Open Network Operating System

Michele Santuari
msantuari@fbk.eu
FBK CREATE-NET - Future Networks research unit
April 28, 2017

#ONOSProject
Short introduction to SDN and network programmability

Introduction to ONOS:
   Architecture, Northbound and Southbound APIs, Roadmap

Hands-on session:
   Running an ONOS cluster, deploy a simple application
Network programmability
Traditional networking paradigm

**Forwarding HW**

**Data plane**

- Router or switch

**Control plane**

- **Control functions**
  - E.g. routing, isolation, traffic engineering

- **State distribution mechanism**
  - E.g. topology, link utilization

**State distribution mechanism**

- Standard protocol
- Standard protocol 2
- Standard protocol 3
Software-Defined Networking

Control plane
Logically centralized

Data plane

Network OS

Topology graph + configurations

Forwarding HW

Handles state distribution, data plane configuration management, error recovery...

App

Forwarding HW

Forwarding HW

Forwarding HW

Forwarding HW

Forwarding HW

Forwarding HW
Open Network Operating System
ONOS
What is ONOS?

Open Network Operating System (ONOS) is an open source SDN network operating system. Our mission is to enable Service Providers to build real SDN/NFV Solutions.

Quarterly Releases, Junco (1.9.0) - released 2017-03
Open Network Operating System

Northbound API

Southbound API

Network OS

Forwarding HW

Forwarding HW

Forwarding HW

Forwarding HW

Forwarding HW
Architectural Tenets

● High-availability, scalability and performance
  ○ required to sustain demands of service provider & enterprise networks

● Strong abstractions and simplicity
  ○ required for development of apps and solutions

● Protocol and device behaviour independence
  ○ avoid contouring and deformation due to protocol specifics

● Separation of concerns and modularity
  ○ allow tailoring and customization without speciating the code-base
ONOS Architecture

- **GUI**
- **REST API**
- **ONOS applications**
- **ONOS networking core**
- **ONOS distributed applications platform**
- **OSGI / Apache Karaf**

#ONOSProject
ONOS Architecture

- GUI
- REST API

Northbound extensions
- ONOS Applications
- Providers
- Protocols
- Driver

ONOS networking core
ONOS distributed applications platform
OSGI / Apache Karaf

#ONOSProject
ONOS Architecture

- **ONOS applications**
- **ONOS networking core**
- **ONOS distributed applications platform**
- **OSGI / Apache Karaf**

**Interfaces**
- **GUI**
- **REST API**

**Tools**
- Command Line
Distributed Core
ONOS Distributed Architecture

- **Distributed**
  - Set up as a cluster of instances
- **Symmetric**
  - Each instance runs identical software and configuration
- **Fault-tolerant**
  - Cluster remains operational in the face of node failures
- **Location Transparent**
  - A client can interact with any instance. The cluster presents the abstraction of a single logical instance
- **Dynamic**
  - The cluster can be scaled up/down to meet usage demands
ONOS Cluster

ONOS 1

Master

ONOS 2

Standby

ONOS 3

#ONOSProject
ONOS Cluster
ONOS Distributed Architecture

Apps

NB Core API

Distributed Core
(state management, notifications, high-availability & scale-out)

SB Core API

Providers

Protocols

Providers

Protocols

Providers

Protocols

Providers

Protocols
ONOS offers a set of **distributed primitives** used to manage state and coordination across the cluster:

- mimic normal Java APIs like maps, sets, queues and locks
- provide functionality for central to distributed coordination
- support Eventually Consistent and Strong Consistent
ONOS Core Subsystem Structure

Manager Component

Application Component

Listener

AdminService

Service

Sync & Persist

ProviderRegistry

ProviderService

Provider

Environment

Events

Register & Unregister

Command

Query & Command

Notify

Provider

Component

Drivers

Protocols

ONOS instance

ONOS instance

APPLICATION COMPONENT

LISTENER

ADMIN SERVICE

SERVICE

SYNC & PERSIST

PROVIDER REGISTRY

PROVIDER SERVICE

PROVIDER

ENVIRONMENT

EVENTS

REGISTER & UNREGISTER

COMMAND

QUERY & COMMAND

NOTIFY

APPLICATION COMPONENT

LISTENER

ADMIN SERVICE

SERVICE

SYNC & PERSIST

PROVIDER REGISTRY

PROVIDER SERVICE

PROVIDER

ENVIRONMENT

EVENTS

REGISTER & UNREGISTER

COMMAND

QUERY & COMMAND

NOTIFY

#ONOSProject

21
Northbound
Key Northbound Abstractions

- **Network Graph**
  - Directed graph

- **Device-centric programming data planes**
  - FlowRule/FlowObjective

- **Network-centric abstraction for programming data-plane**
  - Intent
Network graph abstraction

Directed graph comprising of infrastructure devices, infrastructure links and end-station hosts

ONOS Network Graph

Network

#ONOSProject
FlowRule/FlowObjective abstraction

Device-centric abstraction for programming data-plane flows

Device-centric configuration abstraction offered to applications

Abstraction translated into device-specific configurations

FlowRule

FlowObjective

Driver X

Driver Y

Driver Z

Forwarding HW X

Forwarding HW Y

Forwarding HW Z pipeline A

Forwarding HW Z pipeline B

Pipeline A

Pipeline B

Device-specific configurations translated into multiple configurations based on the pipeline of the device.
Intent Framework

