FUJITSU
shaping tomorrow with you

OpenDaylightを用いたグループベースドポリシーによる仮想ネットワークの制御

2015年11月9日

株式会社富士通研究所
李　忠翰　(lee.chunghan@jp.fujitsu.com)
中川 幸洋　(yukihiron@jp.fujitsu.com)

アジェンダ

FUJITSU

- はじめに
- GBP/OpFlexの紹介
- OpenDaylightのGBP projectの紹介
- GBPを用いた仮想ネットワーク制御例
  - 仮想ネットワーク構築
  - トラヒックのQoS制御

はじめに

FUJITSU

- OpenDaylightでGroup based Policy(GBP)を用いて仮想・物理ネットワーク機器を同時に制御してコンテナの仮想ネットワーク設定とトラヒックのQoS制御を行うデモを紹介します

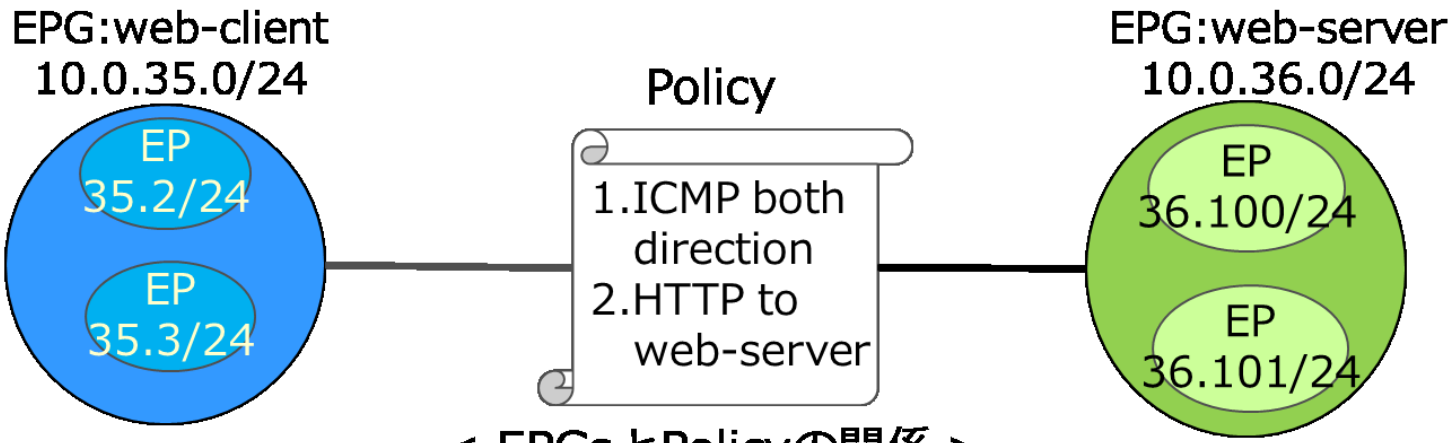
- コンテナとOpenDayligh(ODL)でYang/MD-SALのアプリを開発する人や理解を高めたい人に参考になるかと思います
  - 注) LithiumではAD-SALがなくなり、MD-SALのみになりました
    - MD-SAL: Model Driven Service Abstraction Layer
    - AD-SAL: API Driven Service Abstraction Layer

GBP/OpFlexの紹介

Group based Policy (GBP)

FUJITSU

- Group based Policy(GBP)
  - Endpoint(EP)をグループ化し、Endpoint Group(EPG)間の接続性を記述したPolicyでネットワークを抽象化
- GBPで使用される用語
  - Policy rule
    - Action：EPGのtrafficに対して何かの作業(QoSなど)を実施
    - Classifier：EPGのtrafficに対してdirection/matchingなどを実施
  - Policy(Contract)：Policy ruleの集合 (ODLではRESTconfで記述)
- EPG:web-serverとEPG:web-client間のPolicy例



