一般社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report

OpenFlow スイッチを用いたパッチパネルの検討

仲間 修也† 照屋 保幸‡ 鳥居 隆史‡† 佐藤 陽一‡‡

†NEC ソフト沖縄株式会社 〒900-0015 沖縄県那覇市久茂地二丁目 2 番 2 号
‡一般社団法人 沖縄オープンラボラトリ 〒904-2234 沖縄県うるま市州崎 14-17 沖縄 IT 津梁パーク 中核機能支援施設 112 号
‡†日本電気株式会社 〒108-0014 東京都港区芝五丁目 7 番 1 号
‡‡NTT コミュニケーションズ株式会社 〒108-8118 東京都港区芝浦 3-4-1 グランパークタワー16F
E-mail: †{shuya.nakama, yasuyuki.teruya}@oolorg.mygbiz.com ‡†t-torii@ce.jp.nec.com ‡‡y.sato@ntt.com

あらまし 沖縄オープンラボラトリでは，会員向けに各種検証や PoC（Proof of concept）を行う為のテストベッドを構築し，運用している．また，テストベッドは，セミナーやハンズオン等においても活用され，人材育成にも利用されている．テストベッドを構築するに当たり，SDN 技術やクラウド技術を有効活用し，拡張性，柔軟性，自動化の観点から，機能開発，拡張を行っている．本発表では，テストベッドの概要と，その中で用いている OpenFlow スイッチを用いたパッチパネルに関して説明する．

キーワード SDN， OpenFlow， クラウド， OpenStack， 自動化

1. はじめに

インターネットやスマートフォンを代表とするデータ通信のトラヒックは増加を続けている．近年ではさらに，IoT（Internet of things）やビッグデータ解析，機会学習による人工知能の実用化等，ICT 高度化技術の進展が目覚ましく，それを支える基盤であるネットワークやクラウド技術も変革を迫られている．SDN やクラウド技術の特徴はネットワークや計算資源を抽象化し，API 等を提供することで，プログラムから利用・制御を容易にする点にある．これによりアプリケーションの特性に合った網，計算資源の利用がより柔軟にかつ人間の操作を介さずに自動で制御できるようになり，新しい機能開発やコストの低減が期待されている．

一般社団法人沖縄オープンラボラトリ[1]では，オープン・ソース・ソフトウェアを中心とする SDN とクラウドを融合させることで，ICT 基盤技術がより使いやすくなることを目指し，テストベッドの構築・運用を行っている．テストベッドの運用に際しては，各種検証等で構成変更，設定変更が頻繁に発生するため，それらに伴う人手による作業を極小化し，少ない人的リソースで効率的に運用できる仕組みを導入することを目標にしている．特に検証ごとに発生する機器構成の接続変更に関しては，パッチパネルの代用を OpenFlow スイッチで実現し（以下 OF-Patch），構成変更をプログラマブルに出来るように工夫を取り入れている．

本発表では 2 章でテストベッドの概要を説明し，3 章で OF-Patch の概要を述べ，4 章で実装方式について詳細を述べる．5 章で OF-Patch の評価結果について述べる．

2. テストベッドとは

OpenStack[2]と SDN を連携させた検証・評価や PoC をオンラインで実行するためのプラットフォームとして，沖縄オープンラボではテストベッドを構築している．テストベッドは現在沖縄オープンラボの会員向けのサービスとして提供されている．

2.1. モチベーション

・SDN 装置を用いた検証環境の提供

SDN は通信事業者や大規模データセンタで使われるというイメージも強いが，応用次第で様々なシステムで使うことができる技術である．しかし現状，SDN 装置は通常のネットワーク装置よりも高価で敷居が高く，ユースケースやアプリケーションが普及しないという問題がある．そこで，SDN 装置をオンラインで提供し，装置を購入しなくても SDN とクラウド（OpenStack）が連携したシステムやアプリケーションの評価検証が可能になれば，幅広いユースケースが実現されることが期待できる．

・物理ネットワークの制御と OpenStack（クラウド制御）を連携する仕組みの提供

OpenStack の仮想ネットワーク制御（Neutron）は論理的な API のみで，物理的なネットワークを制御するインタフェースは定義されていない．物理的なネットワーク装置を制御するのは SDN コントローラとされている．しかしそれでは物理的なネットワークをソフトウェアで制御できるという SDN のメリットを使いこなすことができない．ネットワークの状態をリアルタイムに把握し，IT・ネットワークリソースの制御に利用するためには，OpenStack と物理ネットワーク制御を連携させる仕組みが必要である．

