ONOS WG Meeting
16th Dec. 2016

Path Orchestration REST APIs by using ONOS

(Part of IITP Future Internet Project: multiFIA)

Presented by: Syed Asif Raza Shah
Affiliation: (KISTI/UST, Student Researcher)
Supervised by: Seo Young Noh & Prof. Woojin Seok
email: asif@kisti.re.kr

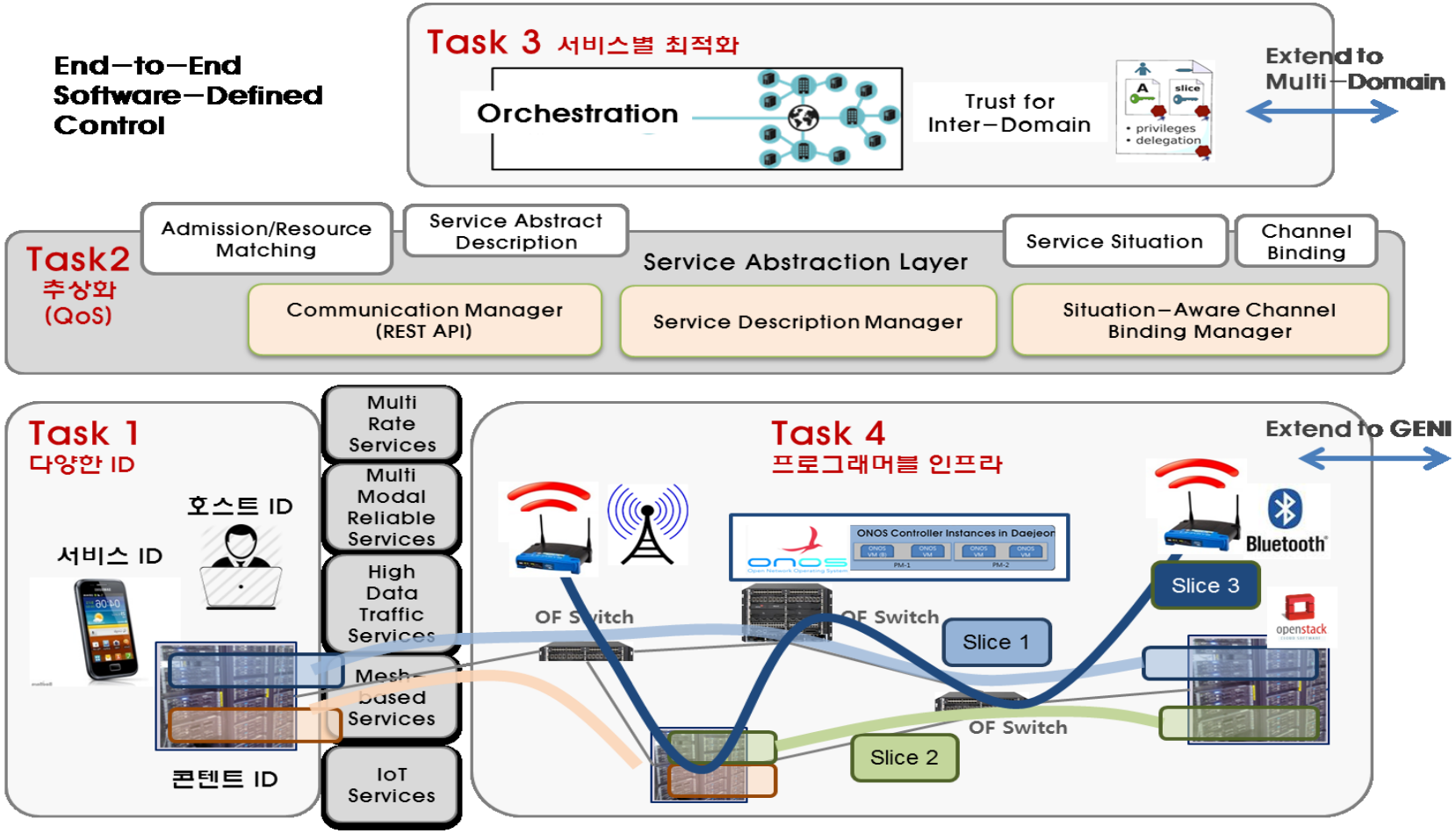
Outline

- Introduction and Background
- Testbed in KISTI
- Path Orchestration REST APIs
- Future Work (QoS REST APIs)

Introduction and Background

- A consortium for Multi Future Internet Architecture (multiFIA):
  - Four Universities + KISTI
  - Working on End-to-End SDC
- End-to-end Software-Defined Control based on a programmable infrastructure:
  - Including various ID processing,
  - Integrated QoS abstraction representation,
  - Service-specific optimization orchestration, terminal / network / cloud resources