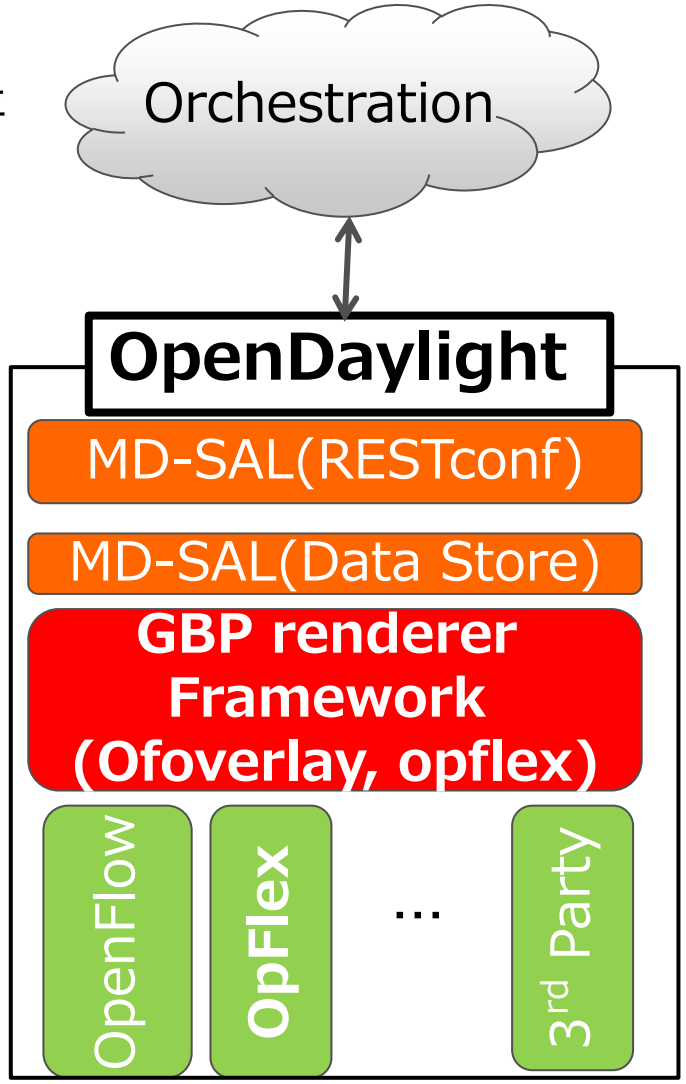
< EPGsとPolicyの関係 >

OpenDaylight, GBP, and OpFlex

FUJITSU

- ODLにおけるGBPの実装
  - RESTconf：Northbound protocol
  - Data Store：RESTconfからのデータを保存 (Yangで記述)
  - Renderer Framework
    - Ofoverlay, opflex rendererが存在
  - OpenFlow, OpFlex, …, 3rd party：Southbound protocol

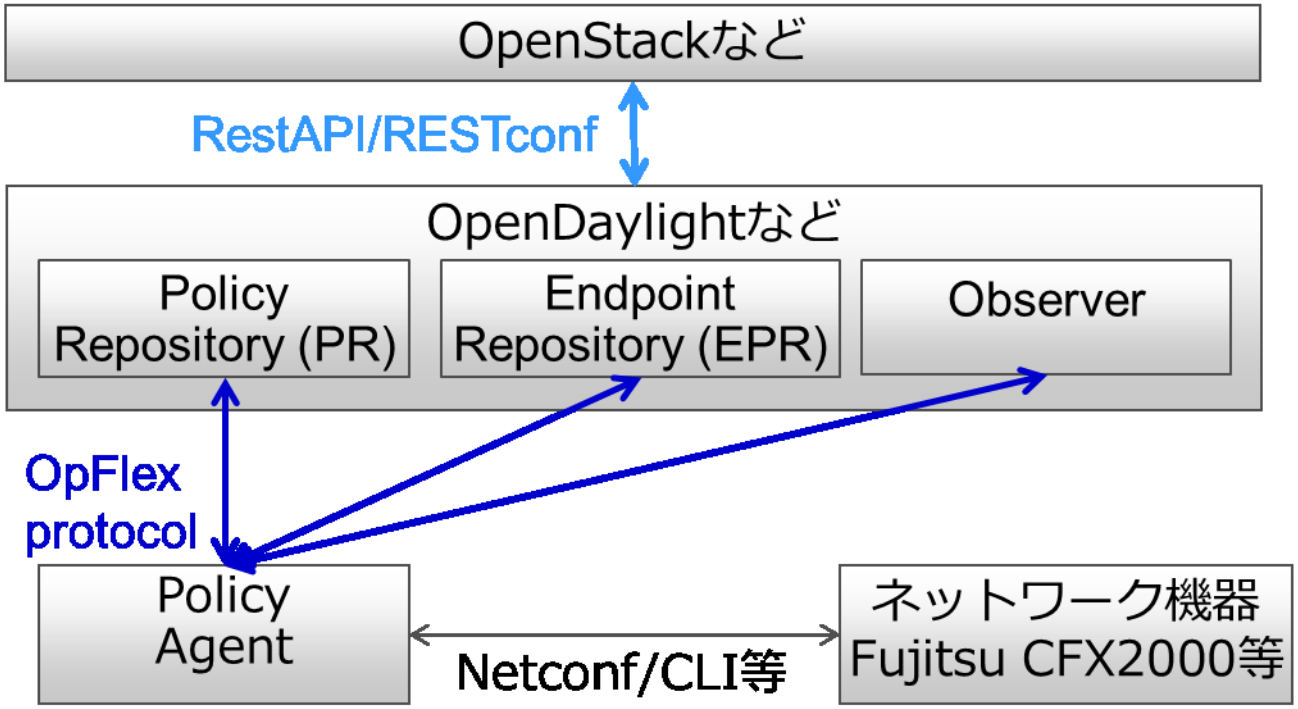
- OpFlexの特徴
  - OpenFlowはFlowという具体的なものを制御するが、OpFlexは抽象度が高いPolicyに基づいて制御する



