Adaptive Flow Monitoring & Selective DPI for ONOS

ONS2016

2016. 3. 17

Taesang Choi

ETRI

Topics

- OPEN-TAM ONOS Subproject Overview
- Development Phase 1
  - Adaptive(Effective) Flow Monitoring
- Development Phase 2
  - Selective-DPI (Deep Packet Inspection)
- Demo & Use-Cases

Open-TAM Subproject Overview

- ONOS Architecture Review and OPEN-TAM Architecture Design
  - ONOS wiki page : beginner, user, developer, architecture guide
  - ONOS Sub Component : analysis of the ONOS architecture and associated source code
  - OPEN-TAM Architecture Design
- OPEN-TAM Subproject Setup
  - Kick-off Conference Call with ONOS TSR
  - Proposal of a Subproject: OPEN-TAM
  - Creation of OPEN-TAM Future Project Wiki Page
- Development and Release Plans
  - Started with Blackbird 1.1.0 release
  - Phase 1 result (AFM) was incorporated in EMU release
  - Phase 2 will be in GoldenEye release

Phase 1: Adaptive Flow Monitoring

Adaptive Flow Monitoring Motivation

- Default ONOS Flow Monitoring Issues
  - Default 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 causes another critical issue: **lack of accuracy**

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

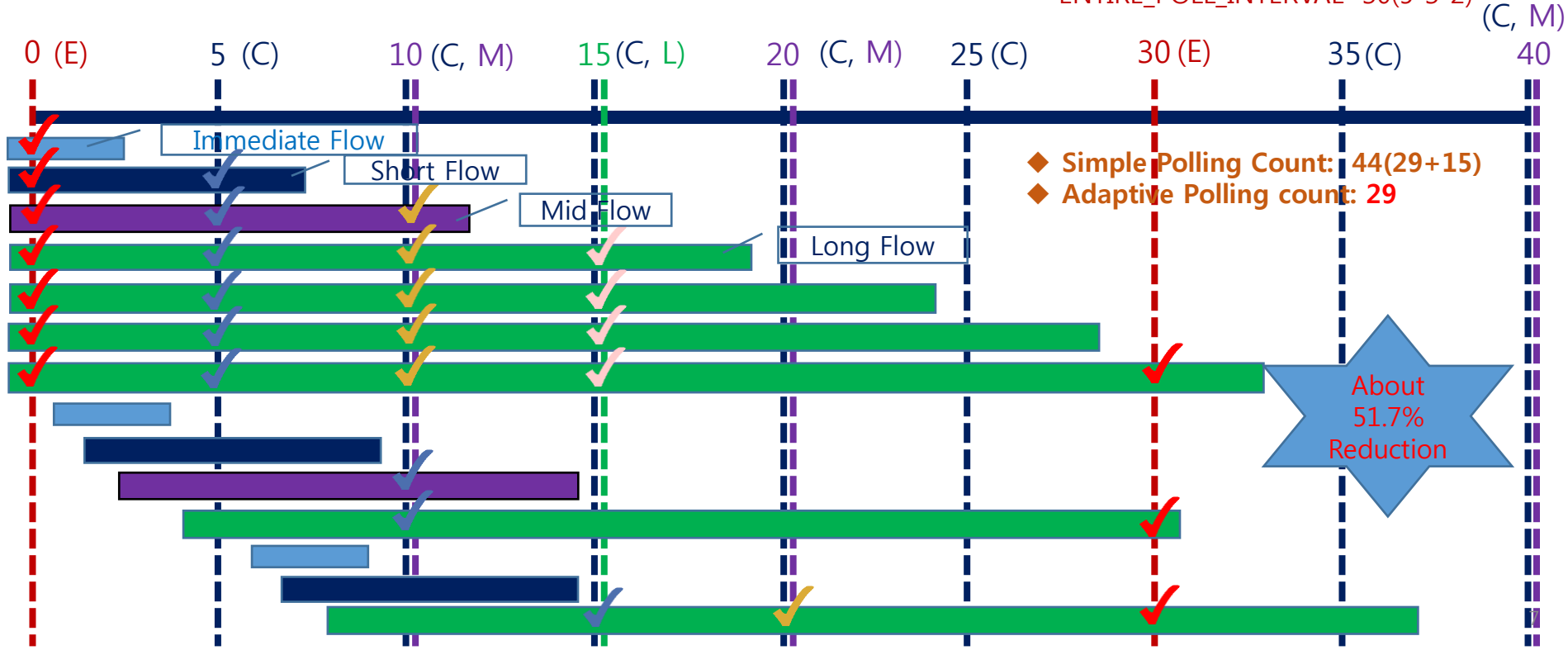
Adaptive Flow Monitoring Algorithm

1. Initialize Variables and Setup Tasks() {
   - CAL_AND_SHORT_POLL_INTERVAL= 5, MID_POLL_INTERVAL=10, LONG_POLL_INTERVAL= 15, ENTIRE_POLL_INTERVAL=30
   - Set each tasks being executed at every corresponding time interval }
2. CAL_AND_SHORT_FLOWS_TASK() {
   - IF at first time call or ENTIRE_POLL_INTERVAL, send FlowStatsRequest message getting all flow entries.
   - Else at every call, calculates FlowLiveType and save it appropriate tables
   - Sends FlowStatsRequest message only for SHORT_FLOWs entries
3. MID_FLOWS_TASK() {
   - If at every time call and not ENTIRE_POLL_INTERVAL, sends FlowStatsRequest message only for MID_FLOWs entries }
4. LONG_FLOWS_TASK() {
   - If at every time call and not ENTIRE_POLL_INTERVAL, sends FlowStatsRequest message only for LONG_FLOWs entries }

