ホップバイホップ方式およびオーバーレイ方式に対応した OpenFlow ネットワーク移行支援システム

野村圭太[†1]　谷口義明[†2]　井口信和[†2]　渡辺健次[†3]

概要：今後，企業や大学等の既存ネットワークにおいて OpenFlow ネットワークへの移行が進められると予測される．しかし OpenFlow ネットワークへの移行には，OpenFlow コントローラの設定にコストを要すると考えられる．そこで本稿では，従来型のネットワークから OpenFlow ネットワークへの移行を支援するシステムを開発する．本システムは，OpenFlow ネットワークの主要な実現方式であるホップバイホップ方式，およびオーバーレイ方式への移行に対応している．本システムを用いることにより，従来型のネットワーク上の機器から自動的に設定情報の取得，取得した設定情報の変換，および OpenFlow ネットワークへの反映を行える．これにより，従来型のネットワークと同等のパケット制御を行う OpenFlow ネットワークを自動的に構築できる．本システムを用いることにより，従来型のネットワークをホップバイホップ方式，およびオーバーレイ方式の OpenFlow ネットワークへ移行できることを確認した．

キーワード：Software-Defined Networking, OpenFlow, NETCONF

A System for Supporting Migration to Hop-by-hop/Overlay OpenFlow Network

KEITA NOMURA[†1]　YOSHIAKI TANIGUCHI[†2]　NOBUKAZU IGUCHI[†2]
KENZI WATANABE[†3]

Abstract: It is expected that traditional networks of various organization will migrate to OpenFlow networks in the future. However, it takes time and costs for setting OpenFlow controller for migration. In this report, we develop a system for supporting migration from a traditional network to an OpenFlow network. Our system supports migration to both hop-by-hop OpenFlow networks and overlay OpenFlow networks, which are two main implementation methods for OpenFlow networks. Our system automatically acquires setting information from the traditional network, converts the information appropriately, and reflects them to the OpenFlow network. Through experimental verification, we confirm that a traditional network is migrated to a hop-by-hop OpenFlow network and an overlay OpenFlow network by using our system.

Keywords: Software-Defined Networking, OpenFlow, NETCONF

1. はじめに

クラウド環境やサーバ仮想化の普及に伴い，ネットワークに対する要件が変化してきている．例えば，サーバ仮想化技術により，仮想サーバの追加や移動が可能となった．このような環境の変化に伴い，ネットワークの構成やルータ，スイッチなどのネットワーク機器の設定を変更する必要が生じる場合がある．しかし，従来型のネットワーク機器で構成されるネットワークでは，各ネットワーク機器を手動で設定する必要がある．そのため，ネットワークの構成や機器の設定の変更に手間を要し，サーバの追加や移動などの環境の変化に柔軟に対応することが困難となっている．そこで，このような環境の変化に柔軟に対応するための仕組みが求められている．

そうした背景から，Software-Defined Networking (SDN) というコンセプトが注目を集めている [1, 2]．従来型のネットワークでは各ネットワーク機器上に，経路制御を行うソフトウェアや OS に相当する制御部と，フレームやパケットの転送を行うハードウェアに相当する転送部の両方がある．これに対し SDN では，制御部と転送部が分離されている．従来型のネットワークでは，制御部の実装はネットワーク機器のベンダに依存しているため，ベンダが提供する機能以外用いることができない．一方，SDN では，管理者が自由に制御部を開発できるためベンダ依存が解消され，用途に応じた柔軟なネットワークを構築できる．

SDN を実現するための技術のひとつに OpenFlow [3]がある．今後，企業や大学などの既存ネットワークにおいて OpenFlow ネットワークへの移行が進められると予測されている [4-6]．OpenFlow ネットワークの主要な実現方式として，ホップバイホップ方式とオーバーレイ方式の二つがある．ホップバイホップ方式では，全てのネットワーク機器が OpenFlow スイッチにより構成される．そのため，柔軟なネットワーク制御が可能となる．一方，オーバーレイ方式では，ネットワークのエッジにあるサーバなどの機器

†1 近畿大学大学院　総合理工学研究科
Graduate School of Science and Technology，Kindai University
†2 近畿大学　理工学部　情報学科
School of Science and Engineering，Kindai University
†3 広島大学大学院　教育学研究科
Graduate School of Education, Hiroshima University

間で構築したオーバーレイネットワーク上で OpenFlow による制御を行う．ホップバイホップ方式と比較すると柔軟な制御は行えないが, OpenFlow に対応していない既存のネットワーク機器を使用することができるため OpenFlow ネットワークの導入コストが抑えられる．

OpenFlow ネットワークは従来型のネットワークとは異なるアーキテクチャである．そのため，OpenFlow に習熟していない管理者にとって，従来型のネットワークから OpenFlow ネットワークへの移行や OpenFlow コントローラの設定は困難である．また，OpenFlow のアーキテクチャに習熟している管理者においても, OpenFlow ネットワークへの移行や OpenFlow コントローラの設定には時間を要する．そのため，このようなコストを削減するために OpenFlow ネットワークへの移行を支援するシステムが求められる．

我々はこれまでに，従来型のネットワークからホップバイホップ型 OpenFlow ネットワークへの移行を支援するシステムの開発・検証 [7]，および従来型のネットワークからオーバーレイ型 OpenFlow ネットワークへの移行を支援するシステムの検討 [8]を行ってきた．