OpFlexシステム

FUJITSU

- OpFlexシステムアーキテクチャー
  - Policy Repository (PR) ：ポリシーを一元管理するコンポーネント
  - Endpoint Repository (EPR)：エンドポイントを格納するコンポーネント
  - Observer：システムのモニタリングを行うサブシステム
  - Policy Agent：ポリシーを機器に適用するコンポーネント
    「Policy Element (PE)とも呼ばれる」



GBP projectのRenderer紹介

GBP renderers

FUJITSU

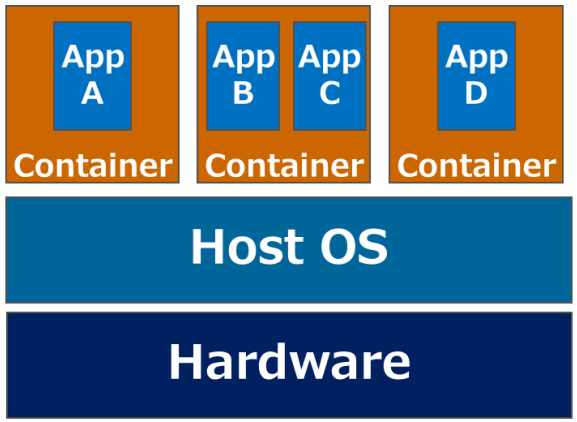
- GBP rendererの紹介
  - 基本的には同じ形式のRESTconf messageを使用してユーザとODL間でやり取りを実施し、Southbound protocol (OVSDB, OpenFlow or OpFlex)に変換してネットワーク機器の制御を行う
  - https://wiki.opendaylight.org/view/Group_Policy:Main
  - https://github.com/opendaylight/groupbasedpolicy
- 主なGBP rendererの特徴

Name	Target	特徴
ofoverlay renderer	OpenFlow &IP-overlay	OpenFlowスイッチ(OVS)とOpenFlow protocolを利用してやり取りを行い、ODLが直接Renderingを実施 (ODL Hackfest Tokyo 2014ではOfoverlay rendererをベースに試作してデモを実施) https://wiki.opendaylight.org/images/c/c2/GBP_demo_HackFest_20141029_v8.pdf
opFlex renderer	OpFlex	一般的なネットワーク機器を制御するためのOpFlex agentとOpFlex protocolを利用してやり取りを行い実際のRenderingはPolicy agentが実施

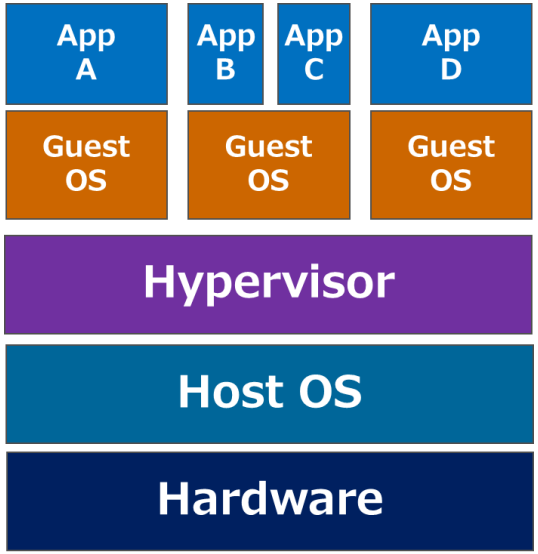
GBPを用いたコンテナの仮想ネットワーク構築の実行例

コンテナとDocker

- Linuxのコンテナ(LXC)の特徴
  - VMより軽くて、起動＆停止が容易
- 最近、話題のDockerとは？
  - 柔軟に複数のコンテナ構築を実現可能したツール
  - 単一OSで簡単に複数のコンテナを起動＆停止することが可能
  - しかし、コンテナ間のネットワークはNATになっており、お客様ごとにネットワークが分離されてあるDCNなどで使用することは困難