・常設の沖縄オープンラボ版 ShowNet の提供

毎年開催されている Interop の ShowNet では，最新の機器を相互接続して大規模なネットワークが実現されている．各社のエンジニアはここで貴重な経験をすることができ，装置ベンダやオペレータはマルチベンダの装置との接続検証をすることができる．しかし ShowNet は Interop の期間だけ設置されるテンポラリな環境であり，期間が終われば解体されてしまう．常設の ShowNet があれば，機器を持ち込んでいつでも検証や評価をすることができ，ベンダだけではなくユーザにとっても価値を提供できる．

2.2. テストベッドの機能

・サーバと SDN 装置を組み合わせた環境をオンライン提供

テストベッドのユーザは，従来のクラウドのようにオンラインでリソースを利用することができる．本テストベッドの特徴は，仮想化されたリソースではなく，物理的な装置単位でリソースを利用できることにある．たとえば SDN 装置として，NEC 製品にするか，Pica8 製品にするかを選択することができる．仮想ネットワークの論木的な検証であれば，どの装置であるかは意識しなくてよいが，物理的な装置に依存する検証評価をしたい場合には十分ではない．本テストベッドはそのようなニーズにこたえることができる．

・OpenStack システムの自動構築

テストベッドのユーザが利用するサーバ装置・ネットワーク装置を選択すると，サーバ群に対して自動的に OpenStack システムがインストールされる．その際，ユーザは OpenStack のバージョン（Folsom，Grizzly，Havana・・）や，Neutron プラグインなどの選択を行うことができる．OpenStack システムの構築は，以前よりも改良されてきたものの，まだ手間のかかる作業である．本テストベッドではこの作業を自動化することで，ユーザ作業の軽減を目指している．また，あらかじめテストベッドで用意した OpenStack システムではなく，ユーザ独自のクラウド基盤や OpenStack バージョンをインストールすることも可能である．

・テストフレームワーク

OpenStack から VM を起動し，VM 上でテストアプリケーションを動かすことでネットワークの検証・評価を行うフレームワークを提供する．これにより，ユーザはテストの内容に集中することができる．また，このフレームワーク上に様々なテストアプリケーションが公開されることで，ユーザはその中から自分たちの実施したいテストやユースケースを見つけて再利用することができる．

・バックアップ・リストア

オンラインシステムであるため，一度構築したシステムを恒久的に維持しておくことは現実的ではないため，検証が終わったシステムの保存や，構築したシステムのスナップショットを保存しリストアしたいという要望がある．本テストベッドでは，構築したシステムを丸ごとバックアップ・リストアできる機能を提供する．

・他拠点リソースの活用

本テストベッドの物理リソースには，沖縄オープンラボが所有するリソースだけでなく，協賛企業を含めた他社や研究機関のリソースをネットワーク経由で接続して利用する機能を備える．これにより，単に使えるリソースが増えるだけでなく，広域ネットワークやインタークラウドの検証が可能になる．他拠点との接続はインターネット経由で VPN を利用する方法と，広域網との連携も視野に入れている．

3. OF-Patch

OF-Patch とは，従来のパッチパネルを SDN 技術である OpenFlow[3]を用いて，OpenFlow スイッチにて実現したものである．従来のパッチパネルはルータやスイッチ，サーバ等の機器をパッチパネルに先行配線しておき，接続構成を変更する場合，パッチパネルで局所的に切り替えるものである．パッチパネルは手動で切り替えるが，OF-Patch は OpenFlow スイッチに置き換え，それをソフトウェアで制御することにより機器同士の接続構成の変更を行う．このため機器同士の接続構成を遠隔よりプログラマブルに変更可能となる．

3.1. 必要性

沖縄オープンラボラトリでは OpenStack/SDN のテストベッドを提供している．検証環境では機器の変更，機器の接続構成の変更などが頻繁に発生するが，従来は物理的な配線を手動で差し替えて変更するため，技術者が現地へ行き配線の変更を行う必要があり，遠隔地に検証環境がある場合は移動する時間やコストがかかっていた．また，検証環境では一時的な環境を構築することが多く，配線が複雑に絡まり，配線にタグがついていない，タグが間違っている，接続先が分からない配線が存在することも良くあり，手動で配線を差し替えるため，接続間違いや，ルーズコンタクトを起こすリスクが高い．テストベッドではこれらの問題を解決し，遠隔よりプログラマブルに機器の接続構成の変更を行う必要がある．このため，沖縄オープンラボラトリでは SDN 技術を用いた OpenFlow パッチパネルを開発した．