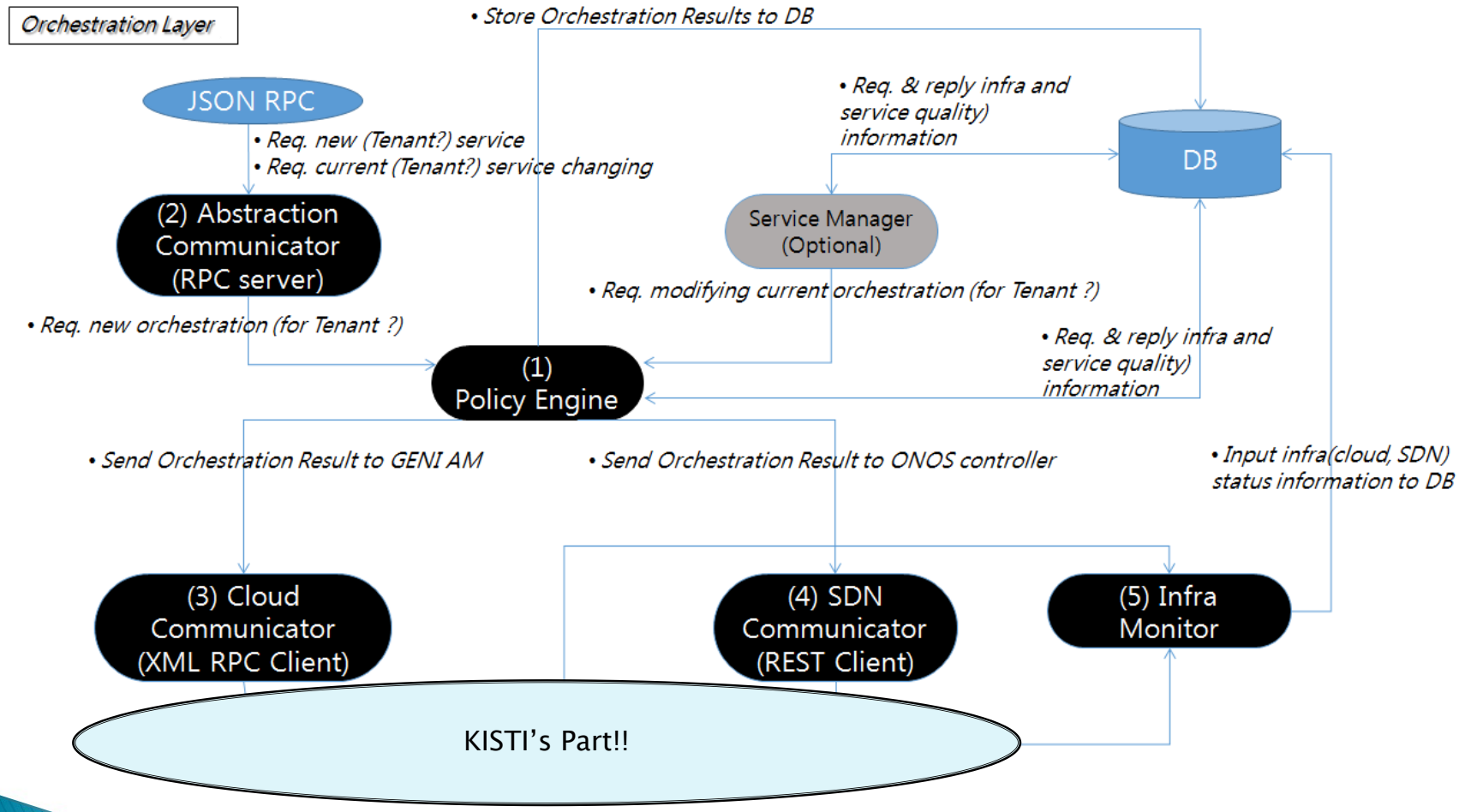
End-to-End Software Defined Control



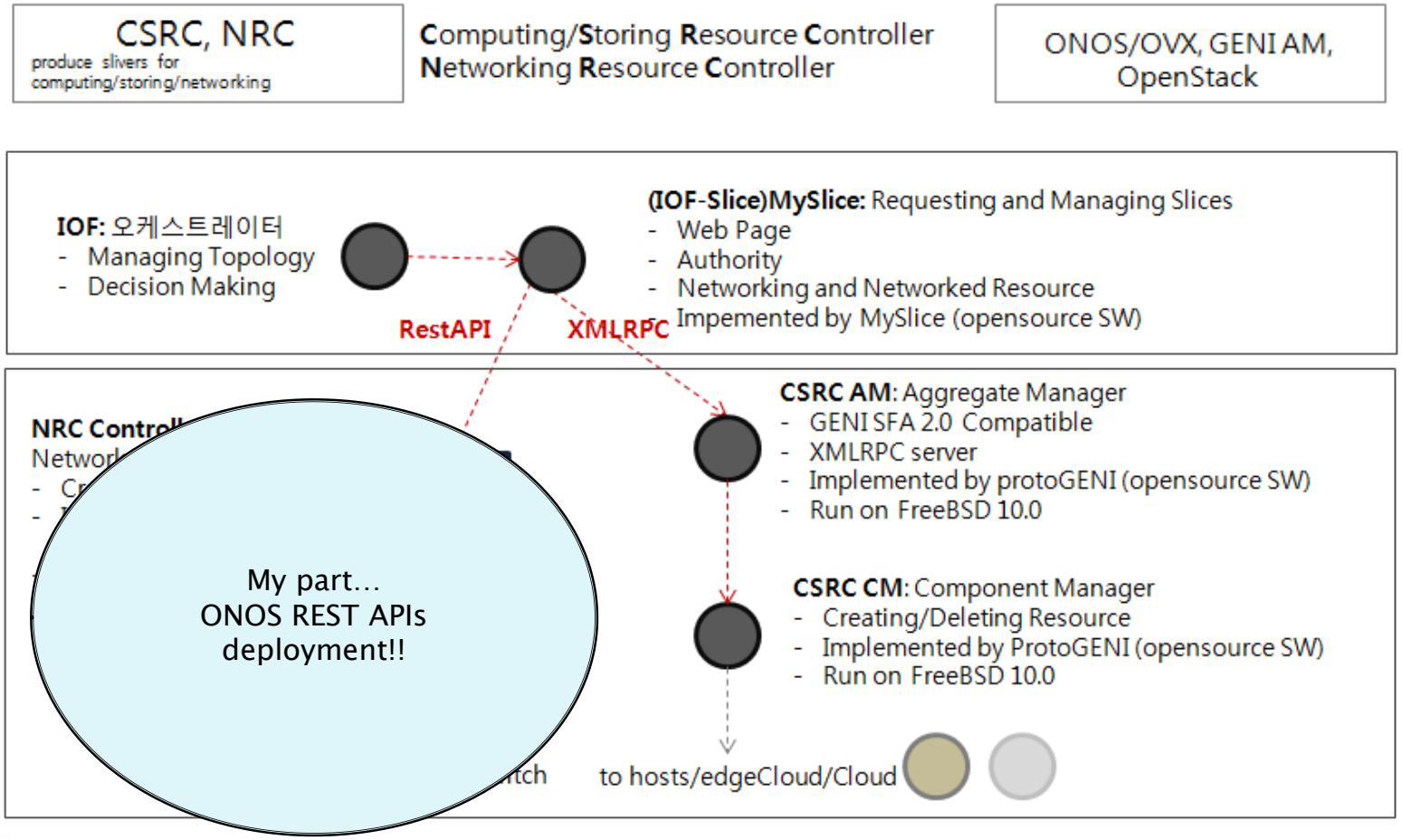
Orchestration Layer

- Intelligent Orchestration for Future application (IOF):
  - Consists of different modules
    - Infrastructure level services
    - Managing topology
    - Decision making
    - Etc…

