ETRI-TAM Subproject Development Plan

SangSik Yoon

Taesang Choi

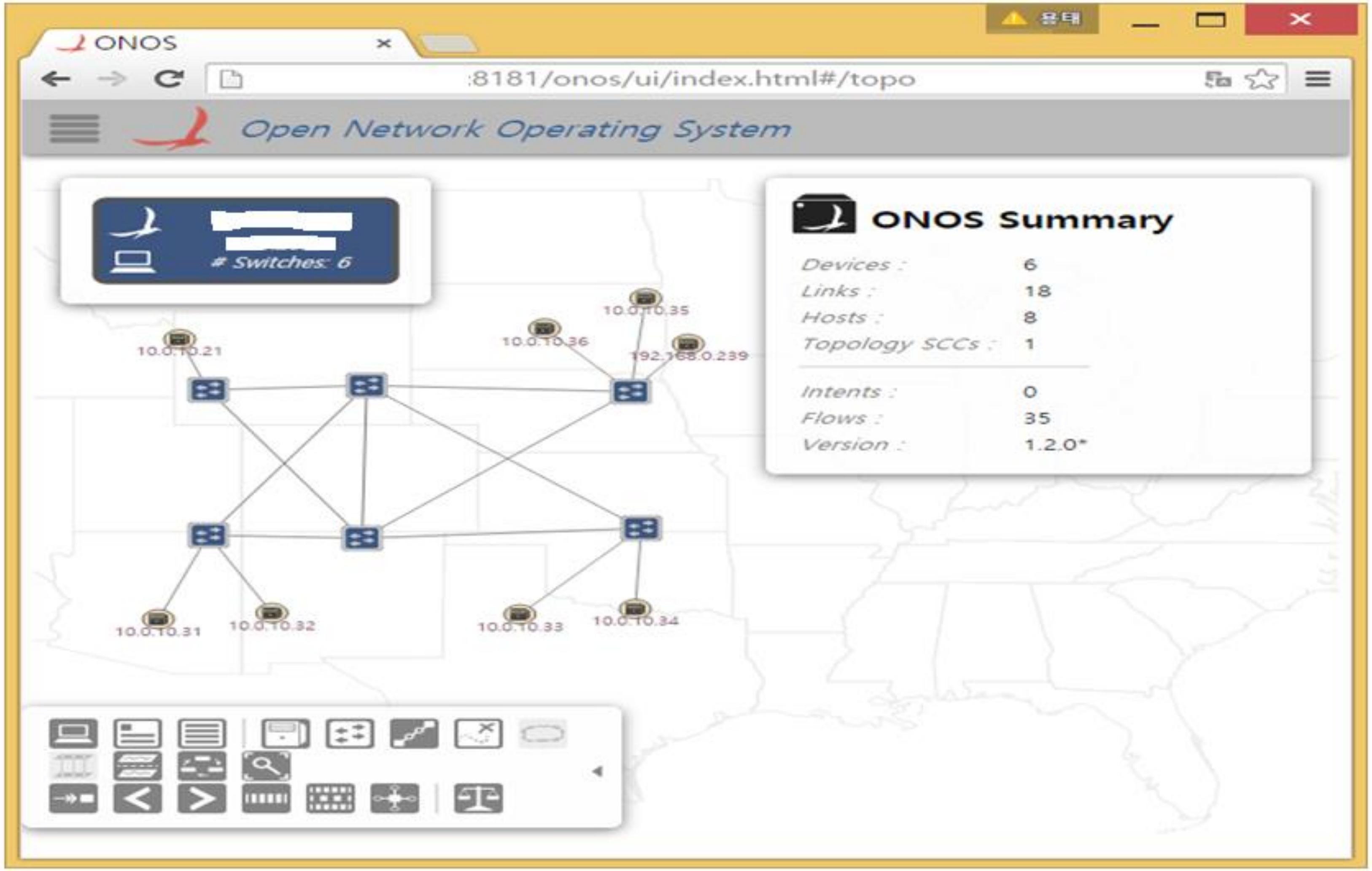
ETRI-TAM Planning Overview

- Preparing and Setup Subproject (~June 30)
- Development Phase 1 (~Aug. 31)
  - Adaptive(Effective) Flow Sampling
- Development Phase 2 (~Oct. 31)
  - Open Selective-DPI(Deep Packet Inspection)