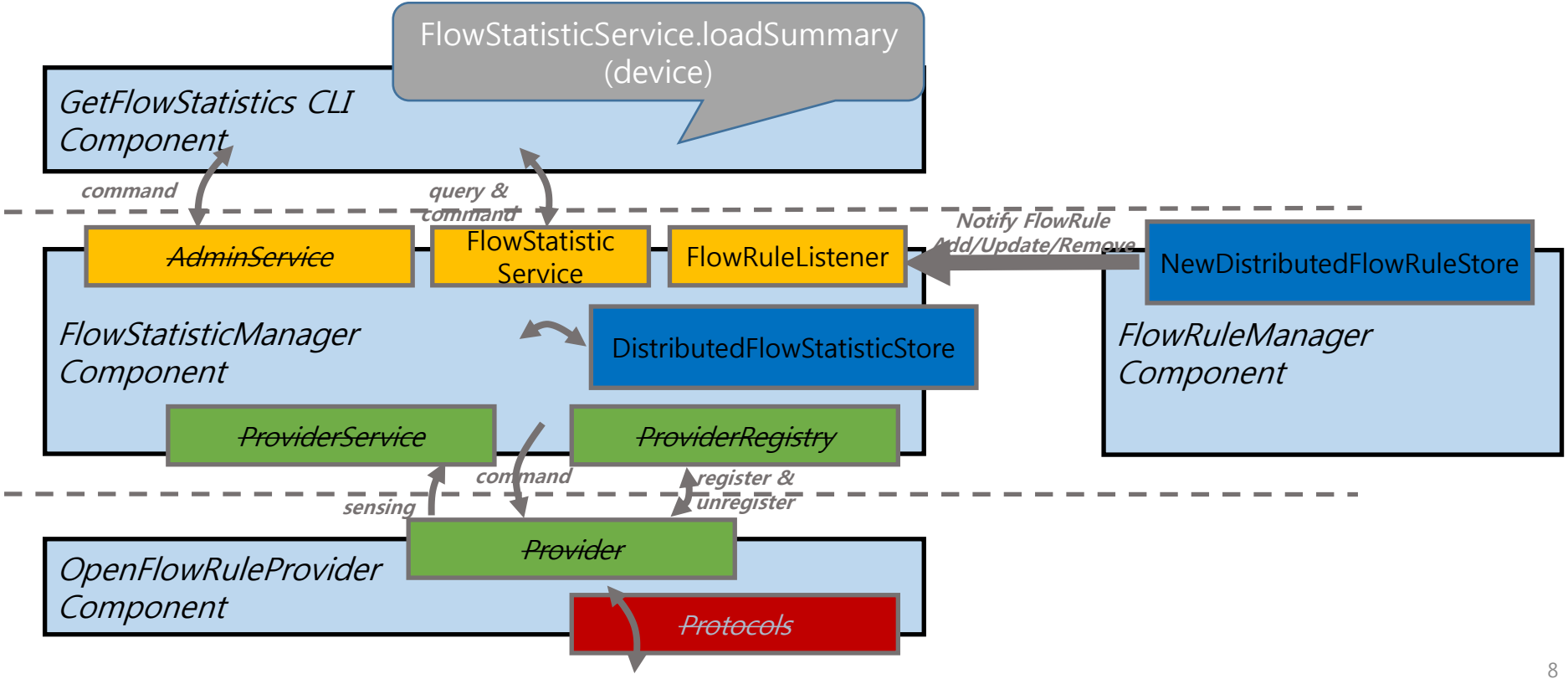
AFM– Flow Polling Example

- CAL_AND_SHORT_POLL_INTERVAL= 5
- MID_POLL_INTERVAL=10(5*2)
- LONG_POLL_INTERVAL= 15(5*3)
- ENTIRE_POLL_INTERVAL=30(5*3*2)

✓CAL_AND_SHORT_POLL, ✓MID_POLL, ✓LONG_POLL, ✓ENTIRE_POLL



FlowStatisticManager Subsystem Design



Adaptive Flow Monitoring Design Strategy

- Re-designing of FlowStatCollector into NewAdaptiveFlowStatCollector used in OpenFlowRuleProvider for each corresponding switch
- Extended StoredFlowEntryWithType from StoredFlowEntry
- Divides FlowEntries into four groups, i.e., IMMEDIATE_FLOW, SHORT_FLOW, MID_FLOW, and LONG_FLOW groups
- And uses four time intervals, i.e., SHORT_POLL_INTERAVAL, MID_POLL_INTERVAL, LONG_POLL_INTERVAL, ENTIRE_POLL_INTERVAL
- At every SHORT_POLL_INTERAL, calculates all flows into appropriate flow groups and send FlowStatsRequest message about all SHORT_FLOWs
- And every MID_POLL_INTERVAL, send FlowStatsRequest message about all MID_FLOWs
- And every LONG_POLL_INTERVAL, send FlowStatsRequest message about all LONG_FLOWs
- And finally every ENTIRE_POLL_INTERVAL, send FlowStatsRequest message all flow entries in the switch

NewAdaptiveFlowStatsCollector

I	StoredFlowEntry

I	TypedStoredFlowEntry
Member	public static enum FlowLiveType { IMMEDIATE_FLOW, SHORT_FLOW, MID_FLOW, LONG_FLOW, UNKNOWN_FLOW }
Method	public int flowLiveType(); public void setFlowLiveType(FlowLiveType liveType);

C	DefaultFlowEntry	Impl	TypedStoredFlowEntry

C	DefaultTypedFlowEntry
Member	private FlowLiveType liveTyep;