本稿では，これら二つの方式に対する OpenFlow ネットワークへの移行支援システムを統合したシステム（以降，本システム）について述べる．本システムは，我々がこれまでに開発してきたシステム [7, 8]の機能拡張および実装を行ったものである．本システムを用いることにより，従来型のネットワークと同等のパケット制御を行うホップバイホップ方式，あるいはオーバーレイ方式の OpenFlow ネットワークへの移行を容易に実現できる．

本稿の構成は以下の通りである．まず，2 章で関連研究について述べる．次に，3 章で本システムの概要について述べ，4 章，5 章で本システムの詳細について述べる．6 章で実験を行った結果とそれに対する考察について述べる. 7 章でまとめと今後の課題について述べる．

2. 関連研究

本章では，本研究と関連する研究について述べ，本システムとの比較を行う．

Exodus [9] は，本稿におけるホップバイホップ型 OpenFlow ネットワークへの移行支援機能と同様に, 従来型のネットワーク上のルータの設定情報を OpenFlow ネットワークで適用可能な形式に変換する．Exodus では，設定情報の自動取得については考慮されておらず, Cisco IOS の設定情報をあらかじめ OpenFlow コントローラで保持しておく必要がある．さらに，現状，Exodus は Cisco IOS に対応しているルータの設定情報のみ変換が可能であり，他のベンダのルータの設定情報を変換するためには，OS の情報を変換するモジュールを追加で開発する必要がある．これに対し，本システムでは，さまざまなベンダのネットワーク機器で採用されている NETCONF [10]を用いてルータから設定情報を取得し，XML ファイルに変換する．そして，取得した XML ファイルに基づき，ルータの設定情報を OpenFlow ネットワークで適用可能な形式に変換する. そのため Exodus に比べ, 他のベンダに対応する際のモジュール開発を容易に行え，マルチベンダの環境に対応できる．

NEC ビッグローブ株式会社が開発したクラウドコントローラである Momiji [11]は，仮想化したサーバとネットワークを一括制御できる. OpenFlow コントローラなどの設定には，ウィザード形式のポータルを使用する．プログラミングによる OpenFlow コントローラ開発と比較した場合，操作が簡略化されるが，設定対象のサーバ数や設定項目数の増加に伴い, 設定に要する時間が増加すると考えられる．これに対し本システムでは, OpenFlow コントローラの制御対象のサーバから設定情報を自動的に取得する．これにより，サーバ数が増加した場合においても設定に要する時間を抑えることができる．

3. システム概要

本章では，本システムの概要について述べる．本システムの構成を図 1 に示す．本システムは，従来型のネットワークの管理用セグメント，および OpenFlow ネットワークにおける OpenFlow チャネルによる接続性が保たれる管理用セグメントの両方に接続されるものとする．本システムは大きく分けて，設定情報取得機能，設定情報変換機能，および設定情報反映機能から構成される．

従来型のネットワークから OpenFlow ネットワークへ移行する場合，まず，利用者（ネットワーク管理者）は，本システムの設定情報取得機能を用いて従来型のネットワーク上の機器から設定情報を取得する．ここで，ホップバイホップ型 OpenFlow ネットワークに移行する場合は，ネットワーク上のルータから NETCONF により経路表や ACL (Access Control List) などの設定情報を取得する．オーバーレイ型 OpenFlow ネットワークに移行する場合は，移行対象のサーバから SSH により仮想マシンや仮想スイッチの

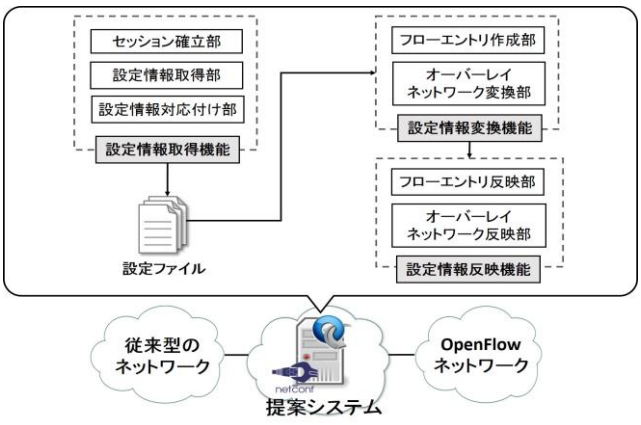

図 1　システムの全体構成
Figure 1　Overall system architecture

設定情報を取得する．どちらのネットワークに移行する場合でも，取得した設定情報は設定ファイルとして保存される．また，取得した設定情報は，設定情報変換機能によりOpenFlow ネットワークの構築に適した形式に変換される．

次に，設定情報反映機能を用いて変換した設定情報をOpenFlow ネットワークに反映する．オーバーレイ型OpenFlow ネットワークに移行する場合は, 変換した設定情報に基づき，適切なオーバーレイネットワークの構築を同時に行う．なお，本システムではプロアクティブ型の制御を想定しており，設定情報反映機能を使用した時点で移行前の従来型のネットワークと同等の動作を行うよう，各OpenFlow スイッチあるいは仮想スイッチである Open vSwitch にフローエントリが書き込まれる．以降，本システムは OpenFlow コントローラとして動作する．

以下, 4 章でホップバイホップ型 OpenFlow ネットワークへの移行支援機能の詳細を，5 章でオーバーレイ型OpenFlow ネットワークへの移行支援機能の詳細を述べる．