Preparing and Setting-up the Subproject (~June 30)

- ONOS Architecture Review and ETRI TAM Architecture Design
  - ONOS wiki page : beginner, user, developer, architecture guide
  - ONOS Sub Component : analysis of the ONOS architecture and associated source code
  - ETRI-TAM Architectural Design
- Setting up Development Environment
  - Downloaded Blackbird 1.1.0 release and installed at Ubuntu 14.04.2
  - Setting Environment & Hands on examples : mininet and OVS
- ETRI-TAM Subproject Setup
  - Kick-off Conference Call
  - Proposal of a Subproject: ETRI-TAM
  - Creation of ETRI-TAM Future Project Wiki Page

ONOS GUI – ETRI SDN Testbed Env.



[redacted]

# Switches: 6

ONOS Summary

Devices :	6
Links :	18
Hosts :	8
Topology SCCs :	1
Intents :	0
Flows :	35
Version :	1.2.0*

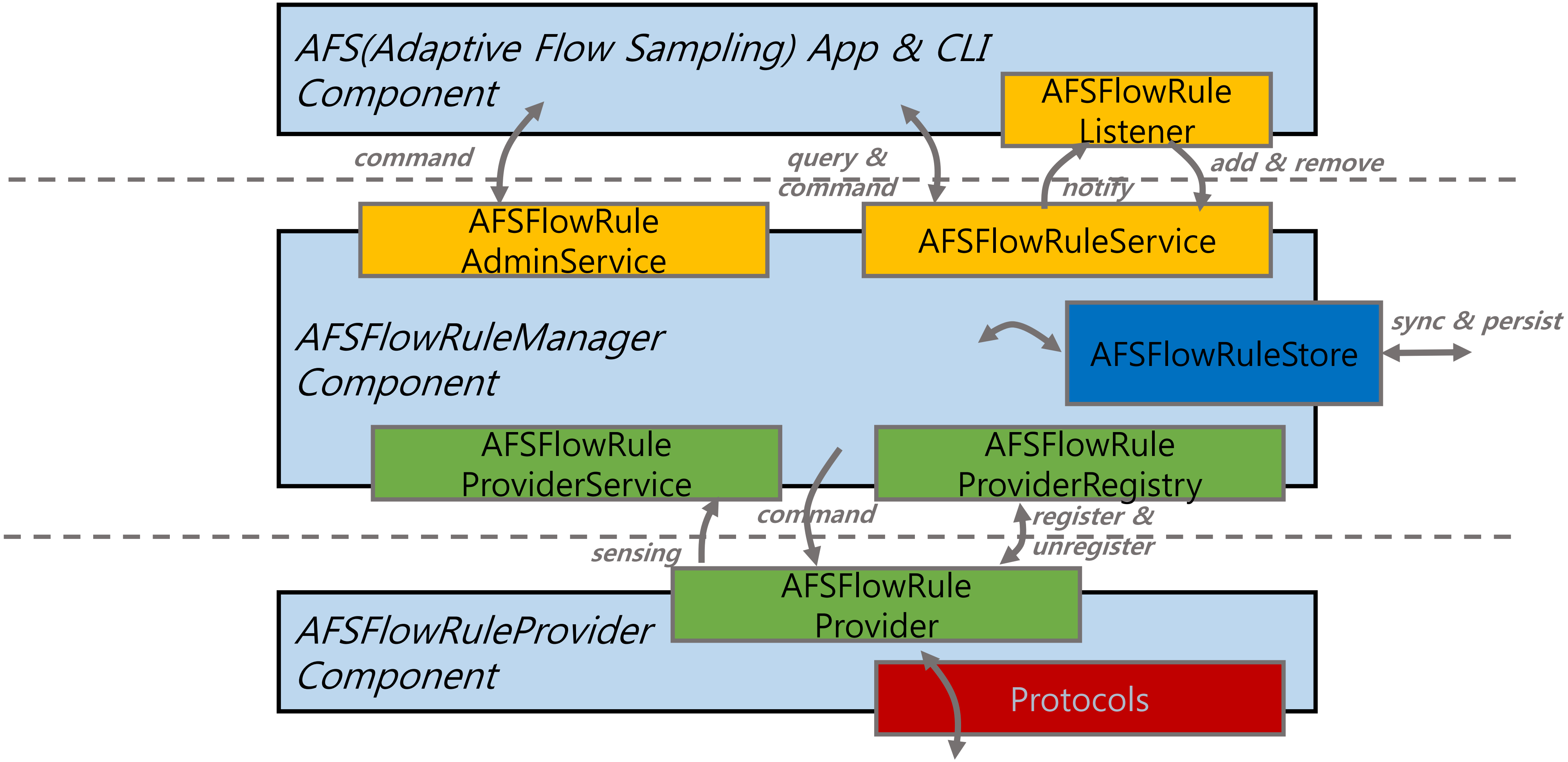
Development Phase 1 (~Aug. 31) -Adaptive(Effective) Flow Sampling Service