Method	public DefaultTypedFlowEntry(FlowRule rule, FlowLiveType liveType);
	public DefaultTypedFlowEntry(FlowEntry fe, FlowLiveType liveType);
	
	@Override
	public int flowLiveType();
	@Override
	public void setFlowLiveType(FlowLiveType liveType);

NewAdaptiveFlowStatsCollector

C	NewAdaptiveFlowStatsCollector

Member	pivate final Logger log = getLogger(getClass());
	private final OpenFlowSwitch sw;
	
	private ScheduledExecutorService adaptiveFlowStatsScheduler = newScheduledThreadPool(4, groupedThreads("onos/flow", "device-stats-collector-%d"));
	
	private CalAndShortFlowsTask calAndShortFlowsTask;
	private MidFlowsTask midFlowsTask;
	private LongFlowsTask longFlowsTask;
	
	private int calAndShortPollInterval = 5;
	private int midPollInterval = 10;
	Private int longPollInterval = 15;
	Private int entirePollInterval = 30;
	private InteralDeviceFlowTable deviceFlowTable = new InteralDeviceFlowTable();

Method	NewAdaptiveFlowStatsCollector(OpenFlowSwitch, int);
	
	synchronized void adjustCalPollInterval(int pollInterval);
	private class CalAndShortFlowsTask implements Runnable { }
	private class MidFlowsTask implements Runnable { }
	private class LongFlowsTask implements Runnable { }
	public synchronized void start();
	public synchronized void stop();
	
	public boolean addWithFlowRule(FlowRule… flowRules);
	public boolean addOrUpdateFlows(FlowEntry… flowRules);
	public boolean removeFlows(FlowEntry… flowRules);
	public boolean pushFlowMetrics(List<FlowEntry> flowEntries);

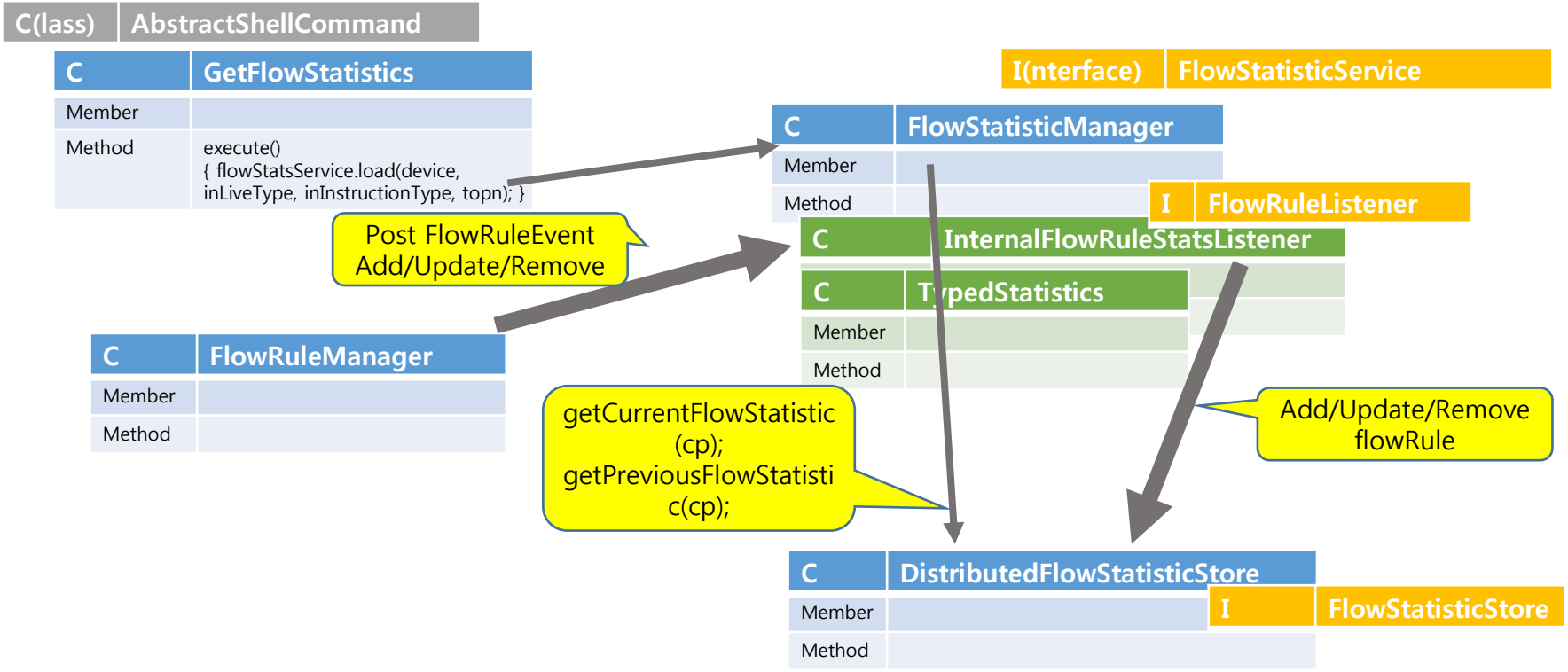
C	OpenFlowRuleProvider
Member	
Method	Added NewAdaptiveFlowStatsCollector method call statements at every flow entry ADD, REMOVED, and UPDATED

Internal C	InteralDeviceFlowTable

Member	private final Map<FlowId, Set<StoredFlowEntryWithType>> deviceFlowEntries = Maps.newConcurrentMap();
	private final Set<StoredFlowEntry> shortFlows = new HashSet<>();
	private final Set<StoredFlowEntry> midFlows = new HashSet<>();
	private final Set<StoredFlowEntry> longFlows = new HashSet<>();