< Linux container >



< Full virtualization >

デモ概要

FUJITSU

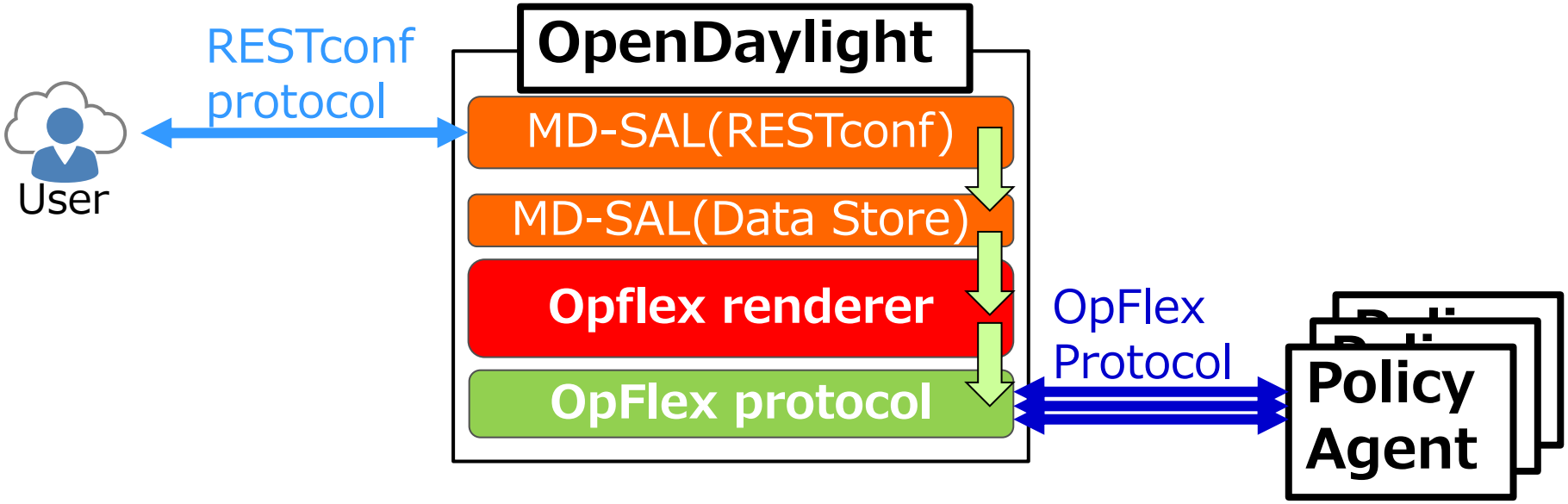
- ねらい
  - OpenDaylightを用いて物理・仮想スイッチを制御して数百ミリ秒で立ち上がるコンテナに合わせてテナント毎の仮想ネットワーク(VLAN)を構築可能なことを示す
- 実現方法
  - テナントをEPGとして扱い、EPGとPolicyを作成してRESTconfを利用して設定
  - テナントの仮想ネットワークをRenderingして物理ネットワークのVLANを設定
    - Endpoint(コンテナ)登録時、VLANを決定
    - ポリシーに基づいてテナントごとに仮想ネットワーク(VLAN情報)を仮想スイッチ(OVS)と物理スイッチ(Fujitsu CFX2000)を事前配布
    - コンテナが起動したら、合わせて仮想ネットワークを構築
    - コンテナが停止したら、合わせて仮想ネットワークも削除

ODLを中心にした仮想ネット構築の手順

FUJITSU

■ ODLを中心にした全体の動作手順

1. UserはEPG[VLAN ID](テナント情報)/Policy[通信可能]を登録するためにRESTconf Message(Northbound)を作成し、ODLに送信
2. ODLのMD-SAL(RESTconf)はMessageを解釈して、Data Storeに保存
3. ODLのOpFlex rendererはData StoreからEPG[VLAN ID]/Policy[通信可能]の情報を取り出してOpflex messageを作成
4. ODLのOpFlex rendererはOpFlex protocol(Southbound)を利用してEPGに関連するPolicy agentにEPG情報とPolicyを送信