- Current Problem
  - Current FlowRule service collects all flow information from all devices at every time interval(default 10 seconds)
  - This mechanism may cause **performance degradation issue** at each collection time in a large-scale real carrier network due to the number of switches and its associated flows (for example; WAN: ~500 Routers, ~10K ports, ~1-10M flows per port)
  - To overcome performance problem in a simple way, we can maintain collection time interval value with a large number. It then generates another critical issue: **lack of accuracy**

  - Our proposal to this problem is an effective flow monitoring scheme called, **Adaptive(Effective) Flow Sampling Service** that can minimize collection computing overhead and provide more accurate flow statistics

Development Phase 1 (~Aug. 31) -Adaptive(Effective) Flow Sampling Service

- Service Architecture and Development Modules



Development Phase 1 (~Aug. 31)
-Adaptive(Effective) Flow Sampling Service

- Development Modules
  - AFSFlows CLI App Component: AFSFlowRuleListener
  - AFSFlowRuleManager Component: AFSFlowRuleAdminService, AFSFlowRuleService, AFSFlowRuleStore, AFSFlowRuleProviderServie, AFSFlowRuleProviderRegistry
  - AFSFlowRuleProvider Component: AFSFlowRuleProvider
- Detailed Development Plan
  - Architecture & Interface Module Design (~July 10)
  - Development of AFSOpenFlowRuleProvider Interface (~July 24)
  - Integration to the existing FlowRuleManager with our Developed Provider and Testing (~July 31)
  - Development of AFSFlowRuleManager and CLI application (~Aug. 14)
  - Integration Testing and Performance Evaluation (~Aug. 31)
  - Code contribution and wiki update (~Aug. 31)