Method	public getFlowCount();
	public TypedStoredFlowEntry getFlowEntry(FlowRule rule);
	public Set<StoredFlowEntry> getFlowEntries();
	public void add(TypedStoredFlowEntry rule) ;
	public void addWithCalAndSetFlowLiveType(TypedStoredFlowEntry rule);
	{ getFlowEntriesInternal(rule.id()).add((StoredFlowEntry) rule); }
	public boolean remove(FlowEntry rule);
	public void checkAndMoveLiveFlowAll();

FlowStatisticManager Class Hierarchy



FlowStatisticManager Class Design

C(lass) AbstractShellCommand

C	GetFlowStatistics
Member	

Method	execute() {
	statisticFlowService.loadSummary(device, port);
	statisticFlowService.loadAllByType(device, port, flow_type, inst_type);
	statisticFlowService.loadTopnByType(device, port, flow_type, inst_type, int topn); }

C	TypedFlowEntryWithLoad
Member	ConnectionPoint cp; TypedFlowEntry typedFlowEntry; Load load;
Method	

C	SummaryFlowEntryWithLoad
Member	ConnectionPoint cp; Load totalLoad; Load immediateLoad; Load shortLoad; Load midLoad; Load longLoad;
Method	

I(nterface) FlowStatisticService

C	FlowStatisticManager
Member	
Method	@Overrride Map<ConnectionPoint, SummaryFlowEtnryWithLoad> loadSummary(Device device) { } SummaryFlowEtnryWithLoad loadSummary(Device device, PortNumer pNumber) { } Map<ConnectionPoint, List<TypedFlowEntryWithLoad>> loadAllByType(Device device, FlowLiveType liveType, InstructionType instType) { } List<TypedFlowEntryWithLoad> loadAllByType(Device device, PortNumer pNumber, FlowLiveType liveType, InstructionType instType) { } Map<ConnectionPoint, List<TypedFlowEntryWithLoad>> loadTopnByType(Device device, FlowLiveType liveType, InstructionType instType, int topn) { } List<TypedFlowEntryWithLoad> loadTopnByType(Device device, PortNumer pNumber, FlowLiveType liveType, InstructionType instType, int topn) { }

I(nterface) FlowStatisticStore

C	DistributedFlowStatisticStore
Member	private Map<ConnectPoint, Set<FlowEntry>> previous = new ConcurrentHashMap<>(); private Map<ConnectPoint, Set<FlowEntry>> current= new ConcurrentHashMap<>();
Method	void removeFlowStatistic(FlowRule rule) { } void addFlowStatistic(FlowEntry rule) { } void updateFlowStatistic(FlowEntry rule) { } Set<FlowEntry> getCurrentFlowStatistic(ConnectPoint connectPoint) { } Set<FlowEntry> getPreviousFlowStatistic(ConnectPoint connectPoint) { }

Advanced Adaptive Flow Sampling

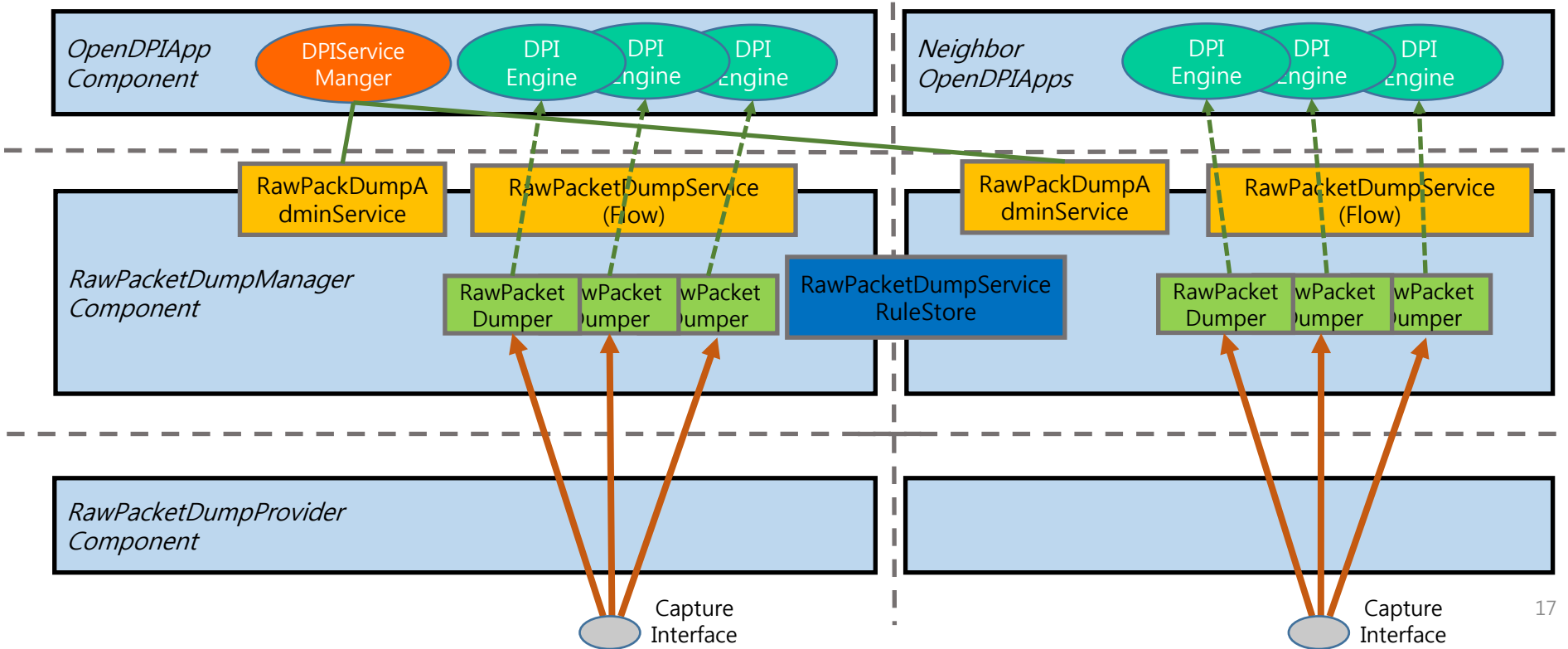
- To reduce performance overhead further,
  - At every polling interval, instead of getting all flow stats of corresponding flow entries (SHORT, MID, and LONG_FLOWs)
  - Apply one of the following sampling methods for each flow table groups, *especially for SHORT or MID groups with large number of flow entries*
    - No sampling (get all, same as current method)
    - Random sampling (at every count (default=100) flow entry number)
    - Top-n sampling (only top-n (default=1000) flows based on Bytes per Second rates)
    - Probabilistic sampling (only probabilistic (default=1/2) success flow entry number)
    - …

