
  - from: 2016-05-31 06:00:00.000000000 +09:00  
  - to: 2016-05-31 12:00:00.000000000 +09:00

2017/11/16

Executed Vnn.rb Command on Each Node and Assigned Names/Addresses to be Used within the NBVN

- InP1 executes vnn.rb on each node

  sv1.inp1# vnn.rb --nodename sv101 --vnno 3 --domain race1 --offer eth1  
  ap1.inp1# vnn.rb --nodename ap101 --vnno 3 --req eth1 --dhcps eth3  
  ap4.inp1# vnn.rb --nodename ap102 --vnno 3 --req eth1,eth2 --dhcps eth3  
  ap5.inp1# vnn.rb --nodename ap103 --vnno 3 --req eth1,eth2 --dhcps eth3  
  ap6.inp1# vnn.rb --nodename ap104 --vnno 3 --req eth1 --dhcps eth3

- IPv6 addresses are automatically assigned to all the network nodes
- IPv6 forwarding tables are also configured
- Name resolution system (DNS) is simultaneously/automatically configured

2017/11/16
We validated that our proposed system practically constructs NBVNs that are used for area/time-bound events.

It is expected to take tens of minutes to construct the NBVN for the event of tens of thousands attendees from our PoC net experiments.

Names and Addresses assigned to Mobile Terminals and IoT Devices

- **ap101.race1** starts a DHCP server with an address space of 2002:db8:3:6::/64, and provides wireless access with SSID containing domain name `race1`.

- Mobile terminals and IoT devices searches SSID containing `race1`, and connect to NBVN race1.

- According to DHCP, access point **ap101.race1** assigns IP addresses and DNS names such as mb1.ap101.race1 and mb2.ap101.race1 to mobile terminals and IoT devices.
We validated that our proposed system practically constructs NBVNs.

It is expected to take tens of minutes to construct the NBVN for the event of tens of thousands attendees from our PoC network experiments.
Conclusions

- Proposed an automatic construction mechanism of NBVNs for IOT
- Re-defined ASP, VNO, and InP, and proposed the roles of ASP/VNO/InP and the required interactions among them
- Developed a proof-of-concept system that implements the operations of ASP/VNO/InP, and automatically constructs NBVNs.
  - IPv6 addresses are automatically assigned to the network nodes and IoT devices
  - The data forwarding and name resolution mechanisms are also automatically configured
- The automatic construction system of NBVNs enables area-/time-bound event-oriented NBVNs for IoT applications such as outdoor concerts and sporting events

Future work
- Network resource (e.g. bandwidth/delay) management
- Function of polling network resources

Packet Formats and Protocol Formats

### VLAN-bound VN

<table>
<thead>
<tr>
<th>Eth header</th>
<th>IP header</th>
<th>Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>VLAN#1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Implemented Name-bound VN

<table>
<thead>
<tr>
<th>Eth header</th>
<th>IP header</th>
<th>Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2002:db8:1</td>
<td></td>
</tr>
</tbody>
</table>

### Essential Name-bound VN

<table>
<thead>
<tr>
<th>Name header</th>
<th>Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain#1</td>
<td></td>
</tr>
</tbody>
</table>

- Current system sets up
- Proposed system sets up

2017/11/16