次世代ネットワーク制御技術（OpenFlow）の検証

○原祐一、伊藤康広、雨宮尚範

工学系技術支援室　情報通信技術系

概要

近年、クラウドサービスの基盤となる仮想技術が目覚ましい進歩を遂げてきた。しかし、この技術の進歩の中で、サーバやストレージ、デスクトップの仮想化に比べて、ネットワークは、仮想化の進みが遅れている。そのため、大規模データセンターなどでは、ネットワークの柔軟性に限界が生じ、新しい技術の進歩を遅らせる原因になりつつある。従来のネットワーク機器は、便利だけれども目的にあったサービスを自由につくれない状態にあると言える。そこで、制約にとらわれない柔軟なネットワークを構築するための技術が求められるようになってきた中で、「ソフトウェアでプログラムするように自由にネットワーク構成を変更できるようなネットワーク」である SDN（Software Defined Networking）という概念が生まれた。現在では、ベンダー固有の機能も含めるため、SDN の対象が拡大している。SDN の導入は、大規模データセンターが中心に進んでいる。今回、SDN の概念を実現する技術の中の 1 つである OpenFlow の研鑽を行う目的で、全学技術センターの技術研鑽プログラムに応募したところ採択された。本研修会では、技術研鑽プログラムで行ったことを報告する。

1　従来のネットワーク機器

従来のネットワーク機器は、RFC の規定どおり動作すれば良く、ソフトウェアやハードウェア、その上で動作するサービスをベンダーが独自開発しているため、ベンダーに依存された垂直統合モデルである。そのため、RFC に規定がないサービスを導入することがとても困難である。

図 1. 従来のネットワーク機器

1.1　仮想化時代における問題点

- ベンダーごとに機器の設定方法等を習得する必要があり、高い技術レベルをもつ要員が必要になる。マルチベンダ構成にすることが難しい。
- 従来の商用ネットワーク機器は、フローテーブルを用いたパケット転送を行う必要があり、L2 スイッチの各ポートに VLAN を設定することになる。そのため、仮想環境下において、VLAN を超えてのライブマイグレーションの実現が難しい。

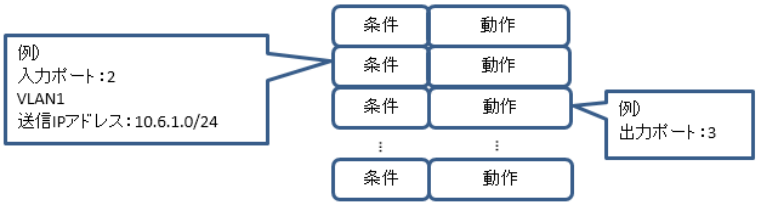

図 2. フローテーブル

- 自由度が低いため、目的にあったネットワークデザインを組むことが難しい。

2　OpenFlow

OpenFlow は、「ソフトウェアでプログラムするように自由にネットワーク構成を変更できるようなネットワーク」である SDN の概念が生まれる前から存在し、2007 年ごろ、スタンフォード大学の研究グループから誕生したプロジェクトである。現在は、2011 年に発足した ONF（Open Networking Foundation）で標準化及び普及が進められている。

2.1　OpenFlow の構成

ネットワーク上でデータを転送するためには、「データプレーン」「コントロールプレーン」「アプリケーション」の３つの要素が必要である。従来のスイッチでは、この３つの要素が一体となって同じ機器で実装されている。

表 1. ネットワークデータ転送の 3 要素

要素名	機能
データプレーン	パケットの転送を行う
コントロールプレーン	パケット転送における経路計算や制御などを行う
アプリケーション	各プロトコルの機能を提供する

OpenFlow では、３つの要素が分離されていることに特徴があり、OpenFlow スイッチ、OpenFlow コントローラー、OpenFlow プロトコルの３つで構成される。「データプレーン」と「コントロールプレーン」は、メーカーが販売しているネットワーク機器である必要がなく、ユーザーが作ったソフトウェアでも動作させることが可能である。