4. ホップバイホップ型 OpenFlow ネットワークへの移行支援機能

本章では，本システムのうち，ホップバイホップ型OpenFlow ネットワークへの移行支援機能（以降，本機能）の詳細を述べる. 本機能は 我々がこれまでに開発したシステム [7]を基に，さらに OSPF の設定情報も移行できるよう拡張したものである．

4.1 前提

初めに，図 2 に本機能の概要を示す．従来型のネットワークは，全てのネットワーク機器がルータで構成されており，任意のルータにホストが接続されているものとする．移行対象とするネットワークのルーティングプロトコルとして RIP，OSPF，同じく ACL として，標準 ACL，標準名前付き ACL, 拡張 ACL, 拡張名前付き ACL を対象とする．本機能では，従来型のネットワークから OpenFlow ネットワークへ完全に移行することを想定し，あらかじめ実機で構成された OpenFlow ネットワークが物理的に結線され，構築されているものとする．移行するホップバイホップ型

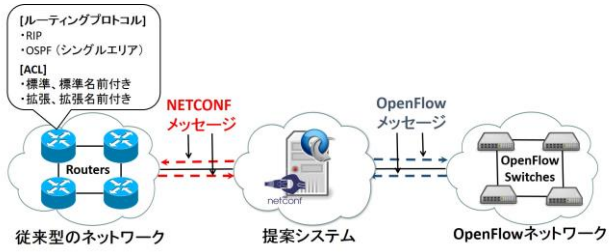

図 2　ホップバイホップ型 OpenFlow ネットワーク移行支援機能の概要

Figure 2　Overview of proposed system for migration to hop-by-hop OpenFlow network

OpenFlow ネットワークは, 従来型のネットワークと同一トポロジであり，従来型のネットワークにおけるルータとホップバイホップ型 OpenFlow ネットワークにおける OpenFlow スイッチが一対一で対応する．

本機能では，ネットワークの経路制御が安定して動作している従来型のネットワークを移行の対象とする．安定して動作しているネットワークでは，ネットワーク障害などによる経路切り替えは起こりづらいため，設定情報取得の同期ずれに伴う問題は想定しない．また，本機能では，RIP や OSPF，ACL の設定が施されたルータで構成された従来型のネットワークを OpenFlow ネットワークへ移行することを移行ケースとして想定している．これは，従来型のネットワークのうち一部分を OpenFlow に移行しても十分な効果が得られるため [6]，企業や大学などの既存ネットワークにおいて，ルータ部分のみを OpenFlow に移行することが考えられるためである．なお，本稿における提案システムを拡張することで，マルチエリア OSPF や IGRP などを用いた大規模なネットワークの移行支援にも対応できる．現状，本機能は，OpenFlow ネットワークへ移行後のトポロジ変化に対応していないが，RIP や OSPF の経路制御機能を移行することで対応可能であり，今後，これらの機能を順次開発する予定である．

4.2 設定情報取得機能

設定情報取得機能は，図 3 に示すシステム GUI により提供される．利用者は，まず，システム GUI を用いて設定情報を取得したいルータの IP アドレスと SSH 接続に必要なパスワードを入力し，取得を行うボタンを押下する（図 3 左）．このことにより，本システムは，指定された IP アドレスを持つルータに対して SSH を用いて接続し，NETCONF により設定情報を自動的に取得する．取得する設定情報は，経路表，標準 ACL，標準名前付き ACL，拡張 ACL，拡張名前付き ACL，およびインターフェイスの設定情報である．ここで，経路表に含まれる経路情報のうち，直接接続経路，RIP または OSPF により学習した経路が移行の対象となる．

さらに，利用者は，システム GUI を用いてルータと OpenFlow スイッチの対応付け情報を入力する（図 3 右）．

図 3　システム GUI

Figure 3　System GUI

また，ルータのインターフェイスと各 OpenFlow スイッチのポートの対応付け情報を自動的に取得する．

4.3 設定情報変換機能

設定情報変換機能は，設定情報取得機能により得た設定情報を OpenFlow ネットワークで適用可能な形式に変換する．具体的には，OpenFlow スイッチごとにフローテーブル 0 からフローテーブル N+1 までの合計 N+2 個のフローテーブルを作成する．ここで N は，移行対象とする経路の種類である．現状のシステムでは，直接接続経路，RIP または OSPF により学習した経路，の 3 種類の経路情報を移行の対象としているため，N=3 である．経路のアドミニストレーティブディスタンス値（AD 値）の順にフローテーブルを作成するため，フローテーブル 1 が直接接続経路用（AD 値 0），フローテーブル 2 が OSPF により学習した経路用（AD 値 110），フローテーブル 3 が RIP により学習した経路用（AD 値 120）のフローテーブルとなる．なお，フローテーブル 0 はインバウンド方向の ACL，フローテーブル N+1 はアウトバウンド方向の ACL，を管理するために用いる．以降，経路表の情報（ホストとの直接接続経路の情報，RIP または OSPF により学習した経路情報）および ACL の情報をフローエントリに変換する手順について述べる．