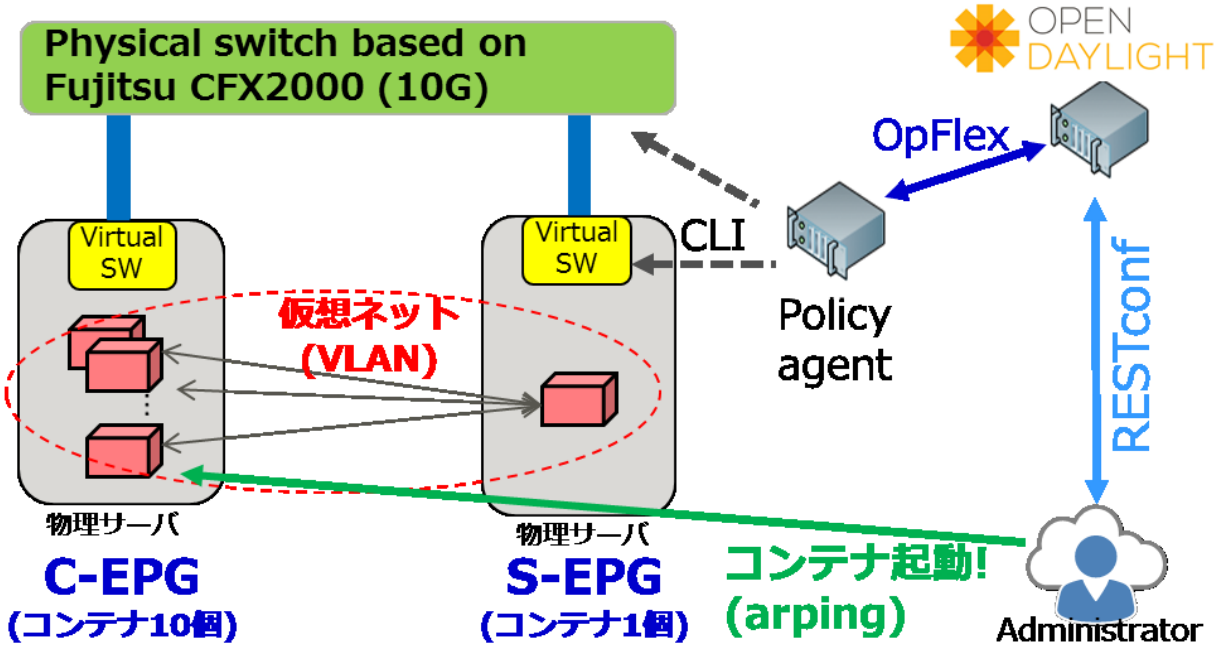


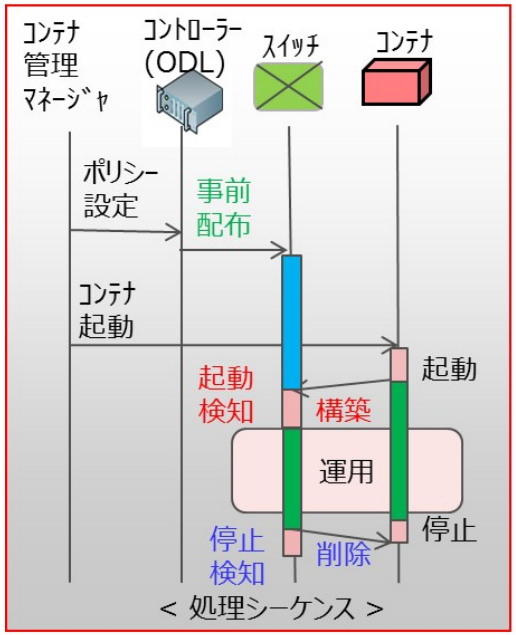
OpFlexを基にした仮想ネットワーク構築

FUJITSU

■ 仮想ネットワーク構築の手順

1. 10個のコンテナをClient EPG(C-EPG)、それに対応する1個のコンテナをServer EPG(S-EPG)として構成し、Policy(通信可能)をODLで設定
2. 仮想ネットを構築するために、Policy agentでは仮想・物理スイッチに対して各EPGが持つ仮想ネットのVLAN IDなどを事前配布
3. C-EPGを起動後、コンテナをNATから分離し、arpingを物理スイッチに伝送
4. 物理スイッチではarpingを受信後コンテナを起動検知して動的にVLANを構築
5. 物理スイッチは死活監視を実施して所定時間通信後、コンテナが停止されたら、VLANを動的に削除