Orchestration Layer for multiFIA

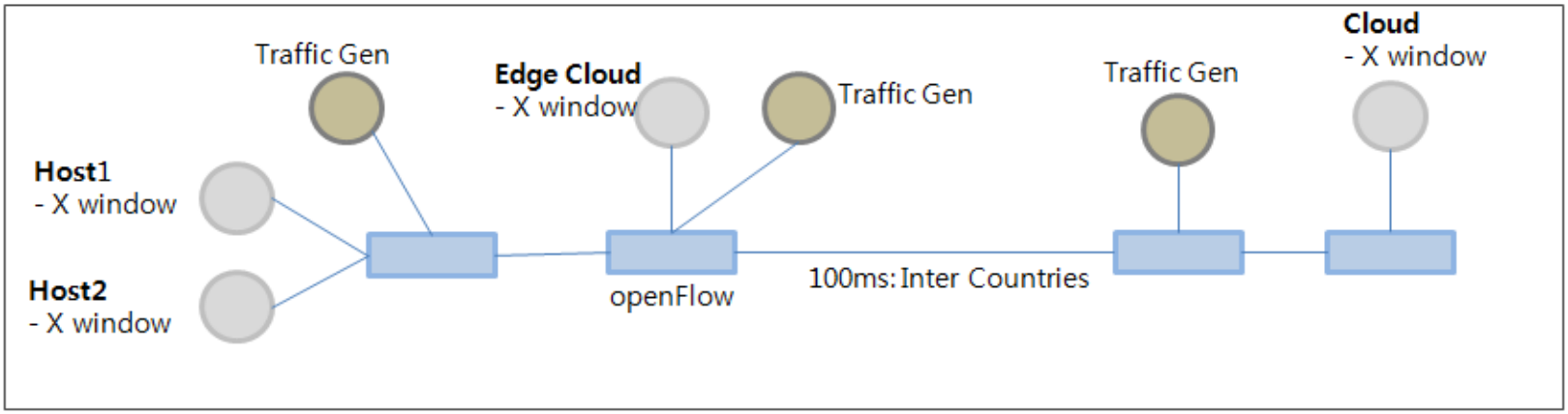


NRC for MultiFIA in KISTI

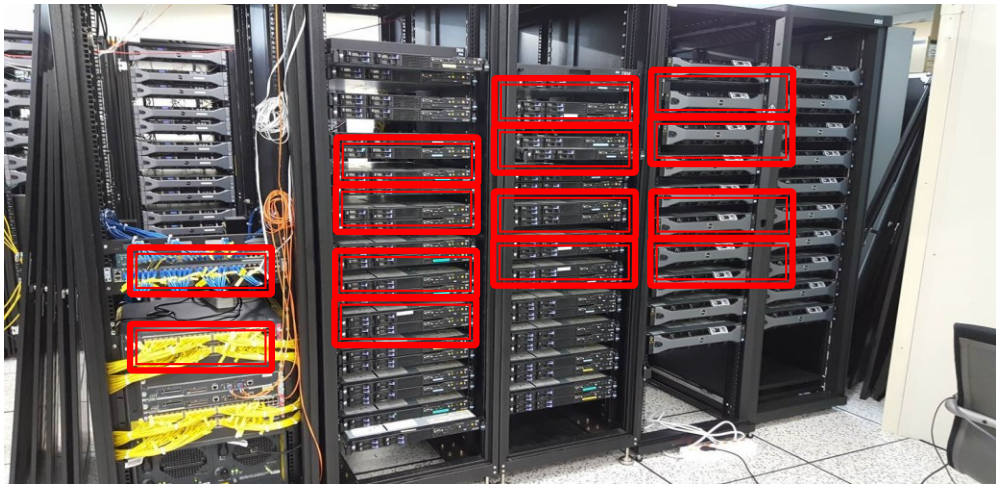


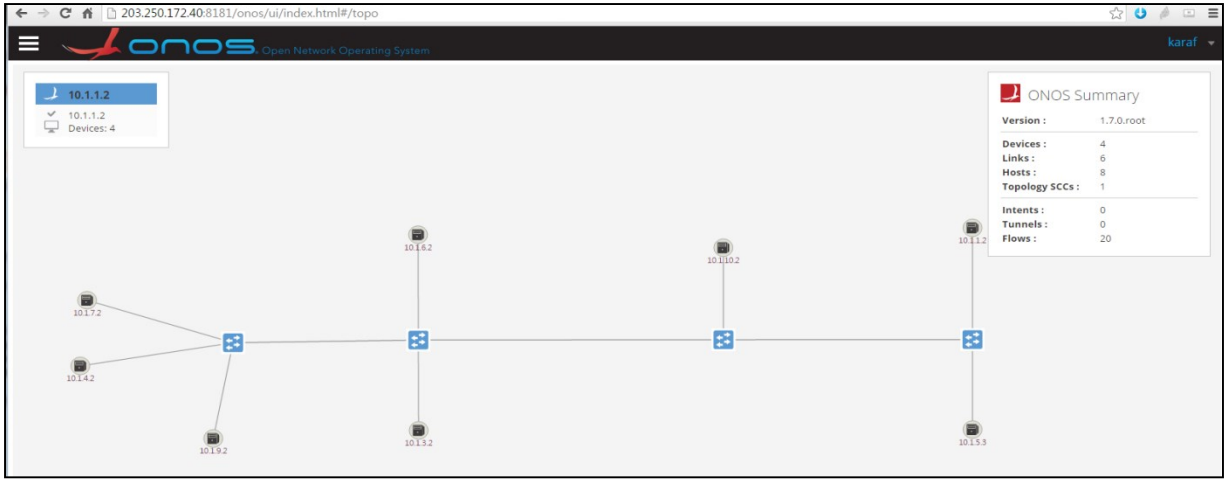
Testbed in KISTI

- Our testbed implemented in Emulab
- It consists of 12 Servers:
  - 1xONOS controller
  - 4xOpenVSwitch
  - 2xHost nodes
  - 2xCloud nodes
  - 3xTraffic Gen. Nodes



Testbed in KISTI (Continue…)