4.3.1 経路表の情報の変換

今回機能拡張を行った，ルータ R1 の OSPF に関する経路情報を OpenFlow スイッチ OFS1 のフローエントリに変換する動作例を図 4 に示す．なお，本機能では，シングルエリア OSPF を対象とするためエリア情報については考慮しない．本機能では，ルータから取得した経路表のうち，RIP または OSPF により学習した経路エントリそれぞれに対して，経路情報に含まれる宛先ネットワークアドレスをマッチフィールド，経路情報のネクストホップ IP アドレスに対応する OpenFlow スイッチとの接続ポートを宛先への出力ポートとするフローエントリを作成する．図 4 の場合，経路表の上から 1 番目のエントリがこれに対応する．また，直接接続経路の場合，ルータから取得した経路表の直接接続経路エントリのうち，ホストとの直接接続経路エントリが存在するかどうかを調べる．存在する場合，それぞれの経路エントリに対して，経路情報に含まれる宛先ネットワークアドレスをマッチフィールド，ホストと接続している接続ポートを宛先への出力ポートとするフローエントリを作成する．また，直接接続経路，RIP，および OSPF の情報を変換したフローエントリのプライオリティは 2 に設定する．このようにして作成した OSPF の経路エントリを変換後のフローエントリの例を図 4 右に示す．

経路情報を管理するフローテーブル M のいずれのフローエントリにもマッチしなかった場合，フローテーブル M+1 のフローエントリとの比較を行う処理に遷移するよう，フローテーブル M 中にフローテーブル M+1 に遷移するフローエントリをプライオリティ 1 で作成する．ここで M は，現在フローエントリとの比較処理を行っているフローテーブルを示し，OSPF であれば M=2 である．

4.3.2 ACL 設定情報の変換

ACL がルータに設定されていた場合には，その情報もフローエントリに変換する．

図 5 にインバウンド方向の ACL をフローエントリに変換する例を示す．インバウンド方向の ACL に対応するフローエントリはフローテーブル 0 で管理される．ルータのインターフェイスにインバウンド方向の ACL が設定されている場合，それぞれの ACL に対して，該当インターフェイスに対応する OpenFlow スイッチのポートを，フローエントリ中のマッチフィールドの inport に設定したフローエントリを作成する．ACL のアクセス条件が permit の場合にはフローテーブル 1 に遷移するようインストラクションを設定し，deny の場合にはパケットを破棄するインストラクションを設定する．マッチフィールドには，ACL のルールを設定する．またプライオリティは，OpenFlow の最大プライオリティである 65535 から ACL の処理順序を決定する番号の値を引いたものを設定する．このことにより，従来型のネットワークと同じ処理順序で ACL を適用できる．なお，経路表を管理するフローテーブルと同様に，フローテーブル 0 においてもフローテーブル 1 に遷移するフローエントリをプライオリティ 1 で作成する．このようにして設定されたフローエントリの例を図 5 右に示す．

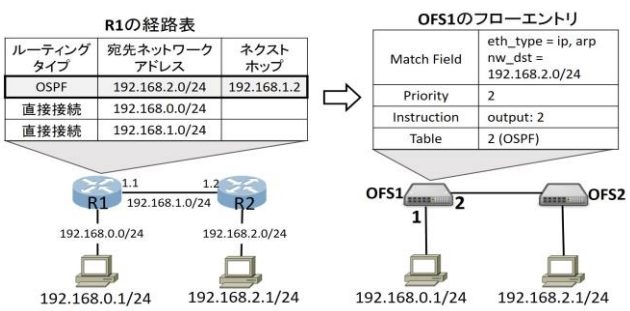

図 4　OSPF 経路情報の変換例

Figure 4　Conversion of an OSPF routing entry to a flow entry

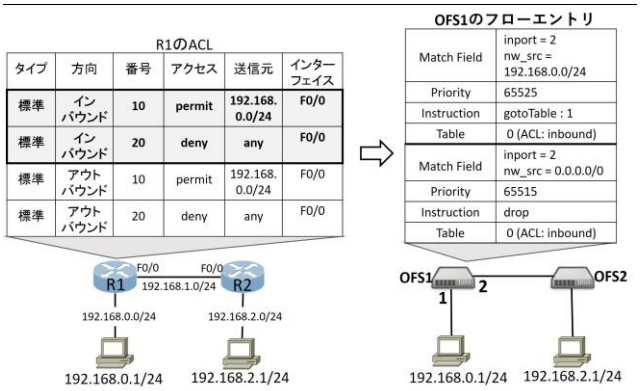

図 5　インバウンド方向 ACL の変換例

Figure 5　Conversion of inbound ACL to a flow entry

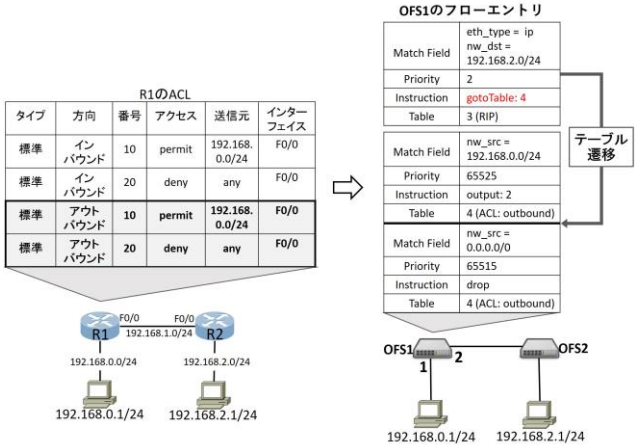

図 6　アウトバウンド方向 ACL の変換例
Figure 6　Conversion of outbound ACL to a flow entry

図6にアウトバウンド方向の ACL をフローエントリに変換する例を示す. アウトバウンド方向の ACL に対応するフローエントリはフローテーブル 4（N+1）で管理される. ルータのインターフェイスにアウトバウンド方向の ACL が設定されている場合，まず，該当インターフェイスに対応する OpenFlow スイッチのポートを取得する. 次に, 4.3.1 節で述べた，経路表のエントリにより作成したフローエントリの中から，取得した OpenFlow スイッチのポートと同一のポートが出力ポートとして指定されているフローエントリを調べる．そして，それらのフローエントリのインストラクションをフローテーブル 4 に遷移するよう変更する. ACL のアクセス条件が permit の場合, 経路表を管理するフローテーブルで本来設定されていたインストラクションをフローテーブル 4 のフローエントリに設定する. ACL のアクセス条件が deny の場合, パケットを破棄するインストラクションを設定する．マッチフィールドとプライオリティはインバウンド方向の ACL と同様に設定する. このようにして設定されたフローエントリの例を図 6 右に示す.