Phase 2: Selective Deep Packet Inspection

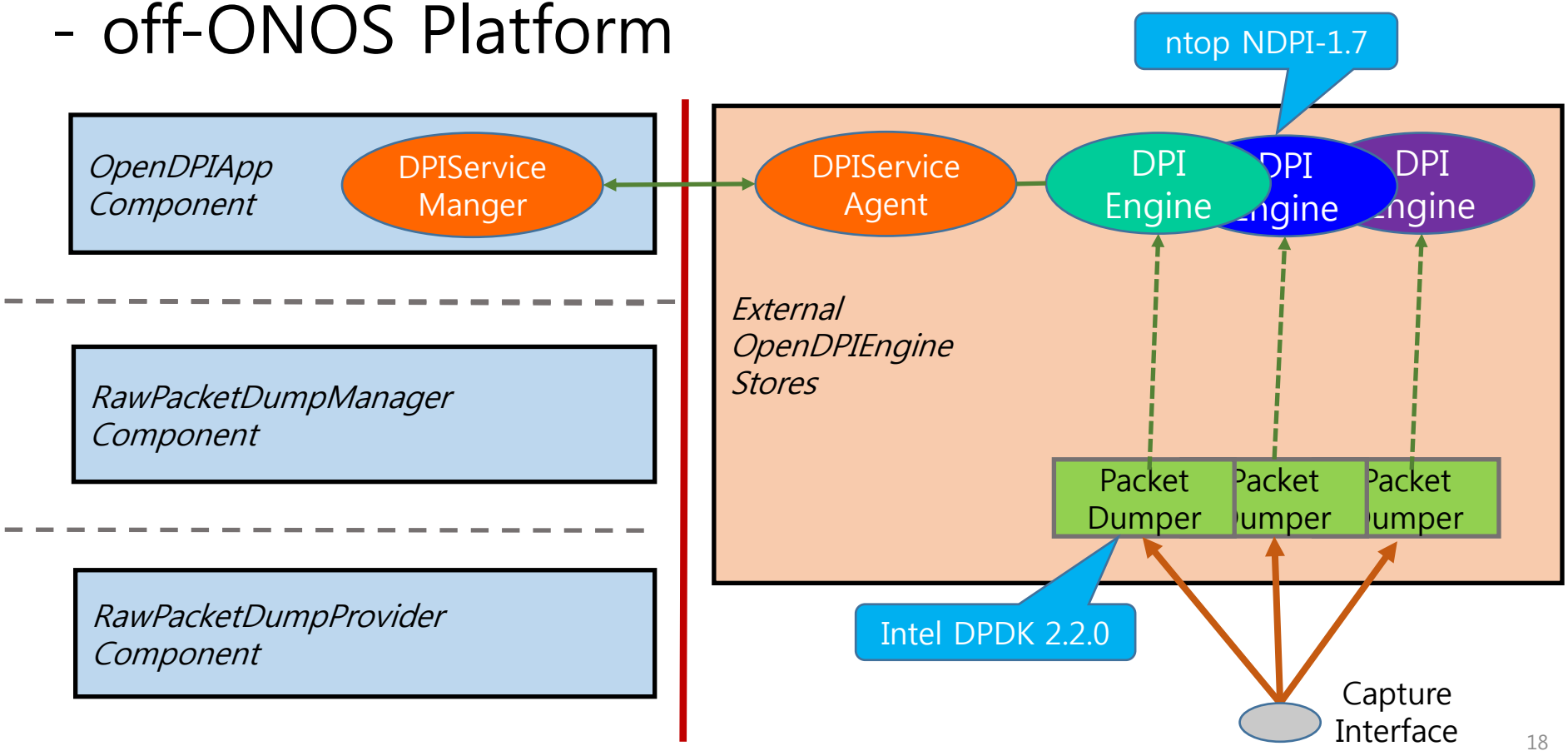
Selective-DPI Motivation

- 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

Selective DPI: OpenDPIManager Architecture – On-ONOS Platform



Selective DPI: OpenDPIManager Architecture - off-ONOS Platform



Selective-DPI: Scalability and Performance

- On-ONOS Platform solution can be used for limited number flows that don't influence performance and reliability of ONOS
  - The scalability limitation of on-ONOS platform can be determined with field tests
  - It is a native ONOS DPI solution which can be used for certain premium service applications or ONOS control & management traffic trouble shooting purposes
- Off-ONOS Platform solution is a full-blown scalable DPI solution
  - It is a non native solution
  - Require a dedicated stand-alone DPI engine which can be a hardware-based solution for the performance assurance or a container-based virtual software function for service agility
- Issue seems to be
  - Not choosing one of them
  - But finding an appropriate ratio of deployment of both solutions depending on the service requirements