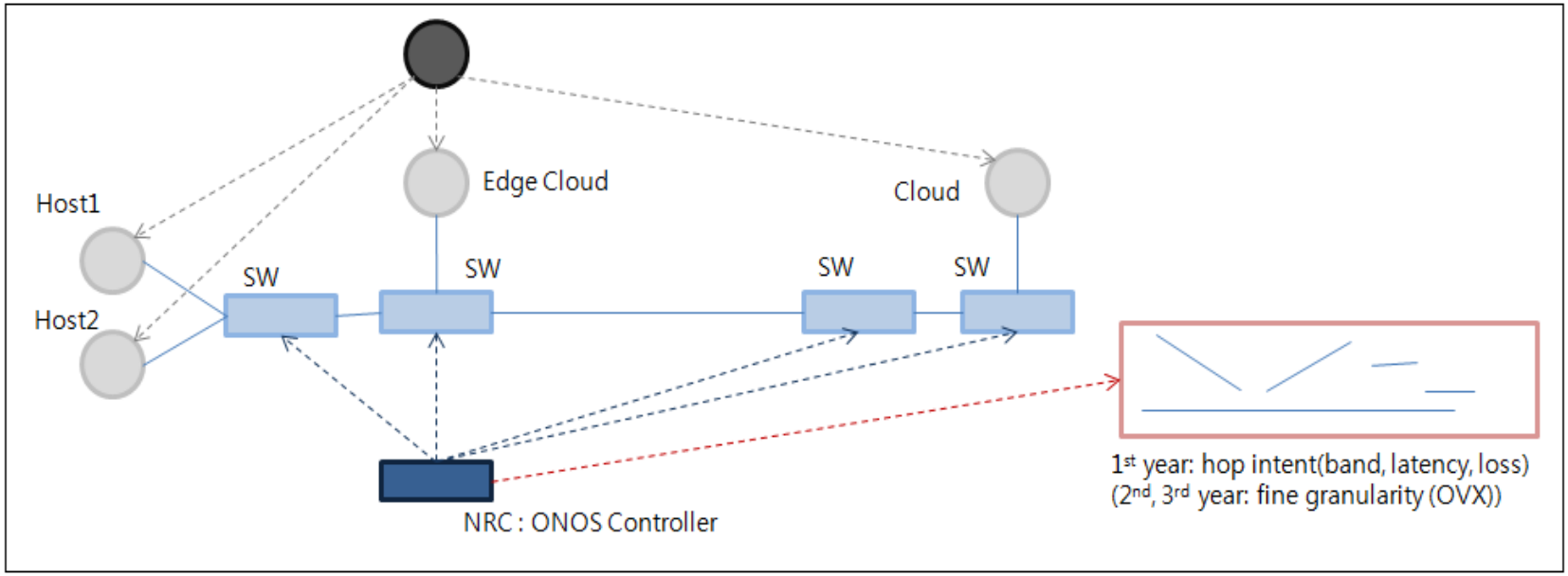




Path Orchestration REST APIs

- **Purpose of REST APIs:**
  - It should provide different information to Policy Engine module of IOF for decision making:
    - Links information
    - Hosts information
    - QoS related information (e.g. Available Bandwidth, Latency, delay etc)
    - Accept path formation request as dictated by Policy engine
    - Etc…

Path Orchestration REST APIs (Cont..)



Path Orchestration REST APIs (Cont..)

- **Initially three Northbound REST APIs for PoC**
- **Rest APIs:**
  - **HOST_info**: providing host information every info (host MAC/PORT, switch ID)
  - **LINK_info:** providing topology information every info (src switch ID/PORT, dst switch ID/PORT)
  - **POST:** request allocating src-to-dst host put (src host info, link_1, link_2, link_n, dst host info)

REST APIs Code

```
@GET
@Path("hosts")
public Response getProjectHosts() {

    final Iterable<Host> hosts = get(HostService.class).getHosts();
    final ObjectNode root = encodeArray(Host.class, "hosts", hosts);
    return ok(root).build();

}
```

Host information Get REST API

```
@GET
@Path("links")
public Response getProjectIntents() {

    final Iterable<Link> links = get(LinkService.class).getLinks();
    final ObjectNode root = encodeArray(Link.class, "links", links);
    return ok(root).build();

}
```

Links information Get REST API

```
@POST
@Consumes(MediaType.APPLICATION_JSON)
@Produces(MediaType.APPLICATION_JSON)
public Response createIntent(InputStream stream) {
    String flowOne=null;
    String flowTwo=null;
    String linkDstSwId=null;
    String ingressPort=null;
    Integer Err=0;

    InputStream flowStream;
    try {
        ObjectNode jsonTree = (ObjectNode) mapper().readTree(stream);
        JsonNode DstHost = jsonTree.path("DstHost");
        JsonNode SrcHost = jsonTree.path("SrcHost");
        ArrayNode PathLinks = jsonTree.get("PathInfo") == null
                ? mapper().createArrayNode() : (ArrayNode) jsonTree.get("PathInfo");

        //When both hosts on same switch
        if(SrcHost.get("SrcSwId").asText().equals(DstHost.get("DstSwId").asText())){
            flowOne = flowSetup(DstHost.get("DstMac").asText(),
                    SrcHost.get("SrcMac").asText(),
                    SrcHost.get("SrcPort").asText(),
                    SrcHost.get("SrcSwId").asText(),
                    DstHost.get("DstPort").asText(),
                    SrcHost.get("SrcSwId").asText());

            flowTwo = flowSetup(SrcHost.get("SrcMac").asText(),
                    DstHost.get("DstMac").asText(),
                    DstHost.get("DstPort").asText(),
                    SrcHost.get("SrcSwId").asText(),
                    SrcHost.get("SrcPort").asText(),
                    SrcHost.get("SrcSwId").asText());

            flowStream = new ByteArrayInputStream(flowOne.getBytes(StandardCharsets.UTF_8));
            pathSetup(flowStream);
            flowStream = new ByteArrayInputStream(flowTwo.getBytes(StandardCharsets.UTF_8));
            pathSetup(flowStream);
        }
```