4.4 設定情報反映機能

設定情報反映機能は，設定情報変換機能で作成したフローテーブルを各 OpenFlow スイッチに反映するために用いられる．各 OpenFlow スイッチにおいて，設定情報変換機能を用いて生成された N+2 個のフローテーブルにより，図 7 に示すような手順でパケットが処理されるようになる．ここで，従来型のネットワークにおけるルータはパケットを受信した際，インバウンド方向の ACL に従った処理，経路表のエントリに従ったパケットのフォワーディング，アウトバウンド方向の ACL に従った処理の順に処理を行う．なお，経路表に同じ宛先への経路が複数存在する場合，最小の AD 値を持つ経路を選択する. OpenFlow スイッチにおいて，図 7 に示される手順でパケットが処理されることにより，従来型のネットワークにおけるルータと同等のパケット処理を OpenFlow スイッチで実現できる.

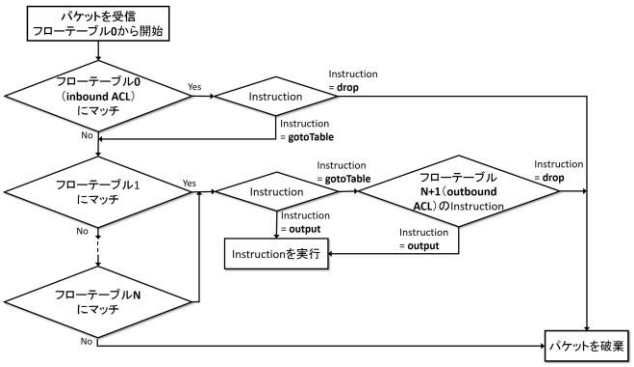

図 7　パケット処理のフローチャート
Figure 7　Flowchart of packet processing

5. オーバーレイ型 OpenFlow ネットワークへの移行支援機能

本章では，本システムのうち，オーバーレイ型 OpenFlow ネットワークへの移行支援機能（以降，本機能）の詳細を述べる．本機能は[8]で検討したシステムに基づいている.

5.1 前提

図 8，図 9 に本機能で移行対象とする従来型のネットワークにおけるサーバ構成と移行後のオーバーレイ型 OpenFlow ネットワークを示す.

従来型のネットワークでは，各サーバが Linux ベースのハイパーバイザである KVM（Kernel-based Virtual Machine) を用いて仮想化されており，複数台の仮想マシンがサーバ上で動作している．また，各仮想マシンは，Linux 標準の仮想スイッチである Open vSwitch に接続されているものとする．各サーバ間は既存ネットワークを用いて接続されており，VLAN によりセグメント分割されている．図 8 の例では，各サーバ上で二つの仮想マシンと一つの Open vSwitch が動作しており，各仮想マシンに異なる VLAN が割り当てられている.

移行後のオーバーレイ型 OpenFlow ネットワークでは，VXLAN により，セグメント分割と Open vSwitch 間のオーバーレイネットワークの構築が行われるものとする. なお，移行後のオーバーレイ型 OpenFlow ネットワークでは，本システムは各サーバとの接続性が保たれる管理用セグメントに配置され，OpenFlow コントローラとして動作する.

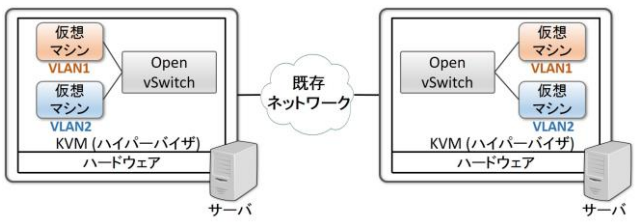

図 8　従来型のネットワークにおけるサーバ構成
Figure 8　Server structures in the traditional network

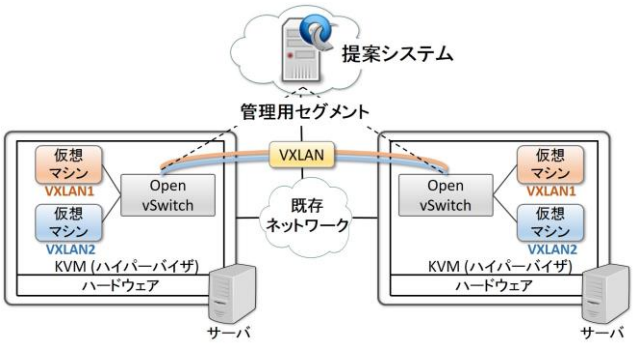

図 9　オーバーレイ型 OpenFlow ネットワーク
Figure 9　Overlay OpenFlow network

5.2 設定情報取得機能

設定情報取得機能は，ホップバイホップ型ネットワークへの移行支援機能と同様，システム GUI により提供される．利用者は，設定情報を取得する全てのサーバの IP アドレスを入力し，取得を行うボタンを押下する．その後，本システムは指定された IP アドレスを持つサーバに対して，SSH により設定情報を自動的に取得する．取得する設定情報は，VLAN タグや仮想マシンとの接続インターフェイス，仮想マシンの IP アドレスと MAC アドレス，Open vSwitch との接続インターフェイスである．