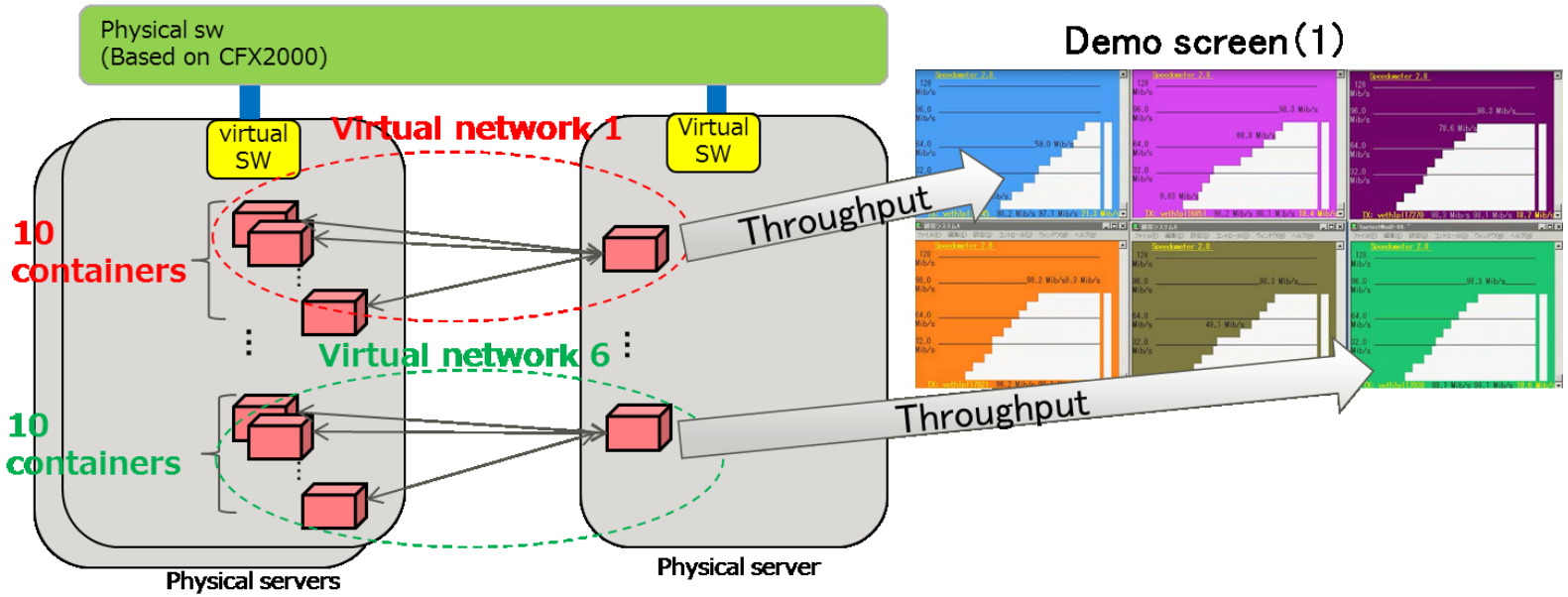


仮想ネットワークのデモ

FUJITSU

■ デモの構成

- 仮想ネットは11個のコンテナ(1 server, 10 clients)として構成しており、6つの仮想ネットを準備
- 各Demo screenは1つの仮想ネットのスループットを示す
- Client側では、60個(6つの仮想ネット)のコンテナを同時に起動
- コンテナの起動後、仮想ネットが構築され10秒間通信を実施後停止



実際デモ

FUJITSU

Throughput
(#of Running Containers)

10.25.227.75 - リモート デスクトップ接続

Virtual networks on containers

Start automatic construction of virtual networks

192.168.1.44:22 - faetest@odl-04: ~ VT

Speedometer 2.8

128 Mib/s
96.0 Mib/s
64.0 Mib/s
32.0 Mib/s

TX: veth1pl5377 2.81 Kib/s 1.91 Kib/s 14.0 Mib/s

TX: veth1pl5852 5.62 Kib/s 1.60 Kib/s 14.0 Mib/s

TX: veth1pl5590 0 b/s 0 b/s 14.3 Mib/s

TX: veth1pl5576 0 b/s 0 b/s 14.4 Mib/s

TX: veth1pl5416 0 b/s 1.69 Kib/s 14.8 Mib/s

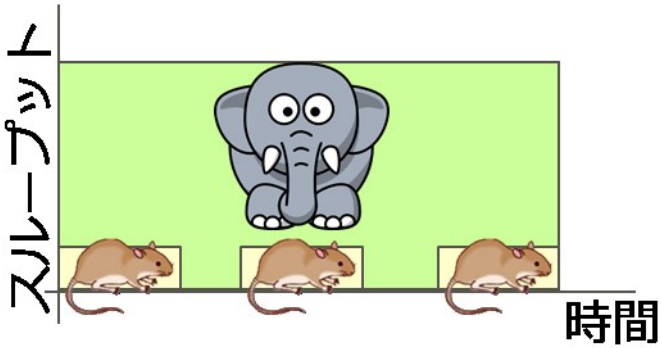
TX: veth1pl5613 2.81 Kib/s 816 b/s 15.0 Mib/s