3.2. OF-Patch の課題

テストベッドは多数のサーバ，スイッチ，テスタを組み込んでおり，OF-Patch はそれらを繋ぎこむため多数のポートが必要になる．通常の OpenFlow スイッチ単体での使用はポート数が不足するため複数の

OpenFlow スイッチを組み合わせて 1 つの大きなパッチパネルとして見せる必要がある．複数のスイッチを組み合わせるためスイッチ間の接続構成の方式と，どのスイッチを経由するかの経路制御，集約スイッチへの Uplink 配線ではトラフィック集約の方式の検討が必要となる．これらは出来る限り検証環境のトラフィックに影響を与えない方式で実装を行う必要がある．複数のトラフィックを集約するため，検証環境毎のアイソレーションが必要であり，別の検証環境の影響を受け，帯域が圧迫されることが無いよう検討する必要もある．また，遠隔操作での機器接続構成の視覚的な把握や変更操作を容易に行うためグラフィカルユーザインターフェースが必要になる．

4. OF-Patch の実装

本章では，OF-Patch の具体的な実装方法に関して述べる．

4.1. OF-Patch の方式

OF-Patch を大規模化するためには複数の OpenFlow スイッチを組み合わせて一つの大きなパッチとして動作させる必要がある．複数スイッチを組み合わせるトポロジとして Leaf-Spine 方式[4]を採用し，Leaf-Spine 間のトラフィック集約方式，経路制御方式について以下に述べる．

4.1.1. Leaf-Spine 方式

Leaf-Spine トポロジにおいては，Spine スイッチと Leaf スイッチの 2 階層の構成をとり，複数の Spine スイッチに対し，Leaf スイッチは全て接続される．Leaf スイッチをまたがる通信は一旦 Spine を介してから宛先の Leaf スイッチに伝達される．Spine スイッチが複数あるので，どの Spine スイッチを通る経路を選択するかは課題となる．

OF-Patch として用いる場合は Leaf スイッチに検証対象のサーバ，スイッチ等の機器を接続する．OF-Patch の物理的な構成例を図 1 に示す．図 1 の通り，標準的な 1Gbps OpenFlow スイッチ（1Gbps ポートが 48 ポート，Uplink 用 10Gbps ポートが 4 ポート）を Leaf スイッチとして使用し，標準的な 10Gbps OpenFlow スイッチ（10Gbps ポートが 48 ポート）を Spine スイッチとして使用した構成を想定している．従来のパッチパネルと同様に各ポートは 1 ポート to 1 ポート接続でポート間は 1Gbps の帯域を保証する．Leaf-Spine 間の接続については，Leaf スイッチの 40 本の 1Gbps ポートを Uplink 用 10Gbps ポート 4 本を使って Spine スイッチへ接続する．10 本の 1Gbps ポートを Uplink 用 10Gbps ポート 1 本に割り当てることで，各ポート 1Gbps の帯域を保証する．Leaf スイッチの各 Uplink 用ポートをそれぞれ別の Spine スイッチに接続することで Spine スイッチは 4 台と決まる．各 Spine スイッチは 48 ポートであるため Leaf スイッチは最大 48 台と決まる．この構成で Leaf スイッチ 48 台，各 40 ポートで最大 1920 ポートをパッチ可能なパッチパネルとなる．