5.3 設定情報変換機能

設定情報変換機能は，設定情報取得機能により得た設定情報を，オーバーレイネットワークの構築に必要な形式およびフローエントリの作成に必要な形式に変換するために用いられる．

まず，取得した設定情報をオーバーレイネットワーク構築のために必要な情報に変換する．設定情報のうち，VLAN タグ，仮想マシンと物理サーバの IP アドレスを確認し，異なるサーバ上に同一 VLAN セグメントに属する仮想マシンが存在する場合，それらに同一の VXLAN ID を割り当てる．また，VLAN によりセグメント分割されていない仮想マシンが複数存在する場合，各仮想マシンを同一セグメントに属するものと判断し，新規に VXLAN ID を割り当てる．

次に，取得した設定情報をフローエントリに変換する．本機能では，同一セグメントに属する全ての仮想マシンの組み合わせに対してアウトバウンド方向とインバウンド方向のフローエントリが作成される．図 10 にフローエントリの例を示す．アウトバウンド方向のフローエントリのマッチフィールドには，送信元仮想マシンを識別する IP アドレスと MAC アドレスの組み合わせ，宛先仮想マシンの IP アドレスを記述する．インストラクションには，Open vSwitch の論理ポートを出力ポートとして記述する．インバウンド方向のフローエントリのマッチフィールドには，アウトバウンド方向のマッチフィールドと同様に記述する．インストラクションには，宛先仮想マシンと接続している Open vSwitch の物理ポートを出力ポートとして記述する．

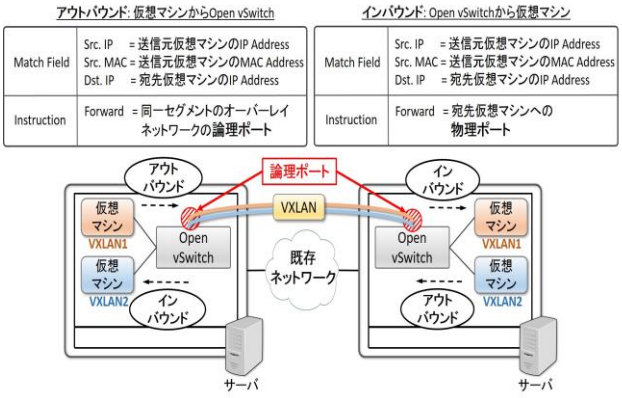

図 10　オーバーレイ型 OpenFlow ネットワークにおけるフローエントリの例
Figure 10　An example of flow entries in the overlay OpenFlow network

5.4 設定情報反映機能

設定情報反映機能は，5.3 節で変換した設定情報に基づき，オーバーレイネットワークを構築し，各 Open vSwitch にフローエントリを反映するために用いられる．オーバーレイネットワークを構築する際には，サーバと SSH を用いて接続し，オーバーレイネットワークの設定情報を Open vSwitch へ反映する．フローエントリを反映する際には，OpenFlow コントローラである本システムと Open vSwitch 間のコネクションを OpenFlow チャネルにより確立し，Flow-mod メッセージを用いてフローエントリを反映する．

6. 検証および考察

本章では，ホップバイホップ型 OpenFlow ネットワークおよびオーバーレイ型 OpenFlow ネットワークへの移行支援機能の動作検証およびフローエントリ数に関する評価を行う．

6.1 ホップバイホップ型 OpenFlow ネットワークへの移行支援機能の動作検証

まず，本機能を用いることにより，従来型のネットワークからホップバイホップ型 OpenFlow ネットワークへ移行できるかについて動作検証を行った．図 11 に示す 8 種類のトポロジで実験を行い，今回機能拡張を行ったシングルエリア OSPF の動作を確認した．

紙面の都合上，動作検証を行った 8 種類のトポロジのうち，最も複雑な複合型トポロジ B における検証結果を示す．なお，複合型トポロジ B は RIP，OSPF，および ACL を用いて構築している．本機能により変換された OpenFlow スイッチ OFS9 のフローエントリの一部を図 12 に示す．ルータの設定情報と図 12 を比較すると，ルータの設定情報に対応したフローテーブルが作成されていることを確認できた．