GBPを用いたトラヒックのQoS制御の実行例

データセンターのトラヒックの特徴

FUJITSU

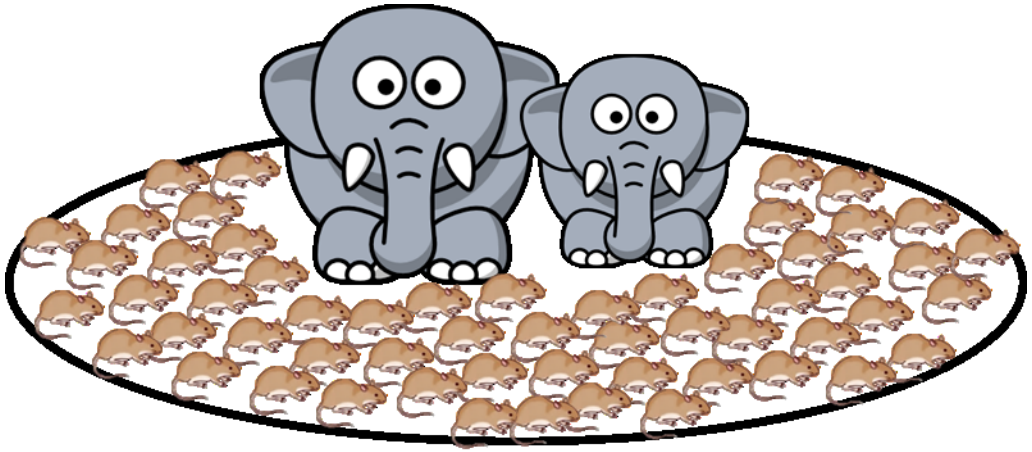
- データセンターのトラヒックの分類
  - 大容量のデータ通信
    - Elephant：スループットが重要



<スループットの比較>

  - 細かい制御メッセージやその応答
    - Mice：遅延が重要

- データセンターのトラヒックの構成[1)]



多数のMice(90%)と少数のElephant(10%)で構成

1) The Nature of Data Center traffic: Measurements & Analysis, In IMC, 2009

デモ概要

FUJITSU

- ねらい
  - OpenDaylightを用いて物理・仮想スイッチを制御して仮想ネットワーク(VLAN)でTarget flowを優先制御が可能なことを示す

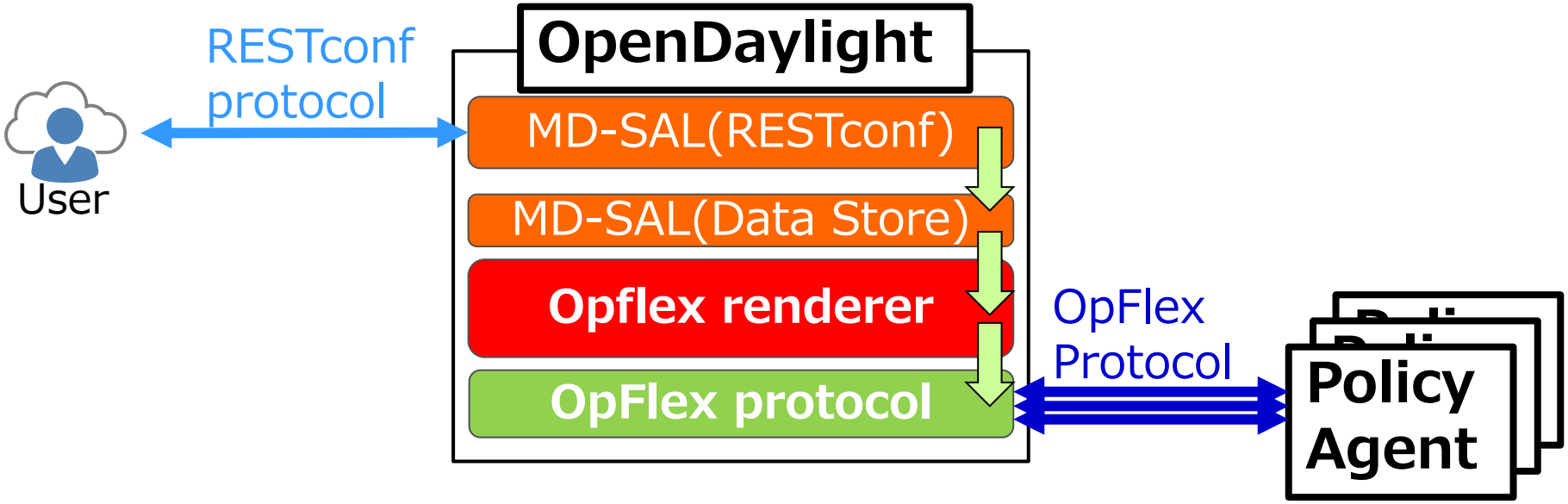
- 実現方法
  - Target flowを優先制御するために、EPGとPolicyを作成してRESTconfを利用して設定
  - Renderingする対象としてVLANによる仮想ネットワーク
    - Endpoint登録時、VLANを決定
    - ポリシーに基づいて特定のVLANのCoSを設定するように仮想スイッチ(OVS)を設定
    - ポリシーのActionを利用してQoSをHighで設定してCoSに基づいて物理スイッチでキューを分ける