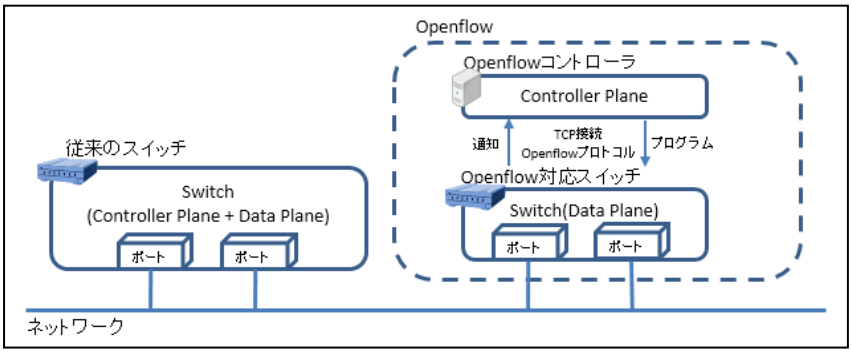

図 3. 従来のスイッチと OpenFlow スイッチ

2.2　OpenFlow のバージョン

現在の最新バージョンは、1.4 であり、各バージョンの間では、互換性がない。また、ipv6 は、1.2 からの実装である。よく使用されるバージョンは、「1.0 + ベンダー拡張」である。それ以外では、「1.3」が使われており、1.4 の利用は少ない。また、1.1、1.2 は使われていない。1.5 の仕様検討も進められており、1.5 が主流になるとも言われている。

2.3　OpenFlow スイッチ

データプレーンの制御を行う。「フロー」と呼ばれる OpenFlow 独自の概念によってデータトラフィックを識別し、フロー毎にトラフィックを制御し、ヘッダ情報（L2 から L4）の書き換え、転送、破棄といった処理を行う。そして、従来のスイッチと違い、スイッチ自身にコントローラーをもたないため、スイッチ自身が隣のネットワーク機器と通信して最適な転送先を選択するといった高度なルーティングプロトコルの動作はしないという特徴があり、高度な制御を行う場合、OpenFlow コントローラーが必要となる。

表 2. OpenFlow スイッチのフロー

フロー（Flow）	スイッチの入力ポート、送信元 MAC アドレス、宛先 MAC アドレス、Ethernet タイプ、VLAN ID、VLAN 優先度、送信元 IP アドレス、宛先 IP アドレス、IP プロトコル、ToS ビット、送信元 TCP/UDP ポート、宛先 TCP/UDP ポート

OpenFlow 1.0 の仕様に基づくと、OpenFlow は、「セキュアチャネル」を使って、「OpenFlow プロトコル」でコントロールとメッセージのやり取りを行い、「フローテーブル」の定義に従ってパケット転送するスイッチとなる。

表 3. OpenFlow スイッチの構成要素

要素名	説明
フローテーブル	各フローの定義と制御処理の設定が格納されるテーブル
セキュアチャネル	OpenFlow プロトコルによって、OpenFlow コントロールからスイッチを制御するための通信路（TCP/IP 通信）
ポート	トラフィックを受け付けるインターフェース
キュー	QoS といったトラフィックを制御するための機構

2.4 OpenFlow プロトコル

OpenFlow プロトコルは、OpenFlow スイッチと OpenFlow コントロール間において、TCP/IP ネットワーク経由で通信するためのプロトコルである。また、通信には、TLS による暗号化と認証が行われる。OpenFlow プロトコルでは、コントローラーからスイッチに対する制御に加えて、スイッチのどのフローテーブルにも合致しないパケットの処理、エラー発生時の処理、新しいスイッチがネットワークに接続された時の処理、スイッチからコントローラーに対する各種イベントの通知などが行われる。

2.5 OpenFlow コントローラー

OpenFlow で定められている仕様は、OpenFlow スイッチと OpenFlow プロトコルのみであり、OpenFlow コントローラーの動作自体には仕様は定められていない。そのため、ユーザーが独自に定義できるようになっている。

OpenFlow コントローラーは、主に表.4 の 2 種類の動作で通信を制御する。

表.4 OpenFlow コントローラーの主な動作

フローエントリの追加	OpenFlow スイッチにルールを書き込む
パケットアウト	パケットを OpenFlow スイッチから送出する

OpenFlow コントローラーをユーザーがフルスクラッチで 1 から作ることは大変である。そのため、いくつかの OpenFlow コントローラーフレームワークが存在する。