Demo and Use Cases

GetFlowStatistics Command

- CLI:onos> **get-flow-stats** *Device*[/*Port*] [--summary | --all | --top *number*] [--flow *type* | --instruction *type*] [--help]
  - Default: onos> get-flow-stats Device –summary
  - -s, --summary: show summary flow rule stats based on flow live type
  - -a, --all: show all flow rule stats
  - -t, --top N : show only top N flow rule stats, 0 < N <= 1000, default = 100
  - -f, --flow type: show only flow rule live type = [IMMEDIATE| SHORT | MID | LONG]
  - -i, --instruction type: show flow rule instruction type = [DROP | OUTPUT | GROUP | L0MODIFICATION | L2MODIFICATION | TABLE | L3MODIFICATION | METADATA]

CLI:onso> get-flow-stats

get-flow-stats --help

```
ssyoon@openiris: ~/onos

Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'system:shutdown' or 'logout' to shutdown ONOS.

onos> get
get              get-flow-stats    get-stats
onos> get-flow-stats --help
DESCRIPTION
        onos:get-flow-stats

        Fetches flow stats for a connection point with given flow type and instruction
type

SYNTAX
        onos:get-flow-stats [options] devicePort

ARGUMENTS
        devicePort
                Device[/Port] connectPoint Description

OPTIONS
        -t, --topn
                Show flow stats topn
        -j, --json
                Output JSON
        -a, --all
                Show flow stats all
        -s, --summary
                Show flow stats summary
                (defaults to true)
        --help
                Display this help message
        -i, --instructionType
                Flow instruction type, It includes DROP, OUTPUT, GROUP, L0MODIFICATION,

                L2MODIFICATION, TABLE, L3MODIFICATION, METADATA, and is valid with -a
                or -t option only
        -f, --flowType
                Flow live type, It includes IMMEDIATE, SHORT, MID, LONG, UNKNOWN, and
                is valid with -a or -t option only

onos>
```

get-flow-stats --summary

```
ssyoon@openiris: ~/onos
onos> get-flow-stats of:0000001517dca3b0
deviceId=of:0000001517dca3b0, show SUMMARY flows
  deviceId/Port=of:0000001517dca3b0/LOCAL, Total=Load{rate=0, NOT VALID}, Immediate=Load{rate=0, NO
T VALID}, Short=Load{rate=0, NOT VALID}, Mid=Load{rate=0, NOT VALID}, Long=Load{rate=0, NOT VALID},
 Unknown=Load{rate=0, NOT VALID}
  deviceId/Port=of:0000001517dca3b0/1, Total=Load{rate=0, NOT VALID}, Immediate=Load{rate=0, NOT VA
LID}, Short=Load{rate=0, NOT VALID}, Mid=Load{rate=0, NOT VALID}, Long=Load{rate=0, NOT VALID}, Unk
nown=Load{rate=0, NOT VALID}
  deviceId/Port=of:0000001517dca3b0/2, Total=Load{rate=0, latest=1748}, Immediate=Load{rate=0, late
st=0}, Short=Load{rate=0, latest=1748}, Mid=Load{rate=0, latest=0}, Long=Load{rate=0, latest=0}, Un
known=Load{rate=0, latest=0}
  deviceId/Port=of:0000001517dca3b0/3, Total=Load{rate=0, latest=1840}, Immediate=Load{rate=0, late
st=0}, Short=Load{rate=0, latest=1840}, Mid=Load{rate=0, latest=0}, Long=Load{rate=0, latest=0}, Un
known=Load{rate=0, latest=0}
  deviceId/Port=of:0000001517dca3b0/4, Total=Load{rate=0, latest=7360}, Immediate=Load{rate=0, late
```

get-flow-stats --all

```
ssyoon@openiris: ~/onos
onos> get-flow-stats -a of:0000001b214edb0c
deviceId=of:0000001b214edb0c, show ALL flows, live type=ALL, instruction type=ALL
  deviceId/Port=of:0000001b214edb0c/LOCAL, 0 flows
  deviceId/Port=of:0000001b214edb0c/1, 2 flows
    flowId=1b00000b22b480, state=ADDED, liveType=SHORT_FLOW -> Load{rate=0, latest=920}
    flowId=1b00005b7d7ae7, state=ADDED, liveType=SHORT_FLOW -> Load{rate=0, latest=1380}
  deviceId/Port=of:0000001b214edb0c/2, 0 flows
  deviceId/Port=of:0000001b214edb0c/3, 0 flows
  deviceId/Port=of:0000001b214edb0c/4, 3 flows
    flowId=1b00004022125d, state=ADDED, liveType=SHORT_FLOW -> Load{rate=184, latest=920}
```

get-flow-stats --topn 10

```
ssyoon@openiris: ~/onos
onos> get-flow-stats -t 10 of:0000001b2163e760
deviceId=of:0000001b2163e760, show TOPN=10 flows, live type=ALL, instruction type=ALL
  deviceId/Port=of:0000001b2163e760/LOCAL, 0 flows
  deviceId/Port=of:0000001b2163e760/1, 3 flows
    flowId=1b0000d574187d, state=ADDED, liveType=SHORT_FLOW -> Load{rate=220, latest=1104}
    flowId=1b000085195216, state=ADDED, liveType=SHORT_FLOW -> Load{rate=184, latest=920}
    flowId=1b0000d57418f9, state=ADDED, liveType=SHORT_FLOW -> Load{rate=0, latest=920}
  deviceId/Port=of:0000001b2163e760/2, 3 flows
    flowId=1b000001a27cf7, state=ADDED, liveType=SHORT_FLOW -> Load{rate=0, latest=1104}
    flowId=1b000001a27d73, state=ADDED, liveType=SHORT_FLOW -> Load{rate=0, latest=736}
    flowId=1b0000b147b690, state=ADDED, liveType=SHORT_FLOW -> Load{rate=0, latest=920}
  deviceId/Port=of:0000001b2163e760/3, 0 flows
  deviceId/Port=of:0000001b2163e760/4, 0 flows
  deviceId/Port=of:0000001b2163e760/5, 0 flows
onos>
```