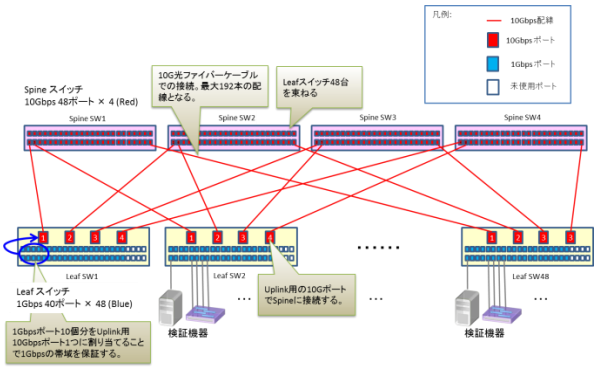


図 1 Leaf-Spine 方式

4.1.2. Leaf-Spine 間のトラフィック多重方式

Leaf スイッチの 10 本分の 1Gbps ポートを Uplink 用の 10Gbps ポート 1 本に多重するため，通信元の機器が接続された Leaf スイッチでトラフィックを多重して Spine スイッチへ送り，Spine スイッチでそれぞれのトラフィックを通信先の機器が接続された Leaf スイッチへ振り分ける必要がある．また，通信先の Leaf スイッチで通信先の機器が接続されたポートに振り分ける必要がある．トラフィックの多重方式は従来のパッチパネルのように検証環境に影響を与えないよう考慮する必要があり，VLAN 等での多重はタグ分フレームサイズが増加するため検証環境の帯域に影響を与えてしまう．また，検証環境が VLAN を使用している場合に，OpenFlow スイッチが 2 重タギングに対応している製品でない場合に VLAN を使用できない懸念などがある．検証環境に影響を与えない Leaf-Spine 間のトラフィック多重方式として MAC アドレス書換え方式を採用した．これは，送信元 MAC アドレス，送信先 MAC アドレスを OF-Patch の内部的な管理アドレスに書き換え，Spine スイッチで管理アドレスを基にそれぞれのトラフィックごとに各 Leaf スイッチへ転送を行う．通信先の Leaf スイッチで管理アドレスを基に元の MAC アドレスに書き直して検証機器へ転送を行う．この方式の場合，MAC アドレスのフィールドを書き換えるのみであるためフレームサイズの増加が無く検証環境の帯域に影響を与えない．また，MAC アドレスのフィールド書換えは OpenFlow1．0 から対応している基本機能で，標準的な OpenFlow スイッチで基本的に備えている機能であるため，現在市場に出回っている多くの OpenFlow スイッチを OF-Patch に使用可能と考える．MAC アドレスを全て内部的な管理アドレスに変換するためデータベースのテーブルのレコード数の増加が懸念される．通常スタンドアロンのマシンでデータベースを動作させた場合，10 万レコード程データがあ

ると検索に数秒かかるようになる．現在，OF-Patch は 200 ポート規模で運用しており管理アドレスは常時 2 千レコード程で検索は数十 msec と言う状況である．

4.1.3. 経路制御方式

Spine スイッチは最大 4 台あり，Leaf スイッチから Spine スイッチに上げる際に，どの Spine スイッチを経由するか経路を選択する必要がある．例えば 1 番目の Spine スイッチから順に埋めていく方法や各 Spine スイッチに均等にトラフィックを振り分ける方法等がある．一つの Spine スイッチに片寄せすると，そのスイッチに障害が発生した場合に影響が大きい．OF-Patch は Leaf-Spine 間の帯域使用率に応じて各 Spine スイッチに均等にトラフィックを振り分ける方式を採用した．データベース上 Leaf-Spine 間の Link に帯域使用率(使用量／容量)を設定する．具体的には分母に Leaf-Spine 間の帯域である 10Gbps を設定し，分子に Leaf スイッチの 1 ポート分使用する毎に 1Gbps を加算することで重み付けを行う．辺の重みを基に経路選択を行うダイクストラ法を用いて，Leaf-Spine 間の Link に設定した重み（帯域使用率）を用いて経路選択を行う．帯域使用率の算出について，現在 1 ポート使用する毎に単純に 1Gbps を加算しているが，今後，ポートのトラフィックを測定し実測に基いた値を加算することで，より正確なトラフィック均等振り分けを行う課題がある．