なお，いずれのトポロジの場合においても移行に必要な時間は 6 分以内であった．これらの結果から，本システム

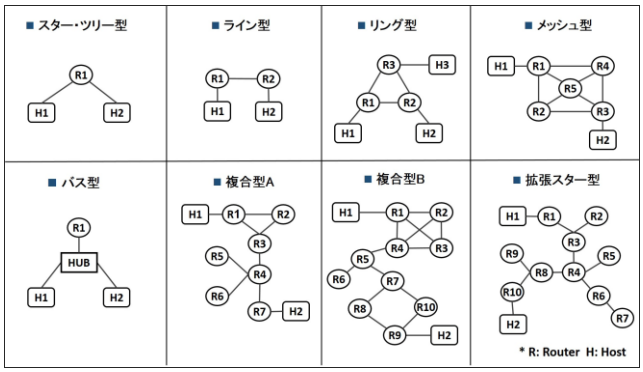

図 11　動作検証に使用したトポロジ
Figure 11　Topologies for evaluation

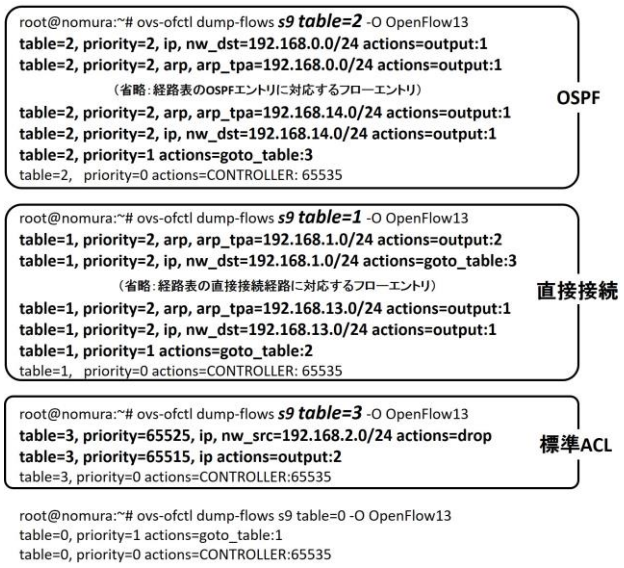

図 12　ルータ R9 の設定情報を変換した
OFS9 のフローエントリ
Figure 12　Flow entries of OFS9 (corresponding to R9)

を用いることにより，複数のルーティングプロトコルが動作する従来型のネットワークにおいても短時間で OpenFlow ネットワークへ移行できるといえる．ただし，本機能では設定情報の取得に NETCONF を用いており，移行対象の各ルータに対して SSH による接続が必要となる．そのため，ネットワーク規模の違いにより OpenFlow ネットワークへの移行時間が変動すると考えられる．今後，図 11 のトポロジに加え，さらに規模が大きく複雑なトポロジにおいても実験，評価を行う予定である．

6.2 ホップバイホップ型 OpenFlow ネットワークへの移行支援機能のフローエントリ数に関する評価

次に，開発したシステムを用いて変換したフローエントリ数に関する考察を行う．今，あるルータの経路表のエントリ数を P，ACL のルール数を Q とし，このルータを OpenFlow スイッチに移行する場合を考える．この場合，まず，4.3.1 節で述べたように経路表から IP と ARP パケット処理のために 2P 個のフローエントリが作成される．また，4.3.2 節で述べたように ACL の設定情報から Q 個のフローエントリが作成される．さらに，4.3 節で述べたようにフローテーブルを順に遷移するために，フローテーブル 0 からフローテーブル 3 にそれぞれ遷移用のフローエントリが作成される（合計 4 個）．加えて，OpenFlow 1.3 では，Packet-In メッセージを OpenFlow コントローラに送信するフローエントリがあらかじめプライオリティ 0 で全てのフローテーブルに書き込まれている（合計 5 個）．したがって，移行後の OpenFlow スイッチにおけるフローエントリ数は合計で 2P+Q+9 となる．すなわち，経路表のエントリ数と ACL のルール数に比例してフローエントリ数が増える．

フローエントリ数に関する検証を行うため，図 11 に示す全てのトポロジに対して移行後の OpenFlow スイッチのフローエントリ数を計測した．その結果，2P+Q+9 で計算される数のフローエントリが作成されていることを確認した．例えば，経路表のエントリ数および ACL のルール数が最も多い複合型トポロジ B のルータ R9 は，経路表のエントリ数 P が 15，ACL のルール数 Q が 2 であり，対応する OFS9 のフローエントリ数は 41 であった．なお，今回の検証でフローエントリ数が最も少ない OpenFlow スイッチのフローエントリ数は 14 であった．

6.3 オーバーレイ型 OpenFlow ネットワークへの移行支援機能の検証および考察

次に，本機能を用いることにより，従来型のネットワークからオーバーレイ型 OpenFlow ネットワークへ移行できるかについて動作検証を行った．

図 13 に実験で使用したトポロジを示す．本実験では，2 台の物理サーバを使用した（ホスト 1. CPU: Core i7 950, Mem: 6 GB, OS: CentOS-7, ホスト 2. CPU: Core i3 540, Mem: 4 GB, OS: CentOS-7）．各物理サーバは KVM により仮想化され，各サーバ上で 2 台の仮想マシン（guest OS）が動作しており，各仮想マシンは VLAN によりセグメント分割されているものとする．また，Open vSwitch はブリッジとして使用され，物理サーバの物理 NIC に接続されている．

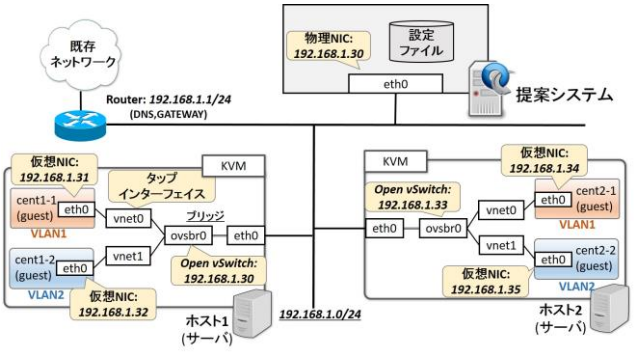

図 13　実験トポロジ
Figure 13　Experimental topology

仮想サーバの仮想 NIC は guest OS のタップインターフェイスに接続されている．このような環境の下，まず本システムが動作するサーバ（CPU: Core i5 4460, Mem: 16 GB, OS: CentOS-7）をホスト 1 とホスト 2 が属するネットワークに追加する．次に，本システムを用いてホスト 1 とホスト 2 の設定情報を自動的に取得し，その情報を基にオーバーレイネットワークを自動で構築する．その後，Open vSwitch 間のオーバーレイネットワークが正しく構築されているかを同一セグメントに属する guest OS 間で ICMP パケットを送受信することにより確認する．この結果，同一セグメントに属する guest OS 間でのみ ICMP パケットを送受信できることを確認した．さらに上記の実験に加え，図 13 と同一のトポロジで，VLAN によりセグメント分割されていない guest OS が存在するネットワークにおいても同様の実験を行った．その結果，同一セグメントに属する guest OS 間でのみ ICMP パケットを送受信できることを確認した．