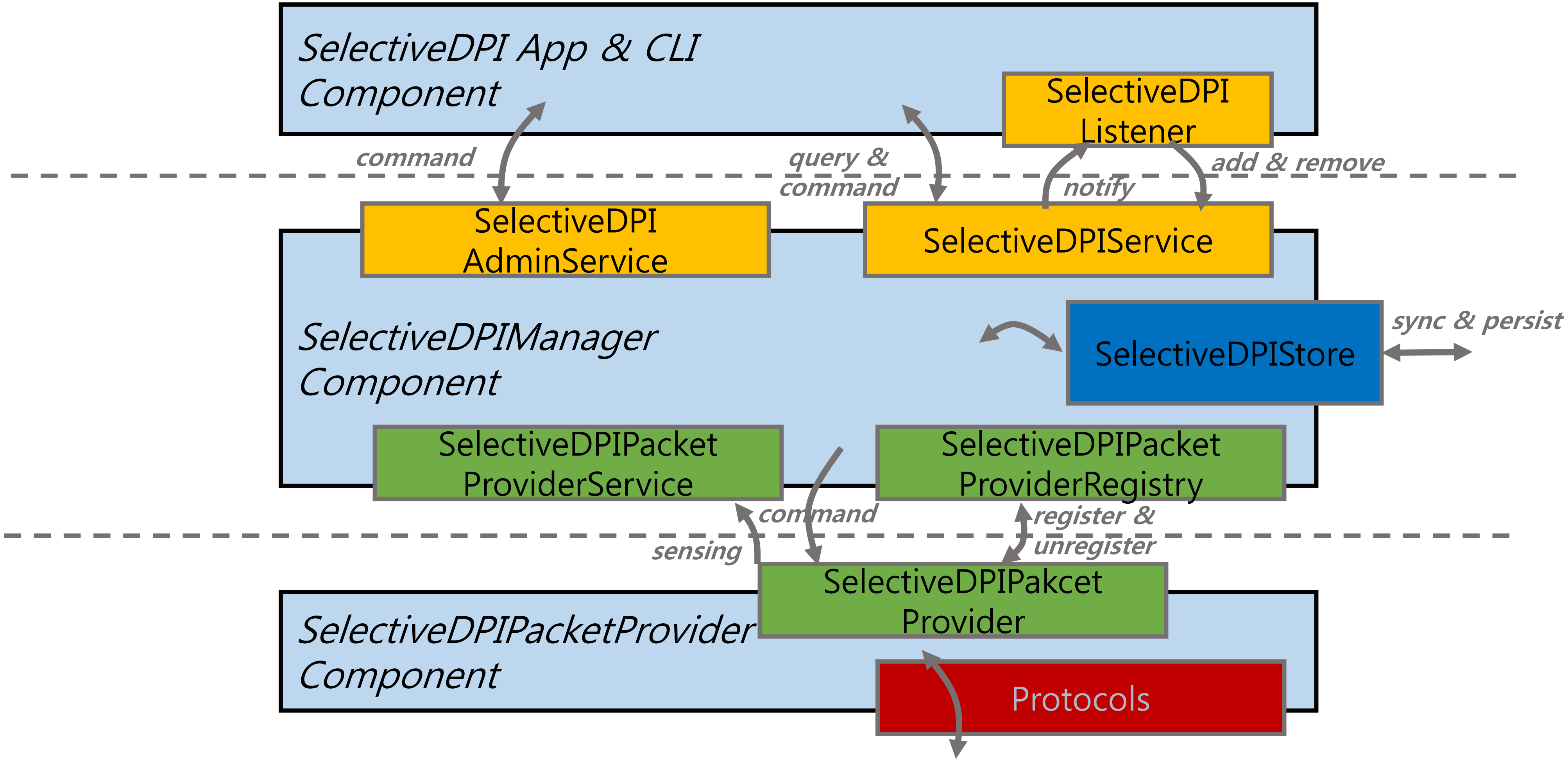
Development Phase 2 (~Oct. 31) -Open Selective-DPI(Deep Packet Inspection)

- Current Problem
  - Current ONOS flow can be classified and selected by lower-level FlowSelection criteria based on FlowRule entry (eg., ports, ether_type, vlan_id, 5-tuple, etc.)
  - There is **no application classification service** for ONOS data plane user-data

  - We propose to add a **Selective DPI service** that can filter data plane user-data from controller traffic and classify them with application level granularity by using a open source DPI s/w
  - Application of S-DPI can be application traffic analysis, service chaining classification, etc. We will take phased approach from a simple application to complex ones

Development Phase 2 (~Oct. 31)
-Open Selective-DPI(Deep Packet Inspection)

- Service Architecture and Development Module


SelectiveDPI App & CLI Component
SelectiveDPI Listener
command
query & command
add & remove
notify
SelectiveDPI AdminService
SelectiveDPIService
SelectiveDPIManager Component
SelectiveDPIStore
sync & persist
SelectiveDPIPacket ProviderService
SelectiveDPIPacket ProviderRegistry
command
register & unregister
sensing
SelectiveDPIPakcet Provider
SelectiveDPIPacketProvider Component
Protocols

Development Phase 2 (~Oct. 31)
-Open-DPI(Deep Packet Inspection)

- Development Modules
    - OpenDPI CLI App Component: OpenDPIListener
    - OpenDPIManager Component: OpenDPIAdminService, OpenDPIService, OpenDPIStore, OpenDPIPacketProviderServie, OpenDPIPacketProviderRegistry
    - OpenDPIPacketProvider Component: OpenDPIPacketProvider

- Detailed Development Plan
    - Architecture & Interface Module Design (~Sept. 11)
    - Development of OpenDPIPacketProvider Interface (~Sept. 25)
    - Integration to existing OpenDPI s/w with our Developed Provider and Testing (~Oct. 9)
    - Development of OpenDPIManager and CLI application (~Oct. 23)
    - Integration Testing and Performance Evaluation (~Oct. 31)
    - Code contribution and wiki update (~Oct. 31)