4.2. OF-Patch Manager の設計

4.2.1. モジュール構成

図 2 に OF-Patch のモジュール構成を示す．

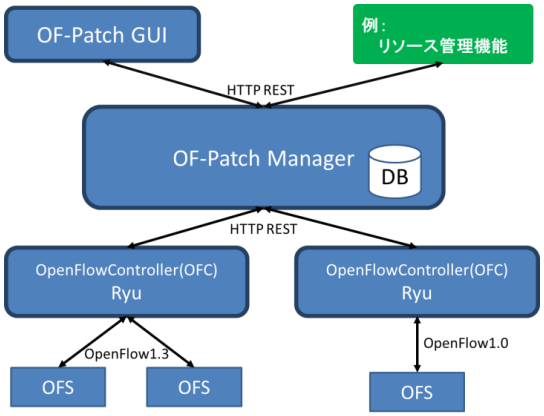


図 2 OF-Patch のモジュール構成

OF-Patch は， OF-Patch GUI， OF-Patch Manager，OpenFlowController（OFC）， OpenFlowSwitch（OFS）の機能から構成される．図 2 の太枠で示すモジュールについて開発を行った．OF-Patch GUI モジュールについては，Web ブラウザ上で動作するアプリケーションであり，グラフィカルな表示を行いユーザビリティなインタフェースを実現する． OF-Patch Manager は OF-Patch GUI からの設定指示を受け，データベースへのデータ登録を行うとともに，OpenFlowController に対しポート接続の設定変更指示を行う．データベースについてはグラフデータベースである OrientDB[5]を用いており，現在主流のリレーショナルデータベースに比べ高速に配線の経路計算をおこなうことができる．OpenFlowController には Ryu[6]を用いて開発を行っている．

4.2.2. OF-Patch Manager API

OF-Patch Manager は上位アプリに対し API を提供している．テストベッドのリソース管理機能からこれを使用し，テストベッドに組み込み，検証機器の接続構成変更の自動化を実現している． また，OF-Patch GUI も本 API を利用している．下記の機器接続構成情報の取得，更新の 2 つの REST API を実装している．

・機器接続構成情報の取得 API

Method：GET

URL：http://[host_name]:[port]/ofpm/logical_topology

URL Param：機器リスト

・機器接続構成の更新 API

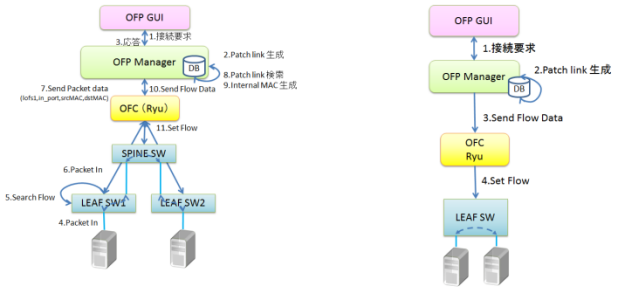
Method：PUT

URL：http://[host_name]:[port]/ofpm/logical_topology

Body：JSON 形式で機器リスト，接続リストを転送

4.2.3. 動作フロー

OF-Patch の動作フローは接続動作時，切断動作時，OpenFlow スイッチが OpenFlow コントローラに接続した時の 3 パターンあり，接続動作は Leaf スイッチを跨ぐ場合のパッチ接続，同一 Leaf スイッチ内でのパッチ接続の 2 パターンがある．図 3 に接続時の動作フローを示す．



(a)LeafSW を跨ぐ場合　　(b)同一 LeafSW 内の場合

図 3 接続時の動作フロー

同一 Leaf スイッチ内でのパッチ接続であるか，Leaf スイッチを跨ぐ接続であるかの判断は OF-Patch Manager にて，データベース内に格納している接続情報よりプログラムにて判断する．API を使用する側は単純に接続元機器，接続元ポートと接続先機器，接続先ポートの情報を OF-Patch Manager へ要求するのみである．

以下，図 3 (a)の Leaf スイッチを跨ぐ接続時動作フローについて説明する．

1. 接続要求を受信

2．Leaf スイッチを跨ぐ場合のパッチ接続の場合は Patch link を接続元 Leaf スイッチ，経由する Spine スイッチ，接続先 Leaf スイッチの 3 組生成し，接続元，接続先それぞれの Leaf スイッチに表 2 Leaf スイッチのフローテーブル No.2 のパケットインを OpenFlow コントローラへ通知するフローを設定する

3．OF-Patch GUI へ応答

4．接続元機器からの通信が発生

5．マッチするフローがあるか検索される．初回の場合，2.で設定したパケットインフローがマッチする

6. OpenFlow コントローラへパケットイン通知が発生する

7．OpenFlow コントローラより OF-Patch Manager へパケットイン情報（Leaf スイッチの dpid, インポート，送信元 MAC アドレス，送信先 MAC アドレス）を通知する