CLI:onso> dpis & -j

- Lists the DPI results
- Received from external DPI engine with table or json type.

dpis --help

3. ssyoon@raptor: ~/onos (ssh)

```
onos> dpis --help
DESCRIPTION
        onos:dpis

        Fetches the Dpi result entries that is received from Dpi engine server

SYNTAX
        onos:dpis [options]

OPTIONS
        -j, --json
                Output JSON
        -n, --lastn
                Show the last N Dpi result entries, MAX_REQUEST_ENTRY = 100
        --help
                Display this help message
        -p, --permanent
                Show the latest dpi result entry permanently at 5 second, use
                Ctrl+C for quitting
        -l, --latest
                Show the latest dpi result entry
                (defaults to true)

onos> ›
```

2. ssyoon@raptor: ~/onos (ssh)

```
onos> dpis
<--- (    0) DPI Result Time [2016-03-12 07:09:15] --->
Traffic Statistics:
        ethernet.bytes:                 413829
        discarded.bytes:                0
        ip.packets:                     2541
        total.packets:                  3422
        ip.bytes:                       352845
        avg.pkt.size:                   103
        unique.flows:                   407
        tcp.pkts:                       0
        udp.pkts:                       2419
        dpi.throughput.pps:             1.07 pps
        dpi.throughput.bps:             1.37 Kb bps
        traffic.throughput.pps:         1.07 pps
        traffic.throughput.bps:         1.37 Kb bps
        traffic.duration.sec:           2365.000 sec
        guessed.flow.protos:            0

Detected Protocols:
        Unknown                 packets: 25307          bytes: 19002572         flows: 7373
        MDNS                    packets: 9005           bytes: 2092966          flows: 464
        NetBIOS                 packets: 314020         bytes: 29053080         flows: 673
        SSDP                    packets: 33014          bytes: 10508504         flows: 1621
        IGMP                    packets: 10808          bytes: 649622           flows: 413
        ICMPV6                  packets: 3329           bytes: 249038           flows: 1419
        DHCPV6                  packets: 5987           bytes: 939959           flows: 433
        LLMNR                   packets: 148999         bytes: 11389759         flows: 145350
<--------------------------------------------------->
onos>
onos> dpis -j
<--- (    0) DPI Result Time [2016-03-12 07:09:23] --->
      { "traffic.statistics": { "ethernet.bytes": 413829, "discarded.bytes": 0, "ip.packets": 2541, "total.packets": 3422, "ip.bytes":
352845, "avg.pkt.size": 103, "unique.flows": 407, "tcp.pkts": 0, "udp.pkts": 2419, "dpi.throughput.pps": "1.07 pps", "dpi.throughput.bp
s": "1.36 Kb bps", "traffic.throughput.pps": "1.07 pps", "traffic.throughput.bps": "1.36 Kb bps", "traffic.duration.sec": "2375.000 sec
", "guessed.flow.protos": 0 }, "detected.protos": [ { "name": "Unknown", "breed": "Unrated", "packets": 25415, "bytes": 19102712, "flow
s": 7395 }, { "name": "MDNS", "breed": "Acceptable", "packets": 9055, "bytes": 2104126, "flows": 466 }, { "name": "NetBIOS", "breed": "
Acceptable", "packets": 316386, "bytes": 29272550, "flows": 677 }, { "name": "SSDP", "breed": "Acceptable", "packets": 33166, "bytes":
10553614, "flows": 1627 }, { "name": "IGMP", "breed": "Acceptable", "packets": 10914, "bytes": 655994, "flows": 415 }, { "name": "ICMPV
6", "breed": "Acceptable", "packets": 3349, "bytes": 250534, "flows": 1425 }, { "name": "DHCPV6", "breed": "Acceptable", "packets": 600
1, "bytes": 942157, "flows": 435 }, { "name": "LLMNR", "breed": "Acceptable", "packets": 149797, "bytes": 11451547, "flows": 146120 } ]
 }
<--------------------------------------------------->
onos>
```