Post API to accept path information and create path

REST APIs Code

```
for (JsonNode node : PathLinks) {
    //When Source host's Switch ID and Link's Source Switch ID same
    if(SrcHost.get("SrcSwId").asText().equals(node.get("SrcSwId").asText())){
        flowOne = flowSetup(DstHost.get("DstMac").asText(),
                SrcHost.get("SrcMac").asText(),
                SrcHost.get("SrcPort").asText(),
                node.get("SrcSwId").asText(),
                node.get("SrcPort").asText(),
                node.get("SrcSwId").asText());

        flowTwo = flowSetup(SrcHost.get("SrcMac").asText(),
                DstHost.get("DstMac").asText(),
                node.get("SrcPort").asText(),
                node.get("SrcSwId").asText(),
                SrcHost.get("SrcPort").asText(),
                node.get("SrcSwId").asText());

        flowStream = new ByteArrayInputStream(flowOne.getBytes(StandardCharsets.UTF_8));
        pathSetup(flowStream);
        flowStream = new ByteArrayInputStream(flowTwo.getBytes(StandardCharsets.UTF_8));
        pathSetup(flowStream);
    }

    //When Destination host's Switch ID and Link's Destination Switch ID same
    if(DstHost.get("DstSwId").asText().equals(node.get("DstSwId").asText())){
        flowOne = flowSetup(DstHost.get("DstMac").asText(),
                SrcHost.get("SrcMac").asText(),
                node.get("DstPort").asText(),
                node.get("DstSwId").asText(),
                DstHost.get("DstPort").asText(),
                node.get("DstSwId").asText());

        flowTwo = flowSetup(SrcHost.get("SrcMac").asText(),
                DstHost.get("DstMac").asText(),
                DstHost.get("DstPort").asText(),
                node.get("DstSwId").asText(),
                node.get("DstPort").asText(),
                node.get("DstSwId").asText());

        flowStream = new ByteArrayInputStream(flowOne.getBytes(StandardCharsets.UTF_8));
        pathSetup(flowStream);
        flowStream = new ByteArrayInputStream(flowTwo.getBytes(StandardCharsets.UTF_8));
        pathSetup(flowStream);
    }
```

```
//When Destination host's Switch ID and Link's Destination Switch ID same
if(DstHost.get("DstSwId").asText().equals(node.get("DstSwId").asText())){
    flowOne = flowSetup(DstHost.get("DstMac").asText(),
            SrcHost.get("SrcMac").asText(),
            node.get("DstPort").asText(),
            node.get("DstSwId").asText(),
            DstHost.get("DstPort").asText(),
            node.get("DstSwId").asText());

    flowTwo = flowSetup(SrcHost.get("SrcMac").asText(),
            DstHost.get("DstMac").asText(),
            DstHost.get("DstPort").asText(),
            node.get("DstSwId").asText(),
            node.get("DstPort").asText(),
            node.get("DstSwId").asText());

    flowStream = new ByteArrayInputStream(flowOne.getBytes(StandardCharsets.UTF_8));
    pathSetup(flowStream);
    flowStream = new ByteArrayInputStream(flowTwo.getBytes(StandardCharsets.UTF_8));
    pathSetup(flowStream);
}

// when Source Switch Id of link and Dst of link are same then create link path between two switches
if(node.get("SrcSwId").asText().equals(linkDstSwId)){
    flowOne = flowSetup(DstHost.get("DstMac").asText(),
            SrcHost.get("SrcMac").asText(),
            ingressPort,
            node.get("SrcSwId").asText(),
            node.get("SrcPort").asText(),
            node.get("SrcSwId").asText());

    flowTwo = flowSetup(SrcHost.get("SrcMac").asText(),
            DstHost.get("DstMac").asText(),
            node.get("SrcPort").asText(),
            node.get("SrcSwId").asText(),
            ingressPort,
            node.get("SrcSwId").asText());

    flowStream = new ByteArrayInputStream(flowOne.getBytes(StandardCharsets.UTF_8));
    pathSetup(flowStream);
    flowStream = new ByteArrayInputStream(flowTwo.getBytes(StandardCharsets.UTF_8));
    pathSetup(flowStream);
}
linkDstSwId = node.get("DstSwId").asText();
ingressPort = node.get("DstPort").asText();
  }
  return Response.status(Err).entity(stream).build();
}
  catch (IOException e) {
      throw new IllegalArgumentException(e);
}
public void pathSetup(InputStream stream) {
    try {
        IntentService service = get(IntentService.class);
        ObjectNode root = (ObjectNode) mapper().readTree(stream);
        Intent intent = codec(Intent.class).decode(root, this);
        service.submit(intent);
    } catch (IOException ioe) {
        throw new IllegalArgumentException(ioe);
    }
}
```