• network-centric programming abstraction
  ○ intents allow network configuration interface that focuses on what should be done rather than how it is specifically programmed

• device-agnostic behavior
  ○ abstracts unnecessary network complexity from applications

• persistency
  ○ maintains requested semantics as network changes
Intent Example

Intent Service API

Host to Host Intent

Link Collection Intent

Flow Rule Batch

Flow Rule Batch

Flow Rule Batch

Flow Rule Batch

COMPILATION
Path selection

INSTALLATION
Network Programming

Abstract to concrete

Intent
- DC Clos Fabric
- Packet/Optical WAN
- Enterprise Campus

Flow Objective
- OFDPA Pipeline
- Single Table Pipeline
- SpringOpen Pipeline

Flow Rule
- OF 1.0
- OF 1.3
- Netconf
- TL1

#ONOSProject
Interact with ONOS: GUI

UI: <onos-ip>:8181/onos/ui
Interact with ONOS: REST and GRPC

REST APIs: `<onos-ip>:8181/onos/v1/docs/`

<table>
<thead>
<tr>
<th>flows</th>
<th>Query and program flow rules</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DELETE</strong></td>
<td><code>/flows</code></td>
</tr>
<tr>
<td><strong>GET</strong></td>
<td><code>/flows</code></td>
</tr>
<tr>
<td><strong>POST</strong></td>
<td><code>/flows</code></td>
</tr>
<tr>
<td><strong>DELETE</strong></td>
<td><code>/flows/{deviceId}/flowId</code></td>
</tr>
<tr>
<td><strong>GET</strong></td>
<td><code>/flows/{deviceId}/flowId</code></td>
</tr>
<tr>
<td><strong>GET</strong></td>
<td><code>/flows/{deviceId}</code></td>
</tr>
<tr>
<td><strong>POST</strong></td>
<td><code>/flows/{deviceId}</code></td>
</tr>
</tbody>
</table>

Northbound GRPC with protobuf for ONOS network model → in progress
Interact with ONOS: CLI

$onos <controller_address>
Southbound
Southbound Architecture

• Southbound abstractions, modularity
  • customization without changing the core
• Protocol and device model independency
  • avoid specifics and dependencies in the core
  • hidden complexity to upper layers
  • testability, extensibility and performance
ONOS Providers

Providers are used by the core to **(re)act on the network:**

- Up/down of device, links
  - DeviceProvider, LinkProvider
- Provisioning of rules, paths, tunnels
  - FlowRuleProvider, TunnelProvider
- Receive notifications/alarms
  - AlarmProvider

Translate to and from Core abstractions into device specific commands
ONOS Protocols

- Features and modules to communicate with devices
- Expose the standard set of APIs and enabled operations. I.e:
  - OpenFlow: FlowMods, GroupMods, etc
  - Rest: implements CRUD operations (GET, POST, DELETE, etc...)
  - Netconf: Open/close session, setConfiguration, getConfiguration
- Usually leverage 3rd party communication libraries → openflowj, snmp4j, thrift
ONOS drivers

• Device specific driver
  • collection of behaviors
  • on-demand activation

• Abstraction via behaviors
  • define specific capabilities offered by the device
  • encapsulate specific logic and code
    • ports, controller, flowrule, power…

• Encapsulate single interaction
  • protocol
  • information

```xml
<driver name="default" manufacturer="ON.Lab"
  hwVersion="0.0.1" swVersion="0.0.1">
  <behaviour api=InterfacePath
    impl=ImpementationPath />
</driver>
```
Southbound overview

Southbound protocols in 1.9.0:

- OpenFlow until 1.3 + optical extension → 1.5 is in the works.
- OVSDB
- NETCONF + YANG → Yang tools and Yang management system
- SNMP
- P4 → thrift api for bmv2 softswitch from barefoot networks.
- BGP, ISIS, OSPF → interoperability with legacy network.
- PCEP → Path computation element protocol (IETF)
- REST and RESTCONF
- LISP
- TL1
ONOS Roadmap
How to get involved

● **Open Source software** → scratch your own itch
● **Bug Bounty** → start small with a simple bug
  ○ [Jira bugs](#)
● **Application or Use Case** → create your own app to deploy your use case
  ○ [Creating and deploying and ONOS App](#) and [Template application tutorial](#)
● **Brigades** → dynamic configuration, virtualization, GUI, deployments
  ○ [Brigades wiki](#)
● **Collaborator proposal** → create, use and maintain your own ONOS subsystem
ONOS Brigades

Active Brigades

- Dynamic configuration
- ONOS Deployments
- Virtualization
- GUI

New Brigades (since Q1 of 2017)

- Northbound
- Teaching
- gRPC
- Build and Package Infrastructure
- P4
- ECOMP/Mano/Open-O integration brigade
ONOS Ambassadors

https://ambassadors.onlab.us
Thank! Questions?

Michele Santuari
msantuari@fbk.eu
FBK CREATE-NET - Future network area