また，フローエントリ数を計測したところ，各 Open vSwitch に対して，セグメントごとに IP と ARP 用に 2 個のフローエントリが作成されていることを確認した．

これらの結果から，本機能を用いることにより，従来型のネットワークから，同等のパケット制御を行うオーバーレイ型 OpenFlow ネットワークへ自動的に移行できることを確認した．手動で移行する場合は設定ミスが発生する可能性があることから，本機能は有用であると考えられる．

7. おわりに

本稿では，従来型のネットワークから OpenFlow ネットワークへの移行を支援するシステムについて述べた．本システムは，ホップバイホップ型およびオーバーレイ型 OpenFlow ネットワークへの移行に対応している．

ホップバイホップ型 OpenFlow ネットワークへ移行する場合，従来型のネットワーク上のルータから設定情報を取得し，その設定情報を OpenFlow ネットワークへ自動的に反映する．本稿では，本機能の動作検証およびフローエントリ数に関する評価を行った．その結果，最大 10 台のルータから構成される従来型のネットワークと同じパケット処理を行う OpenFlow ネットワークを 6 分以内で構築でき，フローエントリ数は最大で 41 個であった．

オーバーレイ型 OpenFlow ネットワークへ移行する場合，従来型のネットワークのエンドポイント上のサーバから仮想マシンと仮想スイッチの設定情報の取得，取得した設定情報に基づき，オーバーレイネットワークの構築，OpenFlow コントローラの設定，を自動で行う．本稿では，動作検証およびフローエントリ数の評価を行い，本機能を用いることにより，従来型のネットワークからオーバーレイ型 OpenFlow ネットワークに移行できることを示した．

今後の課題として，ホップバイホップ型 OpenFlow ネットワークへの移行支援機能には，OpenFlow スイッチの実機を用いて構築した実 OpenFlow ネットワーク環境での動作検証，ルータから取得可能な設定情報の追加，L2 スイッチへの対応が挙げられる．また，オーバーレイ型 OpenFlow ネットワークへの移行支援機能には，OpenStack などを用いて構築された複雑な仮想ネットワークへの対応，KVM 以外の他のハイパーバイザへの対応が挙げられる．

参考文献

[1] D. Kreutz, F. M. V. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig. Software-Defined Networking: A Comprehensive Survey. 2015, Proceedings of the IEEE, vol. 103, no. 1, p. 14–76.

[2] A. Blenk, A. Basta, M. Reisslein, and W. Kellerer. Survey on Network Virtualization Hypervisors for Software Defined Networking. 2016, IEEE Communications Surveys & Tutorials, vol. 18, no. 1, p. 655–685.

[3] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. OpenFlow: Enabling Innovation in Campus Networks. 2008, ACM SIGCOMM Computer Communication Review, vol. 38, no. 2, p. 69-74.

[4] “Migration Use Cases and Methods”. 2014, https://www.opennetworking.org/images/stories/downloads/sdn-resources/use-cases/Migration-WG-Use-Cases.pdf, (参照 2016-10-26).

[5] “The Future of Networking, and the Past of Protocols”. 2011, http://opennetsummit.org/archives/oct11/shenker-tue.pdf, (参照 2016-10-26).

[6] D. Levin, M. Canini, S. Schmid, F. Schaffert, and A. Feldmann. Panopticon: Reaping the Benefits of Incremental SDN Deployment in Enterprise Networks. 2014, Proceedings of the 2014 USENIX Annual Technical Conference, p. 333-345.

[7] 野村圭太，堤啓彰，谷口義明，井口信和．ルータの設定を OpenFlow ネットワーク反映可能とするシステムの開発. 2015, 情報処理学会全国大会講演論文集, vol. 77, no. 3, p. 359-360.

[8] K. Nomura, Y.Taniguchi, N. Iguchi, K. Watanabe. A System for Supporting Migration to Overlay OpenFlow Network using OpenStack. 2016, Proceedings of the 2016 10th International Conference on Complex, Intelligent, and Software Intensive Systems, p. 595-598.

[9] T. Nelson, A. D. Ferguson, D. Y, R. Fonseca, and S. Krishnamurthi. Exodus: Toward Automatic Migration of Enterprise Netwrok Configurations to SDNs, 2015, Proceedings of the 1st ACM SICOMM Symposium on Software Defined Netwroking Researh, no. 13, p. 1-7.

[10] “Network Configuration Protocol”. http://tools.ietf.org/html/rfc6241, (参照 2016-10-26).

[11] “BIGLOBE の SDN に対する取り組み”. 2013, https://www.jaipa.or.jp/event/oki_ict2013/biglobe_taguchi_130627.pdf, (参照 2016-10-26).