Source code available on my github:
https://github.com/syedasifraza/project-app

REST API GET Requests

soyeoni — root@dtn2-daejeon:~ — bash — 88×38

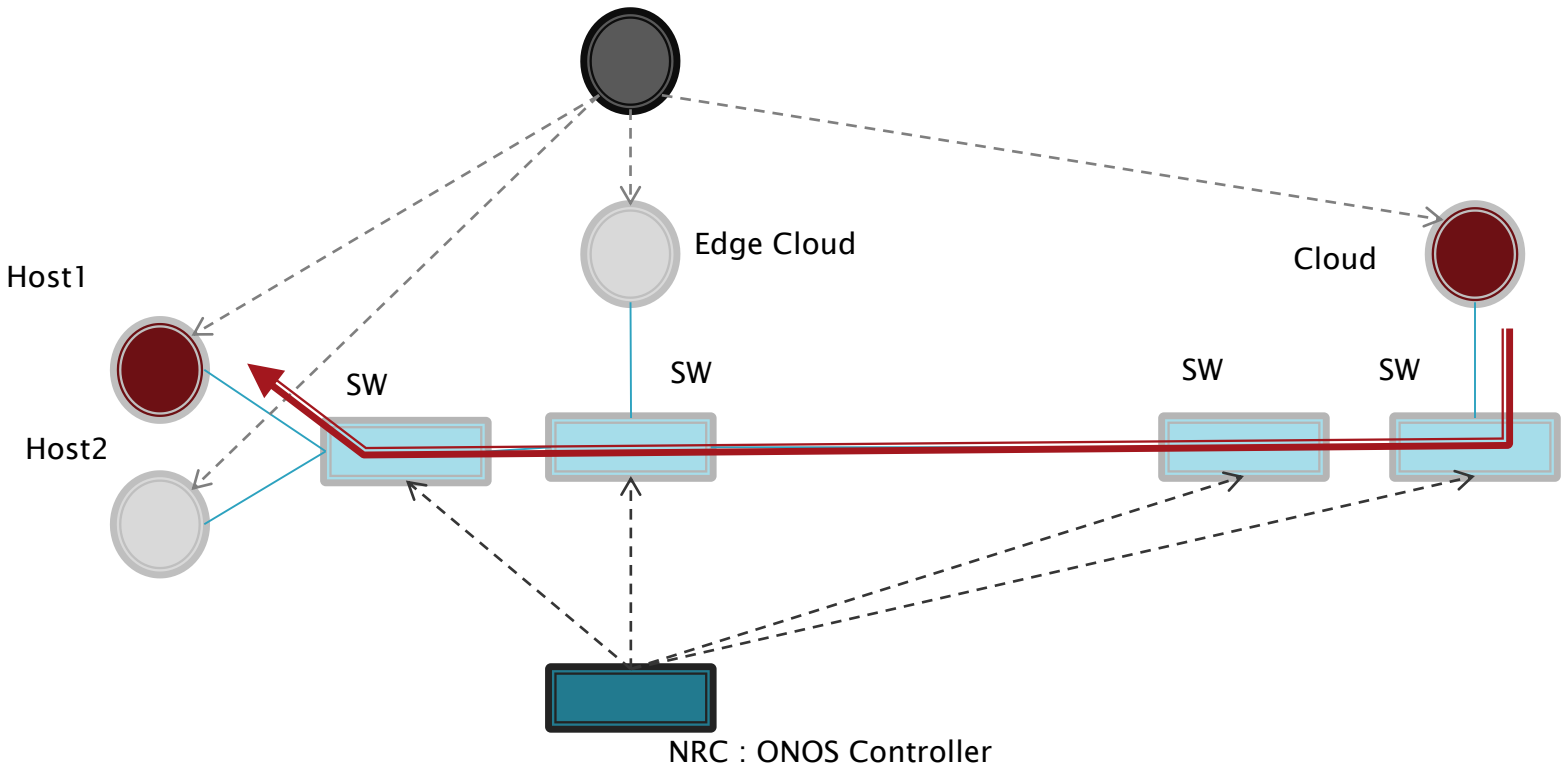
```
soyeon-ui-Mac-Pro:~ soyeoni$
soyeon-ui-Mac-Pro:~ soyeoni$
soyeon-ui-Mac-Pro:~ soyeoni$ curl -u karaf:karaf -X GET --header 'Accept: application/js
on' 'http://203.250.172.22:8181/onos/project-app/path/hosts'
{"hosts":[{"id":"78:2B:CB:13:81:52/None","mac":"78:2B:CB:13:81:52","vlan":"None","ipAddr
esses":["10.1.9.2"],"location":{"elementId":"of:00000010189f19de","port":"2"}},{"id":"78
:2B:CB:28:1A:09/None","mac":"78:2B:CB:28:1A:09","vlan":"None","ipAddresses":["10.1.7.2"]
,"location":{"elementId":"of:00000010189f19de","port":"4"}},{"id":"78:2B:CB:13:81:0A/Non
e","mac":"78:2B:CB:13:81:0A","vlan":"None","ipAddresses":["10.1.1.2"],"location":{"eleme
ntId":"of:0000782bcb138204","port":"1"}},{"id":"78:2B:CB:28:10:9E/None","mac":"78:2B:CB:
28:10:9E","vlan":"None","ipAddresses":["10.1.3.2"],"location":{"elementId":"of:000000101
8a0de5a","port":"1"}},{"id":"78:2B:CB:28:16:51/None","mac":"78:2B:CB:28:16:51","vlan":"N
one","ipAddresses":["10.1.10.2"],"location":{"elementId":"of:0000782bcb281078","port":"2
"}},{"id":"78:2B:CB:28:0F:CF/None","mac":"78:2B:CB:28:0F:CF","vlan":"None","ipAddresses"
:["10.1.5.3"],"location":{"elementId":"of:0000782bcb138204","port":"3"}},{"id":"78:2B:CB
:28:16:36/None","mac":"78:2B:CB:28:16:36","vlan":"None","ipAddresses":["10.1.6.2"],"loca
tion":{"elementId":"of:0000001018a0de5a","port":"3"}},{"id":"78:2B:CB:28:0E:32/None","ma
c":"78:2B:CB:28:0E:32","vlan":"None","ipAddresses":["10.1.4.2"],"location":{"elementId":
"of:00000010189f19de","port":"1"}}]}soyeon-ui-Mac-Pro:~ soyeoni$
```