8．dpid，インポート情報を基に Patch link 情報を検索する

9．dpid，インポート，送信元 MAC アドレス，送信先 MAC アドレスを基に内部 MAC アドレスを生成する

10．送信先 Leaf スイッチ，Spine スイッチ，送信元 Leaf スイッチの順で OpenFlow コントローラへフロー設定要求を行う．このとき設定するフローは，Leaf スイッチに対しては，表 2 Leaf スイッチのフローテーブル No.5 のフローを双方向で設定する．Spine スイッチに対しては, 表 1 Spine スイッチのフローテーブル No.2 のフローを双方向で設定する．

11．OpenFlow コントローラより，各 OpenFlow スイッチへフローを設定する．

以下，図 3 (b)の同一 Leaf スイッチ内接続時動作フローについて説明する．

1．接続要求を受信

2. 同一 Leaf スイッチ内でのパッチ接続の場合は Patch link を Leaf スイッチ 1 つ分の 1 組のみ生成する

3．OpenFlow コントローラへフロー設定要求を行う．このとき設定するフローは表 1 Leaf スイッチのフローテーブル No.4 の単純なポート to ポートのフローを双方向で設定する．

4．Leaf スイッチへフローを設定する

4.2.4. フローテーブル設計

OF-Patch の Spine スイッチと Leaf スイッチに設定するフローについて記載する．Leaf スイッチに設定するフローについては同一 Leaf スイッチ内でパッチリンク接続する場合と，Leaf スイッチを跨いでパッチリンク接続する場合の 2 パターンが存在する．Spine スイッチについては Leaf スイッチから上がってきた複数パッチリンクがトランクされた通信をパッチリンク毎に宛先 Leaf スイッチにパケットを送るため, 内部的な MAC アドレスをマッチ条件に宛先を決めるフローを設定する．

表 1 Spine スイッチのフローテーブル

No.		Match	Action
1	Priority=200	*（全て）	Drop
2	Priority=800, Idle_timeout=800	In_port, Src_mac	Output

表 2 Leaf スイッチのフローテーブル

No.		Match	Action
1	Priority=200	*（全て）	Drop
2	Priority=400	In_port	CONTROLLER
3	Priority=600, Idle_timeout=600	In_port	Drop
4	Priority=800	In_port	Output
5	Priority=800, Idle_timeout=65535	In_port, Src_mac	Output, Mod_src_mac, Mod_dst_mac

4.2.5. データスキーマ

OF-Patch のデータベースとして，機器情報，機器に属するポート情報，物理配線情報，OF-Patch 用スイッチ上のパッチリンク情報，内部 MAC アドレス情報を持つ．

機器情報としては機器名，機器のタイプ，OF-Patch 用スイッチである場合 DatapathID 情報，OFC の IP, ポート情報を保持する．ポート情報としてはポート名とポート番号, どの機器に属するかの情報を保持する．物理配線情報は機器のポートとポート間，内部バスリンク情報は機器と機器に属するポート間の接続情報を両方向で持つ. パッチリンク情報として接続元ポート，接続先ポート，OF-Patch 用スイッチ情報，接続元機器名とポート，接続先機器名とポート，シーケンスナンバー情報を持つ. シーケンスナンバーは同一 Leaf スイッチ内の接続では 1 のみ, Leaf スイッチを跨ぐ場合は，接続先 Leaf スイッチのパッチリンクが 1，経由する Spine スイッチのパッチリンクが 2，接続元 Leaf スイッチのパッチリンクが 3 となる．内部 MAC アドレス情報は接続元 Leaf スイッチの機器名, インポート情報, 発信元 MAC アドレス, 宛先 MAC アドレス, 内部 MAC アドレスを持つ. Leaf スイッチの機器名, インポート，発信元 MAC アドレス, 宛先 MAC アドレスを基に内部 MAC アドレスを生成する．

4.2.6. OF-Patch GUI

ユーザが OF-Patch を操作して，機器の接続構成の変更を容易に行えるよう GUI を開発している．本 GUI

は，各機器の名称と機器に属するポートおよびパッチリンクの表示を行う．機器のポートとポートをクリックすることでパッチリンクによる接続操作が行える．また，パッチリンクをクリックすることで切断操作が行える．スクロール操作で拡大・縮小が行え，機器，ポート，パッチリンクをマウスオーバすることで各種情報を表示できる．