ODLを中心にした優先制御の手順

FUJITSU

■ ODLを中心にした全体の動作手順

1. UserはEPG[VLAN ID]/Policy[優先制御]を登録するために、RESTconf Message(Northbound)を作成し、ODLに送信
2. ODLのMD-SAL(RESTconf)はMessageを解釈して、Data Storeに保存
3. ODLのOpFlex rendererはData StoreからEPG[VLAN ID]/Policy[優先制御]の情報を取り出してOpflex messageを作成
4. ODLのOpFlex rendererはOpFlex protocol(Southbound)を利用してEPGに関連するPolicy agentにPolicy[優先制御]を送信

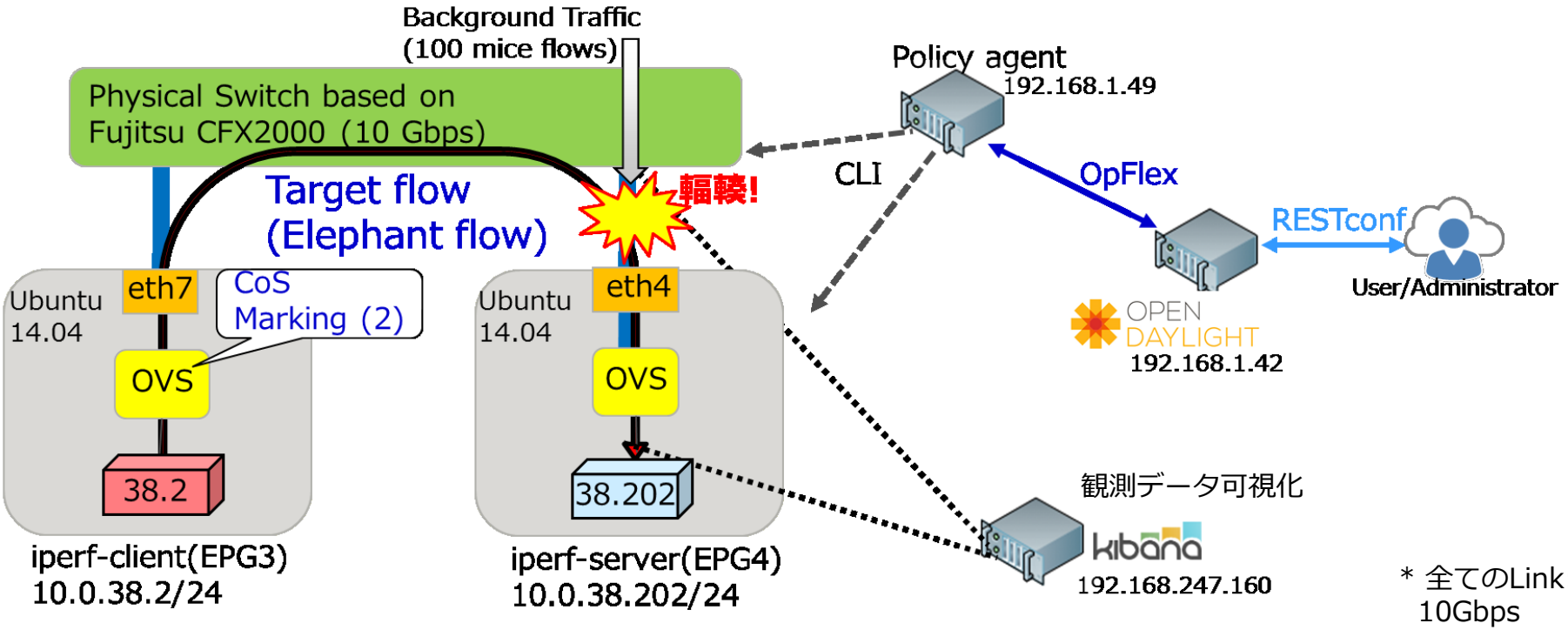


OpFlexを基にしたFlowの優先制御

FUJITSU

■ 優先制御の手順

1. iperf clientのEPG(EPG3)にCoS=2/VLAN(202)を割り当て
   EPG3から待ち受けサーバに(iperf server)のEPG(EPG4)にデータを伝送
2. Background trafficとして毎秒ごとに100個のMice flows(756KB)が
   待ち受けサーバに伝送され、Target flowに対して輻輳を再現
3. EPG3に対して優先制御のPolicyが設定され、Target flowに対しては
   仮想・物理スイッチが連携して輻輳時にキューを分ける優先制御を実施



＊ 全てのLink:
10Gbps

実際デモ

FUJITSU

まとめ

FUJITSU

- OpenDaylightを用いて仮想＆物理スイッチを同時に制御して、コンテナ起動・停止に合わせて動的に仮想ネットワークを生成・削除するデモを実施

- OpenDaylightを用いて仮想＆物理スイッチを同時に制御して、トラフィックのQoS制御するデモを実施

FUJITSU

shaping tomorrow with you