REST API Post Requests

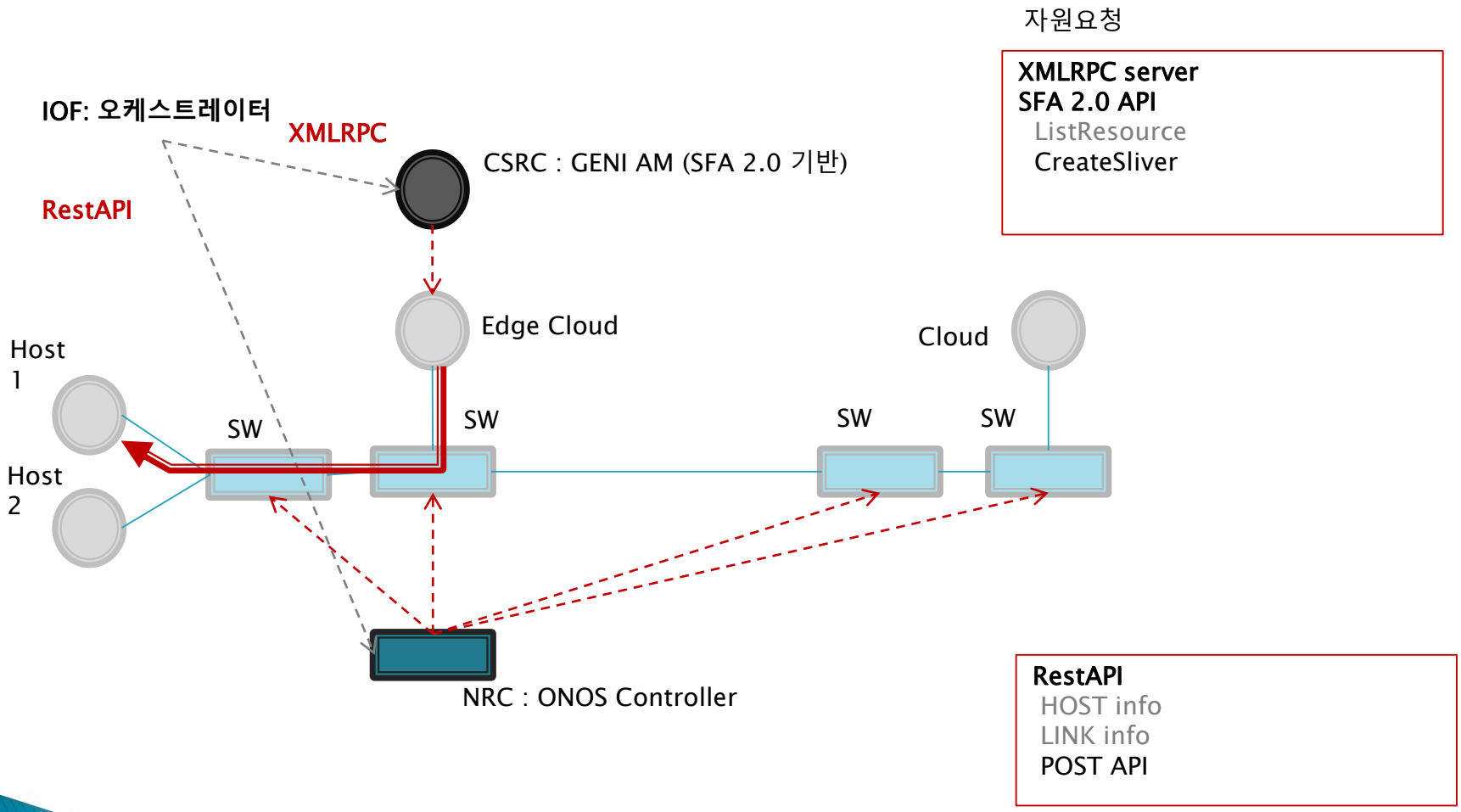
soyeoni — root@dtn2-daejeon:~ — bash — 88×38

```
soyeon-ui-Mac-Pro:~ soyeoni$
soyeon-ui-Mac-Pro:~ soyeoni$
soyeon-ui-Mac-Pro:~ soyeoni$
soyeon-ui-Mac-Pro:~ soyeoni$
soyeon-ui-Mac-Pro:~ soyeoni$ curl -u karaf:karaf -X POST --header 'Content-Type: applica
tion/json' --header 'Accept: application/json' -d '{ \ "SrcHost": \ { \ "SrcPort": "1",
\ "SrcMac": "78:2B:CB:28:0E:32", \ "SrcSwId":"of:00000010189f19de" \ }, \ "DstHost": \ {
 \ "DstPort": "3", \ "DstMac": "78:2B:CB:28:0F:CF", \ "DstSwId":"of:0000782bcb138204" \
}, \ "PathInfo": [ \ { \ "SrcSwId": "of:00000010189f19de", \ "SrcPort": "3", \ "DstSwId"
: "of:0000001018a0de5a", \ "DstPort": "4" \ }, \ \ { \ "SrcSwId": "of:0000001018a0de5a",
 \ "SrcPort": "2", \ "DstSwId": "of:0000782bcb281078", \ "DstPort": "3" \ }, \ \ { \ "Sr
cSwId": "of:0000782bcb281078", \ "SrcPort": "1", \ "DstSwId": "of:0000782bcb138204", \ "
DstPort": "2" \ } \ \ ] \ }' 'http://203.250.172.22:8181/onos/project-app/path'
```

Orchestration Using REST APIs



Orchestration Using REST APIs



Future Work (QoS REST APIs)

- New REST APIs in pipeline:
  - Per Link Latency information
  - End-to-End Delay information
  - Packet loss information
  - Jitter information
  - Link Congestion information
  - Bandwidth allocation using queues
  - Etc..

Thank You!!! ☺
Any Question???