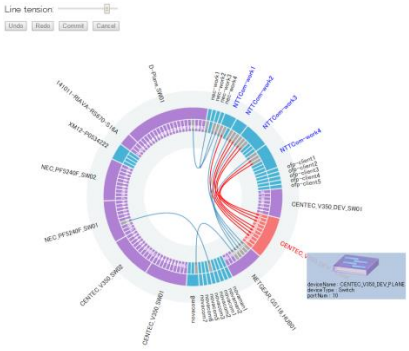


図 4 OF-Patch GUI

5. 動作検証および評価

今回開発した OF-Patch について，動作検証を行い，4 章で述べた設計通り動作することを確認した．まず，市販の OpenFlow スイッチで Leaf-Spine のトポロジを構成し，各 Leaf スイッチに試験用の端末を接続する．OF-Patch Manager のデータベースに機器情報，配線情報を登録する．次に OF-Patch Manager の接続構成更新 API を叩き 2 台の端末間を OF-Patch で接続する．端末が互いに通信可能であることを ping を用いて確認した．通信可能であるとき，Spine スイッチ，Leaf スイッチに期待するフローが設定されていることを確認した．

5.1 拡張性

OF-Patch は Spine スイッチ 4 台，Leaf スイッチ 48 台，最大 1920 ポートのパッチパネルとなる設計を行っている．テストベッドでは当初，Spine スイッチ 2 台，Leaf スイッチ 4 台の構成で OF-Patch を構築した．その後，機器の増加に伴いポートの拡張が必要になり Leaf スイッチを 1 台追加し拡張を行った．Leaf スイッチを追加する場合，データベースに情報を登録し，Leaf スイッチを Spine スイッチに接続するだけで容易に OF-Patch を拡張することができる．現在，テストベッドでは 200 ポート規模で運用を行っており，今後さらに拡張していく予定である．

5.2 スループット

iPerf を用いて OF-Patch の Leaf スイッチを跨ぐ，Spine スイッチ経由でのホスト間の通信速度の測定を実施した．測定方法としてはホスト間で 10 秒間に転送できたデータ量を基に通信速度を算出する方法で，5 回測定の平均速度で約 941Mbps の実効速度が出ることを確認した．ほぼ当初の設計通りの実効速度が出たと考える．今後は精度の高い測定器を用いての測定の実施を検討している．

5.3 遅延

現在，OF-Patch Manager の接続構成更新 API を叩き端末間を OF-Patch Manager 上接続したあと，端末から通信が発生し，端末間の通信が可能となるまで，約 3 秒程度の遅延が発生する．これは OF-Patch Manager が Packet-in した最初のパケットに対して，MAC アドレスを学習し，各 OpenFlow スイッチにフローを設定するまで処理に時間がかかる為である．フローが一旦設定された後は通常のフォワーディング・プレーンのみの遅延となる．

6. まとめ

本稿では，沖縄オープンラボラトリで構築・運用しているテストベッドについて紹介し，その中で用いている OpenFlow スイッチを用いたパッチパネルについて説明した．従来のパッチパネルを SDN 技術である OpenFlow を用いて，OpenFlow スイッチにて実現する方式を検討し，実装してテストベッドに組み込み運用を行っている．今後の課題として，ミラーポートへの転送機能やトラフィックモニタ機能等の検討をおこなうことが上げられる．また，遅延の改善やテストベッドでの運用を通して機能改善を行っていく．

謝辞

本研究の一部は，沖縄県の「クラウドオープンネットワーク国際研究開発拠点形成事業」による補助を受けて実施しています．沖縄オープンラボでは，上記研究成果を世界中のコミュニティへと公開することによりさらに成果を進化・発展させてまいります．

問合先

一般社団法人 沖縄オープンラボラトリ
〒904-2234 沖縄県うるま市州崎 14-17
沖縄 IT 津梁パーク
中核機能支援施設 112 号
TEL：098-989-1940 FAX：098-989-1943
E-mail：info@okinawaopenlabs.org

文 献

[1] www.okinawaopenlabs.org
[2] “OpenStack” https://www.openstack.org/ .
[3] “OpenFlow Switch Specification” https://www.opennetworking.org/ja/sdn-resources-ja/onf-specifications/openflow/ .
[4] http://ccr.sigcomm.org/online/files/p63-alfares.pdf
[5] “OrientDB” http://www.orientechnologies.com/orientdb/ .
[6] “Ryu SDN Framework” http://osrg.github.io/ryu/ .
[7] “iPerf” https://iperf.fr/ .