表.5 OpenFlow コントローラーフレームワーク一覧

名称	プログラミング言語	ライセンス
NOX	C++	GPL v3
POX	Python	GPL v3
Trema	Ruby	GPL v2
Beacon	Java	GPL v2
Ryu	Python	Apache 2.0
OpenDayLight	Java	EPL 1.0
Floodlight	Java	Apache 2.0

本研鑽では、「Floodlight」を利用することにした。

3　検証

3.1　OpenFlow 動作検証環境

本研鑽では、仮想サーバで OpenFlow が利用できることを前提に調査を行うこととし、OpenFlow の L2 の機能を中心に研鑽を行った。仮想サーバは、Citrix 社の XenServer と KVM で評価を行うため、図.4 のように動作検証用の環境を準備した。

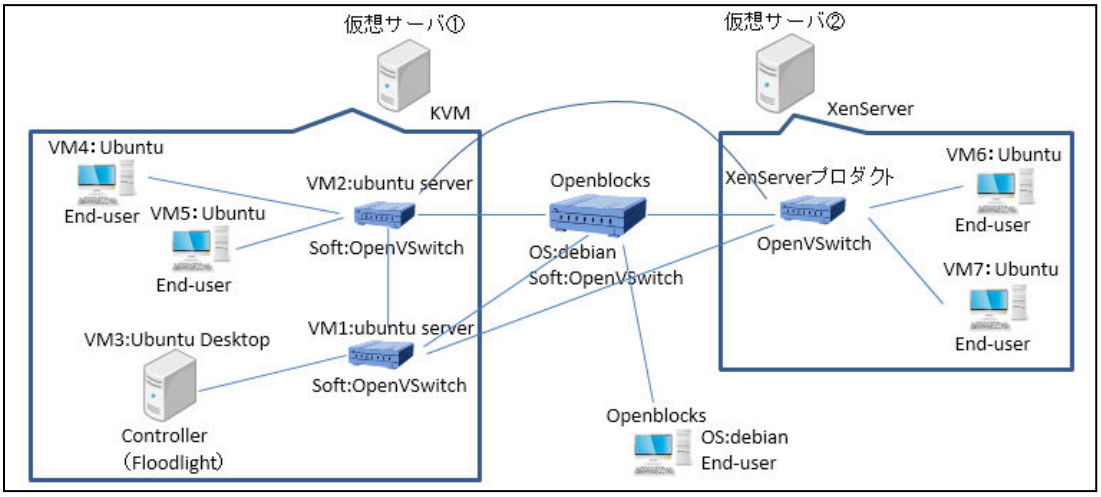

図 4. 動作検証環境イメージ

3.2　プログラミング及び動作確認

OpenFlow でラーニングスイッチを構築し、Floodlight で MAC アドレスをベースにパケット制御を行うようにプログラムを組むことで、パケットが制御できるか、以下の順番で確認を行った。

1) KVM で OpenFlow によるパケット制御を行えるか
2) XenServer で OpenFlow によるパケット制御を行えるか
3) KVM と XenServer の間で、パケット制御を行えるか

1～3 とも、Floodlight でプログラミングしたとおりにパケット制御が行えることを、パケットキャプチャで拾ったパケットの確認を行い、パケットがプログラムどおり動作することを確認した。

4　まとめと今後の展開

OpenFlow で利用できる L2 の機能の動作確認まで完了した。現在、仮想サーバによる管理が増えてきている。今後の展開として、仮想サーバで OpenFlow を利用すると、仮想スイッチ上でパケットが処理されるため、物理的に見えないパケットが多くなり、トラブル時に対応が難しくなる。Hinemos などを使い経路の可視化をできるようにする必要がある。また、既存のネットワークと混在させたときにもトラブルが発生しないようにする必要がある。実際の運用に耐えうるところまで研鑽を行い、OpenFlow をサーバ室でも利用できるようにしたいと思う。

参考文献

[1] 馬場達也・大上貴充・関山宣孝・高畑知也/著、『OpenFlow 徹底入門　SDN を実現する技術と知識』、翔泳社

[2] 石井秀治・大山裕泰・河合栄治/著、『次世代ネットワーク制御技術　OpenFlow 入門』、KADOKAWA/アスキー・メディアワークス

[3] 渡辺和彦・法橋和昌・沢村利樹・池上竜之/著、『ネットワーク仮想化　基礎からすっきりわかる入門書』、リックテレコム