CLI:onso> dpis –n 2

- Lists the latest n (MAX=100) DPI result entries

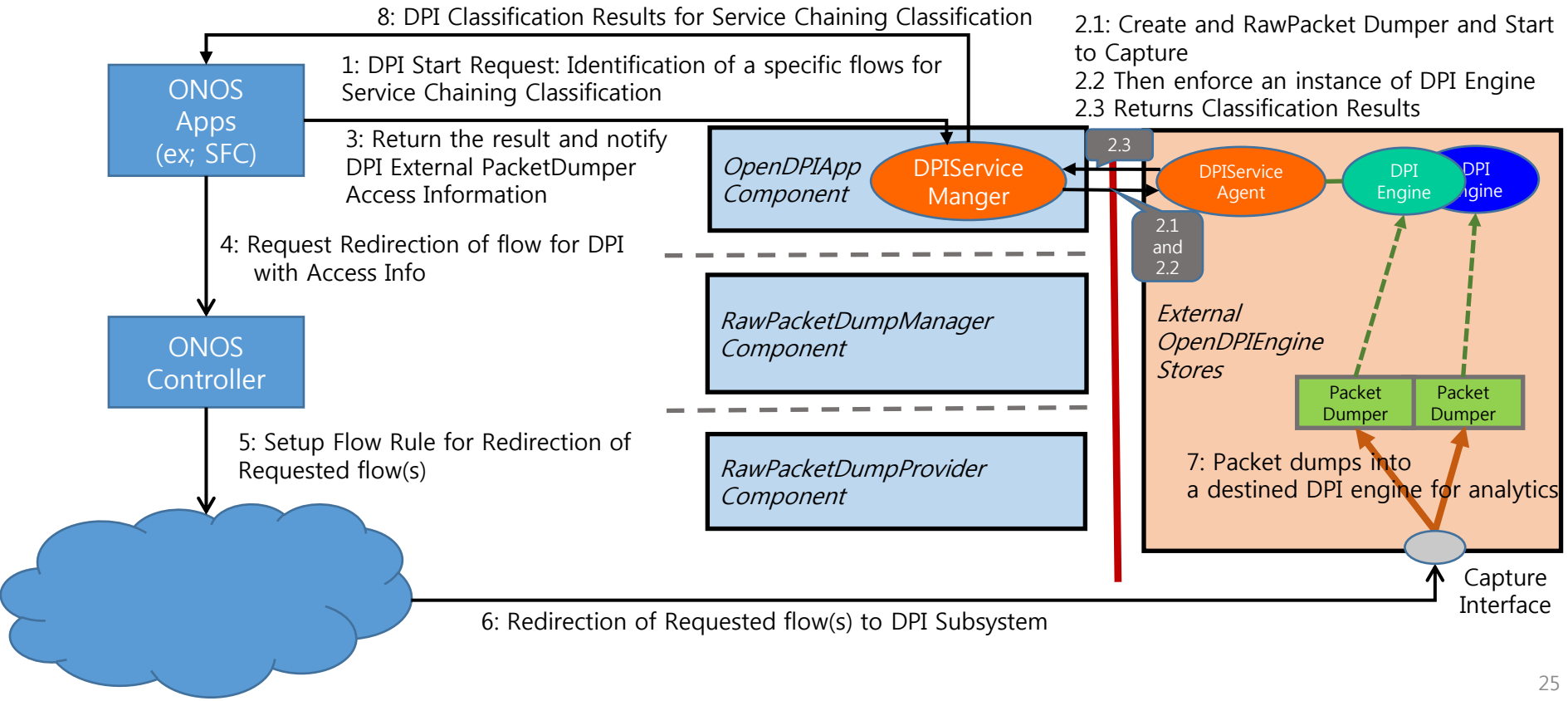
2. ssyoon@raptor: ~/onos (ssh)

```
onos> dpis -n 2
<--- (    0) DPI Result Time [2016-03-12 07:10:26] --->
Traffic Statistics:
        ethernet.bytes:                 416817
        discarded.bytes:                0
        ip.packets:                     2565
        total.packets:                  3446
        ip.bytes:                       355257
        avg.pkt.size:                   103
        unique.flows:                   405
        tcp.pkts:                       0
        udp.pkts:                       2443
        dpi.throughput.pps:             1.05 pps
        dpi.throughput.bps:             1.33 Kb bps
        traffic.throughput.pps:         1.05 pps
        traffic.throughput.bps:         1.33 Kb bps
        traffic.duration.sec:           2450.000 sec
        guessed.flow.protos:            0

Detected Protocols:
        Unknown                 packets: 26225                  bytes: 19853762             flows: 7560
        MDNS                    packets: 9430                   bytes: 2187826              flows: 481
        NetBIOS                 packets: 334223                 bytes: 30927039             flows: 707
        SSDP                    packets: 34306                  bytes: 10891939             flows: 1672
        IGMP                    packets: 11709                  bytes: 703784               flows: 430
        ICMPV6                  packets: 3411                   bytes: 255242               flows: 1448
        DHCPV6                  packets: 6118                   bytes: 960526               flows: 450
        LLMNR                   packets: 155790                 bytes: 11916728             flows: 151903
<--------------------------------------------------->
<--- (    1) DPI Result Time [2016-03-12 07:10:22] --->
Traffic Statistics:
        ethernet.bytes:                 416817
        discarded.bytes:                0
        ip.packets:                     2565
        total.packets:                  3446
        ip.bytes:                       355257
        avg.pkt.size:                   103
        unique.flows:                   405
        tcp.pkts:                       0
        udp.pkts:                       2443
        dpi.throughput.pps:             1.05 pps
        dpi.throughput.bps:             1.33 Kb bps
        traffic.throughput.pps:         1.05 pps
        traffic.throughput.bps:         1.33 Kb bps
        traffic.duration.sec:           2445.000 sec
        guessed.flow.protos:            0

Detected Protocols:
        Unknown                 packets: 26171                  bytes: 19803692             flows: 7549
        MDNS                    packets: 9405                   bytes: 2182246              flows: 480
        NetBIOS                 packets: 333028                 bytes: 30816200             flows: 705
        SSDP                    packets: 34230                  bytes: 10869384             flows: 1669
        IGMP                    packets: 11656                  bytes: 700598               flows: 429
        ICMPV6                  packets: 3409                   bytes: 255086               flows: 1447
        DHCPV6                  packets: 6107                   bytes: 958799               flows: 449
        LLMNR                   packets: 155391                 bytes: 11885681             flows: 151518
```

Selective DPI Use Case for Function Service Chaining Classification Application (off-Platform)



Thanks

for your Attention

choits@etri.re.kr